МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования

«Новосибирский национальный исследовательский государственный университет»

(Новосибирский государственный университет, НГУ)

Структурное подразделение Новосибирского государственного университета – Высший колледж информатики Университета (ВКИ НГУ)

КАФЕДРА ИНФОРМАТИКИ

**РАЗРАБОТКА ПРОГРАММНОГО МОДУЛЯ "ОБЪЕДИНЕНИЕ ЛОКАЛЬНЫХ ФАЙЛОВЫХ БАЗ ДАННЫХ" ИНФОРМАЦИОННОЙ СИСТЕМЫ АНАЛИЗА РЕЗУЛЬТАТОВ ГЕОХИМИЧЕСКИХ СЪЁМОК МЕТОДОМ ИСП-МС**

Квалификация программист

|  |  |
| --- | --- |
| Руководитель  к. г. -м. н., с. н. с. ИНГГ СО РАН | Агеенков Е.В.  «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_2025 г. |
| Студент 4 курса  гр. 107д2 | Зеньков И.А.  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_2025 г. |

Новосибирск

2025

**СОДЕРЖАНИЕ**

[ПЕРЕЧЕНЬ СОКРАЩЕНИЙ, УСЛОВНЫХ ОБОЗНАЧЕНИЙ И ТЕРМИНОВ 4](#_jksmpkq6hm7t)

[ВВЕДЕНИЕ 5](#_7u2j8iqcg6w7)

[1 ПОСТАНОВКА ЗАДАЧИ ВКР 6](#_6gfbakhkhlqh)

[1.1 Бизнес-требования 6](#_l087lfnzlnx)

[1.2 Пользовательские требования 7](#_678uowwjw3xw)

[1.3 Системные требования 7](#_otqpre5133s5)

[1.4 Требования к пользовательскому интерфейсу 7](#_lix0jwkbxevk)

[1.5 План-график выполнения ВКР 8](#_lozs0et7gxqe)

[2 АНАЛИЗ ТРЕБОВАНИЙ И ОПРЕДЕЛЕНИЕ СПЕЦИФИКАЦИЙ 9](#_cq8h7y8fjwl0)

[2.1 Описание предметной области задачи ВКР 9](#_e4x55j15g1r4)

[2.1.1 Информационные объекты предметной области и взаимосвязи между ними 9](#_wv1zz15u3qt2)

[2.1.2 Информационные потребности пользователей 10](#_phsuf9zpv55)

[2.1.3 Методы работы с информационными объектами предметной области 10](#_4negj4k4gaag)

[2.1.3.1 Способы хранения информации об объектах предметной области 10](#_dk7isc41jlyf)

[2.1.3.2 Математические модели, используемые для обработки информации 11](#_iq6d5gmnyc3g)

[2.1.3.3 Применяемые программные технологии обработки информации, основанные на математических моделях 11](#_qzhytpuz5p2i)

[2.1.3.4 Способы интерпретации и визуального представления информации 11](#_mwzbw4uv95y9)

[2.1.3.5 Технологии получения и передачи информации 12](#_ya3130x05gy0)

[2.1.4 Обзор существующих программных реализаций решения задачи 12](#_2w1pzepiz4x4)

[2.1.5 Концептуальное обоснование разработки 12](#_jbyaxlpdfsun)

[2.2 Классы и характеристики пользователей 12](#_fxumpq1xjbty)

[2.3 Функциональные требования 13](#_viviffkfkxm)

[2.3.1 Определение функциональных возможностей 13](#_g5max2prc46k)

[2.3.2 Описание прецедентов 13](#_bqmozx4l6vun)

[2.4 Нефункциональные требования 14](#_ixpelptk6ozq)

[3 ВЫБОР ПРОГРАММНЫХ СРЕД И СРЕДСТВ РАЗРАБОТКИ 15](#_2y6crclnxuhe)

[3.1 Сравнительный анализ имеющихся возможностей по выбору средств разработки 15](#_xhic8w5kz72k)

[3.2 Характеристика выбранных программных сред и средств 15](#_f6k5y0eejaml)

[4 АЛГОРИТМЫ РЕШЕНИЯ ПОСТАВЛЕННОЙ ЗАДАЧИ 16](#_4afhs1ngg0sw)

[4.1 Этапы реализации 16](#_815nbrde34f)

[4.2 Пользовательский интерфейс 16](#_9nz0xf6lp9v8)

[4.2.1 Описание взаимодействия с пользователем 16](#_rn556faxhwf1)

[4.2.2 Определение операций пользователей и составление функциональных блоков 16](#_h1zkmyv12yag)

[4.2.3 Проектирование структуры экранов и схемы навигации 17](#_sudnaf5olaro)

[4.2.4 Разработка дизайна интерфейса 17](#_ahkilk19ditx)

[4.3 Входные, выходные и промежуточные данные 18](#_14hl94tcjo9m)

[4.4 Реализация используемых методов хранения, обработки и передачи информации об объектах предметной области 18](#_2ez0epf4764h)

[4.4.1 Методы хранения данных 18](#_k57cnwv1658y)

[4.4.2 Алгоритмы реализации используемых математических моделей 19](#_iymb9koqiuyb)

[4.4.3 Алгоритмы использования применяемых программных технологий обработки данных 20](#_pwipbu8l8aaf)

[4.4.4 Алгоритмы применения методов графического анализа данных 20](#_kgqrth6xfa4p)

[4.4.5 Алгоритмы использования технологий передачи данных 20](#_ijep5c4ei1hs)

[4.5 Описание архитектурного решения 21](#_yvb5u2ll05ss)

[4.5.1 Структурная организация программной системы 21](#_m1mfhddkwcvp)

[4.5.2 Архитектура программного кода 21](#_4i9cvqr5z6wz)

[5 ТЕСТИРОВАНИЕ И ОПТИМИЗАЦИЯ 22](#_xrnulbni4zcc)

[5.1 План тестирования 22](#_tf9txavrb4cl)

[5.2 Результаты тестирования и оптимизация 22](#_mys8wve6y5tk)

[6 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 23](#_67z0brmi0q5z)

[ЗАКЛЮЧЕНИЕ 24](#_u79nolbbl4ni)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 25](#_a5wqi763nxt4)

[ПРИЛОЖЕНИЯ 26](#_q6dlgz6smg)

[Приложение А 26](#_vuyjjtpa85rk)

[Приложение Б 26](#_a0zeqncdk3g9)

[Приложение В 26](#_r4s1bq2h7geb)

[Приложение Г 26](#_4jug0r7sju5a)

[Приложение Д 26](#_8kuk7y19n8p6)

[Приложение Е 27](#_keelx4x46mz9)

**ПЕРЕЧЕНЬ СОКРАЩЕНИЙ, УСЛОВНЫХ ОБОЗНАЧЕНИЙ И ТЕРМИНОВ**

* ИСП-МС - масс-спектрометрия с индуктивно связанной плазмой. Метод элементного анализа, используемый для определения концентраций химических элементов в пробах.
* Проба - образец материала, отобранный для геохимического анализа.
* Концентрация химического элемента - количественное содержание химического элемента в пробе.
* Съёмка - комплекс мероприятий по сбору данных в рамках геохимического исследования на определённой территории.
* ИНГГ СО РАН - институт нефтегазовой геологии и геофизики сибирское отделение российской академии наук.
* JSON (JavaScript Object Notation) - текстовый формат обмена данными, используемый для структурированного хранения информации.
* WPF (Windows Presentation Foundation) - платформа Microsoft для создания графических интерфейсов в приложениях под Windows.
* C# - объектно-ориентированный язык программирования, разработанный Microsoft для платформы .NET.
* .NET - платформа для разработки приложений, предоставляющая библиотеки и инструменты для работы с данными, сетью и интерфейсами.
* LINQ (Language Integrated Query) - технология в C# для выполнения запросов к данным напрямую в коде.
* UML (Unified Modeling Language) - язык графического моделирования, используемый для визуализации архитектуры и процессов в программных системах.
* Хеш-таблица - структура данных, используемая для быстрого поиска и устранения дубликатов записей.
* Сериализация - процесс преобразования объектов программы в формат, пригодный для хранения или передачи.
* Валидация данных - проверка корректности структуры и содержимого данных перед их обработкой.
* TreeView - элемент интерфейса для иерархического отображения данных в виде дерева.

ВВЕДЕНИЕ

В современном мире одним из ключевых методов элементного анализа состава веществ является масс-спектрометрия с индуктивно связанной плазмой. Она активно используется в таких областях, как геология, фармацевтика, геохимия и пищевая промышленность. Однако рост объёмов данных, получаемых с помощью ИСП-МС, требует автоматизации процессов их учёта, обработки и анализа. Во многих лабораториях по-прежнему используются ручные методы обработки данных, что негативно сказывается на скорости и точности исследований, а также повышает вероятность потери информации.

В России, где геохимические исследования остаются важными и по сей день, потребность в точных и оперативных средствах обработки данных особенно высока. Несмотря на это, в российских лабораториях внедрение цифровых решений для управления данными, полученными с помощью ИСП-МС, происходит медленнее, чем в мире в целом. Это снижает эффективность обработки больших объёмов информации учёными.

Для Новосибирска, являющегося крупнейшим научным центром Сибири, актуальность автоматизации лабораторных процессов возрастает. Активное использование метода ИСП-МС в научных исследованиях требует внедрения современных инструментов для работы с данными, что особенно важно для учреждений, занимающихся геохимическими исследованиями.

В ИНГГ СО РАН разрозненность данных, хранящихся в многочисленных локальных хранилищах, создает серьезные трудности для комплексного анализа результатов съемок. Для решения этой проблемы необходим инструмент, позволяющий объединять данные из разных источников и быстро извлекать информацию по конкретным исследованиям. Разработка такого программного модуля поможет оптимизировать внутренние процессы и повысить эффективность анализа геофизических данных.

Целью выпускной квалификационной работы является разработка программного модуля для интеграции локальных файловых баз данных формата JSON, используемых при обработке результатов съёмок методом ИСП-МС, с функциями объединения данных и извлечения по идентификатору съёмки. Для достижения поставленной цели необходимо решить следующие задачи:

* Провести анализ научной литературы и исследований в области геохимических измерений с использованием ИСП-МС.
* Провести анализ существующих программных решений для работы с JSON-базами данных, выявить их ограничения в контексте научных исследований и хранения геохимической информации.
* Реализовать алгоритм объединения двух локальных JSON-файлов в новую независимую базу данных с сохранением целостности информации и устранением дубликатов.
* Реализовать механизм извлечения записей по идентификатору съёмки и экспорта полученных данных в отдельный файл.
* Реализовать алгоритмы создания отчётов по результатам интеграции и экспорта данных.
* Разработать пользовательский интерфейс для управления процессами интеграции и экспорта.
* Реализовать обработку типовых ошибок с информированием пользователя о причинах сбоя.
* Провести тестирование производительности программного модуля на больших объёмах данных для оценки времени обработки.
* Разработать демонстрационный сценарий использования программного модуля.

Для реализации поставленных задач планируется использовать современные технологий разработки программного обеспечения. Основной акцент будет сделан на обеспечении удобства работы пользователей, надёжности обработки данных и высокой скорости выполнения операций при работе с большими объёмами информации.

