Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

**ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

|  |
| --- |
| Институт информационных технологий и анализа данных |
| наименование института |

|  |  |
| --- | --- |
| Допускаю к защите |  |
| Руководитель |  |
|  | подпись |
|  | А.Ю. Юрин |
|  | И.О. Фамилия |

|  |
| --- |
| Разработка программного обеспечения с использованием |
| методов и средств программной инженерии |

наименование темы

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

к курсовому проекту по дисциплине

|  |  |  |
| --- | --- | --- |
| Методы и средства проектирования информационных систем и технологий | | |
|  | 1.002.00.00 - ПЗ |  |

обозначение документа

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил студент |  | ИСМб-19-1 |  |  |  | Д.Е. Вовиков |
|  |  | шифр группы |  | подпись |  | И.О. Фамилия |
| Нормоконтроль |  |  |  |  |  | А.Ю. Юрин |
|  |  |  |  | подпись |  | И.О. Фамилия |

Курсовой проект защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Иркутск 2022 г.

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

ЗАДАНИЕ

НА КУРСОВОЕ ПРОЕКТИРОВАНИЕ

|  |  |  |
| --- | --- | --- |
| По курсу | Методы и средства проектирования информационных систем и технологий | |
| Студенту | Вовиков Д.Е. | |
|  | (фамилия, инициалы) | |
| Тема проекта | Разработка программного обеспечения с использованием методов и средств программной инженерии | |
| Исходные данные | |  |
| 1. Произвести проектирование компонента для преобразования (конвертации) файла Enterprise Architect (xmi) в файл формата EKB (согласно варианту 2). | | |

|  |
| --- |
| Рекомендуемая литература: |
| 1. Дородных Н.О., Юрин А.Ю. Технология создания продукционных экспертных систем на основе модельных трансформаций. Новосибирск: СО РАН, 2019. 144 стр. 2. Юрин А.Ю., Грищенко М.А. Редактор баз знаний в формате CLIPS // Программные продукты и системы. 2012. № 4. С. 83–87. 3. Юрин А.Ю. CASE-средства: Методические указания по выполнению лабораторных работ. Иркутск: ИРНИТУ, 2018. 87 c. |

Графическая часть на \_\_\_\_\_\_\_\_\_\_\_\_\_\_ листах.

Дата выдачи задания « » 20 г.

|  |  |  |  |
| --- | --- | --- | --- |
| Задание получил |  |  | Д.Е. Вовиков |
|  | подпись |  | И.О. Фамилия |

Дата представления проекта руководителю « » 20 г.

|  |  |  |  |
| --- | --- | --- | --- |
| Руководитель курсового проектирования |  |  | А.Ю. Юрин |

Содержание

[Содержание 3](#_Toc121482293)

[Введение 4](#_Toc121482294)

[1 Общая часть 5](#_Toc121482295)

[1.1 Описание предметной области 5](#_Toc121482296)

[1.2 Постановка задачи 6](#_Toc121482297)

[2 Специальная часть 7](#_Toc121482298)

[2.1 Техническое задание 7](#_Toc121482299)

[2.1.1 Введение 7](#_Toc121482300)

[2.1.2 Общие сведения 7](#_Toc121482301)

[2.1.3 Назначение и цели создания 8](#_Toc121482302)

[2.1.4 Характеристика объектов автоматизации 9](#_Toc121482303)

[2.1.5 Требования к программе или программному изделию 9](#_Toc121482304)

[2.1.6 Состав и содержание работ по созданию системы 10](#_Toc121482305)

[2.1.7 Порядок контроля и приемки 10](#_Toc121482306)

[2.1.8 Требования к программной документации 11](#_Toc121482307)

[2.1.9 Приложения 11](#_Toc121482308)

[2.2 Описание проекта в Trello 11](#_Toc121482309)

[2.3 Проектирование 12](#_Toc121482310)

[2.3.1 Диаграммы вариантов использования 12](#_Toc121482311)

[2.3.2 Диаграмма последовательностей 18](#_Toc121482312)

[2.3.3 Алгоритмическое обеспечение 19](#_Toc121482313)

[2.3.4 Диаграмма классов 20](#_Toc121482314)

[2.4 Программная реализация 21](#_Toc121482315)

[2.4.1 Автоматически сгенерированный код 21](#_Toc121482316)

[2.4.2 Описание интерфейса 24](#_Toc121482317)

[2.4.3 Тестирование 27](#_Toc121482318)

[Заключение 34](#_Toc121482319)

[Список использованных источников 35](#_Toc121482320)

[Приложение А. Фрагмент сгенерированной документации 36](#_Toc121482321)

[Приложение Б. Листинг программы 37](#_Toc121482323)

Введение

В рамках проекта осуществляется проектирование и разработка программного обеспечения, обеспечивающего:

конвертацию файлов структуры Enterprise Architect в формат редактора продукционных баз знаний EKB.

Актуальность разрабатываемого программного обеспечения определяется временными затратами в рамках ручного переноса проектов между системами.

Разрабатываемый программное обеспечение должно автоматизировать деятельность специалистов, в области проектирования информационных систем, сокращая их временные затраты при переносе и поддержании в актуальном состоянии проектов разной структуры в рамках представленных систем.

1 Общая часть

1.1 Описание предметной области

Программный компонент осуществляет процесс конвертации файлов Enterprise Architect формата редактора продукционных баз знаний EKB.

Редактор баз знаний EKB сохраняет проекты с расширением .ekb и содержит многоуровневую структуру.



Рисунок 1 – Фрагмент структуры файла .ekb

CASE-инструмент для проектирования и конструирования программного обеспечения хранит файлы с расширением .xmi. XMI – стандарт OMG (Object Management Group) для обмена метаданными с помощью языка XML. Может использоваться для любых метаданных, если их метамодель может быть выражена с помощью MOF (Meta-Object Facility). Наиболее часто XMI применяется как формат обмена UML-моделями.



Рисунок 2 – Фрагмент структуры файла .xmi

Дополнительно программный компонент предоставляет возможность выбора пользователем места хранения файла .xmi и места сохранения конвертированного файла .ekb.

1.2 Постановка задачи

Проектирование и программная реализация компонента для преобразования (конвертации) XML-файлов определенной структуры (согласно варианту задания).

Выполнение работы предполагает решение следующих задач:

* Проектирование, включая:
  + Описание предметной области.
  + Формулировка цели работы.
  + Проектирование компонента.
* Программная реализация компонента, включая автоматизированную генерацию программного кода.
* Тестирование.
* Документирование (описание), включая автоматизированную генерацию документации.

2 Специальная часть

2.1 Техническое задание

2.1.1 Введение

Настоящее техническое задание распространяется на разработку программного компонента для конвертации файлов Enterprise Architect в формат редактора продукционных баз знаний EKB.

При проектировании баз знаний в определенном программном обеспечении может потребоваться продолжить работу в другом прикладном средстве. Экспортирование наработок в данной ситуации не всегда представляется возможным.

Разрабатываемая программа позволит экспортировать проекты продукционных баз знаний EKB в файлы для работы в Enterprise Architect в автоматическом режиме.

2.1.2 Общие сведения

**2.1.2.1 Наименование и шифры**

1. Полное название системы

Программный компонент «Транслятор файлов структуры».

1. Шифр системы

Шифр системы – ПК ТФС.

**2.1.2.2 Сведения о заказчиках и исполнителях**

1. Заказчик

Иркутский Национальный Исследовательский Технический университет.

Адрес: 664074 г. Иркутск, ул. Лермонтова, 83.

1. Исполнитель

Студент Вовиков Д.Е.

**2.1.2.3 Основание для разработки**

1. Основания для разработки программного компонента

Основанием для исполнения работ по созданию программного компонента, предусмотренных в настоящем ТЗ, является учебный план «09.03.02 Информационные системы и технологии на предприятиях/в машиностроении» и методические указания курсовому проекту по курсу «Методы и средства проектирования информационных систем и технологий».

1. Основания для разработки документа

Основанием для разработки настоящего документа является п.3 к методическим указаниям.

1. Нормативные документы

Настоящее Техническое Задание разработано в соответствии с требованиями ГОСТ 34.602 89 «Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы».

При разработке и вводе в эксплуатацию автоматизированной системы Исполнитель должен руководствоваться требованиями следующих нормативных документов Госстандарта:

* ГОСТ 34.601.90 – стандарт, устанавливающий стадии и этапы создания АС, а также содержание работ на каждом этапе;
* ГОСТ 34.603.92 – стандарт, устанавливающий виды испытаний АС и общие требования к их проведению.

При создании проектно-эксплуатационной документации Исполнитель должен руководствоваться требованиями ГОСТ РД 50-34.698-90 «Автоматизированные системы. Требования к содержанию документов».

**2.1.2.4 Сроки исполнения работ**

Начало разработки – «01» сентября 2022 г.

Окончание разработки – «25» декабря 2022 г.

**2.1.2.5 Сведения об источниках и порядке финансирования работ**

Источник финансирования – бюджет Иркутского Национального Исследовательского Технического университета.

Порядок финансирования определяется условиями методического указания.

**2.1.2.6 Порядок оформления и представления заказчику результатов работ**

Порядок оформления и предъявления работы Заказчику ПК ТФС производится согласно методическому указанию. Приемка системы осуществляется комиссией, созданной Заказчиком.

2.1.3 Назначение и цели создания

**2.1.3.1 Назначение**

Программный компонент ПК ТФС предназначен для автоматической конвертации файлов прикладное CASE-средства Enterprise Architect в формат редактора продукционных баз знаний EKB.

**2.1.3.2 Цели создания**

1. Цели создания и внедрения ПК ТФС:

автоматизация конвертации файлов структуры;

1. Цели текущего этапа

Целью работ по данному этапу проекта является создание первой экспериментальной версии программного компонента.

2.1.4 Характеристика объектов автоматизации

**2.1.4.1 Краткие сведения об объекте автоматизации**

Объектом автоматизации является учебный процесс Иркутского Национального Исследовательского Технического университета.

Предметом автоматизации является процесс конвертации файлов структуры.

**2.1.4.2 Сведения об условиях эксплуатации**

Программный компонент будет эксплуатироваться в составе программного комплекса Заказчика.

2.1.5 Требования к программе или программному изделию

**2.1.5.1 Требования к функциональным характеристикам**

1. Программный компонент должен обеспечивать возможность выполнения следующих функций:

указание пути к файлу программного средства Enterprise Architect;

указание пути к месту сохранения конвертированного файла;

оповещение пользователя об успешной процедуре конвертации файла;

оповещение пользователя об ошибке при конвертации файла.

1. Исходные данные:

путь к файлу программного средства Enterprise Architect;

пусть к месту сохранения конвертированного файла.

**2.1.5.2 Требования к надежности**

Предусмотреть контроль вводимой информации.

Предусмотреть блокировку некорректных действий пользователя при работе с системой.

**2.1.5.3 Требования к составу и параметрам технических средств**

Система должна работать на IBM совместимых персональных компьютерах.

Минимальная конфигурация:

тип процессора: Celeron и выше;

объем оперативного запоминающего устройства: 2048 Мб и более.

**2.1.5.4 Требования к информационной и программной совместимости**

Система должна работать под управлением семейства операционных систем Windows NT.

**2.1.5.5 Требования к организационному обеспечению**

В ходе разработки должно обеспечиваться постоянное взаимодействие между сторонами, для чего ими должны быть сформированы рабочие группы, решающие следующие вопросы:

- административные вопросы;

- инженерно-технические вопросы;

- вопросы методического обеспечения.

