**РОСЖЕЛДОР**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**ВЫСШЕГО ОБРАЗОВАНИЯ**

**СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**ПУТЕЙ СООБЩЕНИЯ (СГУПС)**

|  |  |  |  |
| --- | --- | --- | --- |
| **К защите:** |  |  | |
| **Заведующий кафедрой** | **Информационные** | |
| **технологии транспорта** | | |
|  | д-р техн. наук, проф. | |
|  |  | В. И. Хабаров | |
| *подпись* |  | *инициалы, фамилия* | |
|  |  |  | |
| *дата* |  |  | |

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Тема:** | Разработка приложения по автоматическому формированию рабочих | | | | | |
|  | программ дисциплин | | | | | |
|  | |  | БР.БИСТ.13.2022 |  |  |
|  | |  | *шифр документа* |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | **Выполнил** |  |  |  | **Руководитель** |
|  |  | Т. Б. Маматов |  |  |  | канд. техн. наук, доц.  А. А. Уланов |
| *подпись* |  | *инициалы, фамилия* |  | *подпись* |  | *инициалы, фамилия* |
|  |  |  |  |  |  |  |
| *дата* |  |  |  | *дата* |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Консультанты по разделам** |  |  |  |  |
| 1 |  |  |  |  |
|  |  |  |  |  |
| 2 |  |  |  |  |
|  |  |  |  |  |
| 3 |  |  |  |  |
|  |  |  |  |  |
| Нормоконтролер работы |  |  |  | ст. преп.  Т. А. Распопина |
|  |  | *подпись* |  |  |
|  |  |  |  |  |
|  |  | *дата* |  |  |

**2022 г.**

**СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**ПУТЕЙ СООБЩЕНИЯ (СГУПС)**

Факультет: Бизнес-информатики

Кафедра: Информационные технологии транспорта

Направление: 09.03.02 «Информационные системы и технологии»

Профиль: Интеллектуальные транспортные системы

|  |  |
| --- | --- |
|  | ***УТВЕРЖДАЮ****: зав. кафедрой «Информационные технологии транспорта»*  д-р техн. наук, проф.  В. И. Хабаров |
|  | *«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_ 2022 г.* |

**З А Д А Н И Е**

**на выполнение выпускной квалификационной работы**

|  |  |  |  |
| --- | --- | --- | --- |
| студенту | Маматову Тимуру Бахромовичу | | |
|  |  | | |
| 1. Тема «Разработка приложения по автоматическому формированию рабочих программ дисциплин» утверждена приказом № 203/с от «30» мая 2022 г. | | | |
| 2. Задание выдано «12» мая 2022 г. | | | |
| 3. Срок сдачи законченной работы на кафедру «17» июня 2022 г. | | | |
| 4. Исходные данные: данные, полученные в ходе прохождения преддипломной практики | | | |
| 5. Содержание расчетно-пояснительной записки | | | |
| Наименование разделов и вопросов | | Примерное количество страниц | График (сроки) выполнения |
| Введение | | 1 | 14.05.2022 |
| Аналитическое исследование | | 5 | 14.05.2022 |
| Проектирование информационной системы | | 13 | 16.05.2022 |
| Создание приложения | | 37 | 28.05.2022 |
| Заключение | | 1 | 01.06.2022 |

6. Содержание и объемы графической части

|  |  |  |
| --- | --- | --- |
| Наименование графического документа (чертежа, схемы, графика) | Количество  листов  формата А1 | График  (сроки)  выполнения |
| Презентация PowerPoint | 26 | 05.05.2022 |

7. Консультанты по разделам

|  |  |  |
| --- | --- | --- |
| Наименование  раздела | Фамилия, И. О.  консультанта | Подпись консультанта,  дата выдачи задания |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| Руководитель |  | А. А. Уланов |
|  | *(подпись, фамилия, И.О.)* |  |
| Задание к использованию принял |  | Т. Б. Маматов |
|  | *(подпись студента)* |  |

УДК 004.41

**АННОТАЦИЯ**

В работе 57 страниц, 29 рисунков, 3 таблицы, 15 источников.

Ключевые слова: *документ, рабочая программа дисциплины, данные, система, компетенция, генерация*.

Предметная область – формирование документов рабочих программ дисциплин. Приложение, на основе введённых данных пользователя (название дисциплины, файл УП, преподаватели-разработчики, файл таблицы компетенций), формирует документ РПД. Информация о преподавателях и кафедрах, на которых закреплены преподаватели, хранится в базе данных и извлекается из неё.

**ABSTRACT**

The work contains 57 pages, 29 figures, 3 tables, 15 sources.

Keywords: *a document, a discipline work program, a data, a system, a competence, a generation.*

The subject area is the formation of documents for the work programs of disciplines. The application, based on the user's entered data (discipline name, curriculum file, teachers-developers, competency table file), generates a DWP document. Information about teachers and departments where teachers are assigned is stored in the database and retrieved from it.

**ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ**

ИС – информационная система.

ОС – операционная система.

ПО – программное обеспечение.

C# – объектно-ориентированный язык программирования высокого уровня.

.NET – программная платформа компании Microsoft.

DB Browser for SQLite – система управления реляционными базами данных, разработанная специально для SQLite.

MS – компания Microsoft.

SQL (Structured Query Language) – декларативный язык программирования, применяемый для управления данными в реляционной базе данных.

Visual Studio (VS) – среда разработки от компании Microsoft.

ER (Entity Relationship) – модель «сущность-связь».

ОП – образовательная программа.

УП – учебный план.

РПД – рабочая программа дисциплины.

Приложение / программа / генератор – программа, генерирующая документы.

Office (Word, Excel) – пакет программ, предназначенный для работы с электронными документами.

**СОДЕРЖАНИЕ**

[Введение 6](#_Toc105364750)

[1 Аналитическое исследование 7](#_Toc105364751)

[1.1 Описание предметной области 7](#_Toc105364752)

[1.2 Определение и решение проблемы предметной области 11](#_Toc105364753)

[1.3 Обзор аналогичных IT-решений 11](#_Toc105364754)

[1.4 Недостатки аналогов 13](#_Toc105364755)

[1.5 Обоснование целесообразности разработки 17](#_Toc105364756)

[1.6 Обзор методологий моделирования бизнес процессов 18](#_Toc105364757)

[1.7 Выводы по первому разделу 21](#_Toc105364758)

[2 Проектирование информационной системы 22](#_Toc105364759)

[2.1 Выбор методологии моделирования и моделирование бизнес-процессов предметной области 22](#_Toc105364760)

[2.2 Выбор и обоснование типа БД информационной системы 26](#_Toc105364761)

[2.3 Язык программирования 26](#_Toc105364762)

[2.4 Среда разработки 38](#_Toc105364763)

[2.5 Система управления базами данных (СУБД) 40](#_Toc105364764)

[2.6 Модель данных (ER-модель) 41](#_Toc105364765)

[2.7 Средства и методы тестирования 42](#_Toc105364766)

[2.8 Выводы по второму разделу 42](#_Toc105364767)

[3 Создание приложения 43](#_Toc105364768)

[3.1 Развёртывание, установка, настройка генератора 43](#_Toc105364769)

[3.2 Описание генератора 43](#_Toc105364770)

[3.3 Руководство администратора (программиста) 48](#_Toc105364771)

[3.4 Руководство пользователя (ответственного за ОП) 51](#_Toc105364772)

[3.5 Условия обеспечения работоспособности генератора 53](#_Toc105364773)

[3.6 Выводы по третьему разделу 54](#_Toc105364774)

[Заключение 55](#_Toc105364775)

[Список использованных источников 56](#_Toc105364776)

ВВЕДЕНИЕ

Целью выпускной квалификационной работы является разработка приложения, которое формирует документы рабочих программ дисциплин.

