2.3 Проектирование

В качестве инструмента для проектирования было использовано CASE-средство Enterprise Architect. Enterprise Architect представляет собой уникальный и многофункциональный инструмент, предназначенный для осуществления визуального моделирования и дизайна. Главная особенность данного программного обеспечения заключается в том, что его система основана на OMG UML – одно из самых эффективных и мощных платформ в современном мире. 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.

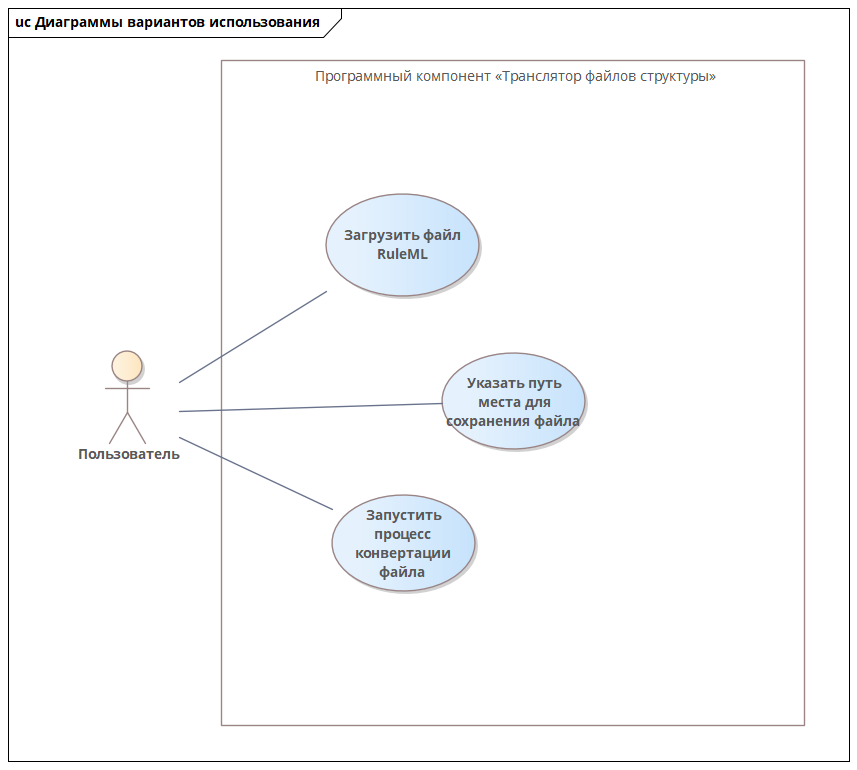


Рисунок N – диаграмма вариантов использования программного компонента

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

**Спецификация варианта использования «Загрузить файл RuleML»**

**Цель**: загрузка файла RuleML для дальнейшей конвертации

**Активные субъекты**: пользователь

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

**Основной поток событий**:

* 1. Пользователю потребовалось конвертировать файл RuleML в файл приложения EKB.
  2. Пользователь запускает приложение посредством исполняемого EXE-файла.
  3. Пользователь запускает механизм загрузки файла RuleML.
  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».

**Специальные требования**: нет.

**Предусловия**: нет.

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

**Дополнительные замечания**: вариант использования «Запустить процесс конвертации файла» не имеет смысла выполнять без завершенных сценариев «Загрузить файл RuleML» 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 RuleML в файл приложения EKB.
  2. Пользователь запускает приложение посредством исполняемого EXE-файла.
  3. Пользователь запускает механизм выбора директории для сохранения.
  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 RuleML».

**Специальные требования**: нет.

**Предусловия**: нет.

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

**Дополнительные замечания**: вариант использования «Запустить процесс конвертации файла» не имеет смысла выполнять без завершенных сценариев «Загрузить файл RuleML» 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 RuleML в файл приложения EKB.
  2. Пользователь запускает приложение посредством исполняемого EXE-файла.
  3. Пользователь запускает механизм загрузки файла RuleML.
  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 запускает механизм выбора директории для сохранения.
  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.
  14. Файл загружается в приложение из указанного пути.
  15. Создается объект класса RuleML с полями файла.
  16. Данные из файла заносятся в соответствующие переменные объекта класса.
  17. Создается объект класса EKB.
  18. Поля класса EKB и RuleML 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:** нет.