2.1.6 Состав и содержание работ по созданию системы

Таблица 1 – Состав и содержание работ

|  |  |  |  |
| --- | --- | --- | --- |
| № | Название этапа | Срок | Отчетность |
| 1 | Выбор языка программирования и подходящей среды для разработки. | 01.09.2022-11.09.2022 | Обоснование выбора языка программирования. |
| 2 | Разработка ядра программного компонента. | 12.09.2022-30.10.2022 | Реализация системы на уровне ядра. Описание методов и алгоритмов. |
| 3 | Разработка базового графического интерфейса приложения. | 31.10.2022- 13.11.2022 | Описание базового графического интерфейса. |
| 4 | Разработка модуля выбора места нахождения файла и пути для сохранения результата. | 14.11.2022- 04.12.2022 | Описание методов и алгоритмов модуля. |
| 5 | Разработка расширенного графического интерфейса с возможностью выбора местоположения файла и пути для сохранения. | 05.12.2022- 11.12.2022 | Описание расширенного графического интерфейса. |
| 6 | Тестирование программ­ного продукта и составление программной документации. | 12.12.2022-25.12.2022 | Тесты. Документация. Про­граммный продукт. |

По окончанию каждого этапа заказчику будет предоставлен отчёт о выполненной работе.

2.1.7 Порядок контроля и приемки

**2.1.7.1 Сдача-приемка работ**

Сдача-приёмка работ производится поэтапно, в соответствии с Календарным графиком. Основанием для сдачи-приёмки работ служит Отчёт о завершении работ по этапу.

**2.1.7.2 Предварительные испытания**

При сдаче-приёмке готовых подсистем ПК ТФС, создаваемых в рамках настоящей работы, проводятся предварительные испытания с целью подтверждения работоспособности соответствующей подсистемы и соответствия требованиям ТЗ. Предварительные испытания должны проводиться представителями Заказчика по программе тестирования, составленной Исполнителем и согласованной с Заказчиком.

По итогам испытаний по мере необходимости Исполнителем проводится корректировка проектной и эксплуатационной документации и устранение выявленных недостатков программного обеспечения.

**2.1.7.3 Опытная эксплуатация**

Передача системы в опытную эксплуатацию производится приемо-сдаточной комиссией на основании положительного заключения, сделанного по итогам предварительных испытаний.

Опытная эксплуатация должна проводиться на технических средствах Заказчика.

2.1.8 Требования к программной документации

Разрабатываемые программные модули должны быть самодокументированы, т. е. тексты программ должны содержать все необходимые комментарии.

В состав сопровождающей документации должны входить:

Пояснительная записка на 25-30 листах, содержащая описание разработки.

Руководство пользователя.

2.1.9 Приложения

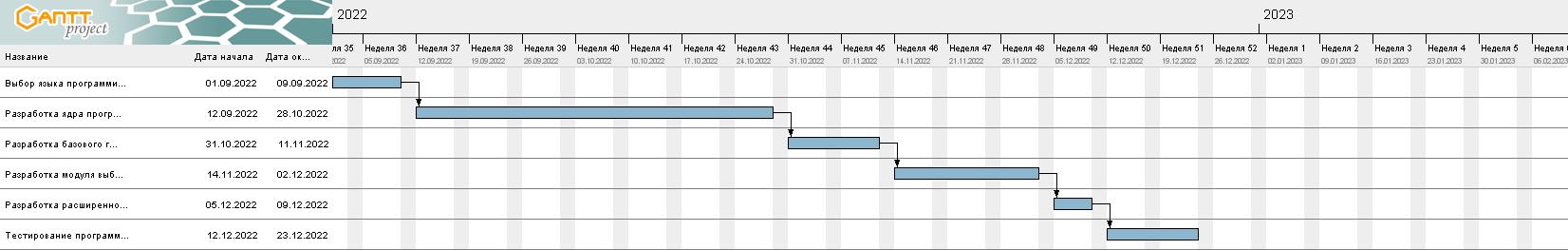


Рисунок 3 – Диаграмма Ганта

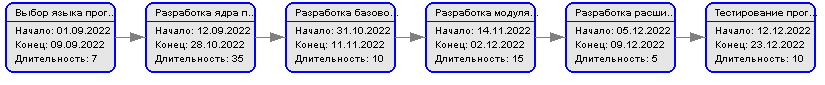


Рисунок 4 – Перт диаграмма

2.2 Описание проекта в Trello

В качестве программы для управления проектом был выбрана Trello. Для ведения прогресса выполнения курсового проекта было создано четыре колонки с соответствующими заголовками. Колонка «Сделано» содержит в себе выполненные и проверенные карточки. Колонка «На проверке» содержит карты с отданными на проверку частями курсового проекта. Колонка «В работе» предназначена для хранения карт с пунктами текущей разработки. Последняя колонка создана для хранения карт с предстоящими задачами.

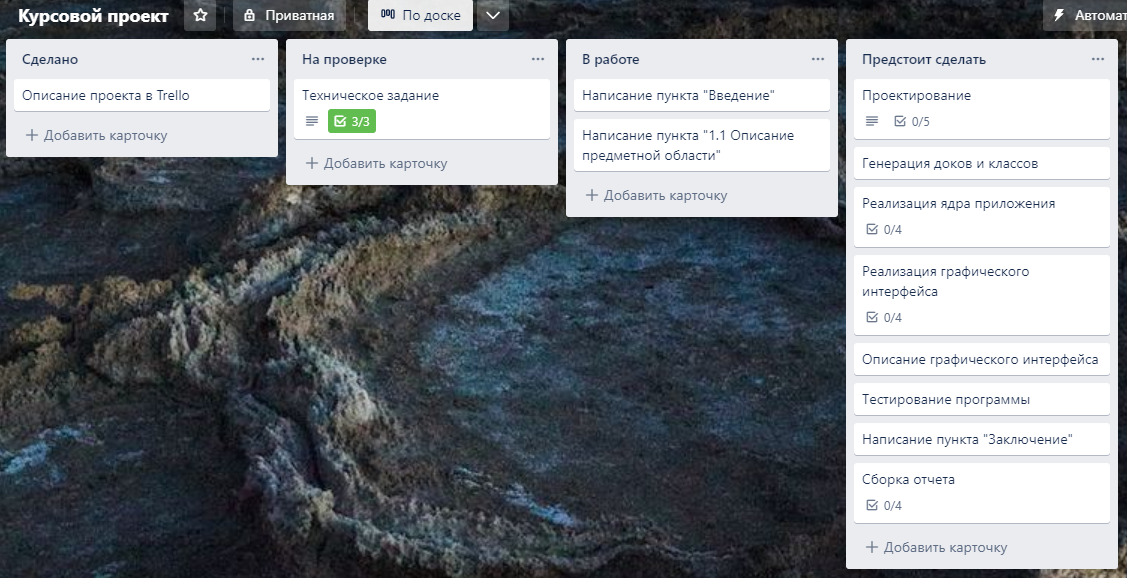


Рисунок 5 – Проект в программе Trello

2.3 Проектирование

В качестве инструмента для проектирования было использовано CASE-средство Enterprise Architect. Enterprise Architect — это инструмент визуального моделирования и проектирования, основанный на OMG UML. Платформа поддерживает: проектирование и построение программных комплексов, моделирование бизнес-процессов и моделирование отраслевых доменов.

2.3.1 Диаграммы вариантов использования

На основе составленного технического задания была составлена диаграмма вариантов использования программного компонента.

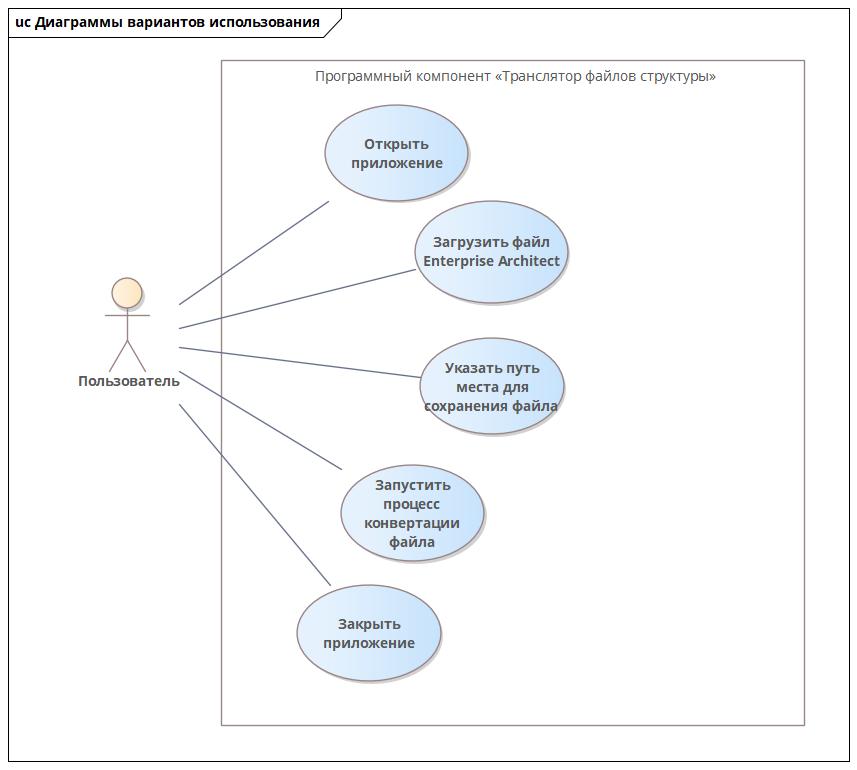


Рисунок 6 – Диаграмма вариантов использования программного компонента

Основываясь на разработанной диаграмме, были описаны основные сценарии вариантов использования приложения.

**Спецификация варианта использования «Открыть приложение»**

**Цель**: открыть программный компонент для конвертации файлов.

**Активные субъекты**: пользователь.

**Краткое описание**: пользователь открывает приложение для дальнейшей конвертации файлов.

**Основной поток событий**:

* 1. Пользователю потребовалось конвертировать файл Enterprise Architect в файл приложения EKB.
  2. Пользователь запускает приложение посредством исполняемого EXE-файла.

**Альтернативные потоки событий:** нет.

**Специальные требования**: операционная система семейства Microsoft Windows.

**Предусловия**: нет.

**Постусловия**: после активации варианта должен быть активирован один или несколько сценариев вариантов использования: «Загрузить файл Enterprise Architect», «Указать путь места для сохранения файла», «Запустить процесс конвертации файла» или «Закрыть приложение».

**Дополнительные замечания**: вариант использования «Запустить процесс конвертации файла» не имеет смысла выполнять без завершенных сценариев «Загрузить файл Enterprise Architect» и «Указать путь места для сохранения файла».

**Спецификация варианта использования «Загрузить файл Enterprise Architect»**

**Цель**: загрузка файла Enterprise Architect для дальнейшей конвертации

**Активные субъекты**: пользователь

**Краткое описание**: пользователь загружает файл Enterprise Architect посредством меню загрузки файла.

**Основной поток событий**:

* 1. Пользователю потребовалось конвертировать файл Enterprise Architect в файл приложения EKB.
  2. Пользователь запускает приложение посредством исполняемого EXE-файла.
  3. Пользователь нажимает на кнопку «Загрузить файл Enterprise Architect».
  4. Приложения открывает окно интерфейса-меню выбора файла.
  5. Пользователь указывает путь до необходимого файла с помощью меню загрузки файла.
  6. Система записывает полный пуль до файла в заранее определенную переменную.
  7. Пользователь получает текстовое уведомление об успешной загрузке файла.

