Министерство науки и высшего образования 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 НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ

НА ТЕМУ:

Обработка и анализ данных принятых с борта космического аппарата

Студент __ИУ5-31М_____ _____ ___С.А.Чупис_____
(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель курсовой работы _____ ___Ю.Е. Гапанюк___
(Подпись, дата) (И.О.Фамилия)

2024 г.

Министерство науки и высшего образования 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)

УТВЕРЖДАЮ
Заведующий кафедрой ___ИУ5____
(Индекс)
_____ ___В.И. Терехов__
(И.О.Фамилия)
«_04_» ____сентября_____ 2024 г.

З А Д А Н И Е
на выполнение научно-исследовательской работы

по теме _____обработка и анализ данных принятых с борта космического аппарата_____
_____

Студент группы _ИУ5-31М__

_____Чупис Сергей Александрович_____
(Фамилия, имя, отчество)

Направленность НИР (учебная, исследовательская, практическая, производственная, др.)
_____ИССЛЕДОВАТЕЛЬСКАЯ_____
Источник тематики (кафедра, предприятие, НИР) _____КАФЕДРА_____

График выполнения НИР: 25% к ____ нед., 50% к ____ нед., 75% к ___ нед., 100% к ____ нед.

Техническое задание _Выполнить обработку и анализ данных принятых с борта космического аппарата_____
_____
_____
_____
_____

Оформление научно-исследовательской работы:

Расчетно-пояснительная записка на _38_ листах формата А4.
Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)
_____
_____
_____
_____

Дата выдачи задания «_04_» ____сентября_____ 2023 г.

Руководитель НИР _____ ____Ю.Е. Гапанюк______
(Подпись, дата) (И.О.Фамилия)
Студент _____ _____С.А. Чупис_____
(Подпись, дата) (И.О.Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

Содержание

ВВЕДЕНИЕ

В современном мире колоссальное внимание уделяется информационным данным, которые поступают из различных источников. Ресурсов, с которых приходит информация существует огромное количество. К ним относятся интернет-сервисы, датчики и устройства, такие как смартфоны, умные дома и другие автоматические системы, базы данных, телевидение, радио и другие. Все эти источники могут содержать различные типы информации, которые разделяются по типу восприятия: визуальная, звуковая, тактильная, обонятельная, вкусовая. Также информация может разделяться по форме представления, а именно: текстовая, графическая, числовая и звуковая [1]. У каждого человека лучше усваивается тот или иной тип информации, однако, так как 80% сведений об окружающем мире мозг получает именно через зрение, самым выгодным типом представления будет графический.

К данным также относятся и те, что приходят с космических аппаратов. Они используются для разнообразных целей, таких как исследование космического пространства, навигация, разведка и другие. В связи с задачами, которые преследует тот или иной космический аппарат, на нем установлено множество датчиков, которые считывают определенную информацию. В данном случае таким аппаратом является “Канопус-В”, с которого были считаны данные о разных параметрах и переданы на Землю, а именно в Наземный Измерительный Пункт, где они проходят некую обработку. Далее эта информация передается в Центр Управления Полетами, где, после нескольких этапов обработки, получается двоичный файл, в котором зашифрованные результаты анализа Телеметрической Информации. Схема функционирования Центра Управления Полётами показана на рисунке 1. Также на этом рисунке указан один из файлов, который будет на входе программы, разработанной в ходе выполнения выпускной квалификационной работы.

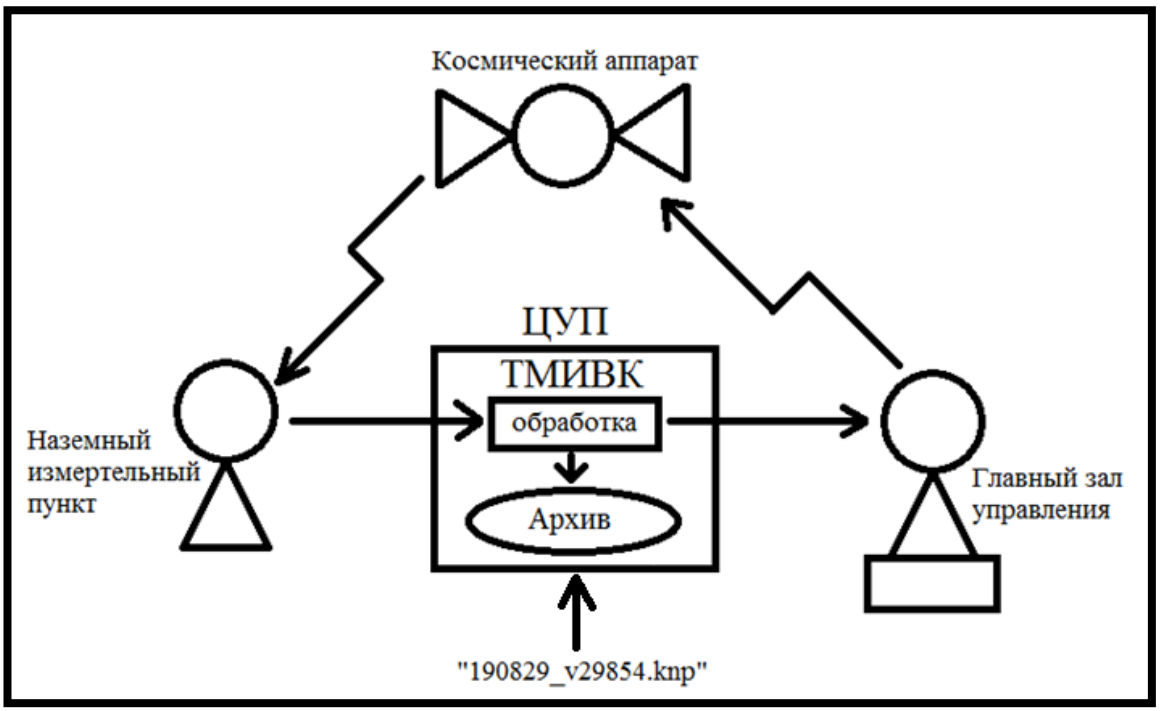

Рисунок 1 – Схема функционирования Центра Управления полётами

Сформированные таким образом файлы используются в отделе Телеметрического Информационно-Вычислительного Комплекса для того, чтобы анализировать эти данные и делать определенные выводы о работе космического аппарата.

Информация в этом файле отображает состояние Космического аппарата за определенный промежуток времени, называемый сеансом связи, который составляет в среднем от десяти до двадцати минут. За это время успевает передаться огромный объем информации, так как установленных датчиков на космическом аппарате сотни и практически все они считывают информацию несколько десятков раз в секунду, а некоторые с частотой пятьдесят герц. Поэтому визуальный просмотр этих данных, как и их анализ, специалистами по управлению Космических Аппаратов, становится трудновыполнимой задачей. Также сложность заключается в том, что рассматриваемый файл является двоичным. В связи с этим, логичным решением будет визуальное отображение информации на графике, что и является задачей данной выпускной квалификационной работы.

1. Общетехническое обоснование

1.1 Цель

Целью работы является доработка программы, для построения графика по времени для данных, принятых с борта космического аппарата.

Для написания выпускной квалификационной работы было необходимо:

1) Изучить структуру входных файлов.

2) Изучить Java-классы и библиотеки, которые наилучшим образом подходят для решения поставленных задач.

3) Провести обзор различных сред разработок и платформ для программирования.

4) Оценить и выбрать оптимальную стратегию для реализации данного приложения.

5) Выбрать СУБД для структурированного хранения данных

1.2 Постановка задачи

Для того, чтобы разработать полезную программу с простым интерфейсом, необходимо выполнить следующие задачи:

1) Разработать структуру (класс) для внутреннего хранения значений функциональных параметров;

2) Из входного двоичного файла с результатами обработки телеметрической информации выбрать телеметрические записи, а также записи, которые были переданы в режиме Непосредственной Передачи;

3) Для каждого параметра сохранить множество значений в разработанной структуре;

4) Изучить Java-классы для работы с графическим интерфейсом пользователя (ГИП);

5) Структурировать полученные данные в СУБД

6) Предоставить пользователю список функциональных параметров с помощью ГИП;

7) Обеспечить пользователю выбор параметра, подлежащего выводу на график;

8) Вывести на график собранные значения для выбранного пользователем параметра.

Из задач, представленных выше, можно выделить отдельные части программы, которые в совокупности будут представлять собой полноценный проект, выполняющий все поставленные цели:

1) Классы обработчики – это классы, предназначенные для обработки входных файлов, а именно: “Dimens.ion”, “KNP-173_14_33_58_dat.xml”, “190829_v29854.knp”, “Config.xml”, и выделения из них необходимой информации.

2) Классы, отвечающие за создание и демонстрацию графического интерфейса на экран.

3) Передачу и структурирование данных в СУБД

1.3 Телеметрия

Для того, чтобы начинать разработку приложения необходимо понимать, что такое телеметрия. Телеметрия – это научное направление, занимающееся разработкой и использованием телеметрических систем – комплексов автоматизированных устройств, которые позволяют удаленно контролировать объекты и процессы путем сбора, передачи, обработки и регистрации информации об измерениях и различных событиях. Телеметрические системы имеют классификацию. Это необходимо для того, чтобы получить представление о той или иной телеметрической системе, а также чтобы охарактеризовать соответствующую телеметрическую систему. Одной из основных классификаций телеметрических систем является классификация по

взаимному расположению передатчика и приёмника [2]. По этому признаку телеметрические системы разделяются на пять классов:

1) Стационарные телеметрические системы – объект изучения находится на удалении от стационарного передатчика и связан с ним проводами. Объект исследования, передатчик и приёмник неподвижны.
2) Транспортируемые телеметрические системы – объект исследования связан с передатчиком с помощью провода. Вместе они находятся на носителе и неподвижны относительно друг друга, однако носитель перемещается относительно приёмника.
3) Динамические телеметрические системы – передатчик закреплен на объекте исследования и перемещается вместе с ним относительно приёмника.
4) Встраиваемые телеметрические системы – передатчик находится внутри объекта и неподвижен относительно его станины.
5) Вводимы телеметрические системы – передатчик находится внутри деталей объекта и перемещается относительно его узлов и приёмника. Объект обычно неподвижен.

В данном случае объектом сбора данных является космический аппарат “Канопус-В”. Он является телеметрической системой второго класса, а именно транспортируемой телеметрической системой. На нём установлено огромное количество разнообразных датчиков, измеряющих определенные параметры. В определенные промежутки времени, называемые сеансами связи, информация с борта передаётся на Землю, где спустя некоторое количество обработок перерабатывается в двоичный файл, содержащий результаты обработки Телеметрической Информации в виде Телеметрических Записей. Записи имеют определенную структуру, разработанную в отделе Телеметрического Информационно-Вычислительного комплекса. Структура представлена в таблице 1. Слева указан номер байта телеметрической записи, сверху номер бита в байте.