1 ПОСТАНОВКА ЗАДАЧИ ВКР

**1.1 Бизнес-требования**

Программный модуль для интеграции локальных баз данных формата JSON будет направлен на устранение ключевых проблем в управлении данными ИСП-МС. Внедрение данного программного модуля должно обеспечить следующие выгоды:

1. Оптимизация процессов объединения данных: Автоматизация объединения разрозненных JSON-баз данных позволит существенно сократить время, затрачиваемое специалистами на ручную обработку и проверку целостности информации. За счёт исключения рутинных операций повысится производительность сотрудников и снизится вероятность ошибок, возникающих при работе с большими объёмами данных.
2. Повышение гибкости работы с данными съёмок: Возможность экспорта информации по конкретному идентификатору съёмки в отдельные независимые базы обеспечит удобство при анализе отдельных этапов исследований, а также упростит повторное использование информации.
3. Прозрачность операций: Автоматическое формирование отчётов по результатам выполнения операций интеграции и экспорта данных позволит фиксировать ключевые сведения о процессе обработки. Это обеспечит возможность быстрой оценки результатов, упрощая контроль и принятие решений на основе полученной информации.
4. Повышение надёжности обработки данных: Проверка целостности и корректности информации позволит своевременно выявлять ошибки в структуре или содержании баз данных и уведомлять пользователя. Это обеспечит стабильную работу модуля и более высокое качество получаемых результатов.

Таким образом, разработка данного программного модуля обеспечит повышение эффективности исследований, снижение временных затрат и минимизацию ошибок при обработке результатов.

**1.2 Пользовательские требования**

Проектируемый программный модуль ориентирован на удовлетворение потребностей сотрудников лаборатории ИНГГ СО РАН. Разработка должна обеспечить специалистам инструмент для управления, интеграции и анализа результатов съёмок. Ключевые требования, удовлетворяющие потребности специалистов, включают:

1. Выбор локальных баз данных: Пользователь должен иметь возможность выбора необходимых JSON-файлов с устройства, для последующей работы с ними.
2. Интеграция баз данных: Объединение выбранных JSON-файлов в новую независимую базу данных с автоматическим устранением дублирующихся записей и сохранением целостности структуры.
3. Выбор идентификатора съёмки: Возможность выбора ID съёмки из доступных в выбранной базе данных для последующего экспорта связанных данных.
4. Экспорт данных по идентификатору съёмки: Создание новой независимой базы данных, содержащей только записи, соответствующие выбранному ID съёмки, для углубленного анализа.
5. Указание пути сохранения: Пользователь сможет задавать путь для сохранения новой базы данных при интеграции или экспорте, что обеспечит удобство организации файлов.
6. Автоматическое создание отчётов: По результатам интеграции или экспорта данных должен создаваться соответствующий отчёт в виде текстового файла.
7. Визуализация процессов: Отображение статуса выполнения операций через прогресс-бар для повышения прозрачности работы.
8. Оповещения об ошибках: Система должна информировать пользователя о возникающих ошибках.
9. Просмотр краткого содержания баз данных: Отдельная панель интерфейса, которая должна отображать сводные параметры загруженных баз данных.

Реализация данных требований оптимизирует рабочие процессы лаборатории ИНГГ СО РАН, сократив время обработки данных и минимизировав риски ошибок. Общие требования и функциональные особенности подробно изложены в техническом задании (Приложение А).

**1.3 Системные требования**

Системные требования определяют общую структуру программного модуля и его разделение на функциональные блоки. В данном случае проектируемый модуль рассматривается как подсистема в рамках более крупной системы, предназначенной для работы с данными, полученными методом ИСП-МС. В рамках этой системы модуль должен обеспечивать выполнение функций интеграции и экспорта данных.

Соответственно в программном модуле должны быть реализованы следующие функциональные блоки:

1. Блок интеграции баз данных:
   * Должен обеспечивать возможность загрузки двух различных локальных баз данных и их объединения в новый файл формата JSON.
   * В процессе объединения требуется проводить проверку на наличие дублирующихся записей по ключевым полям, а также предварительную проверку целостности и корректности данных.
2. Блок экспорта данных:
   * Должен предоставлять возможность выбора конкретного идентификатора съёмки из списка, автоматически формируемого на основе содержимого базы данных.
   * После выбора идентификатора должен обеспечивать возможность извлечения всех связанных с ним записей и их сохранения в виде нового отдельного файла формата JSON.

Архитектура программного модуля должна обеспечивать чёткое разделение функциональности между блоками. Каждый блок должен выполнять свои задачи независимо, а связь между ними должна быть организована таким образом, чтобы обеспечивать удобную работу модуля в целом. Такой подход позволит упростить доработку и расширение модуля при необходимости.

**1.4 Требования к пользовательскому интерфейсу**

Пользовательский интерфейс играет важную роль в обеспечении эффективного взаимодействия пользователей с программным модулем. Корректная реализация интерфейса позволяет минимизировать время на освоение системы, снизить количество ошибок при работе и повысить общую производительность пользователей. Требования к пользовательскому интерфейсу определяют основные принципы проектирования графических форм, размещения элементов управления, а также доступ к функциональности системы через элементы интерфейса.

Требования к пользовательскому интерфейсу можно условно разделить на две группы:

1. Требования к внешнему виду пользовательского интерфейса
   * Основные цвета интерфейса:
     + Фоновый цвет: #343A40 (тёмно-серый).
     + Цвет текста: #FFFFFFFF (белый)
     + Цвета активных элементов: #36889b (зеленовато-синий), #FFFFFF (белый)
   * Шрифты:
     + Заголовки: Tahoma 18px
     + Текст функциональных и нефункциональных элементов: Tahoma 14px
   * Элементы интерфейса:
     + Элементы интерфейса должны быть четкими, хорошо видимыми и легко различимыми.
     + Размещение элементов интерфейса должно быть логичным и удобным для пользователя.
2. Требования по доступу к функциональности системы через интерфейс
   * Для каждой из основных функций программного модуля (интеграция, экспорт) необходимо реализовать свой отдельный интерфейс.
   * Переключение между интерфейсами должно быть реализовано с помощью соответствующих кнопок.
   * Оповещение об ошибках должно выводиться в виде диалоговых окон.
   * Для интеграции баз данных необходимо реализовать:
     + Кнопки выбора баз данных для интеграции, а также элементы интерфейса для отображения путей к выбранным базам данных.
     + Индикатор выполнения интеграции баз данных.
     + Кнопку для интеграции баз данных.
     + Кнопку открытия папки с отчётами по результатам интеграции баз данных.
   * Для экспорта данных по ID съёмки необходимо реализовать:
     + Кнопку выбора базы данных для экспорта, а также элемент интерфейса для отображения пути к выбранной базе данных.
     + Элемент комбинированного списка для выбора ID съёмки.
     + Индикатор выполнения экспорта данных.
     + Кнопку для экспорта данных.
     + Кнопку открытия папки с отчётами по результатам экспорта данных.
   * Также для удобства работы необходимо реализовать интерфейс панели краткого содержания:
     + Кнопки раскрытия/закрытия панели краткого содержания.
     + Интерфейс для иерархического отображения содержимого баз данных.

Соблюдение требований к пользовательскому интерфейсу обеспечит удобство работы с программным модулем, повысит его доступность для пользователей и снизит количество ошибок при эксплуатации. Правильная организация интерфейса будет способствовать эффективному выполнению задач по обработке и систематизации данных, минимизируя необходимость в дополнительном обучении пользователей.

**1.5 План-график выполнения ВКР**

Качественное выполнение работы невозможно без детального планирования этапов разработки. В таблице 1 представлен план выполнения выпускной квалификационной работы, включающий виды работ с их этапами и сроками выполнения.

Таблица 1 – План выполнения выпускной квалификационной работы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Вид работ | Этапы работы | Срок выполнения, день | Даты |
| 1 | Анализ требовании | * Анализ требований ТЗ и предметной области. * Определение целей и задач разработки. | 14 | 03.02.2025  –  16.02.2025 |
| 2 | Изучение необходимой информации | * Изучение научной литературы и принципов работы ИСП-МС. * Изучение файловых баз данных. * Изучение существующих решений для работы с JSON и масс-спектрометрией. | 14 | 17.02.2025  –  02.03.2025 |
| 3 | Разработка механизмов программного модуля | * Разработка механизма для интеграции JSON-файлов с устранением дубликатов. * Разработка механизма экспорта данных по ID съёмки. * Разработка панели просмотра краткого содержания баз данных. * Разработка алгоритмов генераций отчётов интеграции и экспорта данных. | 21 | 03.03.2025  –  23.03.2025 |
| 4 | Разработка пользовательского интерфейса | * Разработка интерфейса для интеграции файловых баз данных JSON. * Разработка интерфейса для экспорта данных по ID съёмки. * Разработка интерфейса для просмотра содержания баз данных. | 21 | 24.03.2025  –  13.04.2025 |
| 5 | Тестирование | * Проверка корректности интеграции баз данных. * Тестирование экспорта по ID на данных разного объема. * Проверка интерфейса на удобство. | 7 | 14.04.2025  –  20.04.2025 |
| 6 | Доработка и оптимизация | * Исправление выявленных ошибок. * Оптимизация производительности при работе с большими массивами данных. | 21 | 21.04.2025  –  11.05.2025 |
| 7 | Оформление документации и внедрение программного модуля | * Написание технической документации. * Составление руководства пользователя. * Внедрение программного модуля в проект. | 14 | 12.05.2025  –  25.05.2025 |

2 АНАЛИЗ ТРЕБОВАНИЙ И ОПРЕДЕЛЕНИЕ СПЕЦИФИКАЦИЙ

**2.1 Описание предметной области задачи ВКР**

***2.1.1 Информационные объекты предметной области и взаимосвязи между ними***

Информационные объекты в рамках обработки данных, полученных методом масс-спектрометрии с индуктивно связанной плазмой (ИСП-МС), представляют собой структурированные элементы, необходимые для хранения, обработки и анализа результатов исследований. К основным информационным объектам относятся:

1. Исходные JSON-базы данных: Представляют собой структурированные файлы, содержащие информацию, полученную в результате съёмок ИСП-МС. Каждый файл включает записи с результатами измерении и метаданными.
2. Объединённая база данных: Представляет собой сводный файл, формируемый сотрудниками на основе нескольких исходных баз. Обычно создаётся сотрудниками при необходимости сопоставления или совместного анализа данных из разных съёмок. Формируется через копирование и проверку информации вручную.
3. Идентификатор съёмки: Уникальный номер, присваиваемый каждой съёмке. Используется специалистами для поиска, сортировки и фильтрации информации в рамках ручной обработки.
4. Файл с записями конкретной съёмки: Представляет собой структурированный файл, формируемый сотрудниками на основе извлечения данных, относящихся к конкретному идентификатору съёмки.

В настоящее время взаимосвязь между информационными объектами осуществляется полностью вручную, что требует значительных трудозатрат со стороны специалистов. Основные взаимосвязи выглядят следующим образом:

1. Связь между исходными и объединёнными базами данных реализуется за счёт ручного объединения записей из разных файлов. Специалисты самостоятельно отслеживают дублирующиеся записи и следят за целостностью структуры итогового файла.
2. Файл с записями конкретной съёмки формируется вручную на основании анализа содержимого исходных или объединённых файлов. Отбор записей производится на основе идентификатора съёмки, что требует поиска нужных значений и копирования соответствующих данных в отдельный файл.

***2.1.2 Информационные потребности пользователей***

Разработка программного модуля для интеграции и экспорта данных ИСП-МС ориентирована на удовлетворение актуальных информационных потребностей сотрудников лаборатории, занимающихся анализом результатов съёмок. Предполагается, что операторы, работающие с JSON-базами данных, получат доступ к инструментам, позволяющим структурировать, объединять и извлекать необходимые данные с минимальными трудозатратами и сниженным риском ошибок.