**Альтернативные потоки событий:**

1. Активация сценария «Указать путь места для сохранения файла».
2. Активация сценария «Закрыть приложение».

**Специальные требования**: нет.

**Предусловия**: выполнен сценарий «Открыть приложение»

**Постусловия**: после активации варианта должен быть активирован один или несколько сценариев вариантов использования: «Указать путь места для сохранения файла», «Запустить процесс конвертации файла» или «Закрыть приложение».

**Дополнительные замечания**: вариант использования «Запустить процесс конвертации файла» не имеет смысла выполнять без завершенных сценариев «Загрузить файл Enterprise Architect» и «Указать путь места для сохранения файла».

**Спецификация варианта использования «Указать путь места для сохранения файла»**

**Цель**: указание места сохранения файла.

**Активные субъекты**: пользователь.

**Краткое описание**: пользователь указывает директорию для сохранения файла посредством меню сохранения файла.

**Основной поток событий**:

* 1. Пользователю потребовалось конвертировать файл Enterprise Architect в файл приложения EKB.
  2. Пользователь запускает приложение посредством исполняемого EXE-файла.
  3. Пользователь нажимает на кнопку «Выбрать место для сохранения файла».
  4. Приложения открывает окно интерфейса-меню выбора директории.
  5. Пользователь указывает путь до необходимой директории с помощью меню.
  6. Система записывает полный пуль до конечной директории в заранее определенную переменную.
  7. Пользователь получает текстовое уведомление об успешном выборе места сохранения файла.

**Альтернативные потоки событий:**

1. Активация сценария «Загрузить файл Enterprise Architect».
2. Активация сценария «Закрыть приложение».

**Специальные требования**: нет.

**Предусловия**: выполнен сценарий «Открыть приложение».

**Постусловия**: после активации варианта должен быть активирован один или несколько сценариев вариантов использования: «Загрузить файл Enterprise Architect», «Запустить процесс конвертации файла» или «Закрыть приложение».

**Дополнительные замечания**: вариант использования «Запустить процесс конвертации файла» не имеет смысла выполнять без завершенных сценариев «Загрузить файл Enterprise Architect» и «Указать путь места для сохранения файла».

**Спецификация варианта использования «Запустить процесс конвертации файла»**

**Цель**: конвертация файла.

**Активные субъекты**: пользователь.

**Краткое описание**: пользователь запускает механизм конвертации файла.

**Основной поток событий**:

* 1. Пользователю потребовалось конвертировать файл Enterprise Architect в файл приложения EKB.
  2. Пользователь запускает приложение посредством исполняемого EXE-файла.
  3. Пользователь нажимает на кнопку «Загрузить файл Enterprise Architect».
  4. Приложения открывает окно интерфейса-меню выбора файла.
  5. Пользователь указывает путь до необходимого файла с помощью меню загрузки файла.
  6. Система записывает полный пуль до файла в заранее определенную переменную.
  7. Пользователь получает текстовое уведомление об успешной загрузке файла.
  8. Пользователь нажимает на кнопку «Выбрать место для сохранения файла».
  9. Приложения открывает окно интерфейса-меню выбора директории.
  10. Пользователь указывает путь до необходимой директории с помощью меню.
  11. Система записывает полный пуль до конечной директории в заранее определенную переменную.
  12. Пользователь получает текстовое уведомление об успешном выборе места сохранения файла.
  13. Пользователь запускает конвертацию файла нажатием на кнопку «Конвертировать файл».
  14. Файл загружается в приложение из указанного пути.
  15. Создается объект класса Enterprise Architect с полями файла.
  16. Данные из файла заносятся в соответствующие переменные объекта класса.
  17. Создается объект класса EKB.
  18. Поля класса EKB и Enterprise Architect соотносятся и перезаписываются
  19. Конечный объект EKB дополняется необходимыми конструкциями для работоспособности файла.
  20. Объект EKB с дополненными конструкциями сохраняется в формате .ekb в заданной пользователем директории.
  21. Пользователь получает текстовое уведомление об успешной конвертации файла.

**Альтернативные потоки событий:**

1. Активация сценария «Закрыть приложение».

**Специальные требования**: нет.

**Предусловия**:

1. Выполнен сценарий «Открыть приложение».
2. Выполнен сценарий «Загрузить файл Enterprise Architect».
3. Выполнен сценарий «Указать путь места для сохранения файла».

**Постусловия**: после активации варианта должен быть активирован сценарий вариантов использования: «Закрыть приложение».

**Дополнительные замечания**: вариант использования «Запустить процесс конвертации файла» не имеет смысла выполнять без завершенных сценариев «Загрузить файл Enterprise Architect» и «Указать путь места для сохранения файла».

**Спецификация варианта использования «Закрыть приложение»**

**Цель**: закрыть программный компонент для конвертации файлов.

**Активные субъекты**: пользователь.

**Краткое описание**: пользователь открывает приложение для конвертации файлов.

**Основной поток событий**:

1. Пользователю потребовалось конвертировать файл Enterprise Architect в файл приложения EKB.
2. Пользователь запускает приложение посредством исполняемого EXE-файла.
3. Пользователь нажимает на кнопку «Загрузить файл Enterprise Architect».
4. Приложения открывает окно интерфейса-меню выбора файла.
5. Пользователь указывает путь до необходимого файла с помощью меню загрузки файла.
6. Система записывает полный пуль до файла в заранее определенную переменную.
7. Пользователь получает текстовое уведомление об успешной загрузке файла.
8. Пользователь нажимает на кнопку «Выбрать место для сохранения файла».
9. Приложения открывает окно интерфейса-меню выбора директории.
10. Пользователь указывает путь до необходимой директории с помощью меню.
11. Система записывает полный пуль до конечной директории в заранее определенную переменную.
12. Пользователь получает текстовое уведомление об успешном выборе места сохранения файла.
13. Пользователь запускает конвертацию файла нажатием на кнопку «Конвертировать файл».
14. Файл загружается в приложение из указанного пути.
15. Создается объект класса Enterprise Architect с полями файла.
16. Данные из файла заносятся в соответствующие переменные объекта класса.
17. Создается объект класса EKB.
18. Поля класса EKB и Enterprise Architect соотносятся и перезаписываются
19. Конечный объект EKB дополняется необходимыми конструкциями для работоспособности файла.
20. Объект EKB с дополненными конструкциями сохраняется в формате .ekb в заданной пользователем директории.
21. Пользователь получает текстовое уведомление об успешной конвертации файла.
22. Пользователь закрывает приложение.
23. Операционная система завершает процесс приложения.

**Альтернативные потоки событий:** нет.

**Специальные требования**: нет.

**Предусловия**:

1. Выполнен сценарий «Открыть приложение».
2. Выполнен сценарий «Загрузить файл Enterprise Architect».
3. Выполнен сценарий «Указать путь места для сохранения файла».
4. Выполнен сценарий «Запустить процесс конвертации файла».

**Постусловия**: нет.

**Дополнительные замечания**: нет.

2.3.2 Диаграмма последовательностей

На основе диаграммы вариантов использования была составлена диаграмма последовательностей.

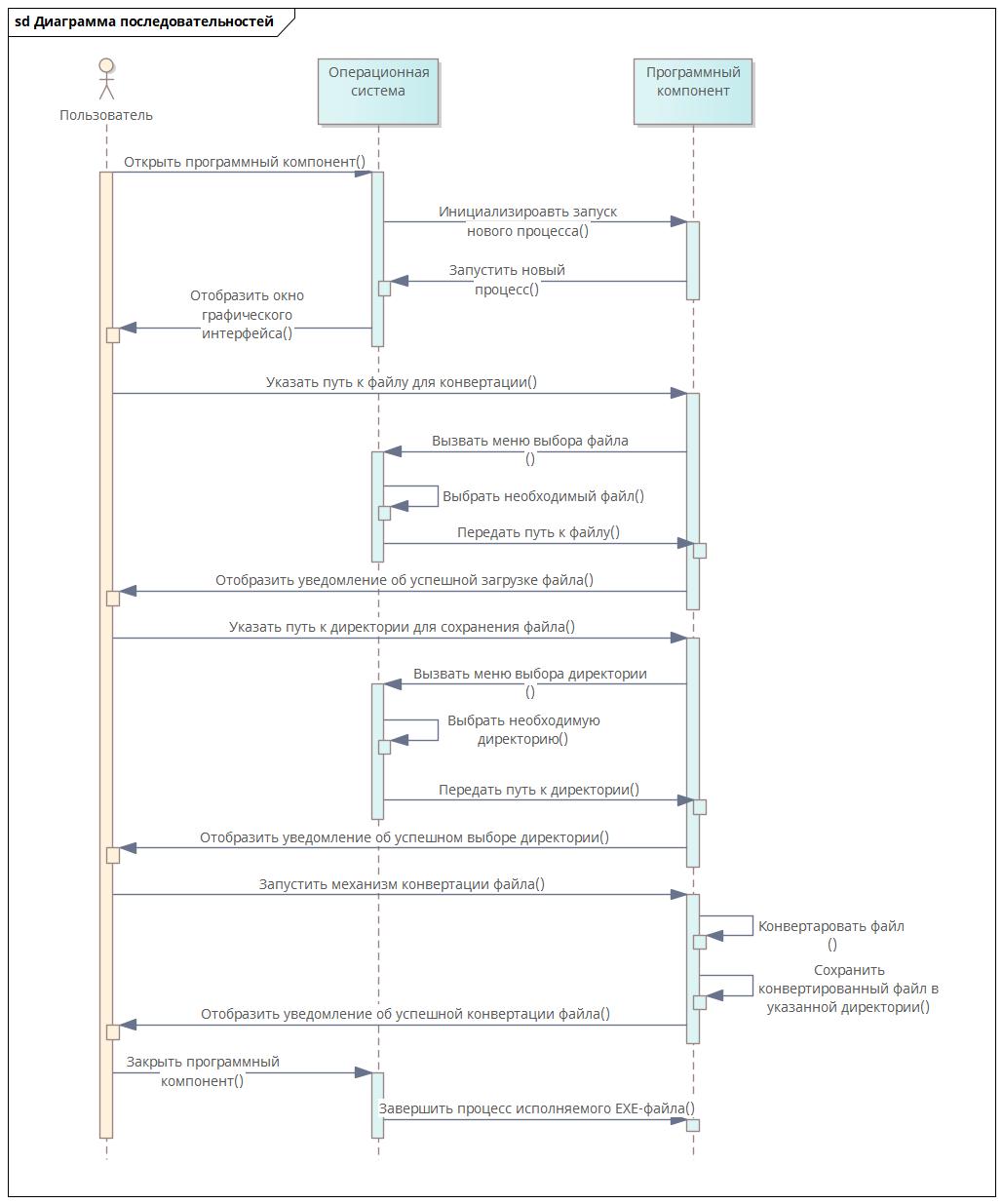


Рисунок 7 – Диаграмма последовательностей

Пользователь с помощью операционной системы запускает исполняемый EXE-файл программного компонента. Запускается окно графического интерфейса приложения. Далее пользователь с помощью меню выбора файла операционной системы задает приложению директорию для загрузки файла и директорию для сохранения результата конвертации. Следующим шагом участник системы запускает механизм конвертации файла посредством нажатия на соответствующий объект графического интерфейса (кнопку). В результате в директории создается файл с соответствующим расширением. После пользователь закрывает приложение.