Таблица 1 – Структура телеметрической записи

<table>
<tr><td></td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>0
1</td><td colspan="8">Номер параметра</td></tr>
<tr><td>2
3
4
5</td><td colspan="8">Время формирования параметра</td></tr>
<tr><td>6</td><td colspan="8">Размерность</td></tr>
<tr><td>7</td><td colspan="4">Атрибут</td><td colspan="4">Тип значения:
0 – “Long” 32-х разрядное целое
1 – “Double” 64-х разрядное целое
2 – “Code” кодовое значение с длиной
3 – “Point” массив данных с длиной</td></tr>
<tr><td>8
9
10
11
12
13
14
15
…</td><td colspan="8">Значение параметра (8 – 4096 байт)</td></tr>
</table>

Структура Телеметрической записи состоит из нескольких полей, а именно:

1) “Номер параметра” – у каждого параметра есть свой номер, по которому можно понять, что это за параметр, какое у него название, описание и так далее. Это поле занимает два байта.

2) “Время формирования параметра” – содержит время формирования параметра в миллисекундах от начала текущих суток. Поле занимает четыре байта.

3) “Размерность” – обозначает тип параметра для его отображения и документирования.

4) “Атрибут” – отображает статус полученной Телеметрической Записи

5) “Тип значения” – занимает восемь бит и показывает какой тип значения имеет запись. Существуют четыре типа значения параметра: Long, Double, Code и Point. Функциональными параметрами считаются те, у которых тип значения “Double”.

6) “Значение параметра” – занимает от 8 до 4096 байт, структура этого поля показана в таблице 2. Слева указывается тип значения, сверху номер байта телеметрической записи.

Таблица 2 – Структура поля “Тип значения”

Тип значения	8	9	10	11	12	13	14	15
Long	Не используется				32-х разрядное целое			
Double	64-х разрядное целое							
Code	Не используется		Длина кода		32-х разрядное целое			
Point	Размер элемента		Длина последовательности в байтах		Последовательность байт			

У всех типов значения отведенные им 8 байт кодируются по-разному, а именно:

- “Long” – первые четыре байта не используются, следующие четыре байта означают 32-х разрядное целое число.
- “Double” – все восемь байт используются для обозначения 64-х разрядного вещественного числа.
- “Code” – первые два байта не используются, 10-11 байты обозначают длину кода, последние четыре байта обозначают 32-х разрядное целое число.
- “Point” – первые два байта обозначают размер элемента, далее два байта обозначают длину последовательности в байтах, далее, начиная с двенадцатого байта телеметрической записи начинается последовательность байт, длина которой указана в десятом и одиннадцатом байтах.

Так как функциональным из них является только тип Double, то записи с другим типом необходимо будет пропустить, а понимание структуры записей позволит сделать это.

Также среди телеметрических записей присутствуют записи, называемые служебными сообщениями. Их структура показана в таблице 3. Слева указан номер байта служебной записи, сверху номер бита в байте.

Таблица 3 – Структура телеметрической записи “Служебное сообщение”

	7	6	5	4	3	2	1	0
0	0xFFFF							
1								
2	Время формирования параметра							
3								
4								
5								
6	Тип сообщения							
7	Тип значения							
8	Сообщение (длиной в 8 (или 24) байт)							
9								
10								
11								
12								
13								
14								
15								
…								

Структура служебного сообщения состоит из нескольких полей, а именно:

1) Первые два байта всегда у служебных записей всегда одинаковые и равны 0xFFFF в шестнадцатеричном виде.

2) “Время формирования параметра” – содержит время формирования параметра в миллисекундах от начала текущих суток. Поле занимает четыре байта.

3) “Тип сообщения” – может принимать значения от нуля до 6 включительно. Подробнее показано в таблице 4.

4) “Тип значения” – отражает тип значения.

5) “Сообщение” – может принимать длину восемь или двадцать четыре байта. Структура первых восьми байтов поля “Сообщение” показана в таблице 4. Слева указан номер байта служебной записи, сверху номер бита в байте.

Таблица 4 – Структура поля “Тип сообщения”

Тип сообщения	0	1	2	3	4	5	6	7
0 – пустая посылка	Не используется							
1 – начало сеанса	Не используется		Длина в байтах		Сообщение “начало сеанса”			
2 – текущее время	Текущая дата в сутках от 06.01.1980				Не используется			
3 – конец сеанса	Не используется				E	N	D	!
4 – смена режима	Не используется				Номер режима (целое)			
6 - ошибка	Не используется				Номер ошибки (целое)			

Числовое поле “номер режима” имеет следующие значения:

- 0 – недостоверный режим;
- 1 – режим непосредственной передачи. Это означает, что информация с борта космического аппарата сразу, или с некоторой задержкой передается на Землю. В итоге данные, получаемые при таком режиме передачи идут последовательно друг за другом.
- все остальные режимы – воспроизведения. Это режимы, при которых данные, накапливаются на внутренних носителях космического аппарата, а при передачи этих данных на Землю они могут приходить в обратном порядке.

Было принято решение ограничиться именно режимом непосредственной передачи.

Числовое поле “номер ошибки” имеет следующие значения:

- 1001 – ошибка в запросе;
- 1003 – нет требуемого сеанса;
- 1004 – ошибка загрузки данных;
- 1010 – потеря источника информации.

Ниже, в таблице 5 показан состав поля “Сообщение” для типа сообщения “начало сеанса”, длина кода равна 24 байтам. Слева указан относительный номер байта ТМ-записи (первого байта в строчке из 8 байтов), сверху – относительный номер байта в строчке.

Таблица 5 – Состав поля “Сообщение” телеметрической записи “Служебное сообщение” для типа сообщения “начало сеанса”

	0	1	2	3	4	5	6	7
8	Не используется		Длина в байтах		Имя борта (текст с завершающим нулём)			
16	Номер борта	Номер витка (текст)						
24	Имя исходных данных						Номер версии 1	Номер версии 2

1.4 Аналоги разработки

В центре управления полётами уже существует программа, которая решает задачу, поставленную в теме данной выпускной квалификационной работы. Такой программой является ТМ-Graph. Она разработана специалистами телеметрического информационно-вычислительного комплекса и имеет

широкий функционал. Однако, она имеет серьезный недостаток, а именно – выбор языка программирования C#, который был выбран для разработки.

Язык программирования C# был разработан компанией Microsoft в 2000 году и с тех пор стал одним из самых популярных языков программирования в мире [3]. Одним из главных преимуществ C# является его удобство в использовании. Синтаксис C# похож на язык программирования Java, но он более читабелен и более прост в использовании.

Однако, сравнивая язык программирования C# с Java, можно заметить несколько недостатков. Java, в отличие от C#, выделяется высокой кроссплатформенностью, то есть приложения, написанные на этом языке, могут работать на различных устройствах и операционных системах, будь то Windows, MacOS, Linux или мобильные устройства на базе Android [4]. Это означает, что разработчики могут создавать универсальные приложения, которые не требуют изменений в коде для работы на каждой платформе. Кроме того, Java – это язык программирования с открытым исходным кодом, то есть разработчики могут создавать свои собственные библиотеки [5]. Это дает возможность разрабатывать приложения любой сложности, в том числе и более крупные, и разветвленные. Также Java имеет большую базу исходного кода и сообщества разработчиков, что позволяет быстро решать проблемы и находить готовые решения.

В ходе решения задачи, поставленной в теме выпускной квалификационной работы, было принято решение выбрать язык программирования Java для того, чтобы создать аналог программы, разработанной в отделе телеметрического информационно-вычислительного комплекса, который не будет привязан ни к одной из платформ.

2. Решение задачи

2.1 Выбор среды программирования

Для того, чтобы приступить к разработке программы, необходимо выбрать среду программирования. Так как в теме выпускной квалификационной работы указан язык программирования Java, то рассматриваться будут среды, в которых есть возможность использования этого языка [8]. Рассмотрим самые популярные из них:

1) Eclipse – это интегрированная среда разработки программного обеспечения, которая предоставляет различные возможности для разработки, отладки и прототипирования приложений на разных языках программирования, таких как Java, C++, Python и многих других.

2) IntelliJ IDEA – это интегрированная среда разработки (IDE) для различных языков программирования, включая Java, Kotlin, Groovy, Scala, JavaScript и другие. Это мощный инструмент для разработчиков, который предоставляет широкий набор функциональных возможностей, таких как средства для анализа кода, создания автоматических тестов, интеграции с системами контроля версий и многое другое. IntelliJ IDEA доступен в двух версиях: Community Edition (бесплатная версия с открытым исходным кодом) и Ultimate Edition (платная версия с расширенным функционалом).

3) NetBeans – это интегрированная среда разработки (IDE) с открытым исходным кодом, предназначенная для различных языков программирования, таких как Java, HTML, CSS, JavaScript и другие. Она предлагает широкий набор функциональных возможностей для разработки приложений, таких как автоматическое заполнение кода, инструменты для отладки и профилирования, поддержка работы с базами данных и многое другое.

4) jEdit – это бесплатный текстовый редактор с открытым исходным кодом, разработанный на языке Java. Он предназначен для редактирования кода на различных языках программирования, включая Java, C++, HTML, XML, PHP

и другие. jEdit имеет множество функций, которые делают его одним из наиболее популярных текстовых редакторов для программистов, таких как подсветка синтаксиса, авто-завершение кода, авто-отступы, множественная отмена и возврат действий, работа с большим количеством файлов и многое другое.

5) Visual Studio Code – это бесплатный текстовый редактор с открытым исходным кодом, разработанный компанией Microsoft. VS Code предназначен для редактирования кода на различных языках программирования, таких как JavaScript, TypeScript, Java, Python, C# и другие.

Сравнив самые популярные интегрированные среды программирования, было принято решение остановиться на IntelliJ IDEA [9], так как эта среда имеет ряд преимуществ:

1) Удобный и интуитивно понятный интерфейс – IntelliJ IDEA предоставляет удобный пользовательский интерфейс и множество опций настройки для удобства работы с проектом.

2) Автодополнение и быстрые корректировки кода – в IntelliJ IDEA есть множество функций автодополнения, корректировки и проверки кода, которые помогут ускорить работу разработчика.

3) Анализ кода и тестирование – IntelliJ IDEA обеспечивает инструменты для анализа кода и тестирования, что позволяет легко обнаруживать ошибки в проекте.

4) Улучшенная производительность – благодаря оптимизации производительности, IntelliJ IDEA позволяет работать с проектами большого размера и объема.

5) Множество плагинов – IntelliJ IDEA имеет множество плагинов, которые расширяют функциональность IDE и позволяют индивидуализировать работу с проектом в соответствии с потребностями разработчика.