Основные информационные потребности пользователей в рамках разработки данного программного модуля включают:

1. Получение объединённой информации из нескольких источников: Пользователю необходима возможность объединения разрозненных JSON-баз в единый сводный файл для последующего анализа. В результате обработки должна формироваться объединённая база данных, содержащая все уникальные записи из исходных файлов.
2. Выделение информации по конкретной съёмке: Для углублённого анализа данных по отдельным съёмкам необходимо извлекать записи, соответствующие конкретному идентификатору съёмки, и сохранять их в отдельный файл. Это позволяет удобно работать с данными отдельных этапов исследований.
3. Визуализация краткого содержания баз данных: В рамках работы с загруженными файлами пользователю требуется предварительный обзор их структуры и содержимого. Это существенно снизит временные затраты сотрудников на дополнительный анализ данных.
4. Получение сводной информации о ходе операций: Важно, чтобы пользователь мог просматривать информацию о результатах интеграции и экспорта данных. Для этого необходимо формировать отчёты, содержащие данные о ходе выполнения операций и результатах обработки.

Для удовлетворения информационных потребностей пользователей в рамках разрабатываемого программного модуля предполагается использование следующих методов работы с информацией:

* Загрузка данных из локальных JSON-файлов с последующей десериализацией их содержимого для представления в виде удобных для обработки структур.
* Использование механизмов фильтрации и выборки данных по идентификатору съёмки для формирования независимых выборок.
* Сериализация и сохранение объединённых или отфильтрованных данных в новые JSON-файлы, пригодные для последующего анализа или передачи.
* Автоматическое формирование текстовых отчётов, содержащих сведения о ходе и результатах выполняемых операций.
* Визуальное отображение краткого содержания данных в виде иерархического списка для быстрой предварительной оценки структуры и состава загруженных файлов.

Таким образом, реализация указанных методов и инструментов работы с информацией позволит в полной мере удовлетворить информационные потребности пользователей при обработке данных ИСП-МС. Более подробно методы работы с информационными объектами предметной области рассматриваются в пункте 2.1.3.

***2.1.3 Методы работы с информационными объектами предметной области***

***2.1.3.1 Способы хранения информации об объектах предметной области***

Для хранения информации, получаемой в ходе геохимических исследований методом ИСП-МС, специалисты лаборатории используют локальные файлы в формате JSON. Данный формат обеспечивает возможность структурированного хранения сложных иерархий данных, включая численные измерения, метаданные о съёмке, параметры проб и уникальные идентификаторы.

Использование JSON позволяет эффективно обрабатывать данные как вручную, так и с помощью программных средств, благодаря поддержке формата во множестве языков программирования. Кроме того, текстовая структура JSON-файлов делает возможным их длительное хранение, переносимость и независимость от конкретного программного обеспечения.

В рамках проектируемого программного модуля предполагается сохранение объединённых и экспортированных данных также в формате JSON. Это позволит сохранить совместимость с существующей структурой хранения информации и интегрировать программный модуль в уже имеющиеся рабочие процессы.

Отдельным элементом хранения информации станут текстовые отчёты, автоматически формируемые по результатам ключевых операций — интеграции и экспорта данных. Такие отчёты будут содержать сведения о ходе выполнения процесса, перечне обработанных файлов, количестве объединённых или экспортированных записей, а также информацию об успешности или ошибках выполнения. Сохранение этих отчётов в виде отдельных текстовых файлов обеспечит возможность упрощённой фиксации хода работы и последующего анализа действий оператора без необходимости анализа самих JSON-файлов.

***2.1.3.2 Методы обработки информации***

Обработка информации, полученной в результате съёмок методом ИСП-МС, является ключевой задачей программного модуля и требует системного подхода, позволяющего работать с большими объёмами данных эффективно и без потери качества. Для обеспечения корректной и быстрой работы с данными в рамках проектируемого программного модуля необходимо реализовать следующие методы обработки информации:

* Считывание и преобразование данных — загрузка структурированных JSON-файлов с последующей десериализацией содержимого. Это позволит преобразовать данные в формат, удобный для программной обработки.
* Фильтрация по идентификатору съёмки — извлечение из общей базы только тех записей, которые соответствуют выбранному идентификатору. Такой подход позволит быстро формировать отдельные выборки для последующего анализа.
* Интеграция данных — объединение записей из разных JSON-файлов в одну сводную структуру с удалением дублирующихся элементов. Для определения уникальности записей будет использоваться сопоставление по ключевым полям, таким как идентификатор съёмки и идентификатор пробы.
* Валидация данных — проверка структуры данных перед выполнением операций. Проверка будет включать контроль обязательных полей, типов данных и допустимых значений. Это необходимо для исключения ошибок при формировании итоговых файлов.
* Сериализация и сохранение результатов — сохранение объединённых или отфильтрованных данных в формате JSON. Это обеспечит совместимость с существующими методами хранения и облегчит дальнейшую работу с файлами.

Реализация указанных методов позволит организовать надёжную и быструю обработку данных, а также упростит выполнение задач, с которыми ранее специалисты справлялись вручную.

***2.1.3.3 Способы интерпретации и визуального представления информации***

В процессе геохимических исследований большое значение имеет визуальное представление данных, поскольку оно способствует более точному и быстрому анализу информации. На практике широко применяются графики, диаграммы, тепловые карты и другие визуальные средства, позволяющие наглядно отразить распределение элементов, выявить аномалии и закономерности в результатах измерений.

Наиболее распространённые формы визуального представления данных включают:

* Линейные графики – используются для отображения изменения концентраций элементов по координатам, времени или другим параметрам.
* Гистограммы – используются для представления изучаемой выборки в графическом виде и выяснения закономерностей распределения химических элементов.
* Столбчатые графики – используются для наглядного представления уровней содержаний химических элементов в различных объектах.
* Дифференциальные и интегральные кривые – строятся, чтобы выяснить закономерности распределения химических элементов.

В рамках проектируемого программного модуля предполагается реализовать более простой и прикладной способ визуального представления информации. Для предварительного ознакомления с содержимым загруженных баз данных будет разработан компонент, отображающий краткое содержание в виде иерархического текстового списка. Такой подход позволит пользователю быстро оценить структуру, объём и ключевые параметры информации, не открывая содержимое файлов вручную. Несмотря на то, что данный подход не является привычным способом визуального представления данных, его функциональности будет достаточно для дальнейшего принятия решений об объединении или экспорте информации.

***2.1.4 Обзор существующих программных реализаций решения задачи***

Разрабатываемый программный модуль предназначен для обработки данных масс-спектрометрии с индуктивно связанной плазмой в лаборатории ИНГГ СО РАН. Его ключевой особенностью является специализированный функционал для работы с локальными базами данных в формате JSON, включающий интеграцию нескольких файлов и экспорт результатов. Проведенный анализ существующих решений показал отсутствие как полных, так и частичных аналогов.

***2.1.5 Концептуальное обоснование разработки***

Современные исследования в области геохимии, проводимые методами ИСП-МС, сопровождаются накоплением больших объёмов данных. Эти данные нуждаются в надёжной обработке, структурировании и подготовке к последующему анализу. Работа с такими объёмами данных сопровождается целым рядом характерных затруднений, что указывает на необходимость создания специализированного программного инструмента. К числу наиболее актуальных проблем относятся:

1. Отсутствие специализированных инструментов для объединения нескольких JSON-баз данных с контролем целостности информации.
2. Неудобство в выборе и выделении отдельных съёмок из больших массивов данных.
3. Повышенные временные затраты при подготовке данных к передаче или последующему анализу.

Оригинальный замысел заключается в разработке программного модуля, способного автоматически обрабатывать локальные JSON-файлы, объединять их с устранением дубликатов, а также обеспечивать экспорт информации по выбранному идентификатору съёмки. Основное внимание при этом уделяется надёжности алгоритмов обработки, автоматизации рутинных операций и адаптации модуля к большим объёмам входных данных.

Таким образом, разработка программного модуля представляет собой не только технически обоснованную задачу, но и концептуально новое решение, направленное на повышение эффективности работы с результатами съёмок ИСП-МС.

**2.2 Классы и характеристики пользователей**

В рамках комплексного проекта, частью которого является разрабатываемый модуль, предусмотрены различные роли пользователей. Однако разрабатываемый функционал ориентирован только на одну категорию – операторов лаборатории, осуществляющих подготовку и обработку данных. В таблице 2 приведены ключевые характеристики данной пользовательской роли.

Таблица 2.2.1 – Профиль Оператора

|  |  |
| --- | --- |
| Пользователь | Оператор |
| Социальные характеристики | Русскоязычные пользователи  Мужчины, женщины  Средний или высокий уровень владения компьютером  Высшее образование в области геохимии, геофизики, аналитической химии или смежных дисциплин  Работа в научно-исследовательских институтах или лабораториях |
| Мотивационная среда | Оптимизация рабочих процессов  Сокращение временных затрат на подготовку и обработку данных  Заинтересованность в качественной обработке данных  Повышение точности и надёжности получаемых результатов  Интерес к освоению современных программных решений |
| Навыки и умения | Умение работать с локальными файловыми базами данных  Опыт работы с результатами съёмок ИСП-МС  Понимание принципов структурирования данных  Умение выполнять повторяющиеся операции с данными в полуавтоматическом режиме |
| Требования к ПО ИС | Интуитивно понятный интерфейс  Поддержка обработки больших объёмов информации  Минимизация ручного ввода  Визуализация процессов экспорта и интеграции данных  Надежность при выполнении операции интеграции и экспорта  Оповещение об ошибках |
| Задачи пользователя | Загрузка локальных JSON-баз данных  Объединение баз данных  Выбор идентификатора съёмки для экспорта  Экспорт записей определённой съёмки в отдельный файл  Проверка корректности сформированных выходных данных |
| Рабочая среда | Стандартизированный ПК или ноутбук |

**2.3 Функциональные требования**

***2.3.1 Определение функциональных возможностей***

На основе анализа предметной области и требований к автоматизации процессов обработки данных можно сформировать перечень функциональных возможностей разрабатываемого программного модуля. Основная цель модуля — обеспечить упрощённую и надёжную работу с локальными файловыми базами данных, минимизируя количество ручных операций при их структурировании, объединении и экспорте информации.

Основные функциональные возможности для пользовательского профиля оператора лаборатории:

1. Интеграция JSON-баз данных

* Выбор баз данных для интеграции
* Интеграция баз данных
* Выбор пути сохранения объединенной базы
* Автоматическое создание отчёта по результатам интеграции

1. Экспорт данных по идентификатору съёмки

* Выбор базы данных для экспорта
* Выбор ID съёмки
* Экспорт данных по ID съёмки
* Выбор пути сохранения файла с результатами экспорта
* Автоматическое создание отчёта по результатам экспорта данных

1. Просмотр краткого содержания

* Просмотр краткого содержания загруженных баз данных в виде раскрывающегося иерархического списка.

На основе анализа предметной области и пользовательских требований к разрабатываемому ПМ составлен реестр юскейсов для выделенных пользовательских ролей (таблица 2.3.1.1).

Таблица 2.3.1.1 – Реестр юскейсов для пользовательской роли «Оператор»

|  |  |  |  |
| --- | --- | --- | --- |
| Пользовательская роль | Потребность пользователя | Use Case | Сценарий использования |
| Оператор | Объединить две локальные базы данных | Интеграция JSON-баз данных | * Выбрать два JSON-файла * Запустить процесс объединения * Сохранить результат |
| Извлечь данные по конкретному ID съёмки | Экспорт данных по ID съёмки | * Выбрать базу данных * Выбрать ID съёмки * Запустить процесс экспорта * Сохранить результат |
| Ознакомиться с содержанием базы перед выполнением операций | Просмотр краткого содержания базы данных | * Загрузить файл * Просмотреть список записей и связанной информации |
| Ознакомиться с результатами экспорта | Просмотр отчётов по экспорту | * После экспорта открыть папку с отчётами * Открыть необходимый отчёт |
| Ознакомиться с результатами интеграции | Просмотр отчётов по интеграции | * После интеграции открыть папку с отчётами * Открыть необходимый отчёт |

На рисунке 1 представлена UML-диаграмма вариантов использования ПМ, представляющая основные действия для пользователей.