2.3.3 Алгоритмическое обеспечение

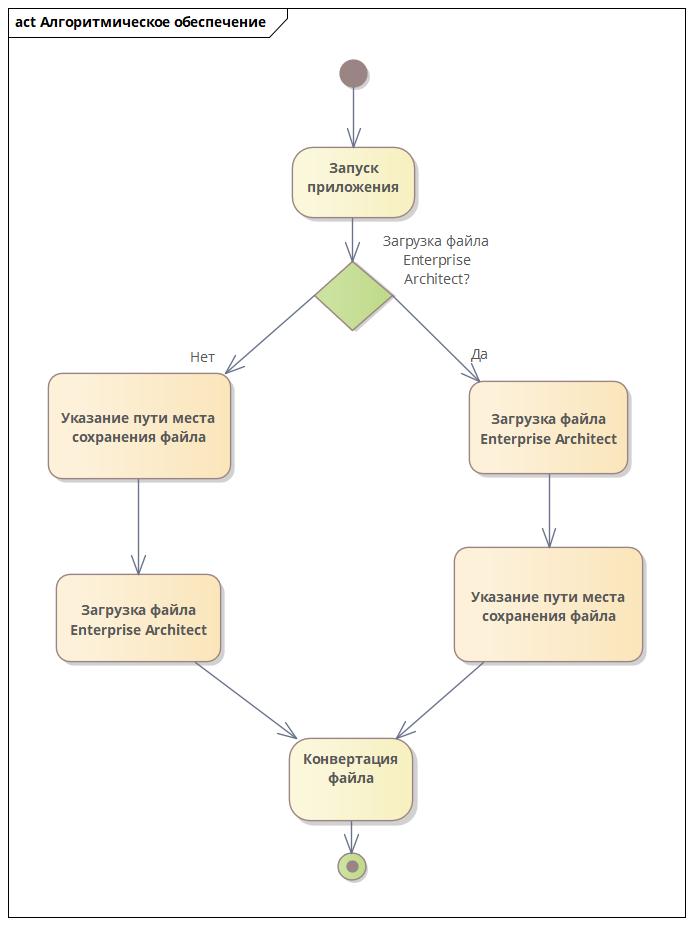


Рисунок 8 – Диаграмма деятельности

Первым этапом после инициализации деятельности выступает блок «Запуск приложения». На данном этапе пользователь запускает программный компонент. После участнику системы представляется выбор очередности действия. В случае выполнения первого вариант пользователь укажет путь к директории для сохранения конвертированного файла, далее загрузит файл для конвертации. В противном случае, пользователь сначала загрузит файл для работы, а после укажет путь к месту сохранения. Ветвления сходятся в блоке «Конвертация файла», в котором осуществляется указанный в названии механизм. Данный блок является финальным.

2.3.4 Диаграмма классов

На основе разработанной проектной документации была создана диаграмма классов программного компонента.

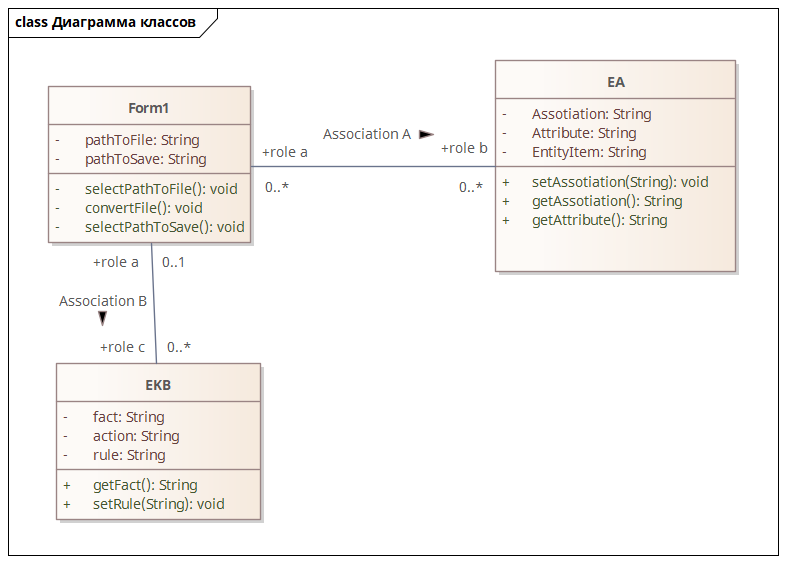


Рисунок 9 – Диаграмма классов

Диаграмма представляет из себя три класса, необходимых для полноценного функционирования приложения с целью выполнения поставленных задач. Класс Form1 представляет из себя контроллер графического приложения. Основными переменными являются: путь до директории с файлом и директория для сохранения. Соответствующие методы предназначены для получения необходимых местоположений. Дополнительно спроектирован метод, реализующий механизм конвертации файла.

Классы EA и EKB предназначены для хранения данных из файлов. При запуске процесса конвертации файл Enterprise Architect будет декомпозирован в отдельные переменные объекта класса. Далее полученные значения будут пересобраны в экземпляре EKB и дополнены необходимыми для структурной целостности файла данными. Спроектированные блоки условных data-классов представлены в виде демонстрационных моделей: отражают неполные наборы переменных и методов getter – setter, так как представленного наполнения достаточно для генерации первичного скелетного кода для последующего расширения.

* 1. Программная реализация

2.4.1 Автоматически сгенерированный код

На основе разработанной диаграммы классов был сгенерирован код. В качестве выбранного языка программирования был установлен C#.

**Код класса Form1**

///////////////////////////////////////////////////////////

// Form1.cs

// Implementation of the Class Form1

// Generated by Enterprise Architect

// Created on: 08-дек-2022 0:16:07

// Original author: worker

///////////////////////////////////////////////////////////

using System;

using System.Collections.Generic;

using System.Text;

using System.IO;

public class Form1 {

private String pathToFile;

private String pathToSave;

public EKB role c;

public EA role b;

public Form1(){

}

~Form1(){

}

///

/// <param name="Parameter A"></param>

private void selectPathToFile(Parameter A){

}

private void convertFile(){

}

///

/// <param name="Parameter"></param>

private void selectPathToSave(Parameter){

}

}//end Form1

**Код класса EA**

///////////////////////////////////////////////////////////

// EA.cs

// Implementation of the Class EA

// Generated by Enterprise Architect

// Created on: 08-дек-2022 0:15:27

// Original author: worker

///////////////////////////////////////////////////////////

using System;

using System.Collections.Generic;

using System.Text;

using System.IO;

public class EA {

private String Assotiation;

private String Attribute;

private String EntityItem;

public EA(){

}

~EA(){

}

///

/// <param name="Parameter A"></param>

public void setAssotiation(String Parameter A){

}

///

/// <param name="Parameter"></param>

public String getAssotiation(Parameter){

return "";

}

public String getAttribute(){

return "";

}

}//end EA

**Код класса EKB**

///////////////////////////////////////////////////////////

// EKB.cs

// Implementation of the Class EKB

// Generated by Enterprise Architect

// Created on: 08-дек-2022 0:15:27

// Original author: worker

///////////////////////////////////////////////////////////

using System;

using System.Collections.Generic;

using System.Text;

using System.IO;

public class EKB {

private String fact;

private String action;

private String rule;

public EKB(){

}

~EKB(){

}

///

/// <param name="Parameter A"></param>

public String getFact(Parameter A){

return "";

}

///

/// <param name="Parameter"></param>

public void setRule(String Parameter){

}

}//end EKB

2.4.2 Описание интерфейса

В качестве среды для создания макетов графического интерфейса был использован онлайн-редактор для создания интерфейсов и прототипов Figma. Основываясь на диаграмме вариантов использования, было спроектировано шесть прототипов возможных состояний программного компонента.

Первый фрагмент отображает первоначальное состояние приложения после запуска. Пользователю доступны возможности загрузки файла и выбора директории для сохранения результата.

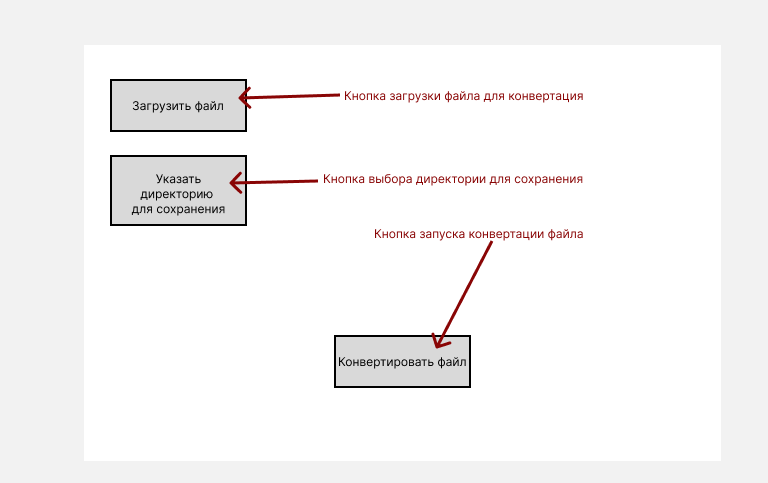


Рисунок 10 – Макет стартового окна

При нажатии на кнопки загрузки или выбора директории открывается окно соответствующего выбора.

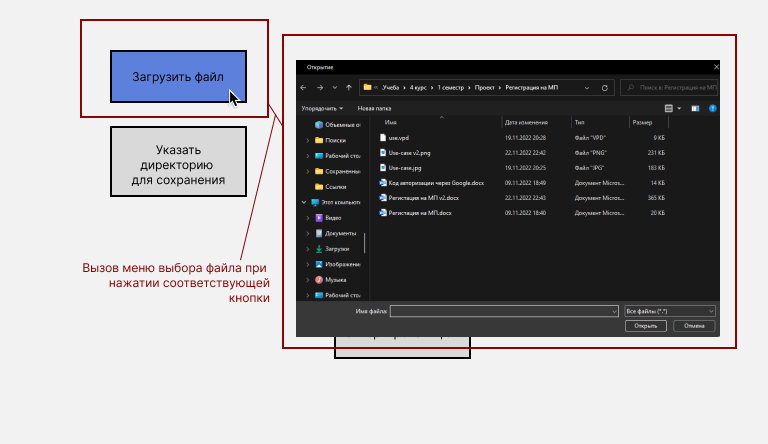


Рисунок 11 – Взаимодействие с кнопками выбора

Текущие состояния работы с файлами и директориями указывают информационные поля. В качестве упрощения визуального восприятия типа состояния текст окрашен соответствующим цветом.