Для достижения поставленной цели необходимо решить следующие задачи:

* провести исследование предметной области;
* провести моделирование бизнес-процессов предметной области;
* выбрать современные инструментальные средства для проектирования и разработки программного обеспечения;
* провести разработку, тестирование и апробацию приложения;
* в соответствии с нормативными документами подготовить программную документацию.

1. **Аналитическое исследование**
2. **Описание предметной области**

В качестве предметной области был выбран процесс формирования документа рабочей программы дисциплины.

Для понимания, введём некоторые определения.

Рабочая программа дисциплины — это нормативный документ, входящий в состав основной образовательной программы высшего образования по соответствующему направлению подготовки и профилю [1].

Учебный план — документ, который определяет перечень, трудоемкость, последовательность и распределение по периодам обучения учебных предметов, курсов, дисциплин (модулей), практики, иных видов учебной деятельности и, если иное не установлено настоящим Федеральным законом, формы промежуточной аттестации обучающихся (п. 22 ст. 2 Федерального закона от 29.12.2012 г. N 273-ФЗ «Об образовании в Российской Федерации») [2].

Далее рассмотрим структуру РПД СГУПСа, её разделы и информацию, которая содержится в этих разделах.

Вначале РПД располагается титульный лист, который оформляется в соответствии со стандартами высшего учебного заведения.

В первом разделе РПД «Цель освоения дисциплины» указывается цель освоения дисциплины.

Во втором разделе РПД «Перечень планируемых результатов обучения по дисциплине (модулю), соотнесенных с планируемыми результатами освоения образовательной программы» указывается, что именно студент должен знать, уметь и какими навыками должен владеть по окончанию освоения дисциплины. Из названия понятно, что эта информация соотносится с информацией о результатах обучения из ОП.

В третьем разделе «Место дисциплины в структуре ОП» указывается к какой категории относится дисциплина и когда она изучается (например, 2 курс 3 семестр или 2, 3 курс 4, 5 семестры).

В четвёртый раздел «Структура и содержание дисциплины» входят следующие подразделы:

* объем дисциплины и виды учебной работы;
* содержание дисциплины.

В подразделе 4.1 объём дисциплины и виды учебной работы представлены таблицей, в которой указаны часы, отведённые на лекции, практики, лабораторные, курсовые или расчётно-графические работы и так далее. Также указан вид промежуточной аттестации (экзамен или зачёт) и общая трудоёмкость в часах и зачётных единицах.

В подразделе 4.2 содержание дисциплины представлено в подразделах 4.2.1 – 4.2.2. Если в качестве итоговой самостоятельной работы студента предусмотрена расчетно-графическая работа или курсовая работа, то это регламентируется в отдельном подразделе 4.2.3. В разделе 4.2.1 «Разделы дисциплины, виды учебной деятельности и формы контроля» представлена таблица с разделами изучаемой дисциплины, в которой прописаны часы, отведенные на каждый раздел и вид учебной деятельности. В разделе 4.2.2 «Содержание дисциплины (модуля), структурированное по разделам (темам)» представлена таблица тем разделов дисциплины с их описанием, разделённые на лекции и лабораторные работы в каждом разделе.

В пятом разделе «Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине (модулю)» представлены ссылки на методические материалы для самостоятельной работы студентов и некоторые положения о самостоятельной работе. Кроме того, там может быть положение, например, о курсовой, если она предусмотрена.

В шестой раздел «Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине (модулю)» входят следующие подразделы: 6.1 Перечень компетенций с указанием результатов обучения по дисциплине, соотнесенные с индикаторами достижения компетенций; 6.2 Типовые контрольные задания или иные материалы; 6.3 Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующие этапы формирования компетенций.

В разделе 6.1 приведена таблица компетенций, которые будут освоены в результате изучения дисциплины, с указанием индикаторов компетенций и результатов обучения. Информация в этой таблице берётся из таблицы компетенций учебного плана. После этого приводится таблица, которая может содержать в себе информацию идентичную информации в разделе 2. Только, помимо дублируемой информации, указываются также оценочные средства (зачёт, экзамен, тест, КР и так далее).

В разделе 6.2 содержится перечень вопросов к зачёту или вопросы и задачи к экзамену, а также может быть перечень вопросов к тесту или темы РГР, КР, если они предусмотрены. Проще говоря, содержание этого раздела зависит от формы проведения аттестации. Стоит отметить, что для каждого этапа аттестации предусмотрен отдельный подраздел. Например, «6.2.1 Зачёт пятый семестр» или «6.2.3 Тестовые задания», «6.2.4 Примерные темы РГР» и так далее.

В разделе 6.3 представлена таблица оценочных средств и описание критериев, шкал, процедур оценивания.

В 7-ом разделе «Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины (модуля)», как можно понять из названия, представлен список основной и дополнительной литературы.

В разделе 8 «Перечень ресурсов информационно-телекоммуникационной сети Интернет, необходимых для освоения дисциплины (модуля)» представлен список интернет-ресурсов, которые нужны для освоения дисциплины, как понятно из названия.

В разделе 9 «Методические указания для обучающихся по освоению дисциплины (модуля)» представлена таблица с видами учебной деятельности и промежуточной аттестацией, а также действия для организации этой деятельности.

В разделе 10 «Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень программного обеспечения и информационных справочных систем (при необходимости)» указаны всё необходимое ПО для осуществления учебной деятельности и ссылки на необходимые интернет-ресурсы.

В разделе 11 «Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю)» как правило представлены требования для занятий в аудитории. Кроме того, здесь может быть ссылка на 10 раздел с необходимым программным обеспечением.

И, наконец, в 12-ом разделе «Иные сведения и (или) материалы» пишется остальная необходимая информация. В большинстве случаев здесь просто делают акцент на том, что при проведении учебных занятий предусмотрено применение различных форм учебных занятий, развивающих у обучающихся навыки командной работы, межличностной коммуникации и так далее.

Теперь перейдём к характеристике процесса формирования документа РПД.