2.2 Идейная реализация приложения

Перед началом разработки приложения нужно определиться, какой функционал оно будет иметь. В основе программы будет лежать построение графиков, что указано в теме выпускной квалификационной работы.

Для того, чтобы можно было построить график, необходимо иметь входные данные, после разбора которых появится возможность структурировать их в виде графика. В качестве входных данных будут четыре файла, а именно:

1) “Dimens.ion” – файл, содержащий соответствие номеров и текстов физических размерностей значений параметров. Этот файл является универсальным для космических аппаратов.

2) “KNP-173_14_33_58_dat.xml” – файл с данными о параметрах (соответствие имени и номера параметра, полное имя параметра, текстовые значения параметра и другое).

3) “190829_v29854.knp” – двоичный файл и результатами обработки телеметрической информации. Он содержит последовательность телеметрических записей.

4) “Config.xml” – файл, в котором ранятся названия всех входных файлов.

Для того, чтобы решить поставленную задачу, необходимо разработать ряд классов, каждый из которых будет выполнять определенные функции. Потребуется четыре класса для разбора входных файлов, как минимум один класс для хранения обработанных данных, а также классы, реализующие графический интерфейс пользователя.

Графический интерфейс пользователя должен представлять собой окно, в котором реализован следующий функционал:

1) Возможность выбора параметра, от которого требуется построить график.

2) Построение графика выбранного параметра от времени.

3) Вывод информации о выбранном параметре в текстовом формате.

4) Возможность вывода нескольких параметров на график.

5) Удаления параметров с графика.

6) Возможность отображения графиков параметров в двух режимах: линейном (точки соединяются прямыми линиями) и ступенчатом (точки соединяются в виде “лесенки”).

7) Возможность открытия и обработки других файлов.

2.3 Разработка алгоритма разбора входных файлов

Перед тем, как начинать разработку графического интерфейса, необходимо произвести разбор входных файлов, для того, чтобы полученные данные использовать в качестве наполнения графического интерфейса. Всего имеется 4 входных файла. Для разбора каждого файла будет разработан отдельный класс.

Первым необходимо обработать файл “Config.xml”, так как он содержит в себе названия других входных файлов. Для того, чтобы использовать названия этих файлов в других классах необходимо создать три поля типа String и применить к ним модификатор public. Также у данного класса нужно сделать конструктор, в котором происходит разбор файла и запись названий других файлов в вышеупомянутые переменные.

Далее необходимо обработать файл “Dimens.ion”, который содержит размерности. В данном классе необходимо создать два поля, первое поле типа String, в которое запишем имя входного файла и второе поле типа TreeMap. Этот класс реализует интерфейс Map, используя дерево. TreeMap обеспечивает эффективное средство хранения пар ключ/значение в отсортированном порядке и позволяет быстро извлекать данные. Этот класс шаблонный, в данном случае первым шаблоном будет тип Integer, а вторым – String. В этом поле будут храниться соответствие размерностей и номеров строк, на которых написаны эти размерности в файле. Отсортированы эти значения будут по номерам строк. Также, в этом классе присутствует конструктор, который производит

инициализации переменных, и метод “load”, производящий обработку входного файла “Dimens.ion”, и заполняющий поле типа TreeMap.

Следующим нужно создать класс, который будет обрабатывать файл “KNP-173_14_33_58_dat.xml”. В нем также, как в предыдущем классе, будет два поля, но в поле типа TreeMap, в качестве второго шаблона будет выступать другой класс, а именно класс Strings, содержащий два поля, которые отвечают за имя и описание параметра. Также у данного класса-обработчика файла будет 3 метода: метод load, метод retName и метод retDesc. Первый метод отвечает за обработку входного файла и наполнения поля типа TreeMap, второй метод возвращает имя параметра по номеру параметра, третий метод возвращает описание параметра по номеру параметра.

Последний из классов-обработчиков – класс, который обрабатывает двоичный файл с ТМ-записями, а именно “190829_v29854.knp”. Этот класс также, как предыдущий, содержит в себе 2 поля, первое – имя входного файла, а второе – объект класса TreeSet [10]. Класс TreeSet также является шаблонным, в качестве шаблона будет выступать класс TmDat, который будет рассмотрен далее. Выше говорилось, что все объекты в классе отсортированы по возрастанию, в данном случае сортировка будет воспроизводиться по имени параметра, то есть все записи будут расположены в алфавитном порядке. Также в этом классе будут конструктор и ряд методов, а именно:

1) Конструктор, в котором будет производиться инициализация поля с типом TreeSet и поля с типом String, в которую записывается имя входного файла.

2) Метод Ret_Str, который будет возвращать сформированную строку, для вывода информации о параметре в текстовом виде (запрос будет осуществляться по имени параметра).

3) Метод Ret_TmDat, который будет возвращать определенный объект из поля с типом TreeSet (запрос будет осуществляться по имени параметра).

4) Метод run, который будет реализовывать обработку входного файла.

Для того, чтобы информацию из файла “190829_v29854.knp” грамотно хранить, необходимо разработать структуру для хранения.

2.4 Разработка структуры для хранения информации о параметрах

Храниться обработанные данные будут в СУБД MySQL, так как данная СУБД является самой популярной среди пользователей и имеет достаточный функционал для использования в моей разработке, а также она изучалась мной ранее и мне знакома.

Для хранения полученных данных удобнее всего использовать несколько таблиц, представленных на рисунке 2.

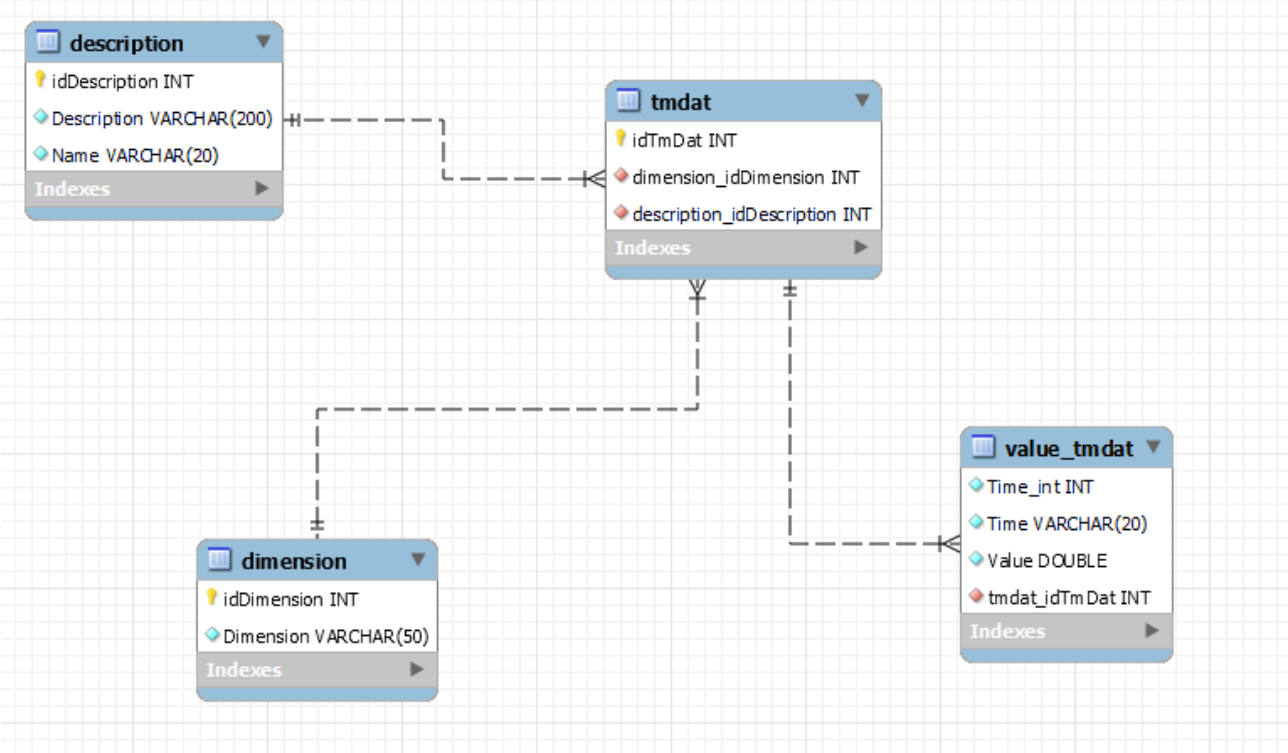

Рисунок 2 – Схема структуры хранения информации о параметрах

Рассмотрим систему хранения данных:

Таблица description – в данной таблице будут храниться имена и описания записей. Данная таблица связана с таблицей TmDat по ключу, который является номером параметра.

Таблица dimension – в данной таблице будут храниться размерностей параметров. Данная таблица связана с таблицей TmDat по ключу, который является номером параметра.

Таблица TmDat – в данной таблице хранятся номера параметров, в также ключи, по которым данная таблица связаны с другими.

Таблица value-tmdat – в данной таблице хранится информация о состоянии какого-либо датчика в определенный момент времени, то есть хранится время в виде числа, время в виде строки, значение параметра, а также ключ, с помощью которого эта таблица связана с таблицей TmDat.

2.5 Разработка графического интерфейса

Для создания графического интерфейса воспользуемся встроенной библиотекой Swing. Swing – библиотека для создания графического интерфейса на языке Java. Программное обеспечение было разработано компанией Sun Microsystems. Оно содержит ряд графических компонентов, таких как кнопки, поля ввода, таблицы и так далее. Также, для создания графиков будет использована библиотека JFreeChart – это библиотека с открытым исходным кодом, разработанная на Java [12]. Главными достоинствами этой библиотеки являются её простота, открытый код, поддержка широкого спектра диаграмм, и обширные настройки графиков, что в данном случае очень важно.

Чтобы сгенерировать основное окно программы нужно создать класс, который необходим для вывода на экран основного окна. Также этот класс будет отвечать за расположение объектов на этом окне и за взаимодействие пользователя с окном. Само окно должно быть разделено на несколько сегментов. В правом сегменте будет располагаться график и все элементы взаимодействия с ним, а в левом – поле с выбором параметра, который должен быть отображен на графике, и текстовое поле, на котором будет отображаться

информация о параметре в текстовом виде. Также должно присутствовать меню, на котором будут вкладки “Файл”, “Настройки графика” и “Справка”. Примерное расположение элементов окна представлено на рисунке 3.