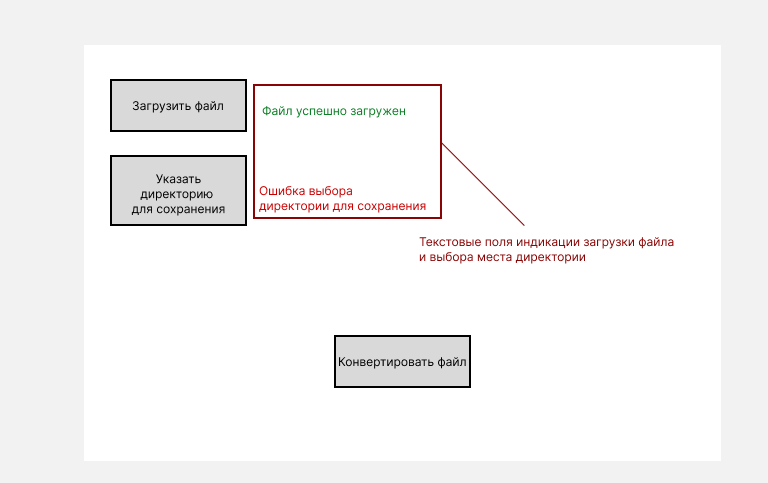


Рисунок 12 – Отображение полей состояния

После запуска пользователем механизма конвертации прогресс процесса отображается визуально с помощью Progress Bar.

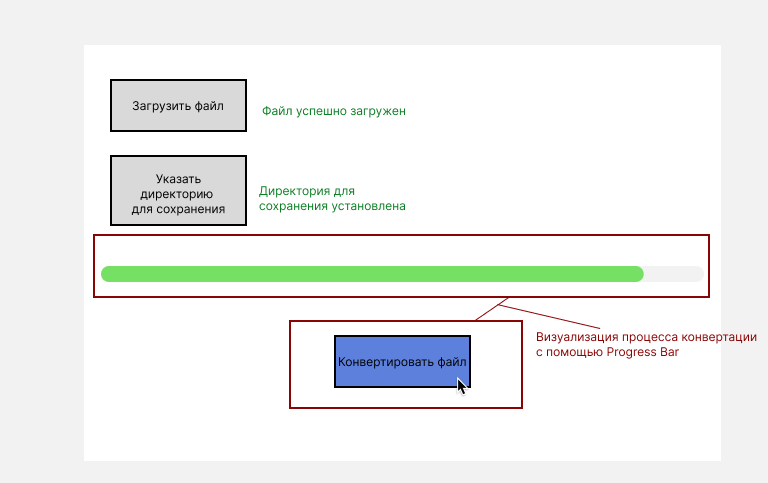


Рисунок 13 – Визуализация процесса конвертации

После завершения данного процесса отображается окно с сообщением об успешной конвертацией или ошибкой

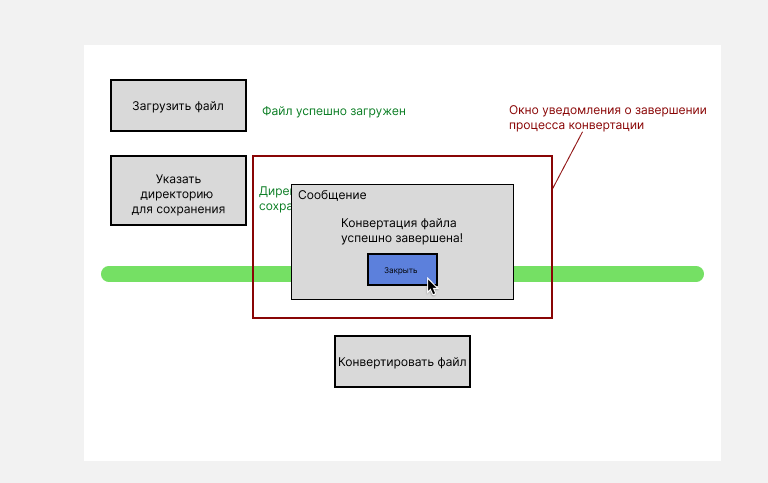


Рисунок 14 – Визуализация завершения процесса (успех)

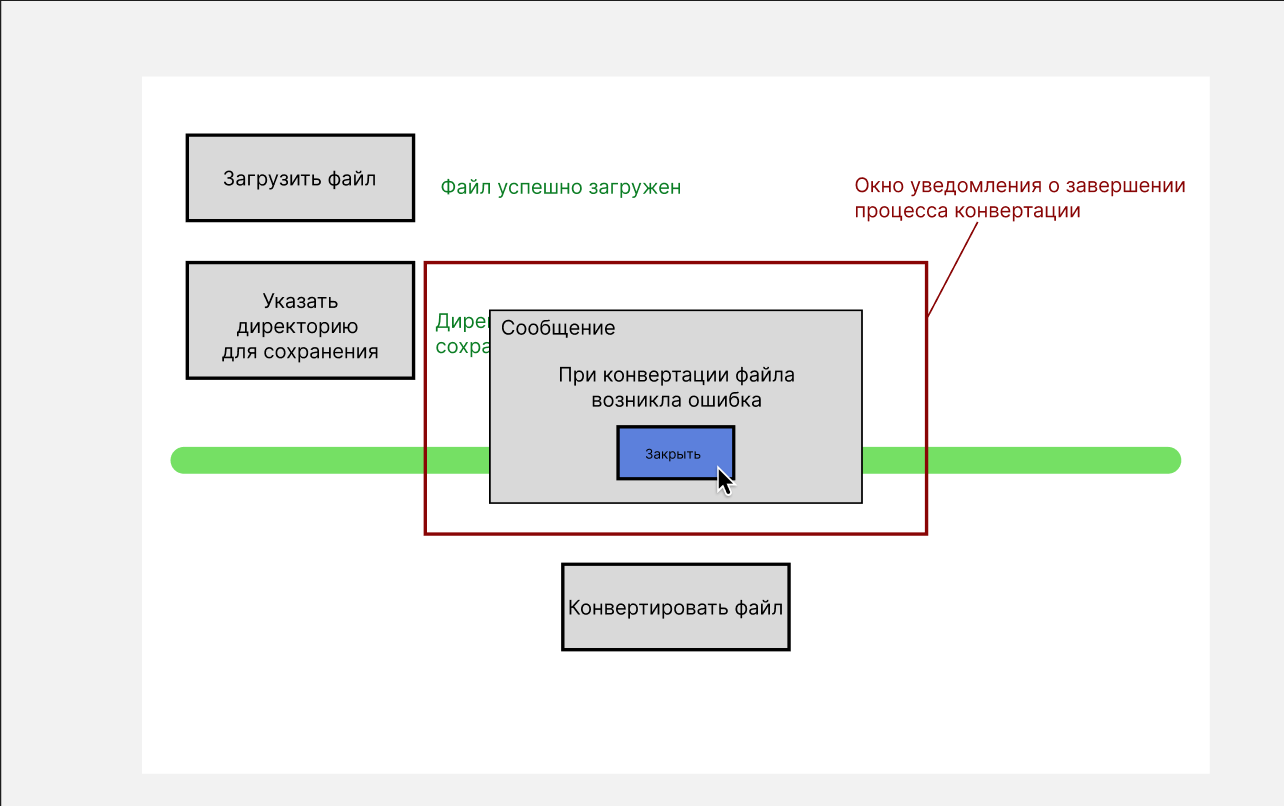


Рисунок 15 – Визуализация завершения процесса (ошибка)

Конечный вариант графического интерфейса создан с помощью элементов окна Windows Forms.

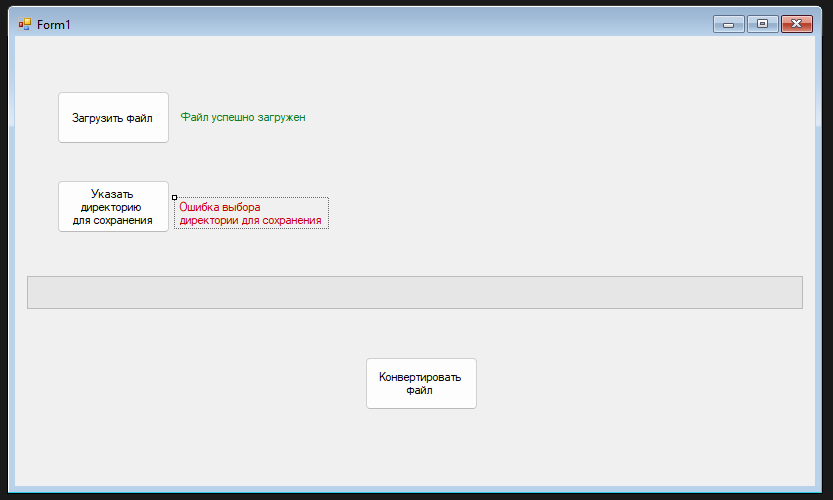


Рисунок 16 – Графический интерфейс в среде разработки

2.4.3 Тестирование

Для тестирования разработанного программного компонента была использована компромиссный стратегия. Компромиссная стратегия – проектирование тестов, исходя из принципов:

1. проверка функции или возможности;
2. проверка каждой области и границы изменения значений какой-либо входной величины;
3. проверка каждого особого случая или исключительной ситуации;
4. каждая команда программы должна проработать хотя бы на одном тесте.

Таблица 2 – Тесты программного компонента

|  |  |  |  |
| --- | --- | --- | --- |
| N | Описание проверяемой ситуации | Тест (входные данные) | Результат |
|  | Проверка работы компонента со стандартным файлом xml. | Полученный из Enterprise Architect файл xml. | Конвертированный файл с расширением ekb. |
|  | Проверка работы компонента с файлом xml нестандартной конфигурации. | Файл xml нестандартной конфигурации. | Окно с уведомлением об ошибке конвертации. |
|  | Попытка конвертировать файл без указания директории для сохранения. | Указан путь к файлу xml. | Кнопка конвертации сохранила неактивное состояние. |
|  | Попытка конвертировать файл без указания файла для конвертации. | Указана директория для сохранения файла. | Кнопка конвертации сохранила неактивное состояние. |
|  | Проверка работы компонента с файлом нестандартного для приложения компонента. | Файл с расширением ekb. | Окно с уведомлением об ошибке конвертации. |
|  | Проверка работы компонента с пустым файлом xml. | Пустой файл xml. | Окно с уведомлением об ошибке конвертации. |
|  | Повторная конвертация загруженного файла без изменения директории. | Полученный из Enterprise Architect файл xml. | Перезаписанный файл с расширением ekb. |

Результаты тестирования представлены на рисунках 17-30.