Преподаватель вручную на компьютере полностью формирует документ РПД. После этого, он отправляет его на проверку и, если нет ошибок, противоречий и прочих недочётов, то документ РПД утверждается. В противном случае, документ отправляется обратно преподавателю на исправление. После того, как документ будет исправлен, он снова отправляется на проверку.

1. **Определение и решение проблемы предметной области**

Нужно автоматизировать процесс формирования документа рабочей программы дисциплины.

Автоматизация процесса позволит минимизировать количество случаев, когда формируется некорректный документ рабочей программы и, как следствие, временные затраты проверяющего; минимизировать трудовые и временные затраты преподавателя. Помимо этого, будет повышена мотивация на дальнейшую работу, так как справляться с работой станет легче и быстрее.

Для решения проблемы нужно разработать приложение, которое на основе введённых данных (название дисциплины, файл УП, преподаватели-разработчики, файл таблицы компетенций) должно будет формировать документ РПД. Причём информация о преподавателях и кафедрах, на которых закреплены преподаватели, должна храниться в базе данных и извлекаться из неё.

Преподавателю останется только заполнить данные, которые относятся непосредственно к самому содержимому дисциплины (банк вопросов к тестам, зачётам, экзаменам; темы РГР, КР, рефератов; список необходимой литературы, ПО и так далее).

1. **Обзор аналогичных IT-решений**

В данном подразделе рассмотрим аналогичные IT-решения, которые применяются другими организациями.

Было найдено несколько аналогов: сервис-конструктор РПД образовательной платформы «Юрайт», «Генератор рабочих программ» (ТУСУР). Интерфейсы сервисов представлены на рисунках 1.1-1.2.

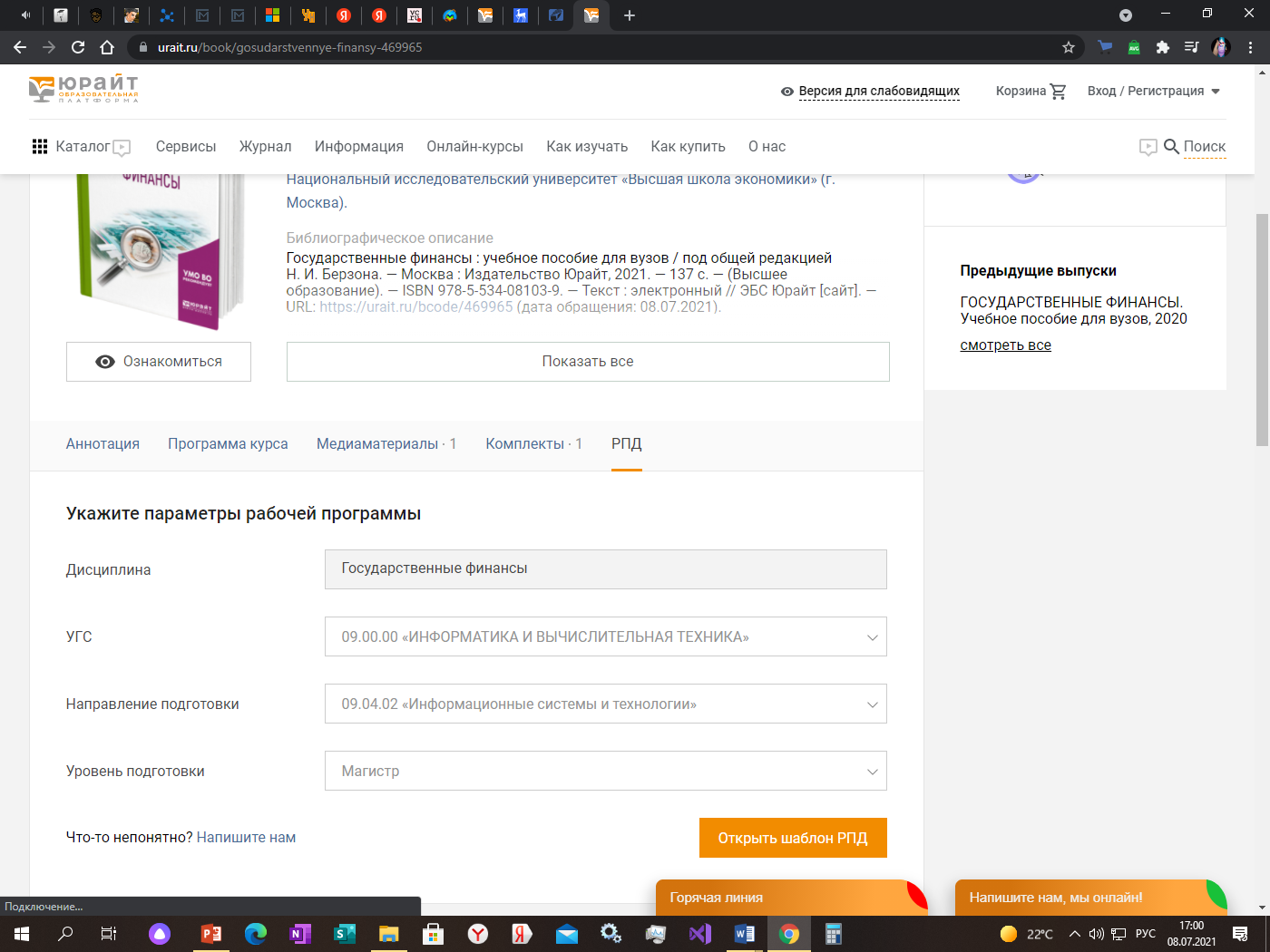


Рисунок 1.1 – Интерфейс сервиса «Юрайт»