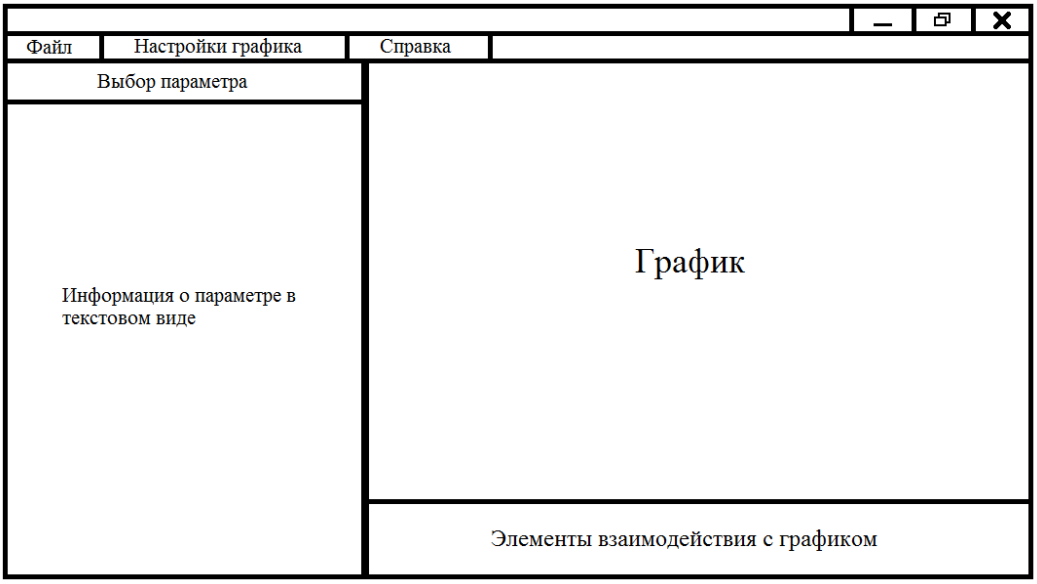

Рисунок 3 – Прототип главного окна

Отталкиваясь от примерного расположения элементов на окне, можно понять, как должен быть написан класс, для создания основного окна программы. На окно были добавлены следующие объекты:

1) Выпадающий список, необходимый для выбора параметра, который будет выводиться на график, а также информацию о нём в текстовом поле слева окна. В списке будут отображаться все параметры, выбранные их входного двоичного файла. То есть в нём не будет параметров, которые не были найдены в файле или параметров, не удовлетворяющих условиям (тип записи “Double” и режим передачи “НП”).

2) Текстовое поле, на котором будет отображена информация о конкретном параметре (или нескольких, в зависимости от количества параметров, отображенных на графике).

3) Поле, на котором будет изображен график. На нём будет присутствовать возможность масштабирования разными способами: прокручиванием колёсика мыши, выделением области левой кнопкой мыши, масштабирование с помощью меню по нажатию правой кнопки мыши.

4) Под графиком будет располагаться поле с выпадающим списком, дублирующим список из первого пункта, однако этот список нужен для добавления новых параметров на график. Также под графиком будет ещё две кнопки: “Добавить на график” и “Удалить последний добавленный график”. Они необходимы для того, чтобы добавить выбранный в нижнем выпадающем списке параметр на график и удалить последний добавленный параметр с графика соответственно.

5) Меню с вкладками “Файл”, “Настройки графика” и “Справка”. По нажатию вкладки “Файл” будет возможность открыть другой входной файл из файловой системы компьютера. Также во вкладке “Файл” будет присутствовать элемент с наименованием “Выход”, по нажатию на который программа завершит работу. На вкладке “Настройки графика” будет присутствовать возможность изменить тип графика со ступенчатого на линейный и обратно (по умолчанию графики имеют ступенчатый тип построения). По нажатию на последнюю вкладку “Справка” будет вызываться диалоговое окно, на котором в текстовом виде будет кратная инструкция по использованию.

Кроме класса, отвечающего за отображения главного окна на экране, необходим класс, который будет преобразовывать данные, находящиеся во внутренней структуре хранения в структуры, предназначенные для отображения на графике. Также методы этого класса будут необходимы для изменения типа графика со ступенчатого на линейный и добавления панели с кнопками под графиком.

Таким образом, в совокупности все эти элементы позволят быстро и эффективно просматривать значения параметров, а также анализировать данные, полученные с борта космического аппарата “Канопус-В”.

2.6 Пример работы программы

Была разработана программа, позволяющая просматривать значения параметров в графическом виде, а также строить графики от нескольких параметров, что облегчит анализ параметров, относительно друг друга.

После того, как программа загружена и все входные файлы обработаны, а данные из этих файлов загружены в СУБД на экране появится основное окно (рисунок 5).

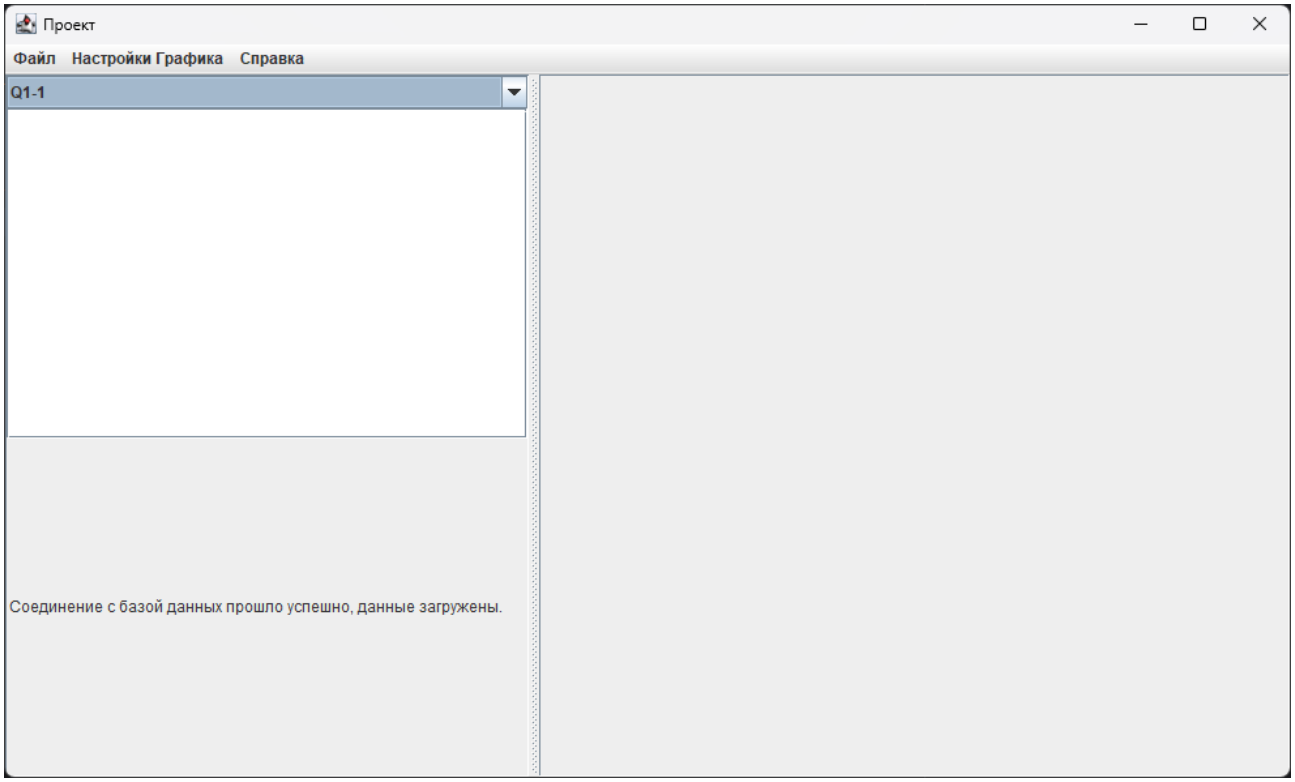

Рисунок 5 – Основное окно программы до выбора параметра

Далее пользователю необходимо выбрать параметр из выпадающего списка, который требуется отобразить на графике (рисунок 6).

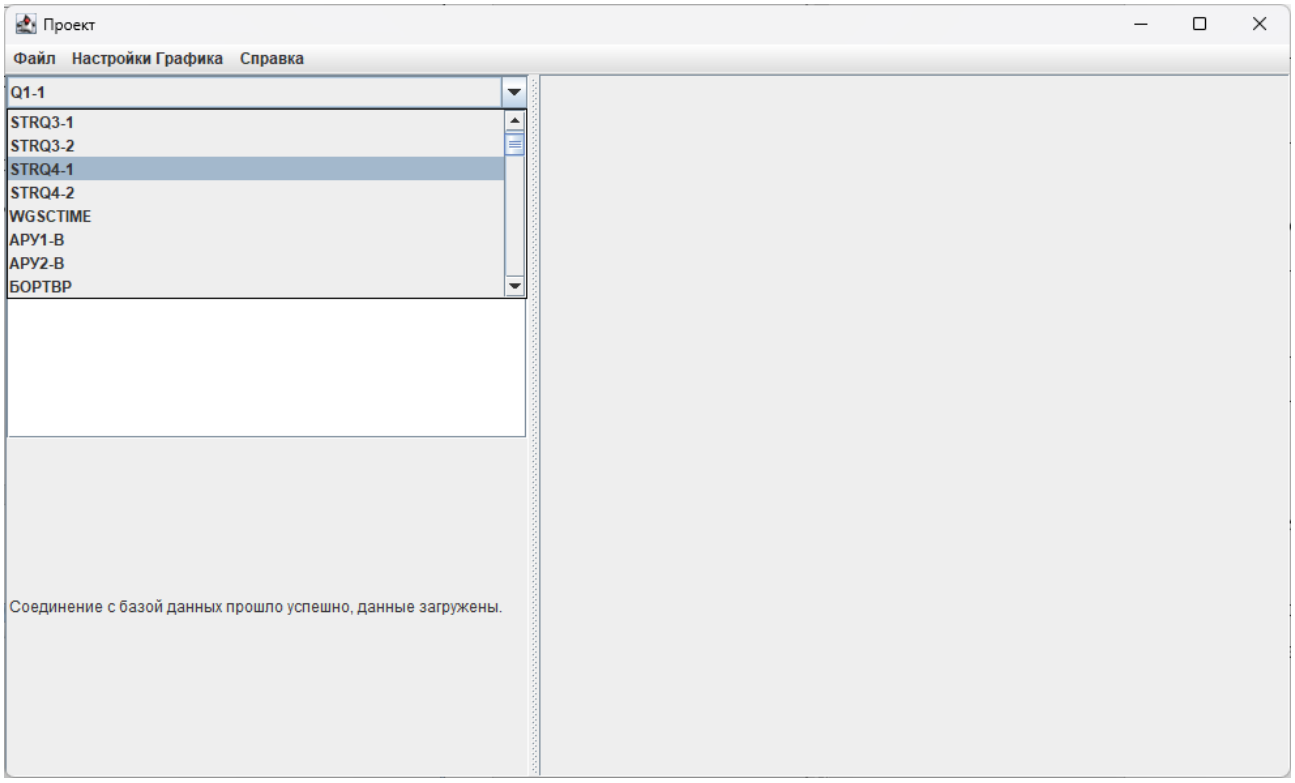

Рисунок 6 – Демонстрация работы выпадающего списка

После выбора параметра на окне произойдут следующие изменения: в правой части появится график с отображенными на нём значениями параметра, под ним появится название графика, отображающее цвет данного параметра, его имя, размерность и описание, также появится панель инструментов в ещё одним выпадающим списком и кнопками “Добавить на график” и “Удалить последний построенный график”. Также в левой части окна на текстовом поле появится информация о выбранном параметре в текстовом виде (рисунок 7).