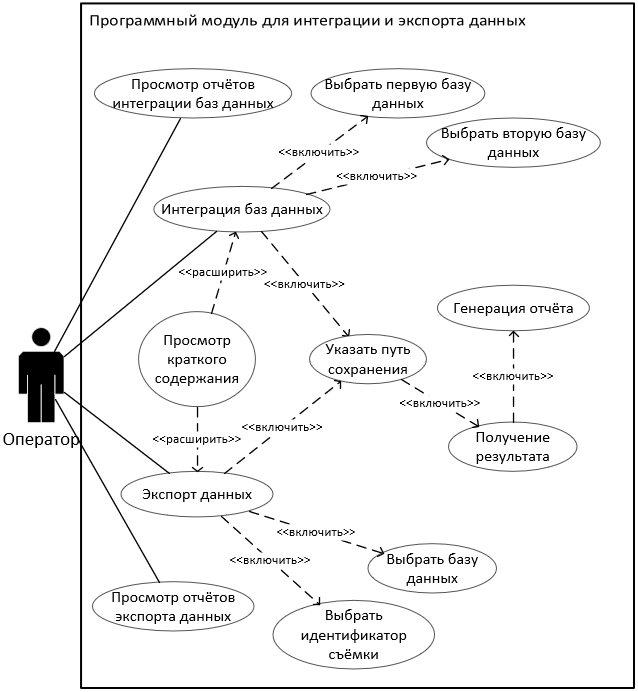


Рисунок 1 - Диаграмма вариантов использования ПМ

***2.3.2 Описание прецедентов***

Описание прецедентов проекта является частью проектирования. Прецеденты позволяют описать взаимодействие пользователя с системой. Каждый прецедент хранит в себе информацию об успешном сценарии взаимодействия пользователя с системой.

Описания прецедентов представлены в таблицах 2.3.2.1 – 2.3.2.5

Таблица 2.3.2.1 – Описание прецедента «Интеграция JSON-баз данных»

|  |  |
| --- | --- |
| Название прецедента | Интеграция JSON-баз данных |
| Исполнитель | Оператор |
| Цель | Объединить данные из двух баз данных в отдельный файл |
| Основной успешный сценарий | * Оператор нажимает на кнопку «Выбрать», после чего открывается окно проводника для выбора первой базы данных для интеграции. * Аналогичное действие выполняется для выбора второй базы данных. * Затем оператор нажимает на кнопку «Создать объединенную базу» и ему предлагается выбрать путь для сохранения файла. * После завершения процесса создается файл и в окне появляется сообщение об успешном объединение данных. |

Таблица 2.3.2.2 – Описание прецедента «Экспорт данных по ID съёмки»

|  |  |
| --- | --- |
| Название прецедента | Экспорт данных по ID съёмки |
| Исполнитель | Оператор |
| Цель | Экспортировать данные по выбранному идентификатору съёмки в отдельный файл |
| Основной успешный сценарий | * Оператор нажимает на кнопку «Выбрать», после чего открывается окно проводника для выбора базы данных, из которой нужно экспортировать данные. * После чего необходимо выбрать ID съёмки по которому будут экспортироваться данные. * Затем оператор нажимает на кнопку «Экспортировать данные» и ему предлагается выбрать путь для сохранения файла. * После завершения процесса создается файл и в окне появляется сообщение об успешном экспорте данных. |

Таблица 2.3.2.3 – Описание прецедента «Просмотр отчётов по интеграции»

|  |  |
| --- | --- |
| Название прецедента | Просмотр отчётов по интеграции |
| Исполнитель | Оператор |
| Цель | Просмотреть отчёты по результатам интеграции баз данных |
| Основной успешный сценарий | * Оператор нажимает на кнопку «Отчёты по интеграции» в окне интеграции баз данных. * Открывается окно проводника со всеми отчётами по интеграции. |

Таблица 2.3.2.4 – Описание прецедента «Просмотр отчётов по экспорту»

|  |  |
| --- | --- |
| Название прецедента | Просмотр отчётов по экспорту |
| Исполнитель | Оператор |
| Цель | Просмотреть отчёты по результатам экспорта данных |
| Основной успешный сценарий | * Оператор нажимает на кнопку «Отчёты по экспорту» в окне экспорта данных. * Открывается окно проводника со всеми отчётами по экспорту. |

Таблица 2.3.2.5 – Описание прецедента «Просмотр краткого содержания»

|  |  |
| --- | --- |
| Название прецедента | Просмотр краткого содержания |
| Исполнитель | Оператор |
| Цель | Просмотреть загруженную базу перед выполнением операций |
| Основной успешный сценарий | * Оператор загружает файл базы данных * Нажимает кнопку для раскрытия панели * В интерфейсе отображается иерархическая структура содержимого базы данных |

**2.4** Нефункциональные требования

Нефункциональные требования играют важную роль в разработке программного модуля, предназначенного для работы с данными, хранящимися в локальных файловых базах формата JSON. Эти требования задают параметры надёжности, совместимости, удобства использования, а также определяют условия, в которых система должна функционировать. Ниже приведены основные нефункциональные требования к программному модулю:

1. Требования к производительности

* Оптимальное время выполнения операций: Операции интеграции и экспорта должны выполняться за минимально возможное время, обеспечивая высокую скорость обработки данных.
* Стабильная работа при высоких нагрузках: Программный модуль должен сохранять стабильность и корректность функционирования при работе с большими объёмами информации.

1. Требования к переносимости и совместимости

* Поддержка операционных систем: Программный модуль должен быть совместим с операционными системами Windows 10 и выше.
* Минимальные системные требования: Программный модуль должен быть оптимизирован для стабильной работы на стандартных рабочих станциях лаборатории.

1. Требования к надежности, доступности и ремонтопригодности

* Оповещение об ошибках: Система должна уведомлять об ошибках, возникающих в процессе выполнения операции.

1. Требования к безопасности

* Проверка корректности данных: Перед началом процесса интеграции или экспорта должна выполняться проверка целостности и структуры данных.
* Исключение функций редактирования данных: Функциональность программного модуля должна исключать возможность изменения или удаления исходных данных.
* Работа без обращения к сети: Полностью автономная работа без подключения к интернету, что снижает риски утечки данных.

1. Требования к локализации

* Интерфейс на русском языке: Весь пользовательский интерфейс программного модуля, включая элементы управления и сообщения, должен быть полностью выполнен на русском языке.

1. Требования к удобству использования

* Интуитивно понятный интерфейс: Интерфейс должен быть интуитивно понятным и доступным для пользователей с различным уровнем технической подготовки.
* Точные обозначения: Все элементы интерфейса должны иметь чёткие и понятные обозначения.
* Отображение прогресса выполнения операций: Наличие индикатора выполнения операций.

Соблюдение нефункциональных требований критически важно для успешного внедрения программного модуля в рабочие процессы лабораторий. Эти требования обеспечивают не только высокую производительность и надежность системы, но и удобство её практического применения в условиях регулярной работы с большими массивами данных.

3 ВЫБОР ПРОГРАММНЫХ СРЕД И СРЕДСТВ РАЗРАБОТКИ

**3.1 Сравнительный анализ имеющихся возможностей по выбору средств разработки**

При создании программного модуля для интеграции и экспорта данных, полученных методом ИСП-МС, особое внимание уделяется выбору подходящих языков программирования, средств разработки и технологий проектирования пользовательского интерфейса. От этого выбора напрямую зависят надёжность, скорость разработки, удобство сопровождения и соответствие функциональных возможностей требованиям лаборатории.

Также необходимо учитывать, что программное средство должно обеспечивать работу с локальными файловыми базами данных в формате JSON. Важную роль играет наличие удобного, интуитивно понятного графического интерфейса, позволяющего пользователю быстро выполнять основные операции. С целью обоснования выбора средств разработки ниже приведено сравнение возможных вариантов по ключевым категориям.

1. Языки программирования:
   * C# – это язык программирования от компании Microsoft, ориентированный на разработку десктопных и корпоративных приложений. C# обладает строгой типизацией, высокой производительностью и тесной интеграцией с платформой .NET, что делает его особенно удобным при создании настольных приложений для Windows. Среди преимуществ – наличие встроенных средств работы с JSON, а также использование LINQ для выборки и обработки данных. Является предпочтительным вариантом для реализации функционально сложных приложений с графическим интерфейсом.
   * Python – это высокоуровневый интерпретируемый язык программирования, широко применяемый в научной среде и в задачах анализа данных. Обладает простым синтаксисом и большим количеством библиотек, включая модули для работы с JSON. Однако Python уступает C# по скорости выполнения, особенно при работе с большими объёмами данных, и менее удобен для создания полноценных графических интерфейсов под Windows без сторонних фреймворков.
2. Среды разработки:
   * Visual Studio – это интегрированная среда разработки от Microsoft, предоставляющая широкие возможности для разработки приложений на различных платформах, включая C# и WPF. Включает средства визуального дизайна интерфейсов, мощный отладчик, систему управления проектами и поддержку подключения сторонних библиотек. Visual Studio тесно интегрирована с WPF и .NET, что делает её оптимальным выбором при создании Windows-приложений.
   * Visual Studio Code – это лёгкий редактор кода, обладающий гибкой системой расширений. Подходит для разработки на Python, JavaScript, C++ и других языках. Может использоваться для C#-разработки с установкой соответствующих расширений, однако не обладает встроенным визуальным дизайнером интерфейса, что затрудняет создание приложений с графическим интерфейсом.
3. Средства разработки пользовательского интерфейса:
   * Windows Presentation Foundation (WPF) – это технология от Microsoft для создания десктопных приложений под Windows с использованием языка XAML. Обеспечивает высокую гибкость в разработке интерфейсов, поддержку стилей, шаблонов, а также привязку данных. WPF хорошо интегрируется с Visual Studio и является стандартом при создании интерфейсов в экосистеме .NET.
   * PyQt – это кроссплатформенный инструмент для разработки GUI-приложений на Python, основанный на Qt. Предоставляет богатый набор компонентов интерфейса и возможности кастомизации. PyQt позволяет разрабатывать приложения для различных операционных систем, однако для Windows-разработки требует дополнительной настройки и уступает WPF по глубине интеграции с системными средствами Windows.

Из сравниваемых языков программирования C# предпочтительнее для разработки настольных приложений под Windows благодаря лучшей производительности и тесной интеграции с такими инструментами, как WPF. Среди сред разработки наиболее удобной для работы с C# является Visual Studio. WPF, в свою очередь, обеспечивает гибкость, поддержку XAML и широкие возможности настройки интерфейса, превосходя аналогичные решения для Python. Таким образом, сочетание C#, Visual Studio и WPF обеспечивает совместимость, стабильность и полноту функциональности при разработке настольного ПО для работы с локальными базами данных.

**3.2 Характеристика выбранных программных сред и средств**

Исходя из сравнительного анализа, для разработки программного модуля были выбраны язык программирования C#, интегрированная среда разработки Visual Studio и технология построения интерфейса WPF. Такой выбор обоснован необходимостью создания настольного приложения, ориентированного на Windows-среду, с возможностью удобной и надёжной работы с файловыми базами данных в формате JSON. Выбранные инструменты хорошо взаимодействуют между собой, обладают высокой производительностью и подходят для реализации требуемого функционала.