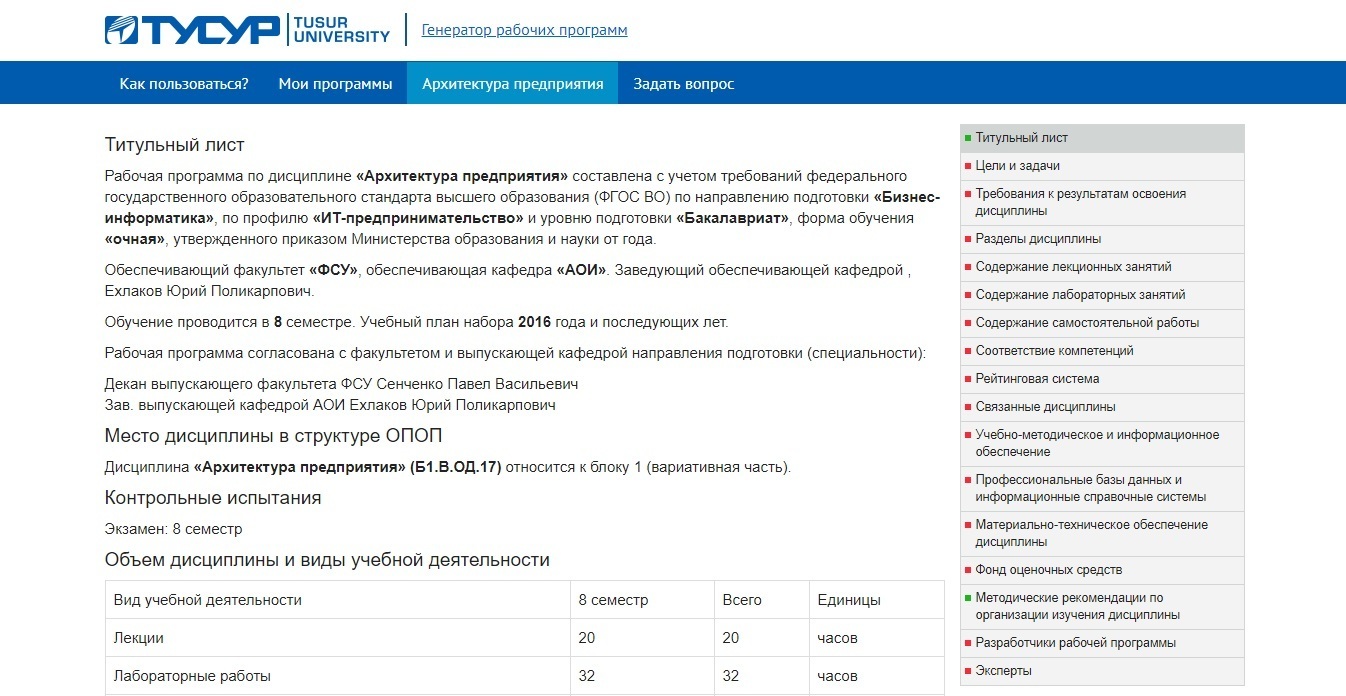


Рисунок 1.2 – Интерфейс сервиса «Генератор рабочих программ» ТУСУРа

И первый, и второй сервисы представляют интуитивно понятный интерфейс. Пользовательский интерфейс первой платформы нацелен на простоту в использовании, в многом исключая появления каких-либо технических проблем и затруднений. Интерфейс второй платформы довольно многофункционален, что позволит эффективно генерировать документы рабочих программ с минимумом правок, а также довольно просто и быстро обновлять эти программы. Сайты работают на компьютере, планшете и смартфоне. Пользователь может посетить платформы в любое свободное время. Сайты удобны и имеют понятный функционал [3, 4].

По данному подразделу можно сделать следующий вывод: в разрабатываемом программном обеспечении должен быть удобный и интуитивно понятный интерфейс, а также следующий функционал: извлечение информации о дисциплине из файла УП; извлечение информации об индикаторах компетенции из файла таблицы компетенций; заполнение результатов обучения по каждому индикатору компетенции; выбор преподавателей в разделе листа согласования; генерация документа.

1. **Недостатки аналогов**

Теперь, в данном подразделе, рассмотрим недостатки аналогичных применяемых решений.

В первом сервисе, вместе с простотой использования, мы получаем неполную информацию. Иными словами, документ формируется лишь частично. За нами останется заполнение титульного листа документа рабочей программы и прочие частные моменты. Помимо этого, сервисом невозможно воспользоваться в офлайн режиме, что может быть необходимо.

В случае со вторым сервисом всё немного по-другому. За счёт большей функциональности, документ формируется со всеми необходимыми шаблонами, но оформление РПД не соответствует стандартам оформления СГУПСа. Это вполне естественно, поскольку сервис создан под стандарты ТУСУРа. Также, как и в первом случае, с сервисом невозможно работать офлайн.

Для более наглядного примера несоответствия оформлений будут представлены титульные листы РПД на рисунках 1.3-1.5.

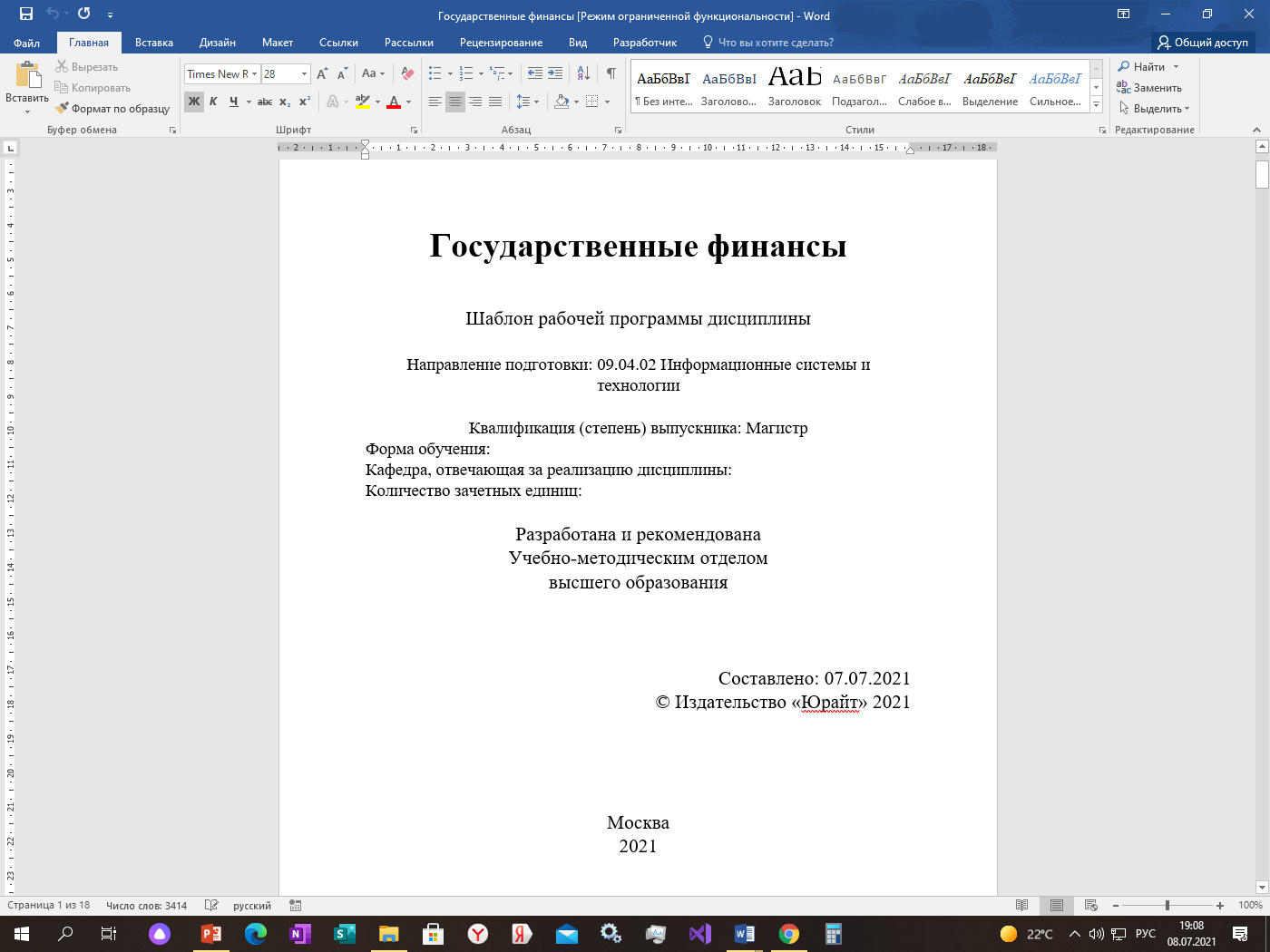


Рисунок 1.3 – Титульный лист РПД созданной в сервисе «Юрайт»