**Тест №1 – Проверка работы компонента со стандартным файлом xml.**

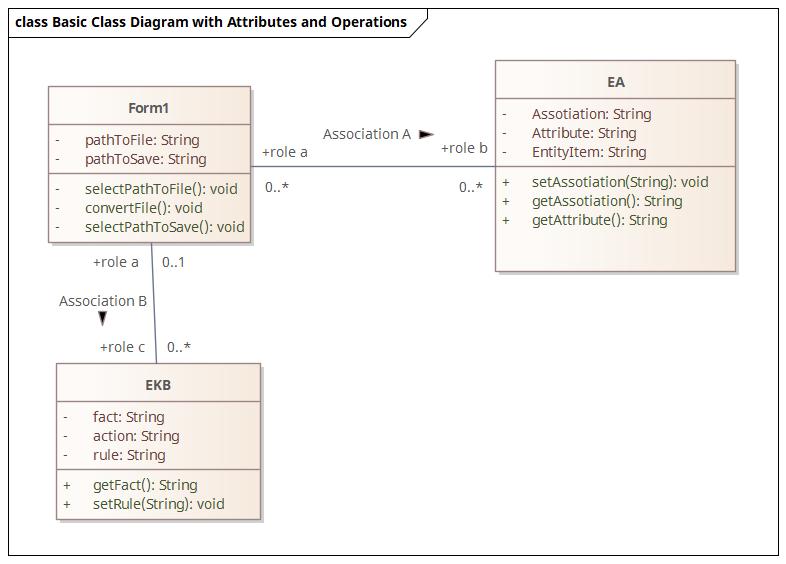


Рисунок 17 – Исходная диаграмма классов

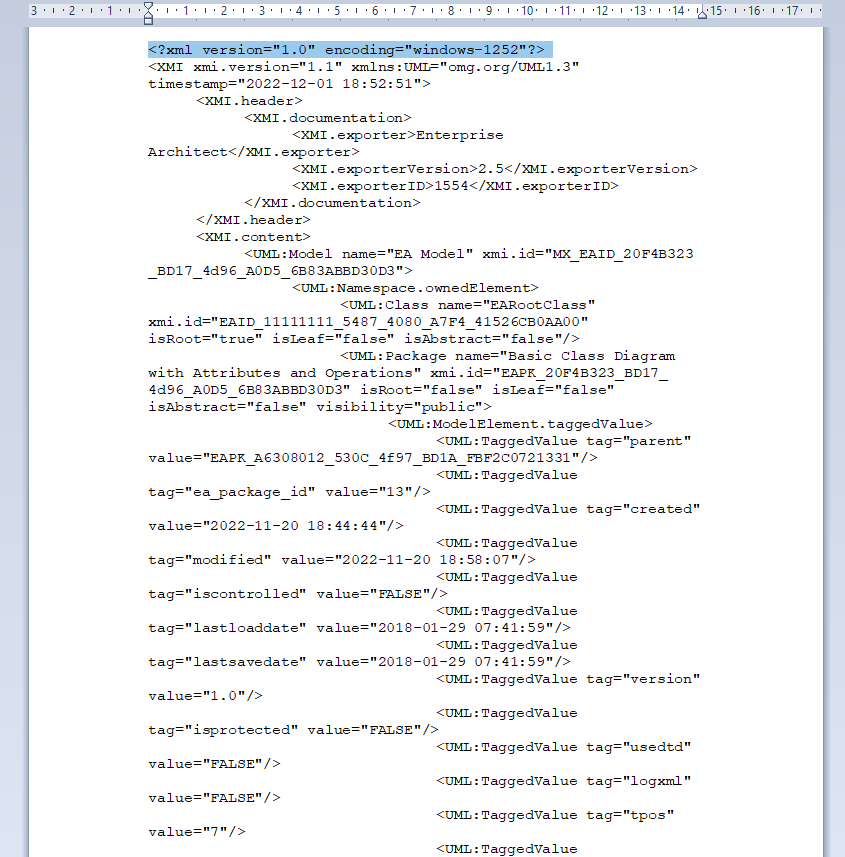


Рисунок 18 – Файл для конвертации

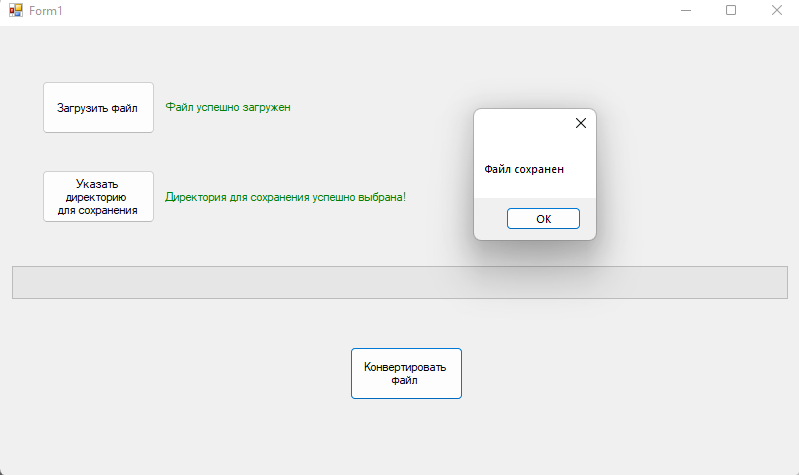


Рисунок 19 – Уведомление об успешной конвертации

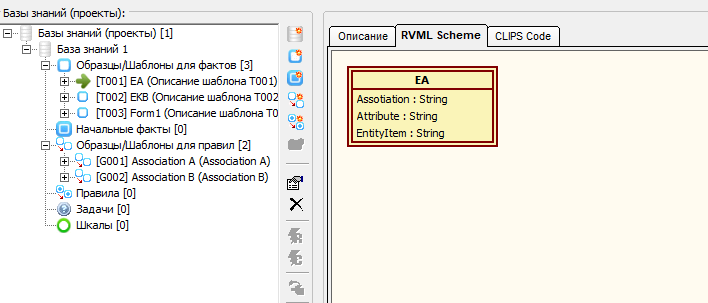


Рисунок 20 – Результат импортирования

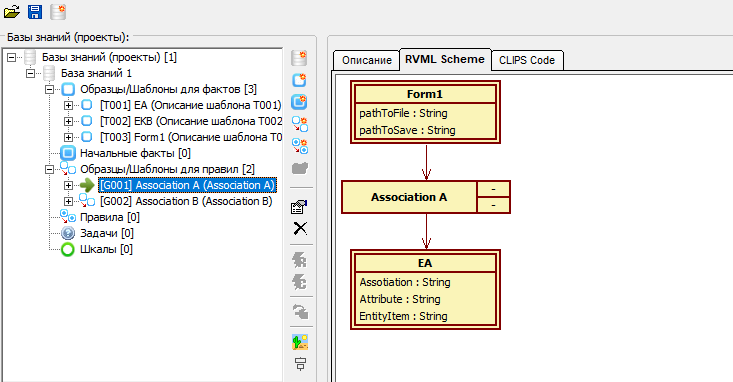


Рисунок 21 – Результат импортирования

**Тест №2 - Проверка работы компонента с файлом xml нестандартной конфигурации.**

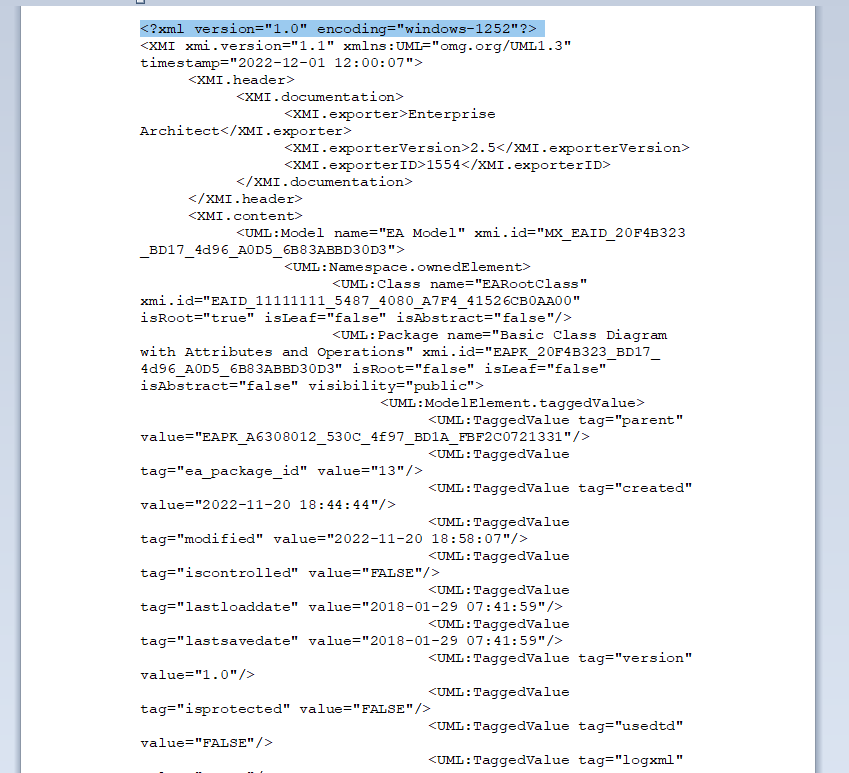


Рисунок 22 – Файл нестандартной конфигурации

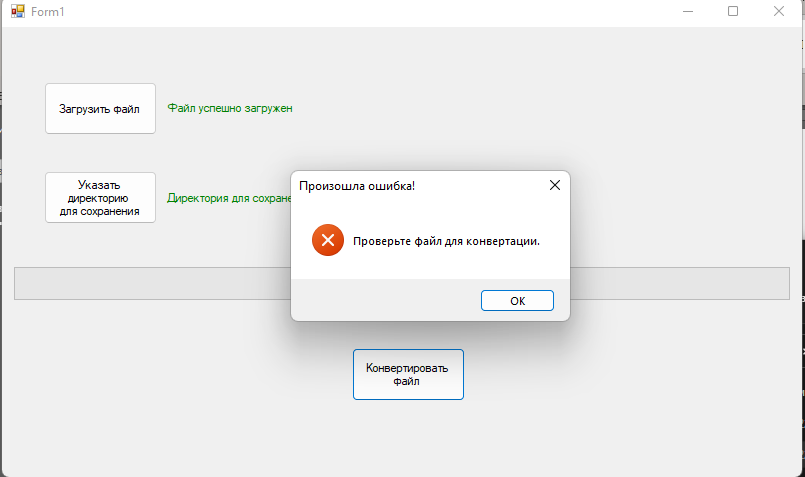


Рисунок 23 – Результат работы приложения

**Тест №3 – Попытка конвертировать файл без указания директории для сохранения.**

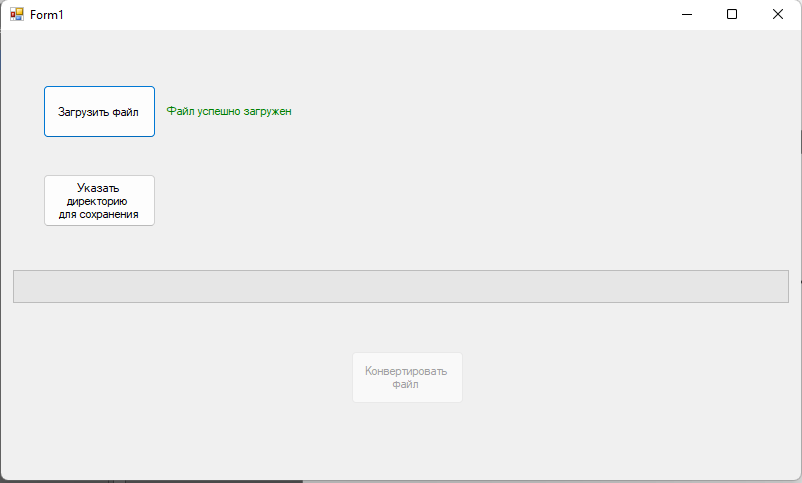


Рисунок 24 – Результат выполнения теста

**Тест 4 – Попытка конвертировать файл без указания файла для конвертации.**

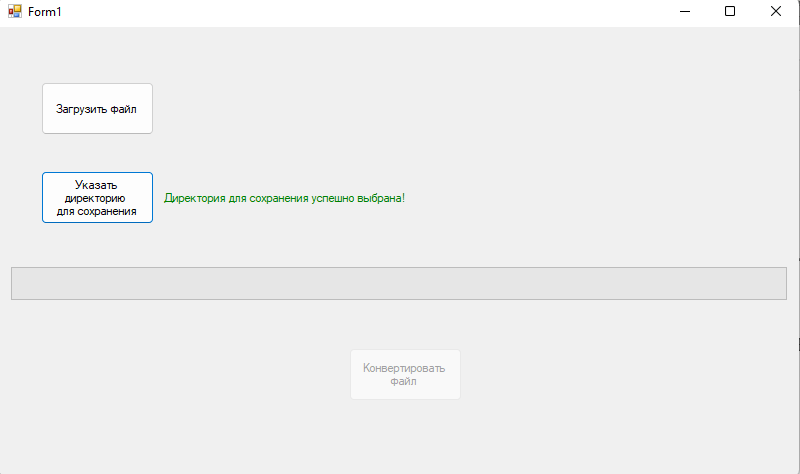


Рисунок 25 – Результат выполнения теста

**Тест №5 – Проверка работы компонента с файлом нестандартного для приложения компонента.**

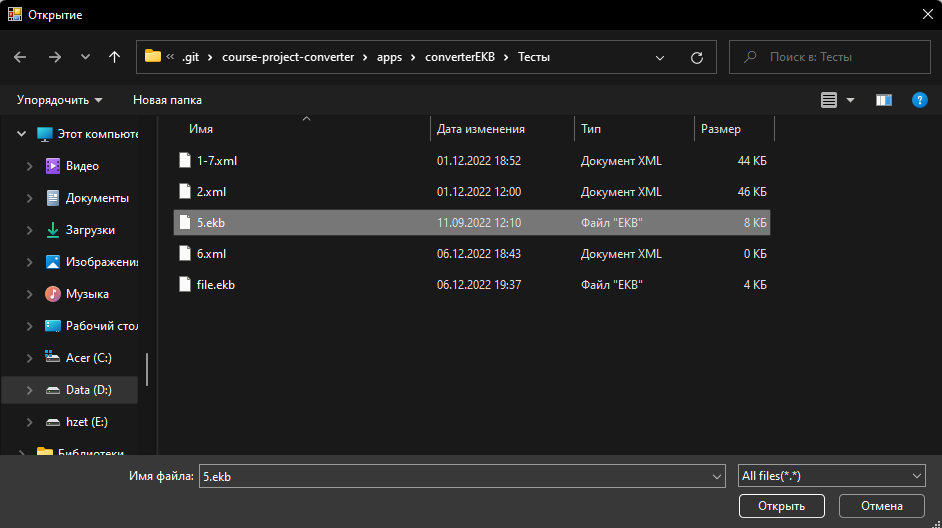


Рисунок 26 – Выбор нестандартного файла

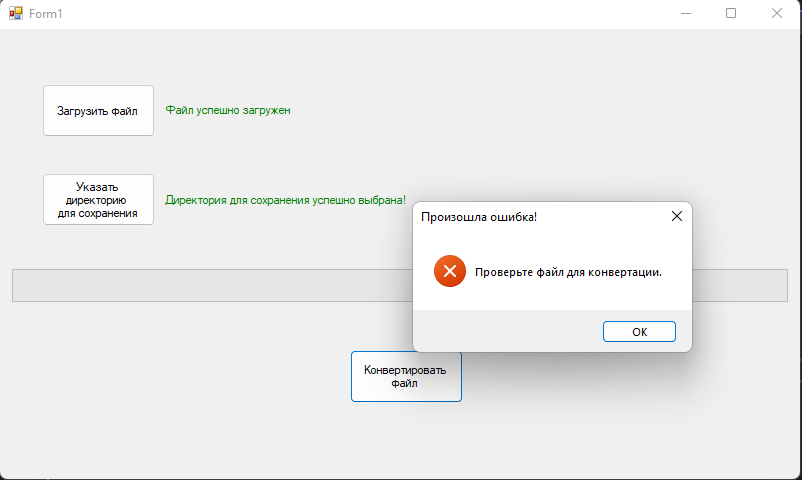


Рисунок 27 – Результат выполнения теста

**Тест №6 – Проверка работы компонента с пустым файлом xml.**

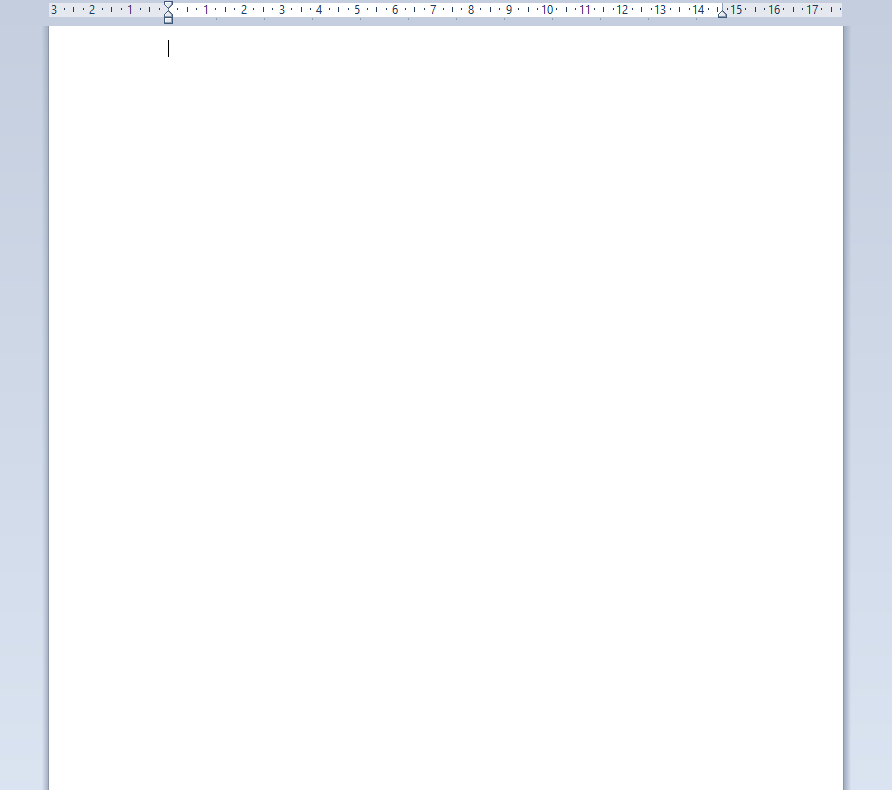


Рисунок 28 – Пустой файл

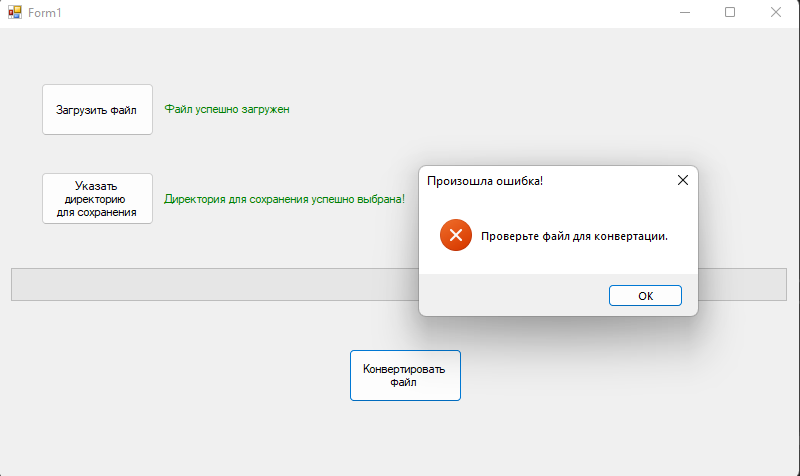


Рисунок 29 – Результат выполнения теста

**Тест №7 – Повторная конвертация загруженного файла без изменения директории.**

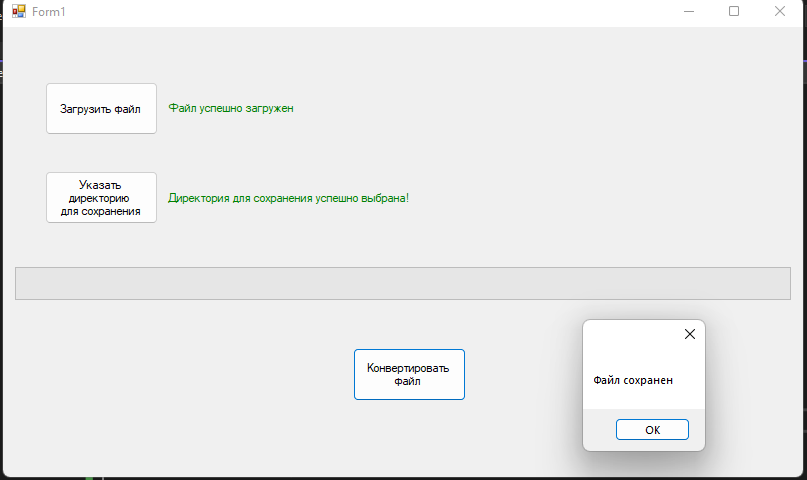


Рисунок 30 – Результат выполнения теста

Заключение

В ходе выполнения проекта было разработано программное обеспечение, обеспечивающее автоматизированное преобразование файлов Enterprise Architect в формат редактора баз знаний EKB.