**Специальные требования**: нет.

**Предусловия**:

1. Выполнен сценарий «Загрузить файл RuleML».
2. Выполнен сценарий «Указать путь места для сохранения файла».

**Постусловия**: нет.

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

### 2.3.2 Диаграмма последовательностей

На основе диаграммы вариантов использования была составлена диаграмма последовательностей.

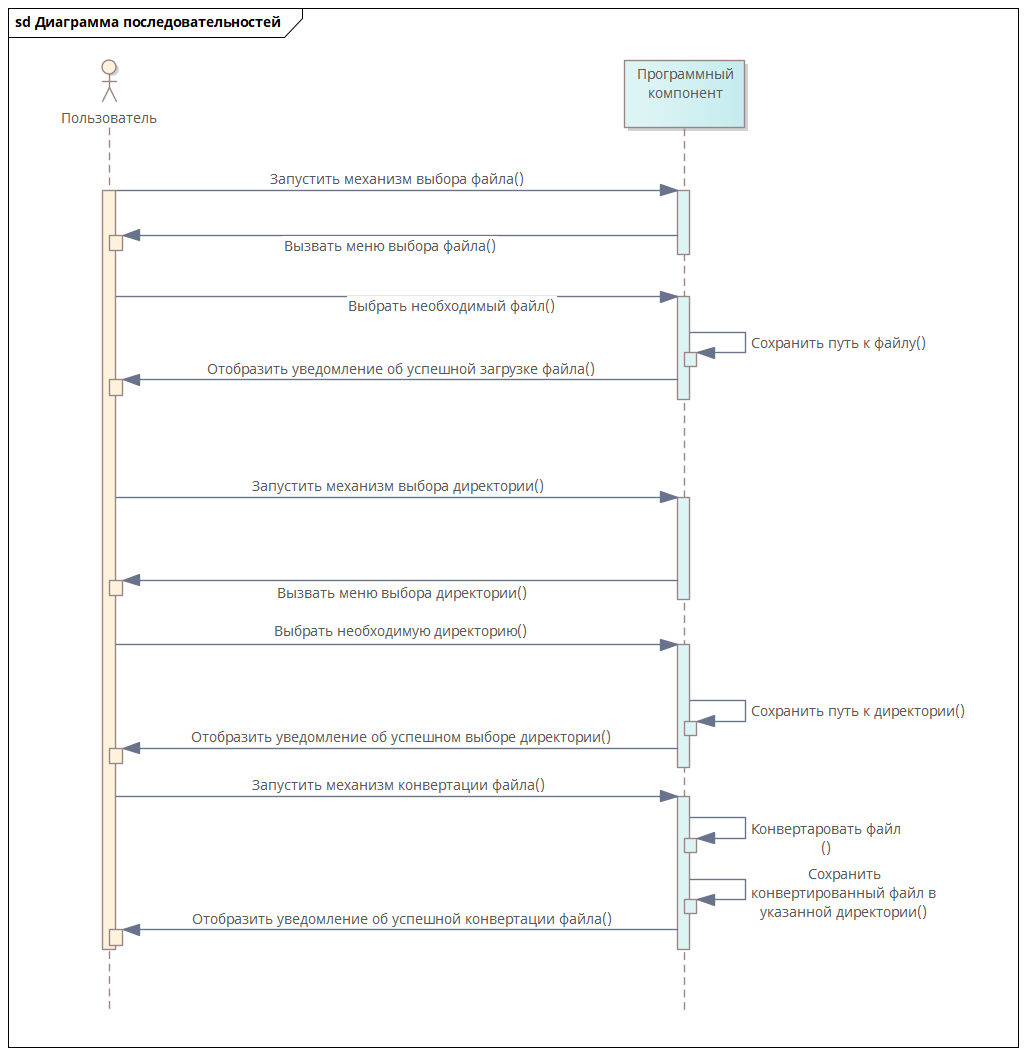


Рисунок N – диаграмма последовательностей

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.