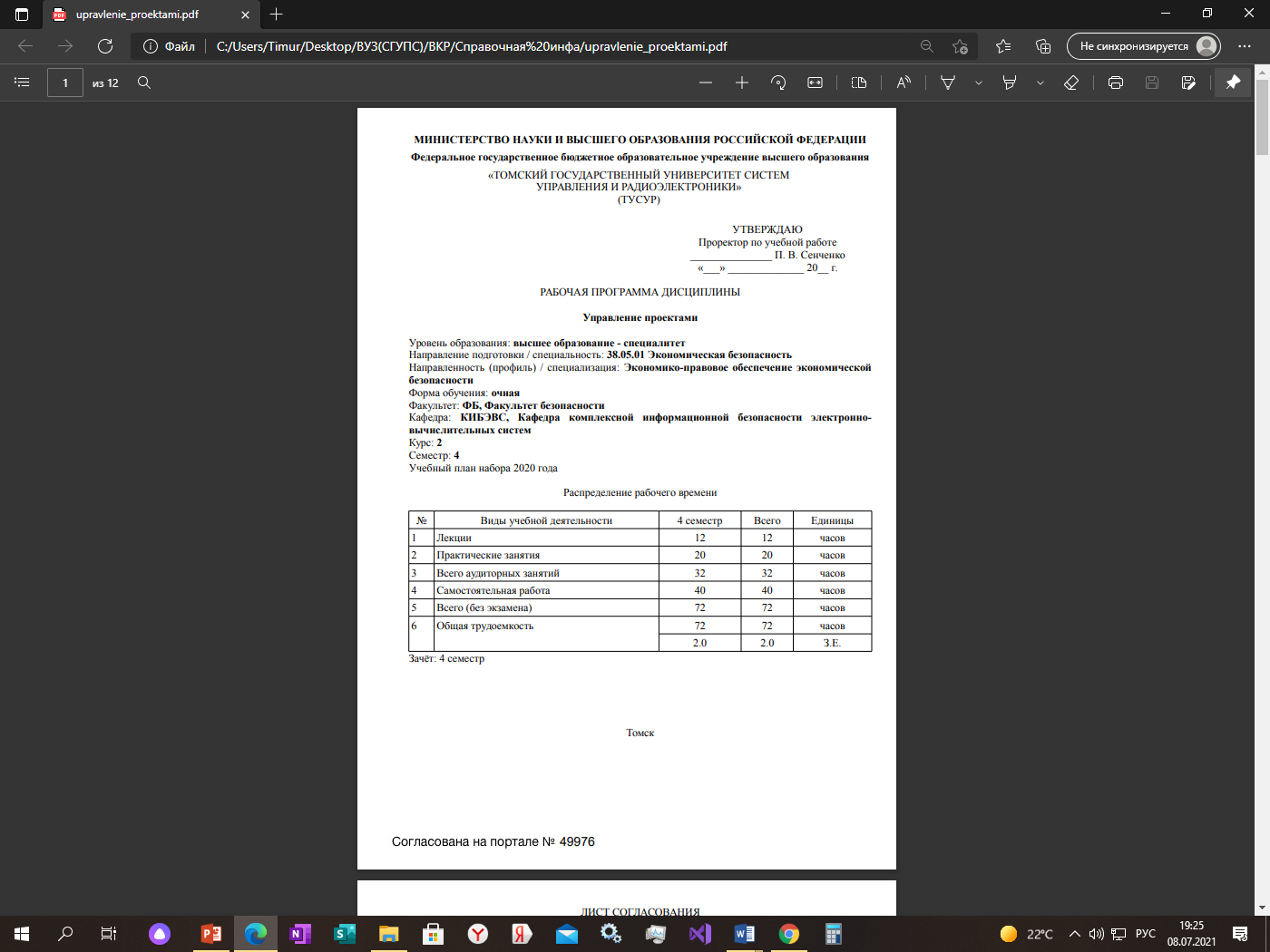


Рисунок 1.4 – Титульный лист РПД созданной в сервисе «Генератор рабочих программ» ТУСУРа