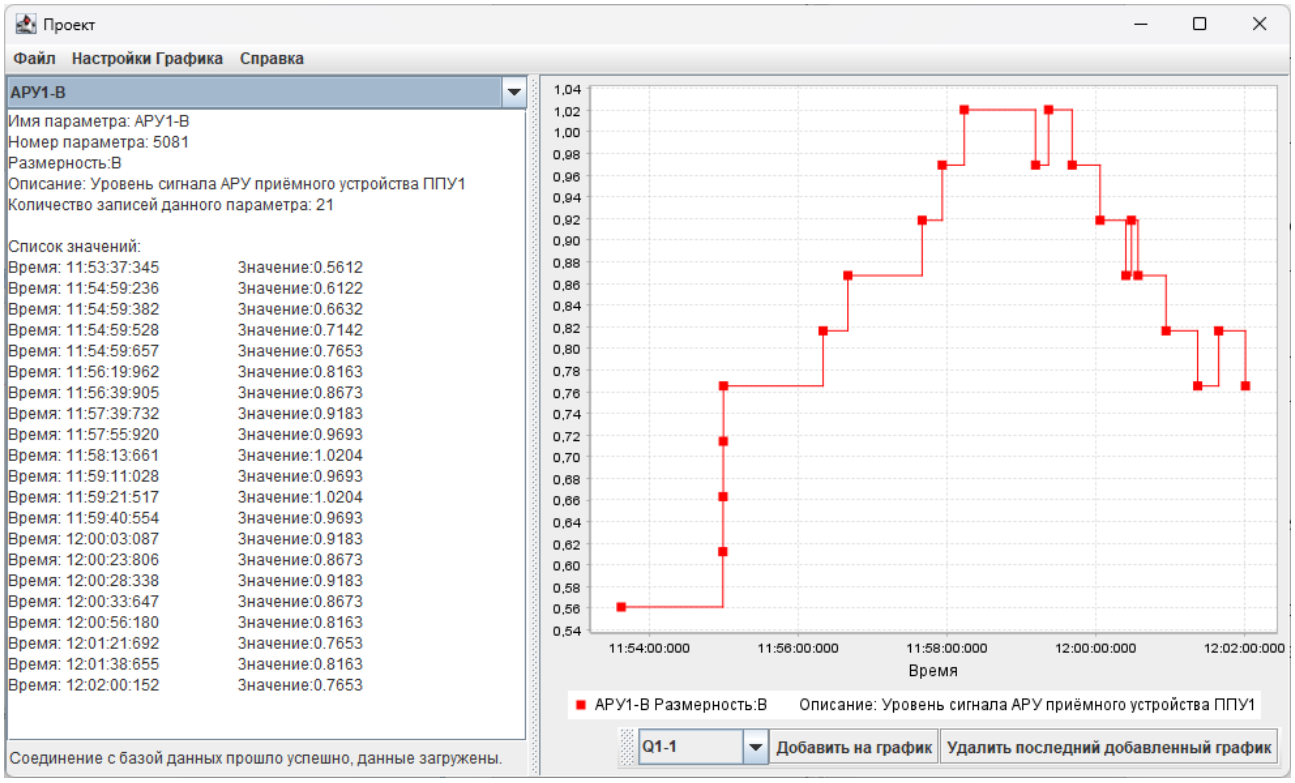

Рисунок 7 – Основное окно после выбора параметра в выпадающем списке

При выборе другого параметра из нижнего выпадающего списка и нажатии на кнопку “Добавить на график”, произойдут следующие изменения: на графике отобразятся значения нового параметра, выделенные другим цветом, также появится его описание под графиком и добавится описание параметра в текстовом виде на панели слева (рисунок 8).

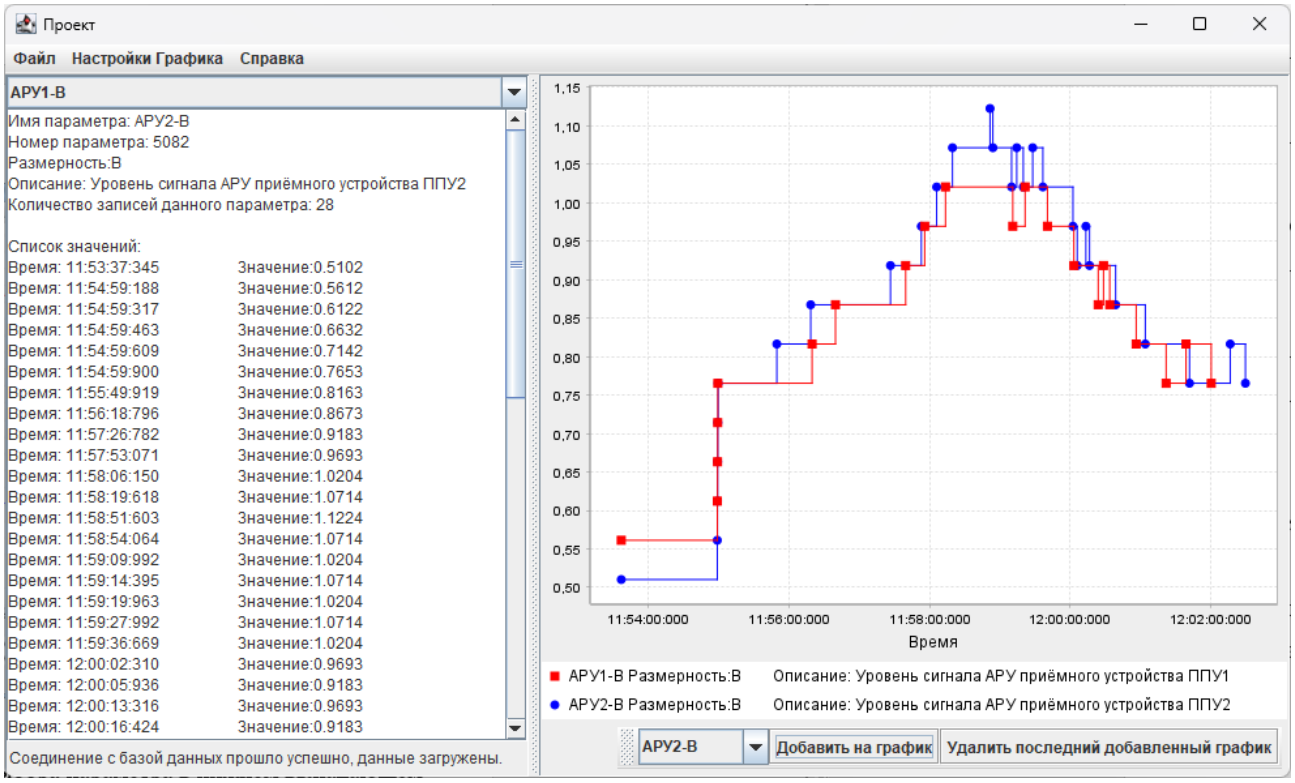

Рисунок 8 – Основное окно после выбора параметра в нижнем выпадающем списке и нажатии на кнопку “Добавить на график”

Для масштабирования графика выделяется область левой кнопкой мыши. Необходимо провести курсором с зажатой левой клавишей мыши от верхнего левого края до правого нижнего края интересующего фрагмента графика (рисунок 9). После этого на графике будет отображена выделенная область (рисунок 10). Если провести с любого другого края, то график вернется в изначальное положение. Также масштабирование работает с помощью прокручивания колёсика мыши.