Язык программирования C# – это современный объектно-ориентированный язык программирования, разработанный Microsoft. Он применяется для создания приложений под платформу .NET и особенно хорошо подходит для разработки Windows-приложений. Основные преимущества C#:

* Статическая типизация: В C# переменные должны быть определены с указанием типа данных, что позволяет выявлять ошибки на этапе компиляции и улучшает производительность и безопасность кода.
* Богатая стандартная библиотека: .NET предоставляет огромное количество классов и методов для выполнения самых различных задач, от работы с файловой системой до сетевого взаимодействия.
* Безопасность и управление памятью: C# включает в себя встроенные механизмы управления памятью и защиту от типичных ошибок программирования, таких как утечки памяти.
* Интеграция с Windows API: язык тесно связан с экосистемой Microsoft, что обеспечивает лёгкий доступ к возможностям Windows, включая работу с интерфейсами, потоками и системными событиями.

Visual Studio – это интегрированная среда разработки (IDE), созданная компанией Microsoft. Она предоставляет разработчикам мощные инструменты для написания, отладки и тестирования кода, что делает процесс разработки более эффективным и удобным. Её основные особенности включают:

* Удобство работы с библиотеками и зависимостями: Visual Studio упрощает управление библиотеками и зависимостями благодаря поддержке пакетов NuGet. Можно легко добавлять, обновлять и удалять пакеты, необходимые для проекта.
* Поддержка множества языков программирования: Visual Studio поддерживает широкий спектр языков программирования, включая C#, C++, JavaScript, Python, TypeScript, F# и многие другие.
* Богатый набор инструментов для разработки: Visual Studio оснащена множеством полезных инструментов, которые помогают ускорить процесс разработки. Сюда входят редактор кода с функцией авто дополнения, встроенные средства рефакторинга, поддержка различных стилей форматирования кода и многое другое.
* Интегрированные средства отладки и тестирования: Visual Studio предоставляет мощные инструменты для отладки и тестирования кода. Встроенный отладчик позволяет выполнять код пошагово, устанавливать точки останова, просматривать значения переменных и отслеживать выполнение программы.

WPF – это платформа для построения пользовательских интерфейсов, предназначенная для приложений под Windows. Она предоставляет широкие возможности для создания гибкого, визуально привлекательного и адаптируемого интерфейса. Преимущества технологии:

* Создание привлекательного пользовательского интерфейса: WPF предоставляет возможность создавать привлекательные пользовательские интерфейсы, позволяя проектировать приложения с эффектной графикой, плавными анимациями и сложными макетами.
* Облегчение работы с привязкой данных: WPF позволяет напрямую связывать интерфейс с источниками данных, что избавляет от необходимости писать лишний код для синхронизации.
* Разделение логики и дизайна: С помощью XAML можно отделить визуальную часть от функциональной, что упрощает обновление внешнего вида приложения без изменения кода, и наоборот.

Таким образом, связка C#, Visual Studio и WPF представляет собой оптимальное решение для создания функционального, стабильного и расширяемого программного модуля, предназначенного для работы с результатами исследований, полученными методом ИСП-МС.

4 **АЛГОРИТМЫ РЕШЕНИЯ ПОСТАВЛЕННОЙ ЗАДАЧИ**

**4.1 Этапы реализации**

Разработка программного модуля для интеграции и экспорта данных, полученных методом ИСП-МС, осуществляется поэтапно с учётом требований к функциональности, надёжности и удобству использования. Ниже приведены ключевые этапы реализации программного средства:

1. Анализ требований и проектирование архитектуры
   * Сбор и уточнение пользовательских требований, в том числе изучение особенностей структуры JSON-баз данных, с которыми предстоит работать.
   * Определение функциональных и нефункциональных требований к модулю.
   * Проектирование общей архитектуры программного модуля с выделением основных компонентов.
2. Разработка пользовательского интерфейса
   * Создание структуры графического интерфейса средствами WPF.
   * Реализация окон интеграции, экспорта и панели краткого содержания.
   * Добавление элементов управления (кнопки, поля выбора, индикаторы прогресса).
   * Обеспечение локализации интерфейса.
3. Реализация логики обработки данных
   * Импорт и десериализация JSON-файлов с использованием встроенной библиотеки System.Text.Json.
   * Разработка алгоритма объединения баз данных с устранением дубликатов по ключевым полям.
   * Реализация фильтрации данных по идентификатору съёмки с сохранением результатов в новый JSON-файл.
   * Формирование логики выбора пути сохранения результатов операций.
4. Реализация вспомогательных функций и отчётности
   * Создание механизма генерации текстовых отчётов по операциям интеграции и экспорта.
   * Сохранение отчётов в соответствующие папки внутри проекта.
   * Добавление кнопок для открытия директорий с отчётами через проводник.
   * Визуализация краткой структуры баз данных с использованием элемента TreeView.
5. Тестирование и отладка
   * Проведение модульного тестирования основных функций.
   * Проверка корректности объединения, фильтрации и сериализации данных.
   * Тестирование обработки исключений и вывода сообщений об ошибках.
   * Оценка времени выполнения операций при работе с большими объёмами данных.

**4.2 Пользовательский интерфейс**

***4.2.1 Описание взаимодействия с пользователем***

Взаимодействие пользователя с программным модулем представляет собой последовательный процесс использования функций, предоставляемых системой. При проектировании пользовательского интерфейса особое внимание уделяется логике взаимодействия пользователя с системой — от выбора исходных файлов до получения результатов операций. Для наглядного отображения этого взаимодействия и понимания последовательности пользовательских действий применяется UML-диаграмма активности. В рамках данной работы оператор взаимодействует с модулем, выполняя такие ключевые операции, как интеграция баз данных и экспорт данных по идентификатору съёмки.

На рисунке 2 представлена диаграмма активностей для Оператора.

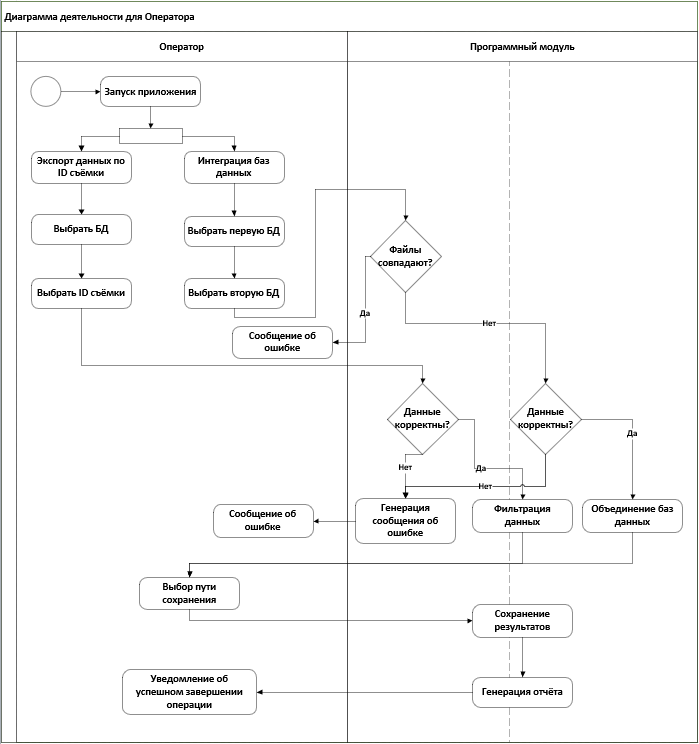


Рисунок 2 - UML диаграмма активностей для Оператора

Для более точного описания взаимодействия с пользователем в рамках отдельных сценариев были разработаны диаграммы последовательности, отражающие логику выполнения ключевых операций. Эти диаграммы иллюстрируют последовательность взаимодействий между участниками процесса. В программном модуле выделены два основных пользовательских сценария:

1. Процесс интеграции баз данных начинается с выбора пользователем двух JSON-файлов. При нажатии кнопки начала операции происходит валидация данных и, если данные корректные выполняется объединение данных с устранением дублирующихся записей. Затем происходит формирование новой базы данных, которая сохраняется по заданному пользователем пути. В завершение автоматически формируется текстовый отчёт. На рисунке 3 представлена диаграмма последовательности для интеграции баз данных.

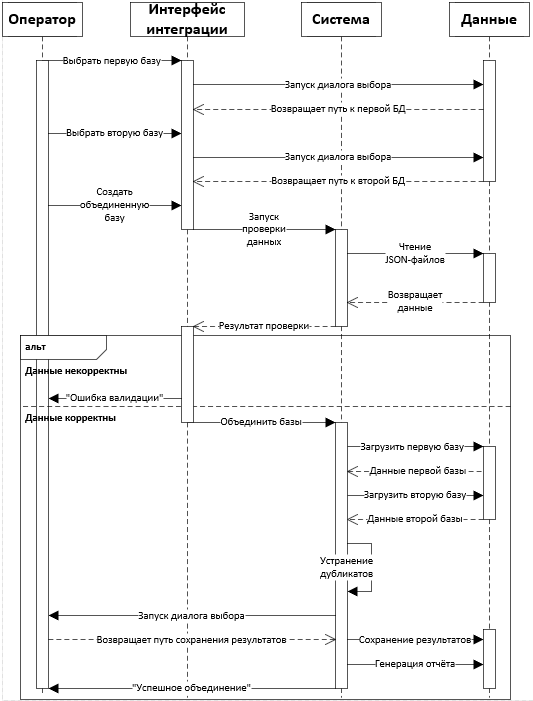


Рисунок 3 - UML диаграмма последовательности "Интеграция баз данных"

1. Процесс экспорта данных по идентификатору съёмки начинается с выбора пользователем JSON-файла. После загрузки базы система автоматически определяет все доступные идентификаторы съёмок, из которых пользователь выбирает нужный. При запуске операции выполняется предварительная валидация структуры и содержимого данных, после чего производится фильтрация записей. Результат сохраняется в новый файл по указанному пути. По завершению операции автоматически формируется текстовый отчёт. На рисунке 4 представлена диаграмма последовательности для экспорта данных по ID съёмки.

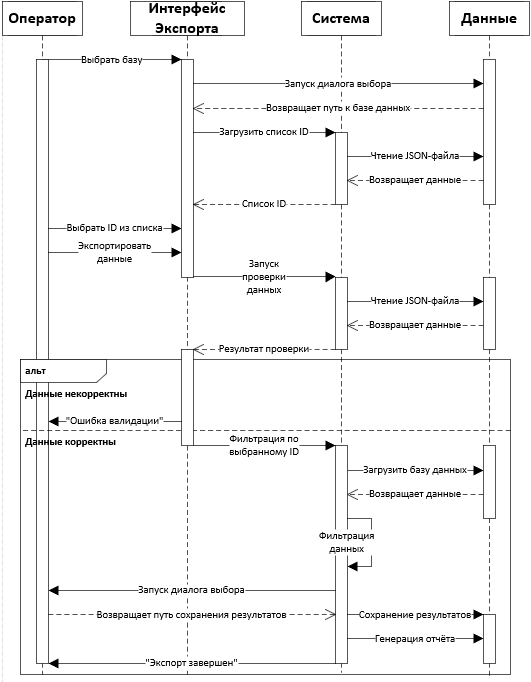


Рисунок 4 - UML диаграмма последовательности "Экспорт данных"

***4.2.2 Определение операций пользователей и составление функциональных блоков***

В рамках программного модуля для обработки данных ИСП-МС пользователи выполняют операции, связанные с интеграцией, фильтрацией и анализом геохимических данных. Ниже приведён список операций, доступных через интерфейс системы, а также функциональные блоки, объединяющие их по назначению:

1. Выбор JSON-файлов для интеграции или экспорта.
2. Выбор идентификатора съёмки для экспорта данных.
3. Интеграция баз данных с устранением дубликатов.
4. Экспорт данных по идентификатору съёмки.
5. Задание пути сохранения результата обработки.
6. Просмотр краткого содержания загруженных баз данных.
7. Открытие папок с автоматически сформированными отчётами по результатам операций.