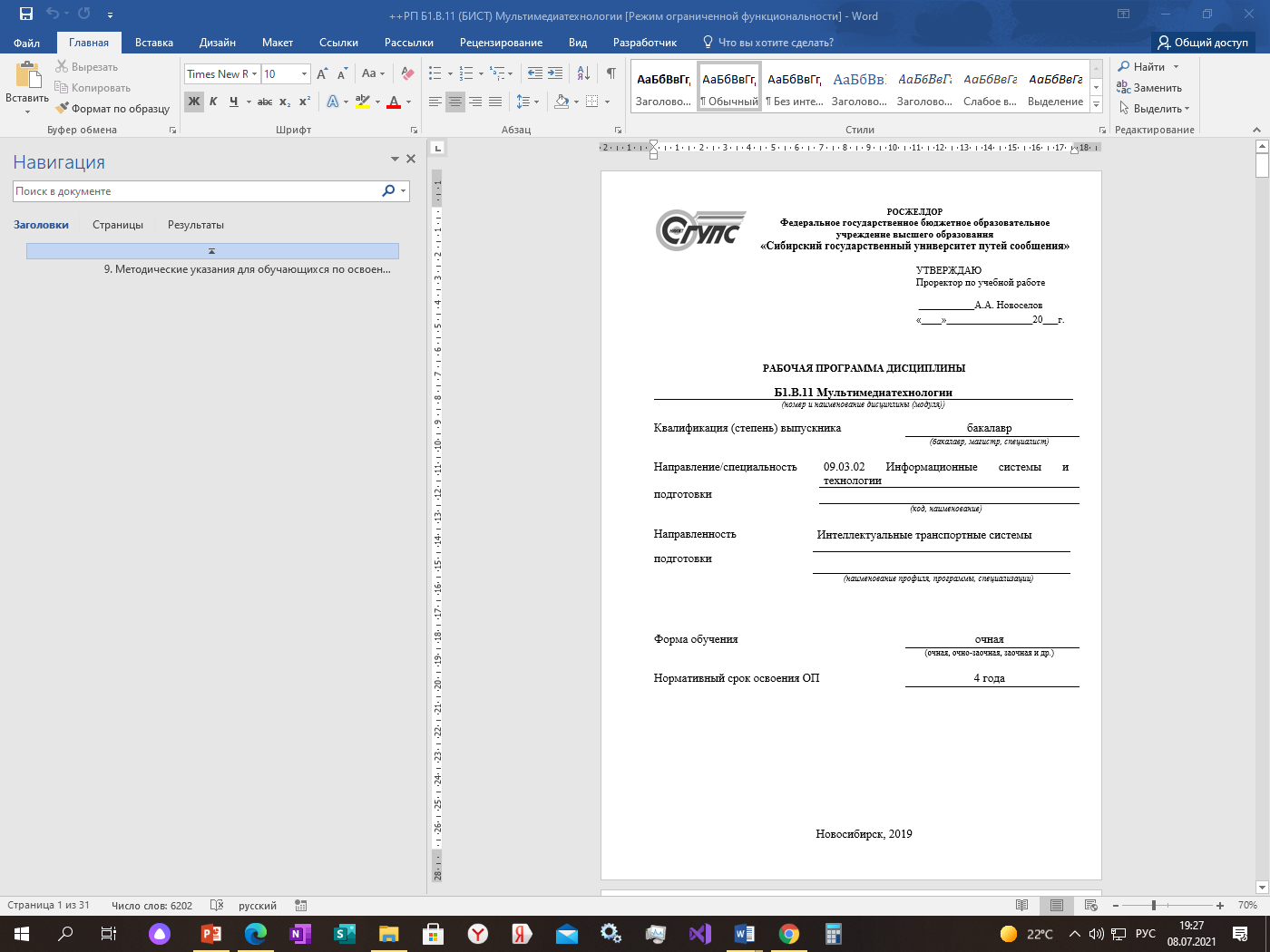


Рисунок 1.5 - Титульный лист РПД СГУПСа

Таким образом, исходя из анализа недостатков сервисов, можно сделать вывод, что разрабатываемое программное обеспечение (ПО) должно формировать шаблон РПД, который будет соответствовать нормам оформления СГУПСа. Помимо этого, должна быть возможность работать с ПО в офлайн режиме.

1. **Обоснование целесообразности разработки**

В данном подразделе рассмотрим целесообразность разработки приложения.

Сначала рассмотрим то, как происходит на текущий момент формирование документа РПД.

Преподаватель сам формирует документ РПД. После этого, он отправляет его на проверку ответственному за ОП и, если нет ошибок, противоречий и прочих недочётов, то документ РПД утверждается. В противном случае, документ отправляется обратно преподавателю на исправление. После того, как документ будет исправлен, он снова отправляется на проверку.

Приложение позволит: минимизировать количество случаев, когда отправляют дефектные документы и, как следствие, временные затраты проверяющего; минимизировать трудовые и временные затраты преподавателя. Помимо этого, будет повышена мотивация на дальнейшую работу, так как справляться с работой станет легче и быстрее.

Теперь подведём итоги анализа аналогов.

Для наглядности представим результаты анализа аналогов в таблице 1.1.

Таблица 1.1 – Результаты анализа аналогов

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Дружественный интерфейс | Отвечает стандартам оформления СГУПС | Наличие офлайн режима |
| Юрайт | Да | Нет, нужно оформлять вручную | Нет |
| Генератор рабочих программ ТУСУРа | Да | Нет | Нет |

Как можно увидеть из результатов анализа аналогов, пользоваться ими, в теории, можно, но за счёт их недостатков эффективность процесса создания РПД может повысится не так сильно, как могла бы повысится, если бы эти проблемы были разрешены. Во время отсутствия интернета процесс может и вовсе полностью приостановится.

Разрабатываемое приложение позволит работать как в онлайне, так и в офлайне, а также будет соответствовать стандартам оформления СГУПСа.

Таким образом, была обоснована целесообразность разработки приложения. Были выделены улучшения, которые принесёт в процесс разрабатываемое приложение. Также были подведены итоги результатов анализа аналогичных решений и были обоснованы преимущества разрабатываемого ПО перед аналогами, что тоже доказывает целесообразность разработки.

1. **Обзор методологий моделирования бизнес процессов**

*Методология IDEF0.*

IDEF0 — [методология](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D1%8F) функционального моделирования и графическая нотация, предназначенная для формализации и описания [бизнес-процессов](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D0%B7%D0%BD%D0%B5%D1%81-%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81). Отличительной особенностью IDEF0 является её акцент на соподчинённость объектов. В IDEF0 рассматриваются логические отношения между работами, а не их временная последовательность ([поток работ](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%82%D0%BE%D0%BA_%D1%80%D0%B0%D0%B1%D0%BE%D1%82)).

Стандарт IDEF0 представляет организацию как набор модулей, здесь существует правило — наиболее важная функция находится в верхнем левом углу, кроме того, существуют правила сторон:

* стрелка входа всегда приходит в левую кромку активности;
* стрелка управления — в верхнюю кромку;
* стрелка механизма — нижняя кромка;
* стрелка выхода — правая кромка.

Описание выглядит как «[чёрный ящик](https://ru.wikipedia.org/wiki/%D0%A7%D1%91%D1%80%D0%BD%D1%8B%D0%B9_%D1%8F%D1%89%D0%B8%D0%BA)» с входами, выходами, управлением и механизмом, который постепенно детализируется до необходимого уровня. Также для того, чтобы быть правильно понятым, существуют словари описания активностей и стрелок. В этих словарях можно дать описания того, какой смысл вы вкладываете в данную активность либо стрелку [5].

Описание методологии IDEF0 содержится в рекомендациях Р 50.1.028-2001 «Информационные технологии поддержки жизненного цикла продукции. Методология функционального моделирования».

*Методология DFD.*

DFD — общепринятое сокращение от англ. data flow diagrams — диаграммы потоков данных. Так называется методология графического структурного анализа, описывающая внешние по отношению к системе источники и адресаты данных, логические функции, потоки данных и хранилища данных, к которым осуществляется доступ.

Диаграмма потоков данных (data flow diagram, DFD) — один из основных инструментов структурного анализа и проектирования информационных систем, существовавших до широкого распространения UML. Несмотря на имеющее место в современных условиях смещение акцентов от структурного к объектно-ориентированному подходу к анализу и проектированию систем, «старинные» структурные нотации по-прежнему широко и эффективно используются как в бизнес-анализе, так и в анализе информационных систем.

Исторически сложилось так, что для описания диаграмм DFD используются две нотации — Йордана (Yourdon) и Гейна-Сарсон (Gane-Sarson), отличающиеся синтаксисом.