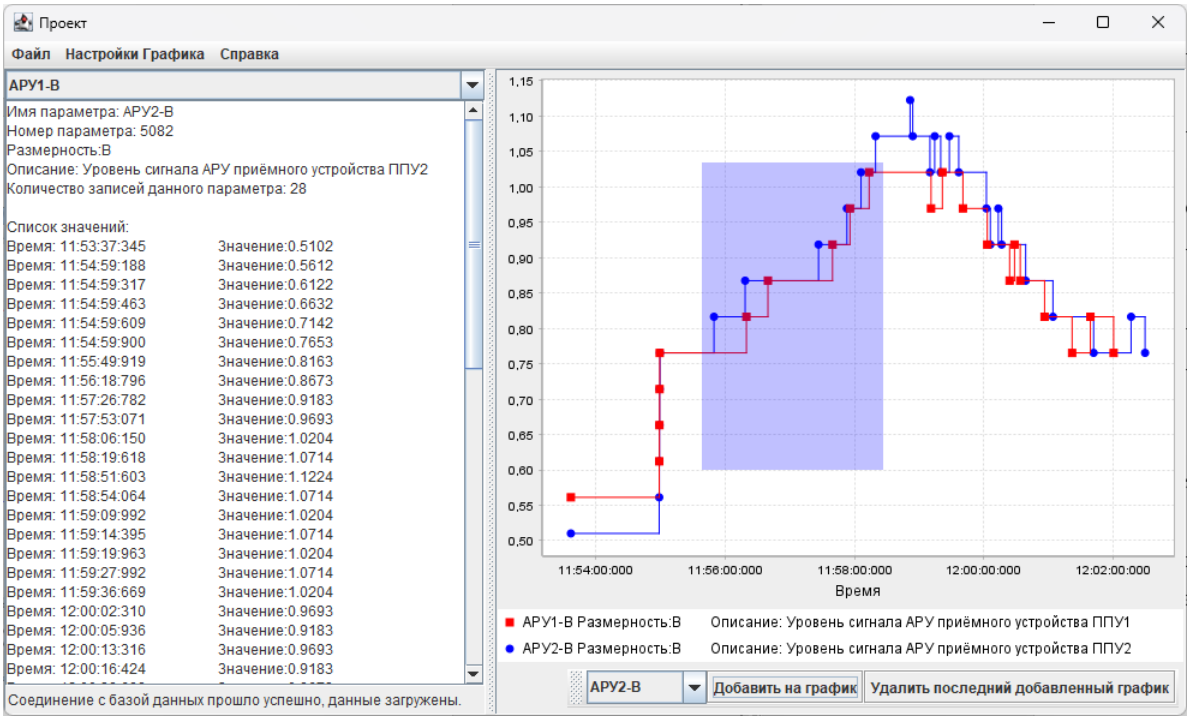

Рисунок 9 – Демонстрация работы масштабирования графика

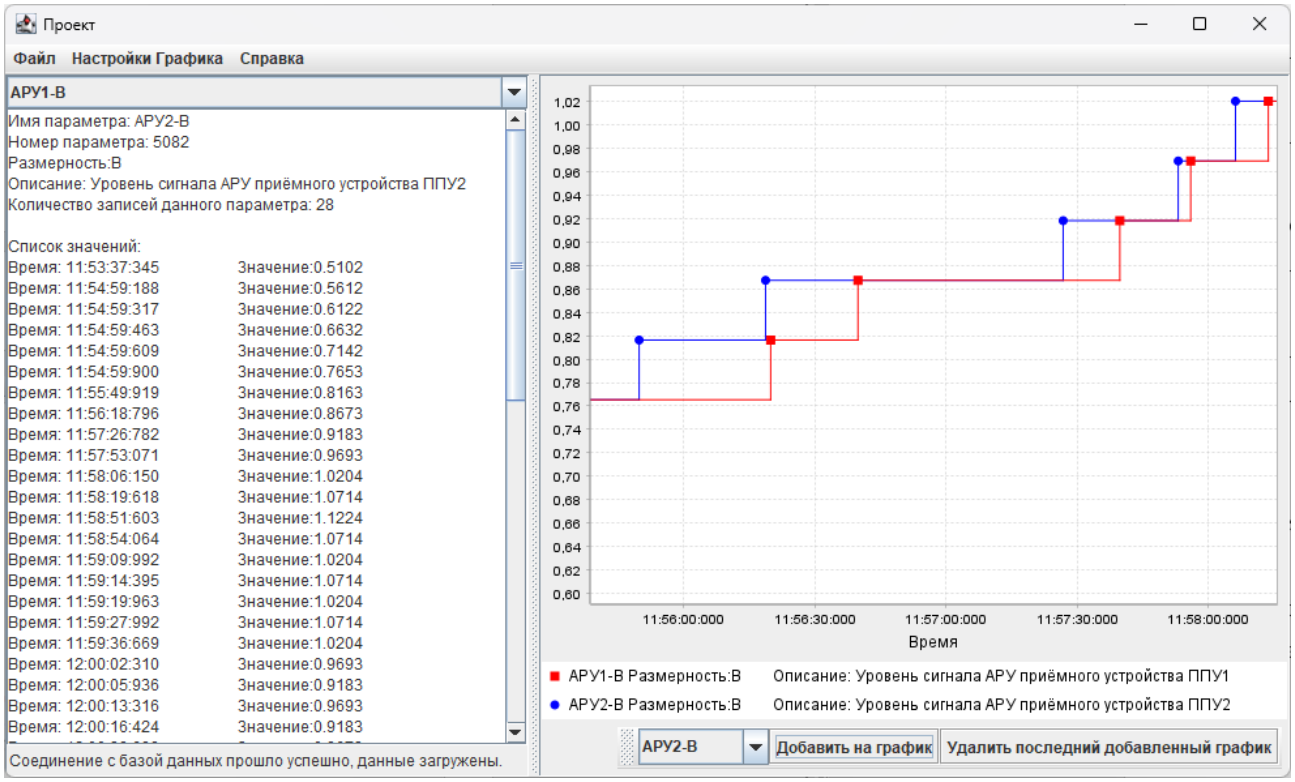

Рисунок 10 – График после масштабирования

Ещё один способ масштабирования – масштабирования с помощью всплывающего меню по нажатию правой кнопки мыши по графику (рисунок 11). Также с помощью всплывающего меню есть возможность настройки графика, а именно: подпись осей, цвета разметки, изменения шрифта и другие.

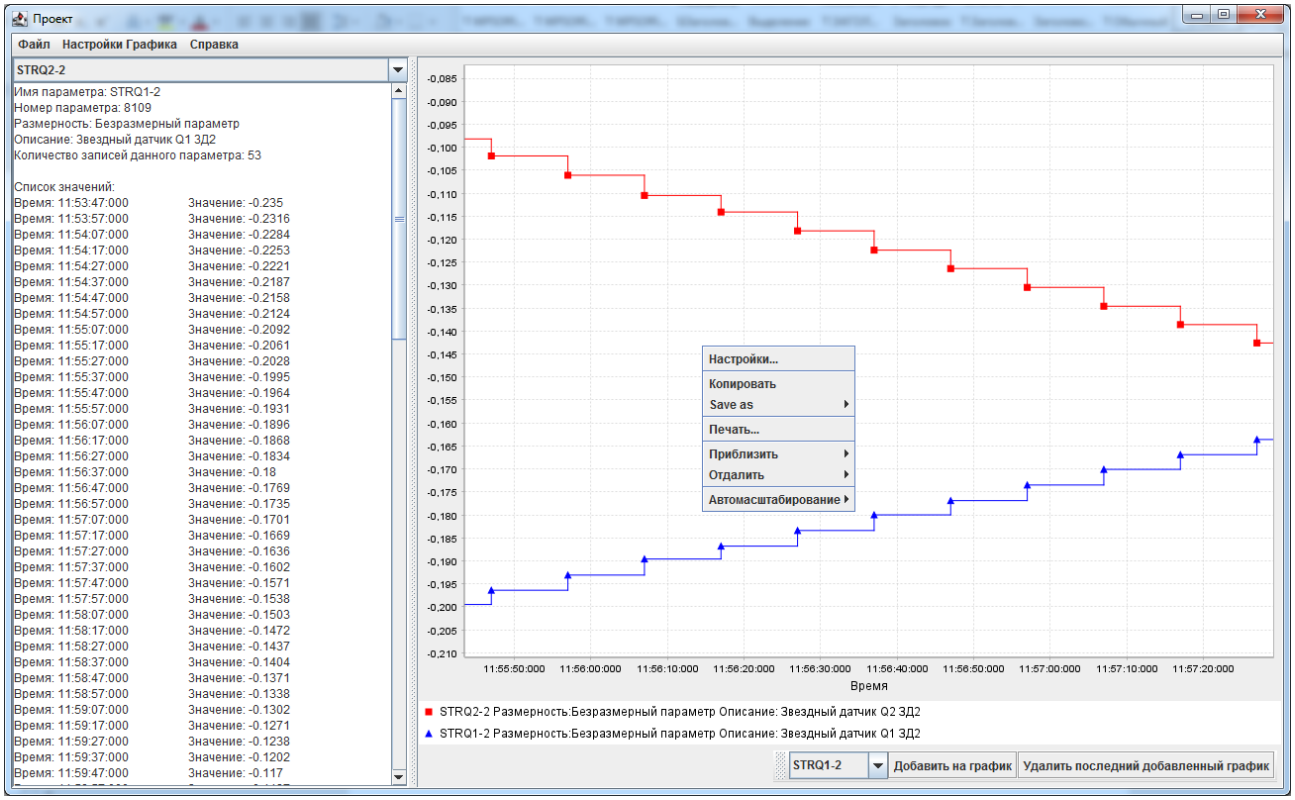

Рисунок 11 – Всплывающее окно по нажатию правой кнопкой мыши по графику

Также с помощью всплывающего меню можно распечатать или сохранить график в .PNG формате. При печати и сохранении в .PNG формате, будет видна только та часть окна, на котором отображён график и описание под графиком. Это отображено на рисунке 12.

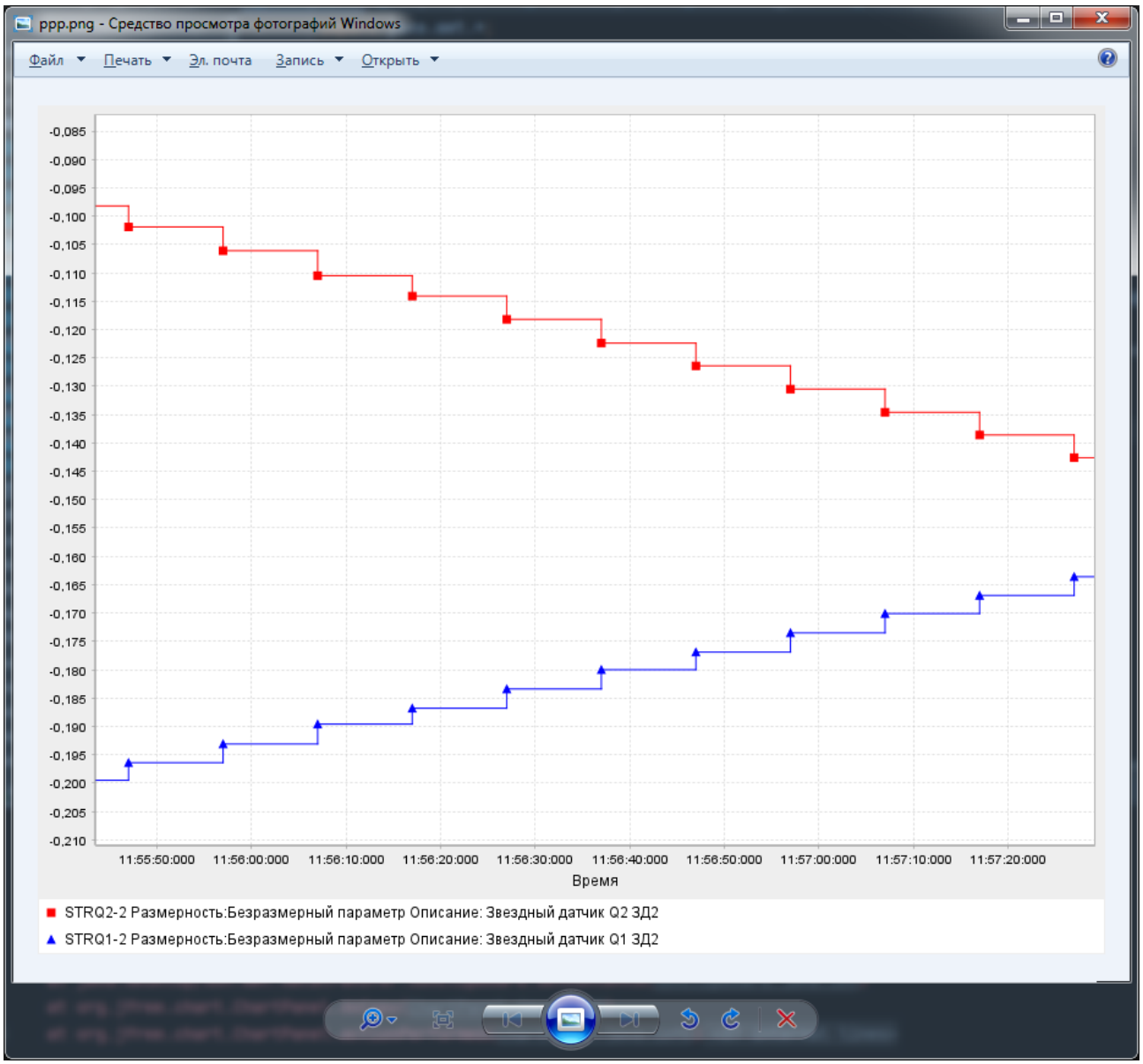

Рисунок 12 – Сохранённый график в формате .PNG

Если нажать на кнопку “Удалить последний добавленный график”, то последний отображенный на графике параметр пропадёт, как и его описание в текстовом виде (рисунок 13).