Для упорядочивания реализованных операций и облегчения понимания логики работы программного модуля, все операции пользователя можно объединить в функциональные блоки по их назначению и характеру выполняемых задач. Ниже представлены основные функциональные блоки программного модуля:

1. Блок интеграции баз данных
   * Включает операции по загрузке двух файлов JSON, выполнению проверки на дубликаты, объединению содержимого в единую структуру и сохранению результата в новый файл. Также обеспечивает генерацию текстового отчёта и доступ к каталогу с отчётами.
2. Блок экспорта данных по ID съёмки
   * Осуществляет загрузку исходной базы данных, выбор идентификатора съёмки, извлечение связанных с ним записей и их сохранение в отдельный JSON-файл. Дополнительно формируется отчёт о выполненной операции и реализована кнопка для перехода к соответствующей папке.
3. Блок визуализации информации
   * Реализует просмотр краткого содержания загруженных баз в виде иерархического списка с помощью боковой панели. Это позволяет пользователю быстро ознакомиться со структурой данных без необходимости вручную открывать и анализировать файлы.

***4.2.3 Проектирование структуры экранов и схемы навигации***

{На этом этапе, проектируются взаимосвязи форм (окон) приложения и взаимосвязи объектов внутри форм.

Основываясь на сценариях работы и ролях пользователей, формируется структура экранов системы, т.е. определяется количество экранов, функциональность каждого из них, навигационные связи между ними, формируется структура меню и других навигационных элементов.

**Привести навигационную схему.** Навигационная схема показывает механизм распределения функций и задач между экранами.

Навигационная схема определяет, каким образом пользователи смогут перемещаться между различными экранами.}

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст

***4.2.4 Разработка дизайна интерфейса***

Одним из ключевых этапов разработки программного модуля является проектирование пользовательского интерфейса, который должен обеспечивать интуитивно понятную и комфортную работу с JSON-базами данных. Визуальное оформление интерфейса ориентировано на практическое применение в лабораторных условиях, где важны лаконичность, читаемость и функциональная логика расположения элементов.

Интерфейс выполнен в тёмных цветах, но с чётким выделением основных элементов, что обеспечивает комфортную работу при слабом освещении. В качестве основного шрифта выбран Tahoma — простой и легко читаемый шрифт, обеспечивающий визуальное единообразие всех текстовых элементов. Интерфейс разделён на вкладки, каждая из которых отвечает за выполнение определённой группы операций. Все кнопки, поля ввода и текстовые метки имеют чёткие подписи, что упрощает взаимодействие с программным модулем.

При запуске программного модуля по умолчанию открывается вкладка "Интеграция баз данных", представленная на рисунке 2. Данное окно предназначено для выбора и загрузки двух JSON-файлов, которые впоследствии объединяются в одну итоговую базу. Интерфейс включает два текстовых поля для отображения путей к выбранным файлам, кнопки выбора файлов, индикатор прогресса объединения и кнопку запуска операции. Ниже располагается кнопка "Отчёты по интеграции", позволяющая открыть директорию с автоматически сгенерированными отчётами.

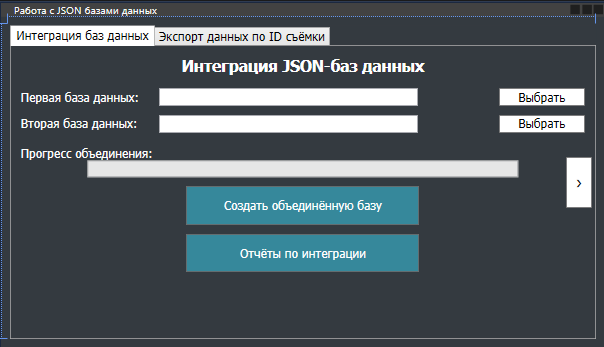


Рисунок 5 - Окно интеграции баз данных

Следующая вкладка — "Экспорт данных по ID съёмки", показана на рисунке 3. В данной вкладке реализован механизм выборки данных на основе уникального идентификатора съёмки. Пользователю предоставляется возможность выбрать JSON-базу, после чего автоматически формируется список ID съёмок. Выбрав нужное значение, пользователь может выполнить экспорт данных и сохранить его в отдельный файл. Также реализована кнопка перехода к отчётам по экспорту данных.

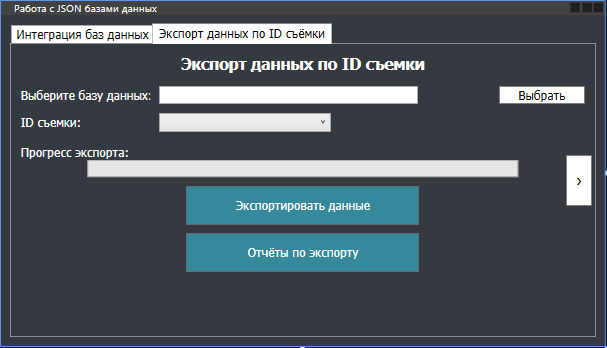


Рисунок 6 - Окно экспорта данных по ID съёмки

С правой стороны окна программного модуля реализована боковая панель, представленная на рисунке 4. Боковая панель открывается нажатием соответствующей кнопки. Она содержит компонент TreeView, предназначенный для визуализации краткого содержания загруженных баз данных в виде иерархического списка. Панель предоставляет пользователю возможность предварительно ознакомиться со структурой данных без необходимости открывать JSON-файл вручную.

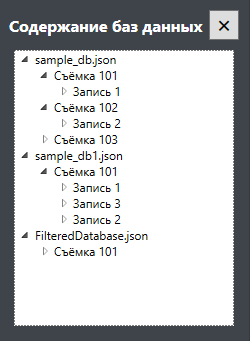


Рисунок 7 - Панель краткого содержания

**4.3 Входные, выходные и промежуточные данные**

Программный модуль работает с тремя категориями данных: исходные файлы, результаты операций и временные структуры в памяти. Их структура и взаимодействие определяются требованиями предметной области и функционалом системы.

Входные данные представлены в виде JSON-файлов, содержащих результаты геохимических съёмок, полученные методом ИСП-МС. Каждый файл включает массив записей, где каждая запись описывает пробу с набором обязательных полей:

* Идентификатор пробы
* Идентификатор съёмки
* Название пробы
* Время сканирования
* Дата добавления записи
* Статус пробы
* Словарь концентраций химических элементов (название элемента, числовое значение)
* Текстовые комментарии
* Серийный номер материнского раствора

Перед обработкой входные JSON-файлы проходят валидацию – процесс проверки структуры и корректности данных. Этот процесс включает: проверку наличия корневого массива, контроль обязательных полей и их типов, проверку уникальности записей по комбинации полей идентификатор пробы и идентификатор съёмки, а также проверку допустимых диапазонов значений. Только после успешного прохождения всех проверок данные десериализуются в коллекции объектов, которые сохраняются в памяти для последующих операций. В ходе выполнения операций формируются промежуточные данные:

* Для устранения дубликатов при объединении используется хеш-таблица, где ключом выступает комбинация идентификатор пробы и идентификатор съёмки.
* Для фильтрации по идентификатору съёмки создаётся временная выборка записей, удовлетворяющих условию, с использованием LINQ-запросов.
* На этапе генерации отчётов промежуточно формируется текстовая сводка с основными показателями выполнения операций.

Выходные данные представляются собой результаты выполнения операций интеграции и экспорта данных:

* Объединённые JSON-файлы, структура которых повторяет входные данные, но исключает дубликаты.
* Отфильтрованные JSON-файлы, содержащие только записи, соответствующие выбранному идентификатору съёмки.
* Текстовые отчёты с ключевыми параметрами операций: количество обработанных записей, список исходных файлов, идентификаторы съёмок, время выполнения операций, ошибки.

**4.4 Реализация используемых методов хранения, обработки и представления информации об объектах предметной области**

***4.4.1 Методы хранения данных***

В рамках программного модуля используются файловые базы данных в формате JSON, хранящие результаты геохимических исследований, полученные методом ИСП-МС. Каждый JSON-файл представляет собой структурированный набор данных, где ключевым элементом является массив объектов. Каждый объект описывает пробу, полученную в ходе геохимической съёмки, и содержит следующие поля:

* id (целое число) – уникальный идентификатор пробы;
* survey\_id (целое число) – идентификатор съёмки;
* name (строка) – название пробы;
* scan\_time (строка) – время сканирования;
* date\_added (строка) – дата добавления записи;
* status (целое число) – статус обработки пробы;
* concentrations (объект) – словарь концентраций химических элементов;
* Remarks (строка) — комментарии;
* Mother liquor serial number (строка) — серийный номер материнского раствора.

На рисунке 2 приведен пример структуры JSON-файла, со всеми обязательными полями.

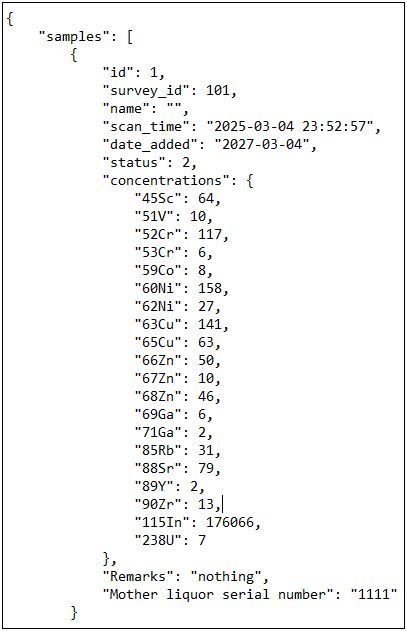


Рисунок 8 - Пример структуры JSON-файла

Кроме того, в программном модуле реализована генерация отчётов для сохранения информации о результатах операций интеграции и экспорта данных. Отчёты сохраняются в текстовом формате и содержат информацию о количестве обработанных записей, идентификаторах съёмок, путях к исходным и результирующим файлам, а также времени выполнения операций. Алгоритм формирования отчётов включает:

* Создание временного файла с использованием класса для работы с текстовыми данными.
* Последовательное добавление метаданных операций через построчное заполнение.
* Сохранение файла в соответствующую директорию (IntegrationReports или ExportReports).

***4.4.2 Реализация методов обработки информации***

Обработка данных в программном модуле направлена на обеспечение точности, устранение дубликатов и формирование структурированных результатов. Основные этапы обработки включают валидацию, интеграцию и фильтрацию данных, что соответствует методам, описанным в разделе 2.1.3 пояснительной записки. Ниже приведены реализованные методы обработки данных:

1. Валидация данных – проверка корректности структуры и содержимого JSON-файлов. Проводится перед выполнением операций и в случае обнаружении ошибок формируется сообщение для пользователя, и операция прерывается. Алгоритм проверки анализирует:
   * Наличие обязательных полей.
   * Корректность типов данных.
   * Уникальность записей по комбинации идентификатор пробы и идентификатор съёмки;
   * Допустимость значений.
2. Интеграция данных – объединение двух баз данных через метод, который принимает два списка объектов. Для устранения дубликатов используется хеш-таблица, где ключом выступает комбинация идентификаторов пробы и съёмки. Алгоритм работает следующим образом:
   * Загружаются JSON-файлы и десериализуются в коллекции объектов.
   * Записи из первого файла добавляются в таблицу.
   * Записи из второго файла проверяются на наличие в таблице по ключевым полям: идентификатор пробы и идентификатор съёмки. Если запись отсутствует, она добавляется.
   * При обнаружении дубликата обновляются поля «Комментарий» и «Серийный номер материнского раствора», если они не заполнены в исходной записи.
   * Происходит сериализация объединённого набора данных в новый JSON-файл с сохранением структуры.
3. Экспорт данных по идентификатору съёмки – извлечение данных, относящихся к конкретной съёмке. Для корректного извлечения данных применяется фильтрация. Алгоритм работает следующим образом:
   * Загружается выбранная база данных и десериализуется в коллекцию объектов.
   * С помощью LINQ-запроса, отбираются записи, соответствующие выбранному идентификатору съёмки.
   * Отфильтрованные данные сериализуются в новый файл.