### 2.3.3 Алгоритмическое обеспечение

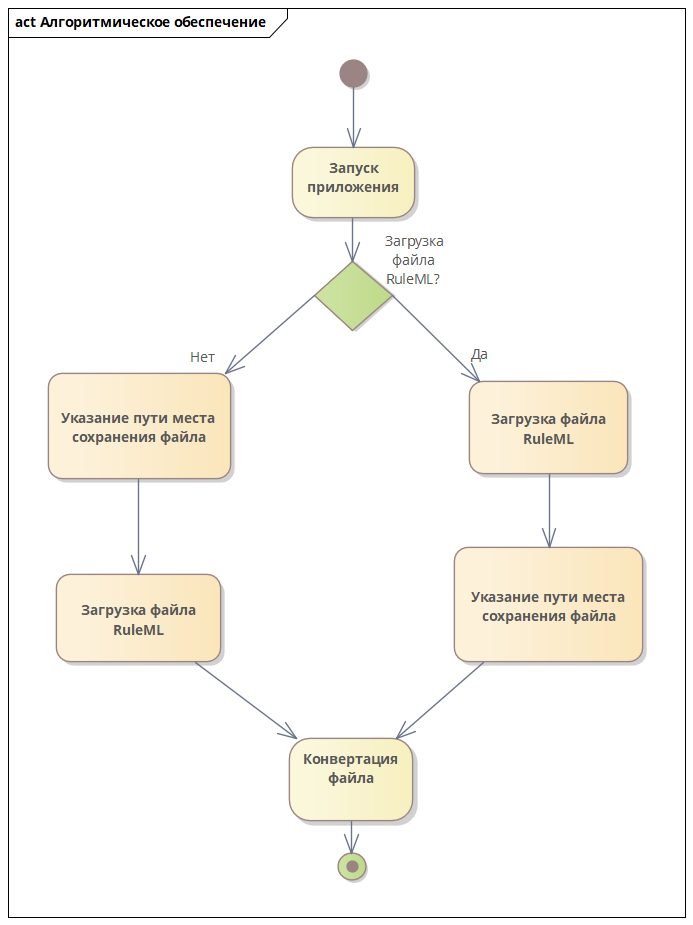


Рисунок N – диаграмма деятельности

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.

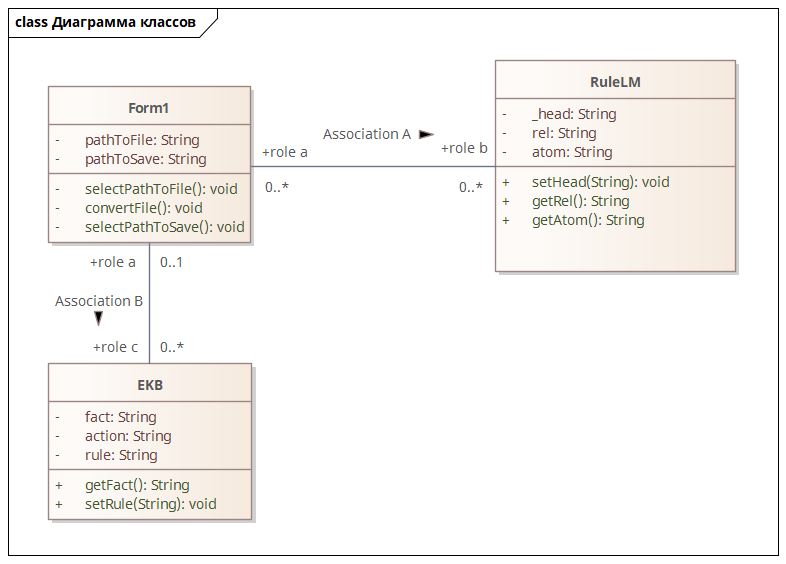


Рисунок N – диаграмма классов

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 RuleML и EKB предназначены для хранения данных из файлов. При запуске процесса конвертации файл RuleML 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.