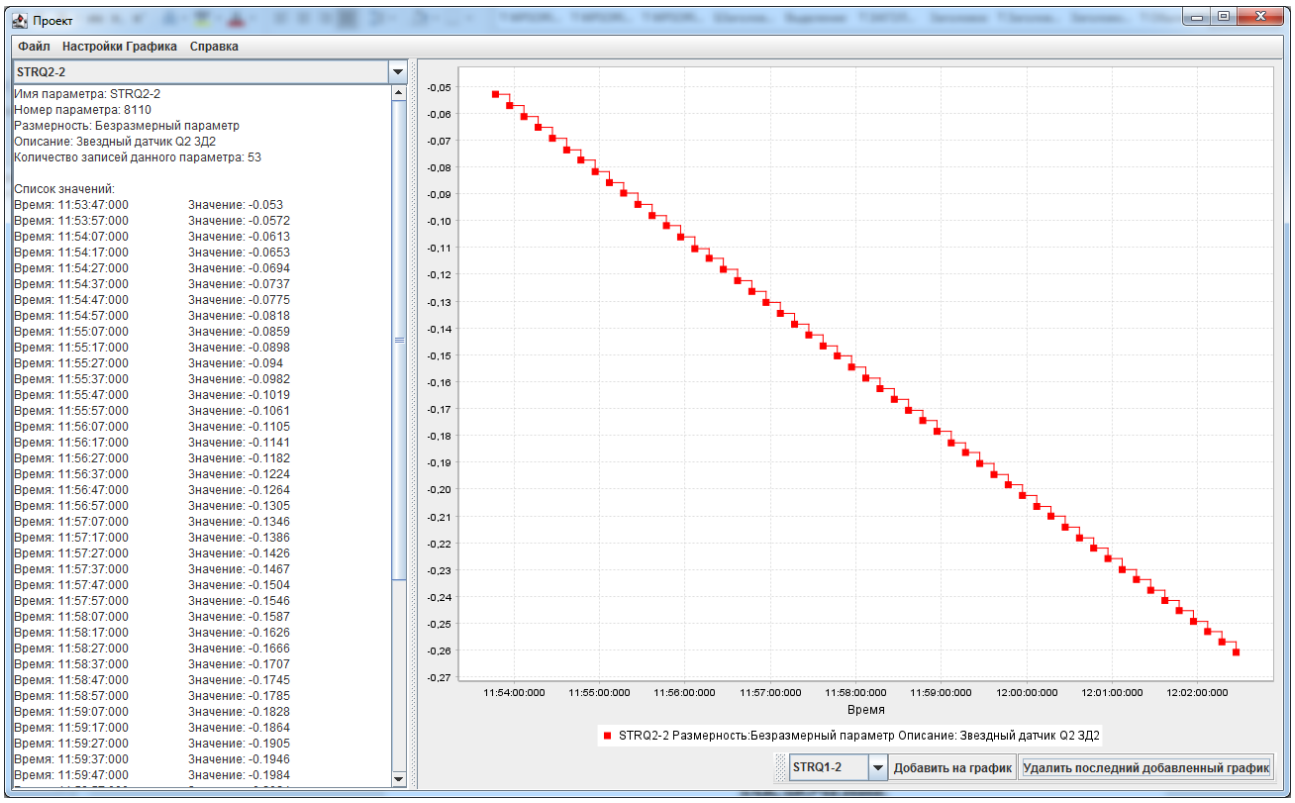

Рисунок 13 – Основное окно после нажатия на кнопку “Удалить последний построенный график”

Также на окне присутствует меню. По нажатию вкладки “Файл” появится список возможностей взаимодействия с файлами, а также кнопка “Выход”, по нажатию на которую программа завершит работу (рисунок 14). При нажатии на любую из трёх возможных кнопок открытия файлов появится проводник (рисунок 15), где пользователь сможет выбрать другой входной файл из файловой системы компьютера. Выбранные файлы сразу пройдут обработку, заменив предыдущие значения новыми.

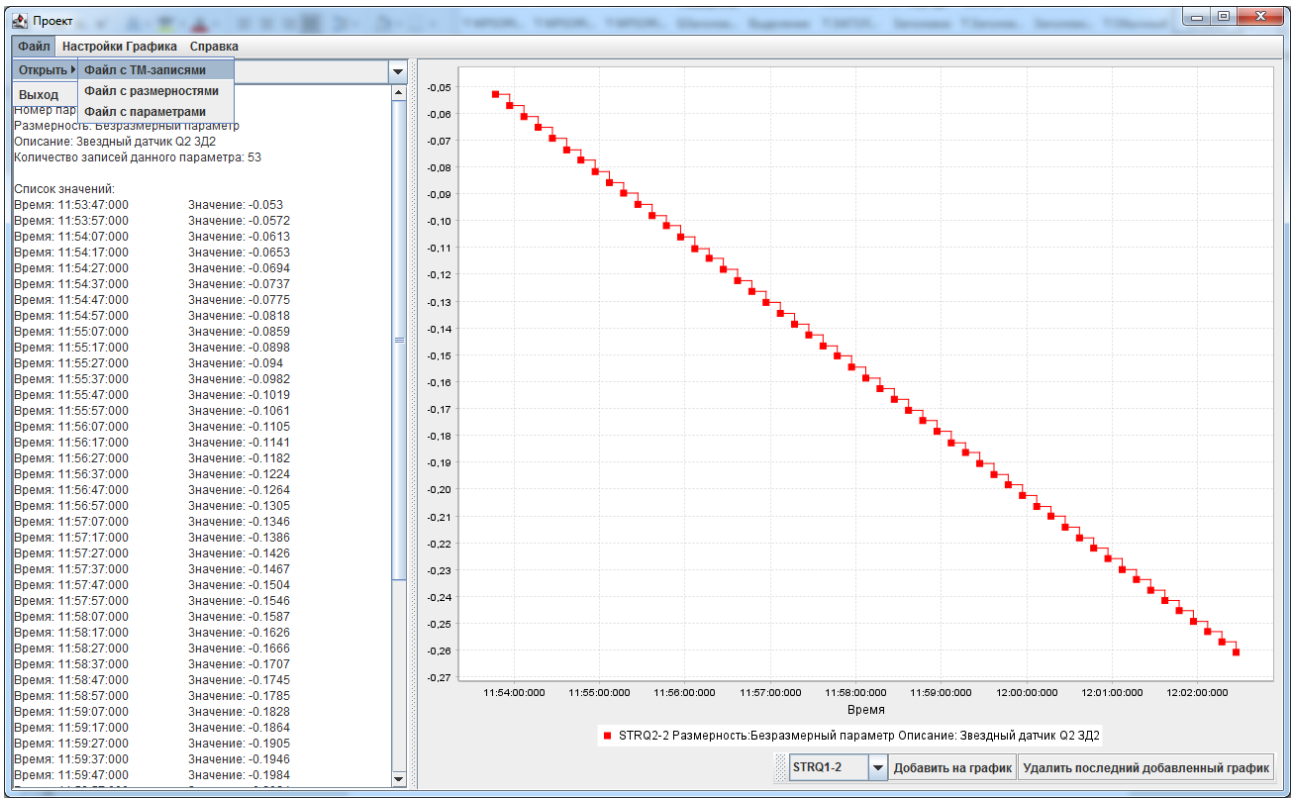

Рисунок 14 – Демонстрация вкладки меню “Файл”

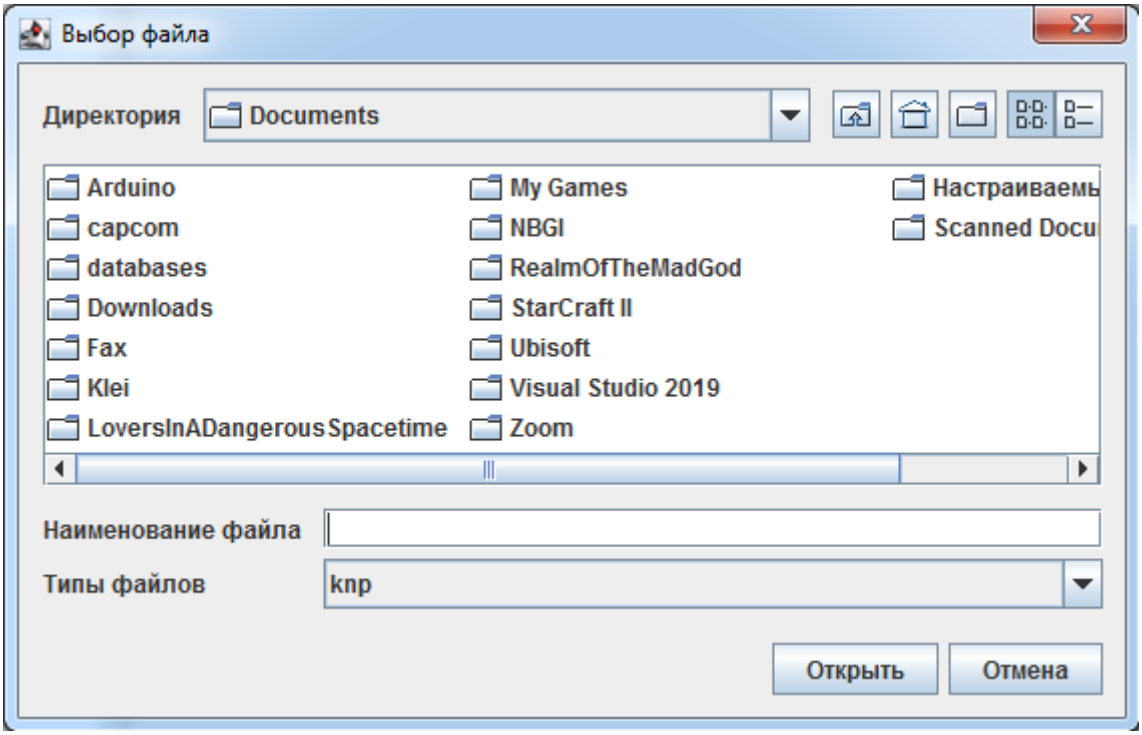

Рисунок 15 – Демонстрация выбора файла из файловой системы компьютера

Во вкладке меню “Настройки графика” находится флажок, отвечающий за тип построенных графиков. Если нажать на настройку “Включить линейное отображение”, то тип графика изменится со ступенчатого на линейный (рисунок 16). Также при нажатии на этот флажок повторно, график изменит свой тип обратно на ступенчатый.