***4.4.3 Реализация методов визуального представления данных***

Визуализация данных играет ключевую роль в обеспечении удобства работы пользователя. В программном модуле реализована панель краткого содержания, которая отображает структуру загруженных баз данных в виде иерархического дерева. Алгоритм построения иерархического представления включает:

1. Группировку записей по идентификатору съёмки: Все пробы, относящиеся к одной съёмке, объединяются в родительский узел.
2. Детализацию проб: Каждая проба отображается как дочерний элемент съёмки и содержит: идентификатор пробы, время сканирования, дату добавления и название.
3. Динамическое обновление: При загрузке новых данных или изменении существующих дерево автоматически перестраивается, отражая актуальное состояние баз.

На рисунке 3 приведён пример того, как выглядит визуализация структуры данных JSON-файла на панели краткого содержания.

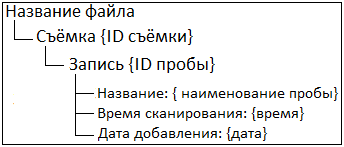


Рисунок 9 - Структура визуализации данных

**4.5 Описание архитектурного решения**

***4.5.1 Структурная организация программной системы***

{В этом разделе следует показать организацию программной системы, структурных элементов и объединение этих элементов в более крупные подсистемы. Также можно показать, какое место разрабатываемое ПС (ПМ) занимает в более крупной системе.

Можно использовать статические UML диаграммы: диаграммы компонентов, диаграммы развертывания. С использованием **диаграммы компонентов UML** (она иллюстрирует архитектуры компонентов программного обеспечения и зависимости между ними) показать, как компоненты соединяются вместе для формирования программной системы. С помощью **диаграммы развертывания UML** показать, как программная система будет физически развернута на аппаратном обеспечении.}

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст

***4.5.2 Архитектура программного кода***

{В этом разделе должна быть описана структура кода ПС, т. е.:

- из каких функциональных блоков (файлов, модулей, процедур, функций, классов) состоит ПС;

- приведено описание каждого блока с его названием и назначением;

- приведена графическая схема взаимосвязи этих блоков.

Для описания структуры кода можно использовать **диаграммы классов UML**.}

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст

5 ТЕСТИРОВАНИЕ И ОПТИМИЗАЦИЯ

**5.1 План тестирования**

Тестирование программного модуля является важным этапом разработки, направленным на обеспечение стабильной и корректной работы программного средства, а также соответствие его функциональных возможностей поставленным требованиям. Программный модуль ориентирован на работу с JSON-базами данных и реализует операции интеграции и экспорта данных по идентификатору съёмки. В связи с этим основное внимание при тестировании будет уделено проверке функциональности, производительности при работе с разными объёмами информации, корректности обработки данных, а также удобству использования интерфейса.

1. Функциональное тестирование – направлено на проверку корректности реализации основных функций программного модуля и их соответствие заданным требованиям. В рамках данного этапа будет протестировано:
   * Выбор исходных баз данных: проверка корректности работы функций выбора файлов для интеграции и экспорта.
   * Интеграция баз данных: проверка успешного объединения данных из разных файлов, устранения дубликатов и сохранения итогового файла.
   * Экспорт данных по ID съёмки: проверка фильтрации и извлечения записей по выбранному идентификатору съёмки, корректность сохранения экспортированного файла.
   * Отображение краткого содержания: проверка корректности работы панели краткого содержания, а также корректности отображаемой информации.
   * Создание текстовых отчётов: проверка автоматической генерации отчётов по операциям, полноты и корректности зафиксированной информации;
   * Работа с путями сохранения: проверка функциональности задания пути сохранения для итоговых файлов и корректности их записи в выбранные директории.
   * Оповещение об ошибках: проверка генерации и отображения информативных сообщений об ошибках при возникновении проблем.
2. Тестирование производительности – направлено на оценку скорости выполнения операций при работе с различными по объёму JSON-базами данных. Тестирование позволит определить, как модуль справляется с нагрузкой, а также выявить потенциальные узкие места в логике обработки. Оценке будут подлежать:
   * Время загрузки и отображения краткого содержания баз: оценка времени от выбора файлов до появления информации в интерфейсе.
   * Скорость выполнения операций интеграции и экспорта: оценка времени, необходимого для обработки различных по размеру баз данных.
3. Тестирование корректности обработки данных – направлено на проверку надежности механизмов обработки данных. В рамках данного этапа будут проверены следующие аспекты:
   * Сохранение структуры данных: проверка сохранения исходной структуры JSON при интеграции и экспорте.
   * Сохранение целостности данных: проверка корректного переноса значений без потерь и искажений.
   * Корректность интеграции баз данных: проверка того, что в файле с результатами интеграции отсутствуют дубликаты записей.
   * Корректность фильтрации данных по идентификатору съёмки: проверка того, что в экспортированный файл попадают только записи, соответствующие выбранному идентификатору.
   * Формирование отчётов с актуальной и достоверной информацией: проверка на соответствие отчётов фактически выполненным действиям.
4. Тестирование пользовательского интерфейса – направлено на проверку удобства и интуитивной понятности пользовательского интерфейса. В рамках данного тестирования будет оценено:
   * Доступность и понятность элементов управления: оценка понятности элементов интерфейса, включая обозначения и сообщения.
   * Простота использования: оценка простоты использования интерфейса без обращения к дополнительной инструкции.

В рамках программного модуля не предполагается проведение тестирования совместимости с различными операционными системами, поскольку программное средство разрабатывается исключительно для среды Windows 10 и выше. Также не планируется проведение нагрузочного тестирования в условиях многопользовательской работы, так как модуль предназначен для локального использования одним оператором без взаимодействия с сервером или сетевой инфраструктурой. Тестирование безопасности, связанное с управлением правами доступа или защитой от внешних угроз, не является актуальным, так как программный модуль функционирует автономно и не предполагает сетевых подключений. Отказ от данных видов тестирования обусловлен спецификой применения и архитектурой программного средства.

**5.2 Результаты тестирования и оптимизация**

В рамках проведения тестирования программного модуля выполняется пошаговая проверка его ключевых функций в соответствии с ранее составленным планом. Целью тестирования является подтверждение корректной работы всех реализованных механизмов, а также проверка устойчивости модуля при работе с различными объёмами информации и оценка удобства пользовательского интерфейса. В соответствие с составленным планом были проведены следующие виды тестирования:

1. Функциональное тестирование:
   * Выбор исходных баз данных: тестирование подтвердило корректную работу механизма выбора JSON-файлов. После выбора файла путь к нему отобразился в соответствующей строке интерфейса. Подтверждение на рисунке 10.

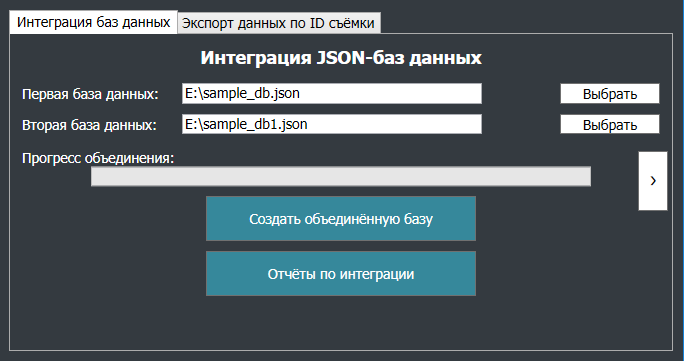


Рисунок 10 - Выбор исходных файлов

* + Интеграция баз данных: проверка показала, что данные из выбранных файлов корректно объединяются, дубликаты устраняются, итоговый файл сохраняется в заданное место. Подтверждение на рисунке 11.

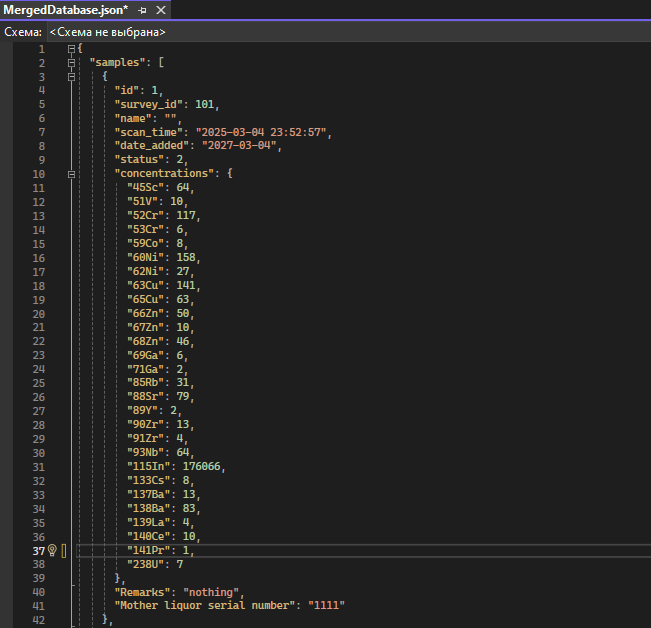


Рисунок 11 - Фрагмент результата объединения БД

* + Экспорт данных по ID съёмки: при выборе идентификатора из списка система извлекает только соответствующие записи и сохраняет их в отдельный файл. Подтверждение на рисунке 12.

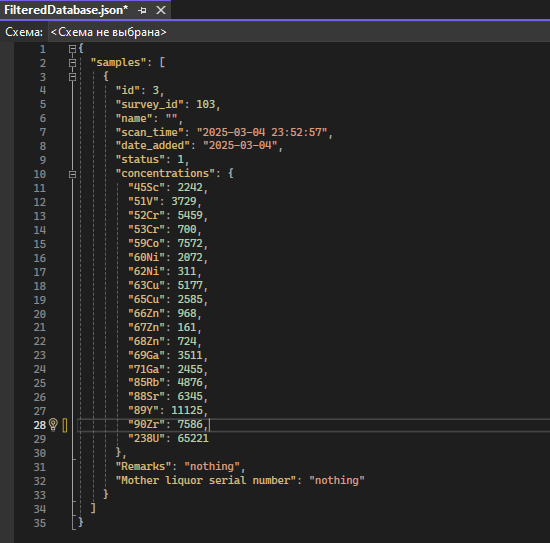


Рисунок 12 - Фрагмент результата экспорта данных

* + Отображение краткого содержания: панель краткого содержания корректно отображает иерархическую структуру загруженных баз данных. Подтверждение на рисунке 13.

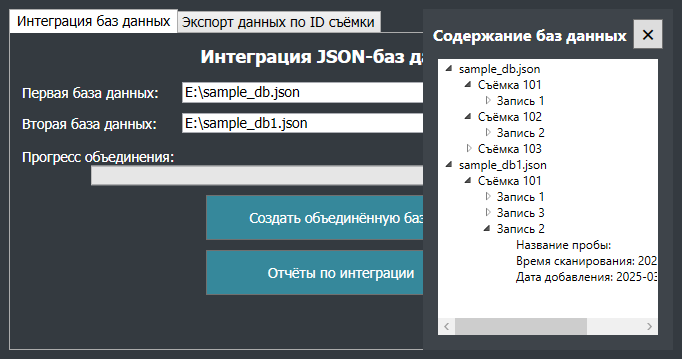


Рисунок 13 - Отображение краткого содержания БД

* + Создание текстовых отчётов: система автоматически генерирует отчёты по каждой операции. Содержание отчётов соответствует фактическому результату операции. Подтверждение на рисунке 14.