Информационная система принимает извне потоки данных. Для обозначения элементов среды функционирования системы используется понятие внешней сущности. Внутри системы существуют процессы преобразования информации, порождающие новые потоки данных. Потоки данных могут поступать на вход к другим процессам, помещаться (и извлекаться) в накопители данных, передаваться к внешним сущностям.

Модель DFD, как и большинство других структурных моделей — иерархическая модель. Каждый процесс может быть подвергнут декомпозиции, то есть разбиению на структурные составляющие, отношения между которыми в той же нотации могут быть показаны на отдельной диаграмме. Когда достигнута требуемая глубина декомпозиции — процесс нижнего уровня сопровождается мини-спецификацией (текстовым описанием).

Кроме того, нотация DFD поддерживает понятие подсистемы — структурного компонента разрабатываемой системы.

Нотация DFD — удобное средство для формирования контекстной диаграммы, то есть диаграммы, показывающей разрабатываемую АИС в коммуникации с внешней средой. Это диаграмма верхнего уровня в иерархии диаграмм DFD. Её назначение — ограничить рамки системы, определить, где заканчивается разрабатываемая система и начинается среда [6].

*Методология объектного проектирования и анализа на языке UML.*

UML (англ. Unified Modeling Language — унифицированный язык моделирования) — язык графического описания для объектного моделирования в области разработки программного обеспечения, для моделирования бизнес-процессов, системного проектирования и отображения организационных структур.

UML является языком широкого профиля, это — открытый стандарт, использующий графические обозначения для создания абстрактной модели системы, называемой UML-моделью. UML был создан для определения, визуализации, проектирования и документирования, в основном, программных систем. UML не является языком программирования, но на основании UML-моделей возможна генерация кода.

UML позволяет также разработчикам программного обеспечения достигнуть соглашения в графических обозначениях для представления общих понятий (таких как класс, компонент, обобщение (англ. generalization), агрегация (англ. aggregation) и поведение) и больше сконцентрироваться на проектировании и архитектуре.

*Методология ARIS.*

ARIS (акроним от англ. Architecture of Integrated Information Systems) — методология и тиражируемый программный продукт для моделирования бизнес-процессов организаций. Продукт и методология принадлежат немецкой компании Software AG как результат поглощения компании IDS Scheer автора методологии Августа-Вильгельма Шеера.

Любая организация в методологии ARIS рассматривается с пяти точек зрения: организационной, функциональной, обрабатываемых данных, структуры бизнес-процессов, продуктов и услуг. При этом каждая из этих точек зрения разделяется ещё на три подуровня: описание требований, описание спецификации, описание внедрения. Для описания бизнес-процессов предлагается использовать около 80 типов моделей, каждая из которых принадлежит тому или иному аспекту. ARIS предоставляет визуальный инструментарий для обеспечения наглядности моделей. Также инструментарий поставляется с набором референтных моделей, заранее разработанных для типичных процессов в различных отраслях. Общий принцип в инструментарии — возможность интеграции моделей разных типов в рамках одного репозитория посредством декомпозиции (детализации) объектов. Таким образом, любую организацию можно описать с помощью иерархии моделей — от обобщения [7].

1. **Выводы по первому разделу**

По итогу первого раздела была описана предметная область, определена проблема предметной области и её решение, проанализированы аналоги, обоснована целесообразность разработки, а также проведён обзор изученных методологий моделирования бизнес-процессов.

# **Проектирование информационной системы**

1. **Выбор методологии моделирования и моделирование бизнес-процессов предметной области**

В качестве нотации была выбрана методология UML. В качестве среды разработки диаграмм использовался Visual Paradigm 16.3.

Перед тем, как сформировать документ, разработчик создаёт саму РПД, затем передаёт её составителю. Далее составитель формирует документ к полученной рабочей программе, после чего отправляет его ответственному за ОП. Далее ответственный за ОП проверяет документ на наличие ошибок и, если всё нормально, то отправляет его в учебный отдел на утверждение, в противном случае документ отправляется на доработку составителю. Соответственно учебный отдел, получив документ, проводит контроль согласования содержания документа РПД и утверждает его.

Разработанные диаграммы процесса представлены на рисунках 2.1-2.9.