Составлено техническое задание для выполнения задания в рамках курсового проекта, отвечающее стандартам и требованиям ГОСТ, описан состав и содержание работы, составлены Перт диаграмма и диаграмма Ганта.

Спроектирована доска в специализированном программном обеспечении Trello для отслеживания процесса выполнения задания в рамках курсового проекта.

С помощью CASE-средства Enterprise Architect спроектированы:

1. Диаграммы вариантов использования в нотации UML.
2. Описаны основные варианты использования разработанного программного компонента.
3. Разработана системная диаграмма последовательности действий.
4. Разработана диаграмма классов уровня проектирования в нотации UML.
5. Произведена генерация скелетного кода на основе диаграмм классов для целевого языка программирования.

Описан графический интерфейс и разработана программная реализация приложения. Разработанное приложение успешно прошло составленные тесты.

Список использованных источников

1. Гутгарц Р.Д. Проектирование автоматизированных систем обработки информации и управления. Методические указания по выполнению курсового проекта. Иркутск, ИрГТУ, 2010. 57 с.
2. Коваленко В.В. Проектирование информационных систем: учеб. пособие. Иркутск: Форум, НИЦ ИНФРА. М, 2014.  320 с.
3. Исаев Г.Н. Проектирование информационных систем: Учебное пособие/ Москва: Изд-во Омега Л, 2015. 424 с.
4. Кватрани Т. RationalRose 2000 и UML. Визуальное моделирование. М.: ДМК Пресс, 2001. 176 с.
5. Дородных Н.О., Юрин А.Ю. Технология создания продукционных экспертных систем на основе модельных трансформаций. Новосибирск: СО РАН, 2019. 144 стр.
6. Юрин А.Ю., Грищенко М.А. Редактор баз знаний в формате CLIPS // Программные продукты и системы. 2012. № 4. С. 83–87.
7. Юрин А.Ю. CASE-средства: Методические указания по выполнению лабораторных работ. Иркутск: ИРНИТУ, 2018. 87 c.