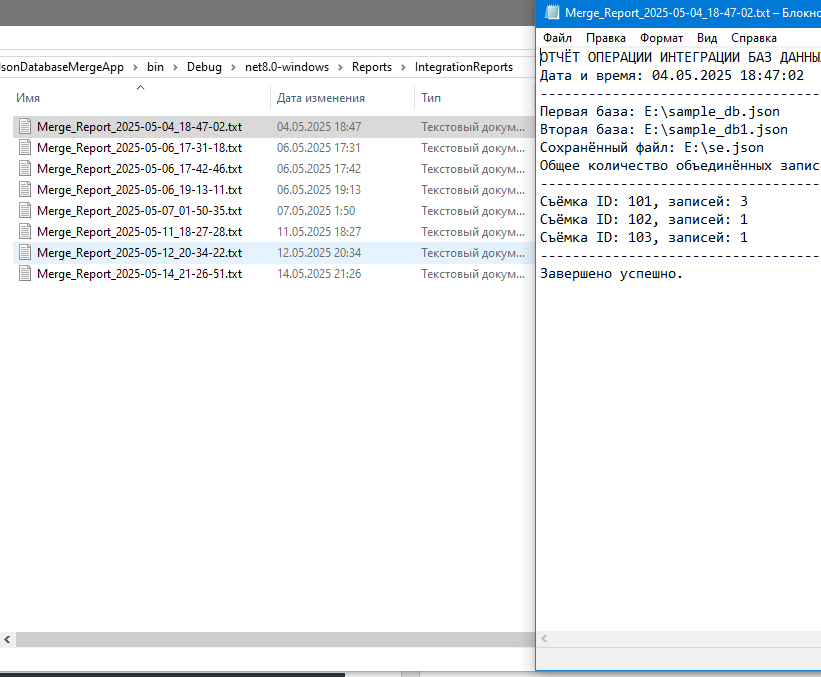


Рисунок 14 - Папка с отчётами по интеграции и фрагмент отчёта

* + Работа с путями сохранения: пользователь может вручную указать путь сохранения результатов, при этом итоговые файлы записываются без сбоев. Подтверждение на рисунке 15.

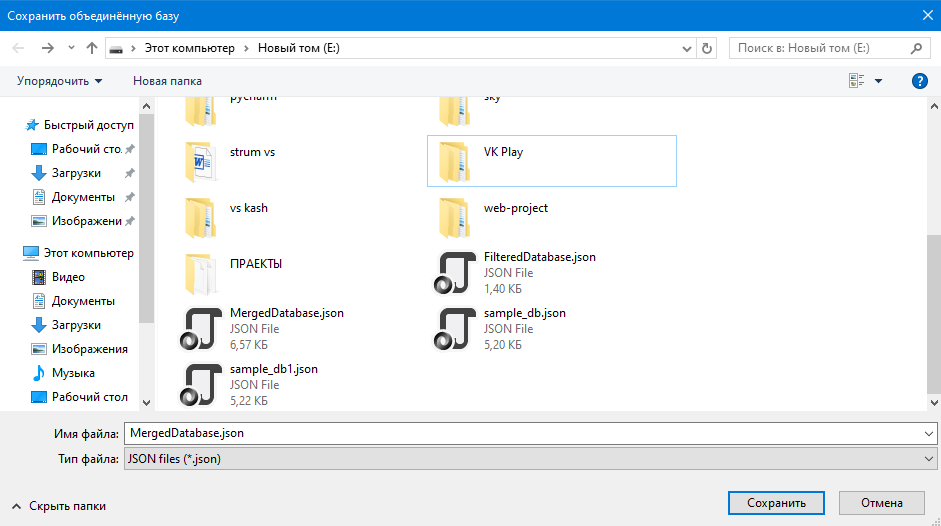


Рисунок 15 - Указание пути сохранения результатов

* + Оповещение об ошибках: при попытке запуска операции без указанных данных или при ошибках структуры система отображает соответствующее сообщение. Подтверждение на рисунке 16.

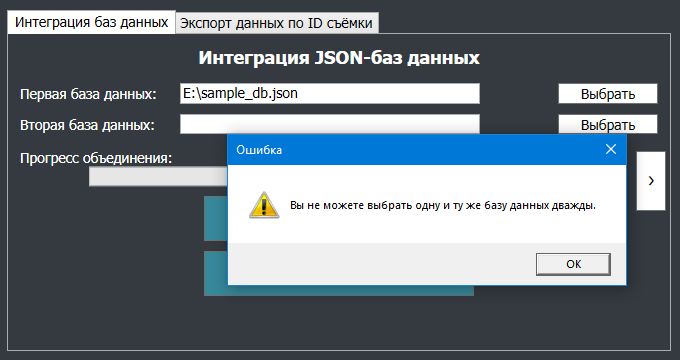


Рисунок 16 - Пример оповещения об ошибке

1. Тестирование производительности:
   * Тестирование проводилось на JSON-файлах различного объёма. При объединении и экспорте файлов объёмом до 5 МБ операции выполнялись в пределах 2–3 секунд. При объёмах более 10 МБ время выполнения возрастает, но остаётся в пределах приемлемого для настольного приложения. Скорость отображения краткого содержания также остается стабильной и не вызывает зависаний.
2. Тестирование корректности обработки данных:
   * В ходе мануальной проверки установлено, что модуль корректно работает с JSON-файлами: структура сохраняется без изменений, числовые и строковые данные переносятся без искажений. Фильтрация по идентификатору съёмки выполняется точно, в экспорт попадают только релевантные записи. Отчеты, генерируемые модулем, содержат полную и достоверную информацию о процессе обработки.
3. Тестирование пользовательского интерфейса:
   * Тестирование проводилось с участием пользователей, ранее не знакомых с программой. Интерфейс показал высокую степень доступности: все элементы подписаны, основные действия легко найти. Навигация по вкладкам и вызов операций проходят без затруднений. Интерфейс оказался интуитивно понятным для пользователей.

В результате проведённого тестирования программного модуля были успешно проверены все ключевые функции, предусмотренные техническими требованиями. Результаты подтверждают стабильную и корректную работу программного модуля при различных сценариях использования. Все функциональные компоненты работают корректно, производительность находится на удовлетворительном уровне, интерфейс отличается доступностью и удобством. Оптимизация программного модуля не требуется.

6 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

{В этот раздел могут быть включены следующие пункты:

- порядок установки и настройки программного средства на компьютер пользователя или первичная настройка технического средства;

- действия пользователя в случае сбоя ПС или ПТС;

- последовательность действий пользователя для решения своих основных задач при работе с программным или техническим средством.

Следует привести образы экранов (ScreenShots) с пояснениями для лучшего восприятия раздела. Приветствуется создание помощи по установке и эксплуатации ПС или ПТС.}

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст

**ЗАКЛЮЧЕНИЕ**

В рамках выполнения выпускной квалификационной работы была решена задача разработки программного модуля для интеграции и экспорта данных, полученных методом ИСП-МС. Разработанный программный модуль обеспечивает автоматизацию обработки данных, что способствует повышению эффективности работы специалистов и снижению трудозатрат при выполнении рутинных операций. Поставленная цель полностью достигнута, а сформулированные задачи, изложенные во введении, были поэтапно реализованы.

В процессе выполнения работы:

* Проведён анализ предметной области и сформулированы функциональные и нефункциональные требования к разработке программного модуля.
* Разработана архитектура программного средства, ориентированного на работу с локальными JSON-базами данных.
* Реализован программный модуль на языке программирования C# с использованием платформы WPF.
* Разработан пользовательский интерфейс с отображением иерархического содержания баз данных и индикаторами прогресса выполнения операций.
* Реализованы основные функции: выбор и загрузка JSON-файлов, их интеграция с устранением дубликатов, экспорт данных по идентификатору съёмки.
* Реализован механизм автоматического формирования текстовых отчётов по результатам операций.
* Проведено тестирование программного модуля, подтверждающее его корректную работу при различных сценариях использования.

Практическая значимость и актуальность разработанного программного модуля обусловлены его способностью повысить эффективность и надёжность обработки данных в лабораторных условиях. Практическая значимость модуля заключается в следующем:

* Повышение эффективности обработки данных за счёт автоматизации ручных операций.
* Снижение риска ошибок при объединении и фильтрации информации.

Программный модуль рекомендуется к внедрению в рабочие процессы специалистов, занимающихся геохимическим и геофизическим анализом. Он может применяться при подготовке данных к дальнейшему анализу, а также использоваться в образовательных целях для демонстрации методов структурирования и обработки аналитической информации. Кроме того, программный модуль может быть интегрирован с другими системами в сфере геохимических исследований, расширяя их функциональные возможности.

В дальнейшем возможны следующие направления развития программного средства:

* Расширение форматов экспортируемых данных (например, поддержка CSV, Excel);
* Внедрение функции предварительного просмотра записей перед выполнением операций;
* Добавление функции объединения данных более чем из двух исходных баз с возможностью гибкой настройки приоритетов.

Таким образом, разработанный программный модуль интеграции и экспорта данных съёмок представляет собой эффективное средство для автоматизации обработки результатов, полученных методом ИСП-МС. Он обладает высокой практической значимостью для лабораторных исследований, упрощает работу с локальными базами данных и способствует снижению количества ошибок при выполнении рутинных операций. Модуль имеет потенциал для дальнейшего развития и может быть адаптирован под расширенные задачи анализа и визуализации геохимических данных.

Общий объём программного кода составил \_\_ строк. Листинг кода представлен в приложении В.

**СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ**

1. Microsoft .NET Framework: официальная документация [Электронный ресурс]. – URL: https://docs.microsoft.com/en-us/dotnet/. – При разработке программного модуля использованы стандартные библиотеки .NET для работы с файлами, сериализации данных и многопоточности.
2. WPF: руководство разработчика [Электронный ресурс]. – URL: https://docs.microsoft.com/en-us/dotnet/desktop/wpf/. – При создании пользовательского интерфейса применены рекомендации по работе с элементами управления XAML, анимацией и привязкой данных.
3. JSON: спецификация формата [Электронный ресурс]. – URL: https://www.json.org/json-ru.html. – В проекте использованы принципы структурирования данных в формате JSON для хранения и обработки результатов геохимических исследований.
4. System.Text.Json: документация [Электронный ресурс]. – URL: https://docs.microsoft.com/en-us/dotnet/api/system.text.json. – Для сериализации и десериализации JSON-данных использована библиотека System.Text.Json.
5. LINQ: официальная документация [Электронный ресурс]. – URL: https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/linq/. – При реализации фильтрации данных по ID съёмки использованы методы LINQ для работы с коллекциями.

**ПРИЛОЖЕНИЯ**

**Приложение А**

{Содержит **техническое задание на реализацию программной системы**, разработанное при непосредственном участии представителей заинтересованных организаций.}

**Приложение Б**

{Содержит примеры входных данных в виде изображений, таблиц, фрагментов текстовых документов и.т.д.}

**Приложение В**

{Содержит фрагменты листинга программного кода.

**Обязательно наличие поясняющего текста (для чего предназначен фрагмент кода, функция, класс)!**

Привести не менее 10 страниц кода. Текстовые пояснения оформляются в соответствии с ГОСТ.

Параметры оформления листинга кода: размер шрифта 10, допускается расположение в 2 столбца, одинарный междустрочный интервал..}