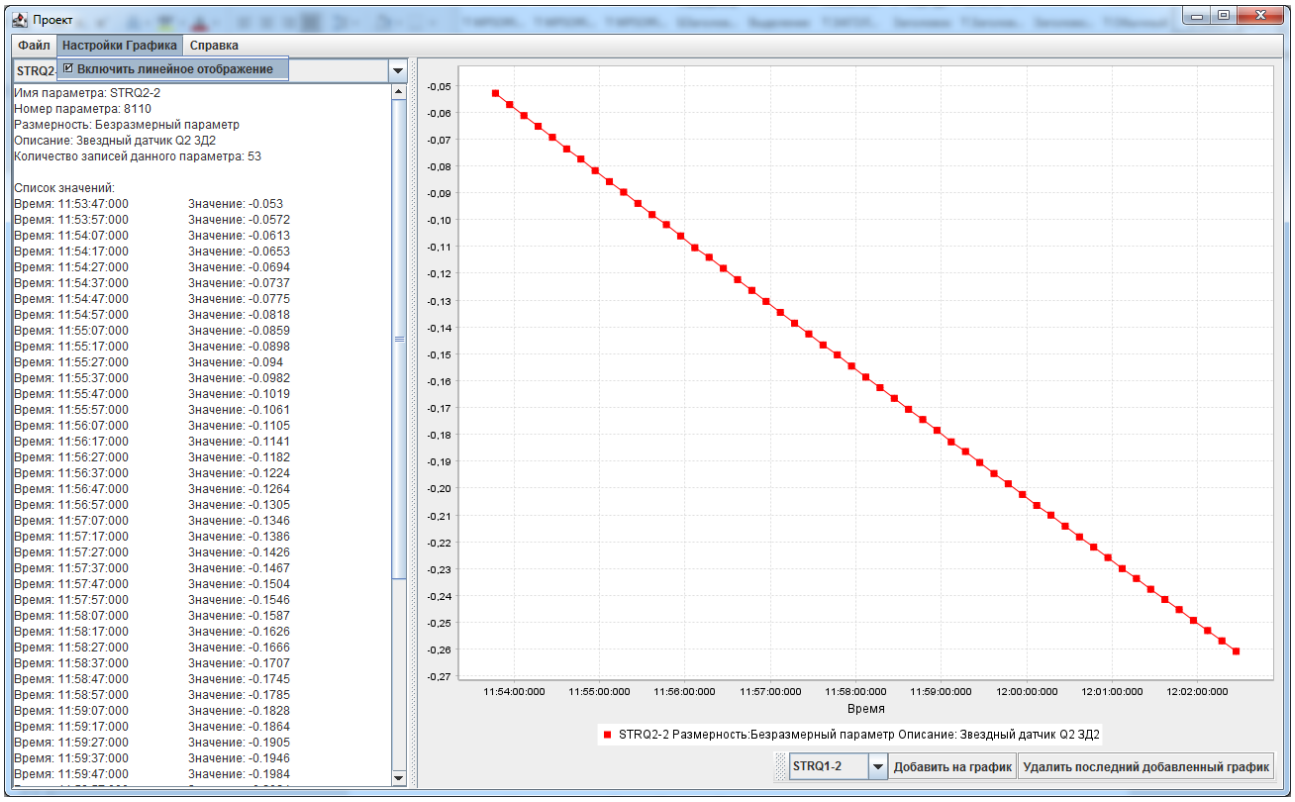

Рисунок 16 – Изменение типа графика со ступенчатого на линейный

При нажатии на вкладку меню “Справка” появится диалоговое окно, на котором будет краткая инструкция по использованию программой (рисунок 17).

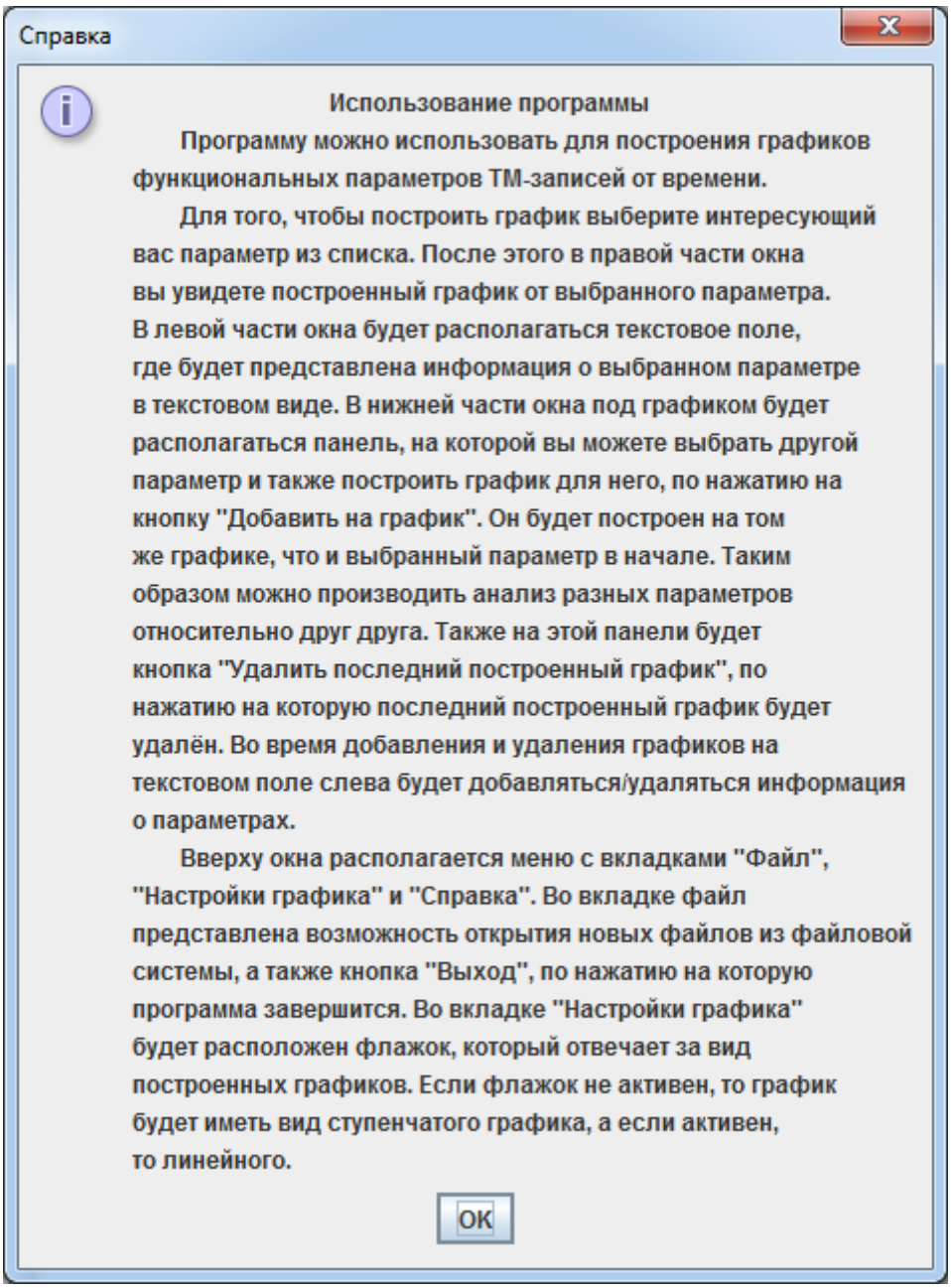

Рисунок 17 – Диалоговое окно “Справка”

ЗАКЛЮЧЕНИЕ

Цель данной работы было разработать программу построения графика по времени для значений функциональных параметров, сформированных в данном сеансе связи.

Я разработал программу на языке программирования Java. Программа выполняет построение графиков, также отображает информацию о выбранных параметрах в текстовом виде.

Для того, чтобы реализовать данный проект мне потребовалось завершить поставленные задачи, а именно:

1) Разработать структуру (класс) для внутреннего хранения значений функциональных параметров;

2) Из входного двоичного файла с результатами обработки телеметрической информации выбрать телеметрические записи функциональных параметров, а также записи, которые были переданы в режиме Непосредственной Передачи;

3) Разработана структура хранения данных в СУБД MySQL

4) Изучить java-классы для работы с графическим интерфейсом пользователя (ГИП);

5) Предоставить пользователю список функциональных параметров с помощью ГИП;

6) Обеспечить пользователю выбор параметра, подлежащего выводу на график;

7) Вывести на график собранные значения для выбранного пользователем параметра.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1) Типы и способы восприятия информации // FB URL: https://fb.ru/article/187059/tipyi-i-sposobyi-vospriyatiya-informatsii (дата обращения: 15.04.2023).

2) Типы телеметрических систем // Лекции по телеметрическим системам URL: https://lms.kgeu.ru/pluginfile.php?file=%2F43198%2Fmod_resource%2Fcontent%2F1%2F%D0%9B%D0%B5%D0%BA%D1%86%D0%B8%D0%B8%20%D0%BA%20%D0%BC%D0%BE%D0%B4%D1%83%D0%BB%D1%8E%201.docx (дата обращения: 21.04.2023).

3) Роберт Лафоре. Структуры данных и алгоритмы в Java. - 5 изд. - СПб: Питер, 2018. - 704 с.

4) Лой Марк, Нимайер Патрик, Лук Дэн. Программируем на Java.. - 5-е межд. изд. - СПб: Питер, 2023. - 544 с.

5) Шилд Герберт. Java. Полное руководство. - 10-е изд. - СПб: Питер, 2020. - 1488 с.

6) Операционная система // skillfactory URL: https://blog.skillfactory.ru/glossary/operaczionnaya-sistema/ (дата обращения: 08.04.2023).

7) Бос Херберт, Таненбаум Эндрю. Современные операционные системы. - 4 изд. - СПб.: Питер, 2022. - 1120 с.

8) TOP 10 лучших сред разработки на Java // Гит Журнал URL: https://gitjournal.tech/10-luchshih-sred-razrabotki-na-java/ (дата обращения: 23.04.2023).

9) Что такое IntelliJ IDEA? // Jet Brains URL: https://www.jetbrains.com/ru-ru/idea/features/ (дата обращения: 24.04.2023).

10) 31.12. Java – Класс TreeSet // ProgLang URL: https://proglang.su/java/treeset-class (дата обращения: 29.04.2023).

11) Петров Алекс. Распределенные данные. Алгоритмы работы современных систем хранения информации. - 9-е изд. - СПб: Питер, 2021. - 336 с.

12) Библиотека JFreeChart // java-online URL: https://java-online.ru/jfreechart.xhtml (дата обращения: 11.05.2023).