Приложение А

Фрагмент сгенерированной документации

**ActivityFinal**

ActivityFinal in package 'Алгоритмическое обеспечение'

| INCOMING BEHAVIORAL RELATIONSHIPS |
| --- |
| ControlFlow from Конвертация файла to ActivityFinal |

**ActivityInitial**

ActivityInitial in package 'Алгоритмическое обеспечение'

| OUTGOING BEHAVIORAL RELATIONSHIPS |
| --- |
| ControlFlow from ActivityInitial to Запуск приложения |

**Запуск приложения**

Activity in package 'Алгоритмическое обеспечение'

| OUTGOING BEHAVIORAL RELATIONSHIPS |
| --- |
| ControlFlow from Запуск приложения to Action1 |
| ControlFlow from Запуск приложения to Открытие графического интерфейса приложения |
| ControlFlow from Запуск приложения to Загрузка файла Enterprise Architect? |

| INCOMING BEHAVIORAL RELATIONSHIPS |
| --- |
| ControlFlow from ActivityInitial to Запуск приложения |

**Конвертация файла**

Activity in package 'Алгоритмическое обеспечение'

Указание пути места сохранения файла

Version 1.0 Phase 1.0 Proposed

worker created on 20.11.2022. Last modified 20.11.2022

Приложение Б

Листинг программы

**Класс Form1**

public partial class Form1 : Form

{

string filename; //Путь к файлу

string pathToDid; //Путь к директории

//Листы для объектов классов и ассоциаций

List<EA> eAs;

List<Association> associations;

string ekbText = "";

bool status = false;

public Form1()

{

InitializeComponent();

openFileDialog1.Filter = "Файл xml (\*.xml)|\*.xml|All files(\*.\*)|\*.\*"; //Инициализация фильтра загрузки файлов

convertBtn.Enabled = false;

eAs = new List<EA>();

associations = new List<Association>();

}

private void selectFileBtn\_Click(object sender, EventArgs e) //Слушатель кнопки выбора файлов

{

if (openFileDialog1.ShowDialog() == DialogResult.Cancel)

{

fileLabel.Text = "Ошибка загрузки файла!";

fileLabel.ForeColor = Color.Red;

filename = null;

}

else

{

// получаем выбранный файл

filename = openFileDialog1.FileName;

fileLabel.Text = "Файл успешно загружен";

fileLabel.ForeColor = Color.Green;

}

checkToReady();

}

private void selectPathBtn\_Click(object sender, EventArgs e) //Слушатель кнопки выбора директории

{

FolderBrowserDialog FBD = new FolderBrowserDialog();

FBD.ShowNewFolderButton = false;

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

{

pathLabel.Text = "Директория для сохранения успешно выбрана!";

pathLabel.ForeColor = Color.Green;

pathToDid = FBD.SelectedPath;

}

else

{

pathLabel.Text = "Ошибка выбора \r\nдиректории для сохранения";

pathToDid = null;

pathLabel.ForeColor = Color.Red;

}

checkToReady();

}

private void checkToReady() //Метод проверки выбора файла и директории

{

if (filename != null && pathToDid != null)

{

convertBtn.Enabled = true;

}

else

{

convertBtn.Enabled = false;

}

}

private void convertBtn\_Click(object sender, EventArgs e) //Слушатель кнопки конвертации

{

convertProgress.Value = 0;

parseTheFile();

convertToEKB();

if (status)

{

saveToEkb();

convertProgress.Value = 100;

}

else

{

convertProgress.Value = 0;

MessageBox.Show("Проверьте файл для конвертации.", "Произошла ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void saveToEkb() //Метод сохранения преобразованного файла

{

string path = $"{pathToDid}/file.ekb";

System.IO.File.WriteAllText(path, ekbText);

MessageBox.Show("Файл сохранен");

}

private void convertToEKB() //Метод сборки полученных данных в формат структуры ekb

{

Random random = new Random();

string id = $"{random.Next(1000000000)}{random.Next(100)}";

//Сборка шапки файла

string header = $"\r\n<Structure>"

+ $"\r\n<KnowledgeBase>"

+ $"\r\n<ID>{id}</ID>"

+ $"\r\n<Name>База знаний 1</Name>"

+ $"\r\n<ShortName>Baza-znaniy-1</ShortName>"

+ $"\r\n<Kind>0</Kind>"

+ $"\r\n<Description></Description>"

+ $"\r\n<Vars/>"

+ $"\r\n<Templates>";

string templates = "";

string slots = "";

string templateEnd = "";

string templatesEnd = "";

string grules = "";

int l = 0;

ekbText = header;

int count = 1;

while (l < eAs.Count) //Сборка классов

{

string idTempStr = "";

if (count < 10)

{

idTempStr = "";

idTempStr = $"00{count}";

}

if (count >= 10)

{

idTempStr = "";

idTempStr = $"0{count}";

}

templates = $"\r\n<Template>" +

$"\r\n<ID>T{idTempStr}</ID>" +

$"\r\n<Name>{eAs[l].ClassName}</Name>" +

$"\r\n<ShortName>{eAs[l].ClassName}</ShortName>" +

$"\r\n<Description>Описание шаблона T{idTempStr}</Description>" +

$"\r\n<PackageName></PackageName>" +

$"\r\n<RootPackageName></RootPackageName>" +

$"\r\n<DrawParams>xT{idTempStr}=15" +

$"\r\nyT{idTempStr}=15" +

$"\r\nw=265" +

$"\r\nh=65" +

$"\r\n</DrawParams>" +

$"\r\n<Slots>";

int p = 0;

ekbText += templates;

while (p < eAs[l].Attribute.Count) //Сборка полей классов

{

slots = $"\r\n<Slot>" +

$"\r\n<Name>{eAs[l].Attribute[p].Item3}</Name>" +

$"\r\n<ShortName>{eAs[l].Attribute[p].Item3}</ShortName>" +

$"\r\n<Description>{eAs[l].Attribute[p].Item3}</Description>" +

$"\r\n<Value></Value>" +

$"\r\n<DataType>{eAs[l].Attribute[p].Item2}</DataType>" +

$"\r\n<Constraint></Constraint>" +

$"\r\n</Slot>";

ekbText += slots;

p++;

}

templateEnd = $"\r\n</Slots>" +

"\r\n</Template>";

ekbText += templateEnd;

l++;

count++;

}

templatesEnd = templatesEnd = $"\r\n</Templates>";

ekbText += templatesEnd;

string facts = "\r\n<Facts/>" +

"\r\n<GRules>";

ekbText += facts;

int q = 0;

count = 1;

while (q < associations.Count) //Сборка ассоциаций файлов

{

string idTempStr = "";

if (count < 10)

{

idTempStr = "";

idTempStr = $"00{count}";

}

if (count >= 10)

{

idTempStr = "";

idTempStr = $"0{count}";

}

grules =

$"\r\n<GRule>" +

$"\r\n<ID>G{idTempStr}</ID>" +

$"\r\n<Name>{associations[q].AssotionName}</Name>" +

$"\r\n<ShortName>{associations[q].AssotionName}</ShortName>" +

$"\r\n<Description>{associations[q].AssotionName}</Description>" +

$"\r\n<PackageName></PackageName>" +

$"\r\n<RootPackageName></RootPackageName>" +

$"\r\n<DrawParams>xG{idTempStr}=26" +

$"\r\nyG{idTempStr}=105" +

$"\r\nw=170" +

$"\r\nh=34" +

$"\r\n</DrawParams>" +

$"\r\n<Conditions>" +

$"\r\n<C0>{associations[q].SourceName}</C0>" +

$"\r\n</Conditions>" +

$"\r\n<Actions>" +

$"\r\n<A0>{associations[q].TargetName}</A0>" +

$"\r\n</Actions>" +

$"\r\n</GRule>";

ekbText += grules;

q++;

count++;

}

string ekbEnd = $"\r\n</GRules>" +

$"\r\n<Rules/>" +

$"\r\n<Functions/>" +

$"\r\n<Tasks/>" +

$"\r\n<FScales/>" +

$"\r\n<TempPackageList/>" +

$"\r\n<FactPackageList/>" +

$"\r\n<RulePackageList/>" +

$"\r\n<GRulePackageList/>" +

$"\r\n</KnowledgeBase>" +

$"\r\n</Structure>";

ekbText += ekbEnd;

}

private void parseTheFile() //Метод парсинга файла

{

try

{

string attr = "";

string func = "";

string start = "";

string end = "";

string typeText = "";

List<(string, string, string)> Attribute;

filename = openFileDialog1.FileName;

XmlDocument xDoc = new XmlDocument();

try

{

xDoc.Load(filename);

}

catch(System.Xml.XmlException ex)

{

status = false;

return;

}

XmlElement xRoot = xDoc.DocumentElement;

if (xRoot != null) //Проверка на наличие корневой директории файла

{

//Разбор дерева структуры

var child = xRoot.ChildNodes;

var content = child.Item(1);

if (content == null)

{

status = false;

return;

}

var model = content.FirstChild;

var ownedElement = model.FirstChild;

var packagesTemp = ownedElement.ChildNodes;

var packages = packagesTemp.Item(1);

var ownedElementTemp = packages.ChildNodes;

var ownedElementClasses = ownedElementTemp.Item(1);

var classes = ownedElementClasses.ChildNodes;

for (int i = 0; i < classes.Count; i++)

{

Attribute = new List<(string, string, string)>();

var childClasses = classes.Item(i);

if (childClasses.Name == "UML:Class") //Получение классов

{

var inside = childClasses.ChildNodes;

var feature = inside.Item(1);

if(feature == null)

{

status = false;

return;

}

var fields = feature.ChildNodes;

for (int j = 0; j < fields.Count; j++)

{

var operation = fields[j];

if (operation.Name == "UML:Attribute") //Поиск полей

{

var typeList = operation.ChildNodes;

var typeTemp = typeList.Item(2);

var type = typeTemp.ChildNodes;

for (int k = 0; k < type.Count; k++)

{

var typeAt = type.Item(k);

if (typeAt.Attributes["tag"].Value == "type")

{

typeText = typeAt.Attributes["value"].Value;

}

}

attr += $"\r\n{operation.Attributes["visibility"].Value} {typeText} {operation.Attributes["name"].Value} ";

Attribute.Add((operation.Attributes["visibility"].Value,

typeText,

operation.Attributes["name"].Value));

}

}

var attribute = childClasses.Attributes["name"];

if (attribute != null)

{

string className = attribute.Value;

eAs.Add(new EA(className, Attribute)); //Компановка класса в лист

}

attr = "";

func = "";

}

else

{

var inside = childClasses.ChildNodes;

var feature = inside.Item(0);

var deepInside = feature.ChildNodes;

for (int k = 0; k < deepInside.Count; k++) //Поиск ассоциаций

{

var links = deepInside.Item(k);

if (links.Attributes["tag"].Value == "ea\_sourceName")

{

start = links.Attributes["value"].Value;

}

if (links.Attributes["tag"].Value == "ea\_targetName")

{

end = links.Attributes["value"].Value;

}

}

var attribute = childClasses.Attributes["name"];

string assocName = attribute.Value;

associations.Add(new Association(assocName, start, end)); //Компановка ассоциаций в лист

}

}

}

status = true;

}

catch (System.NullReferenceException ex)

{

status = false;

return;

}

}

}

**Класс EA**

internal class EA

{

public string ClassName { get; set; }

public List<( string, string, string)> Attribute { get; set; }

public EA(string className, List<( string, string, string)> attribute)

{

ClassName = className;

Attribute = attribute;

}

}

**Класс Association**

internal class Association

{

public string AssotionName { get; set; }

public string SourceName { get; set; }

public string TargetName { get; set; }

public Association(string assotionName, string sourceName, string targetName)

{

AssotionName = assotionName;

SourceName = sourceName;

TargetName = targetName;

}

}