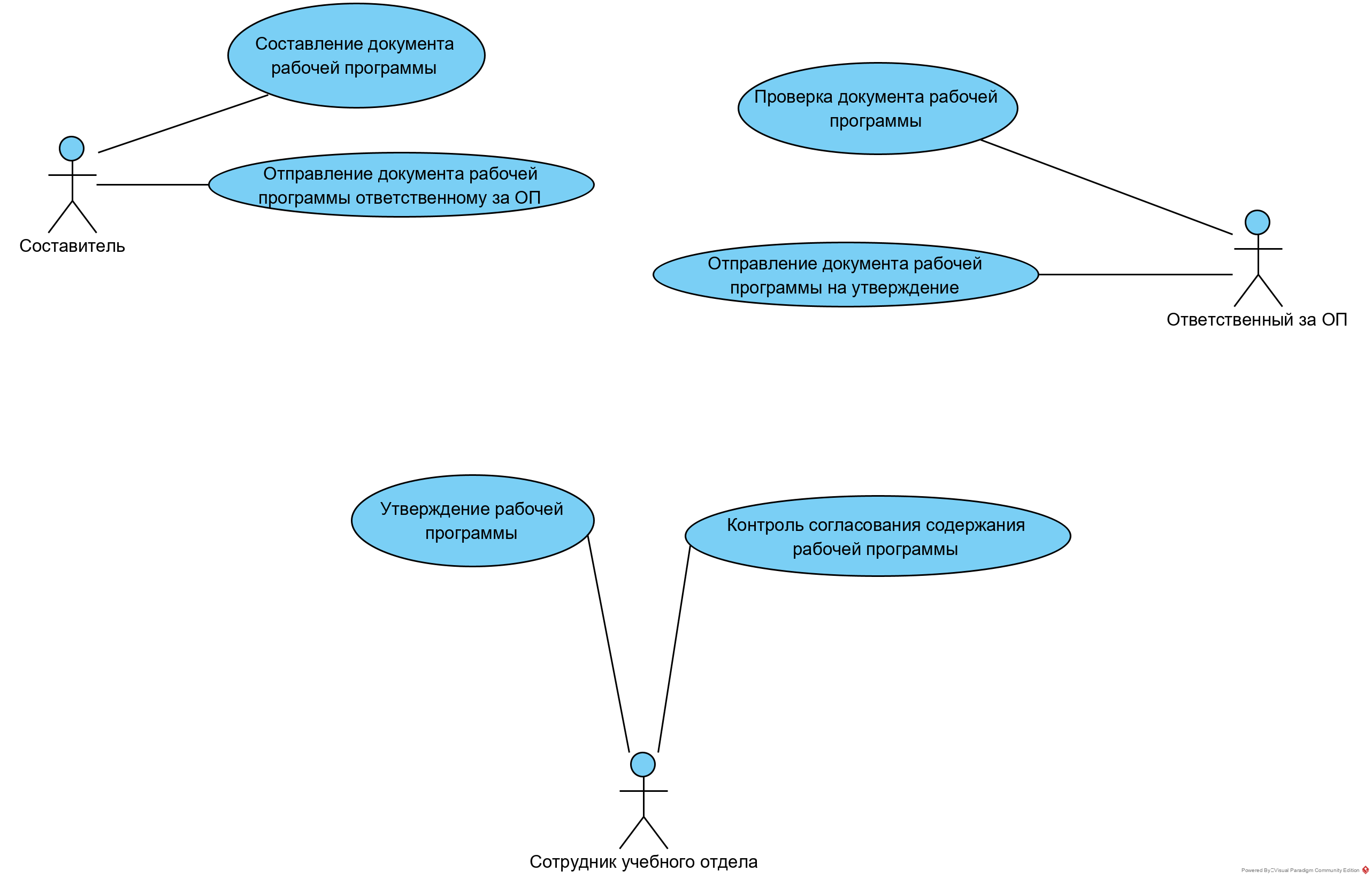


Рисунок 2.1 – Use case диаграмма участников создания документа РПД

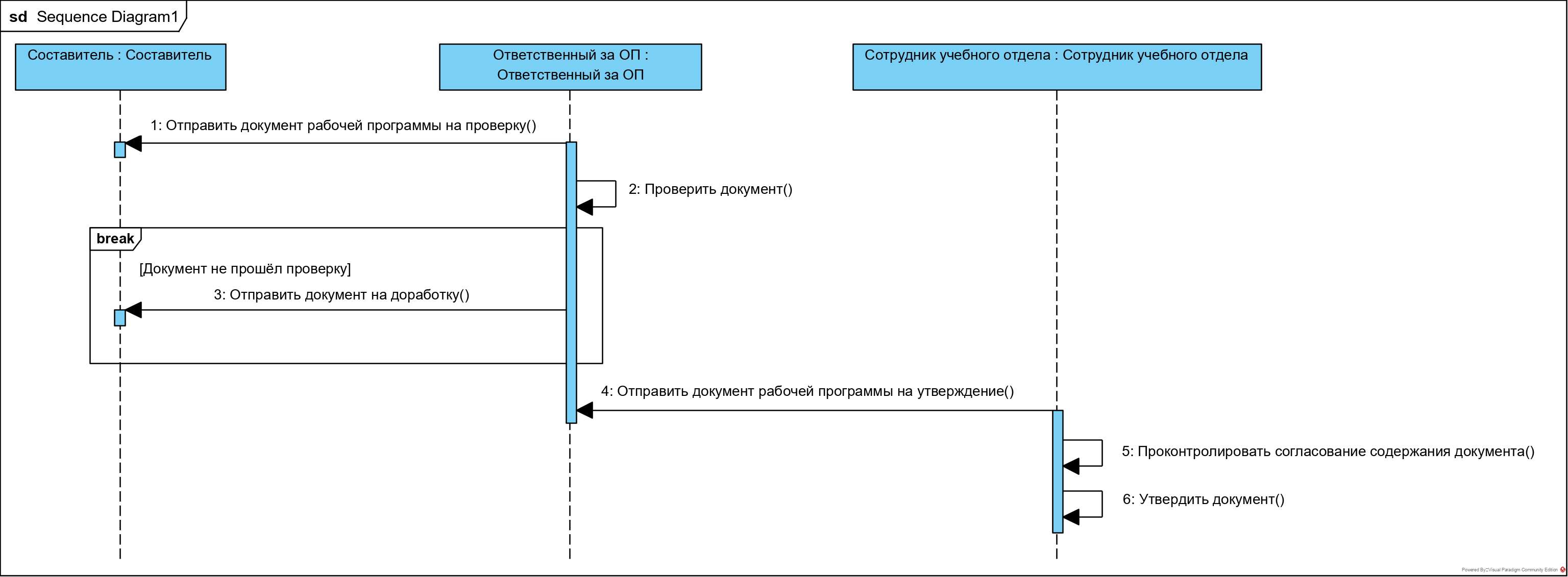


Рисунок 2.2 – Sequence диаграмма процесса верификации документа

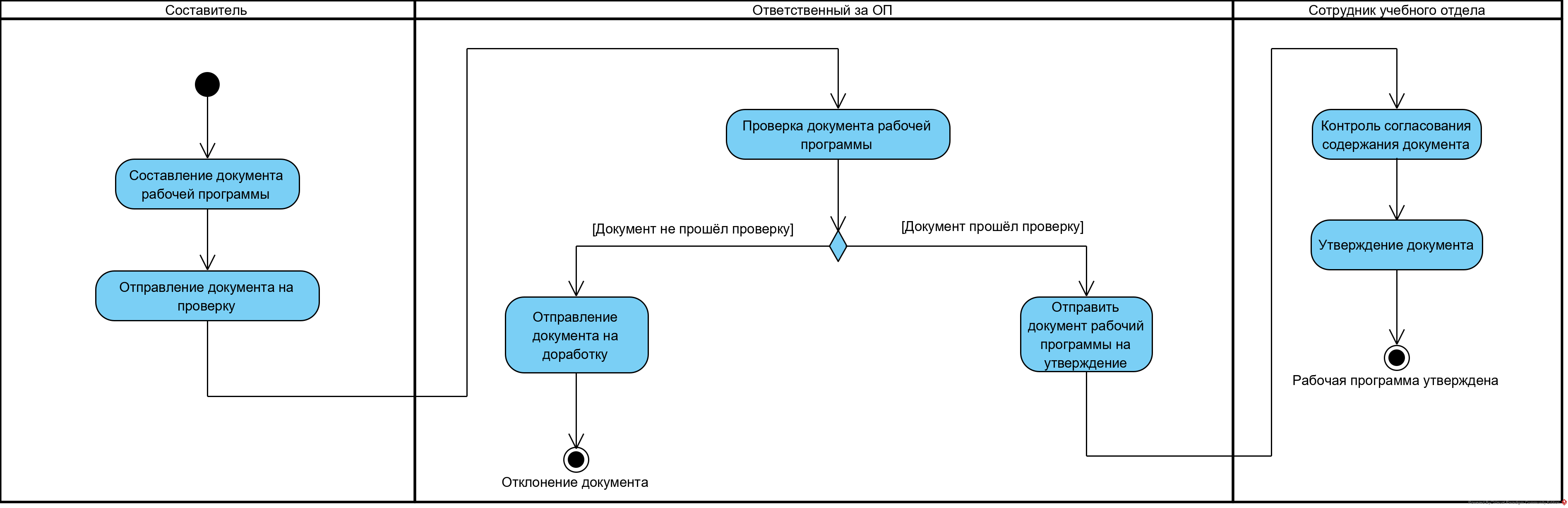


Рисунок 2.3 – Диаграмма активности (activity) процесса верификации документа

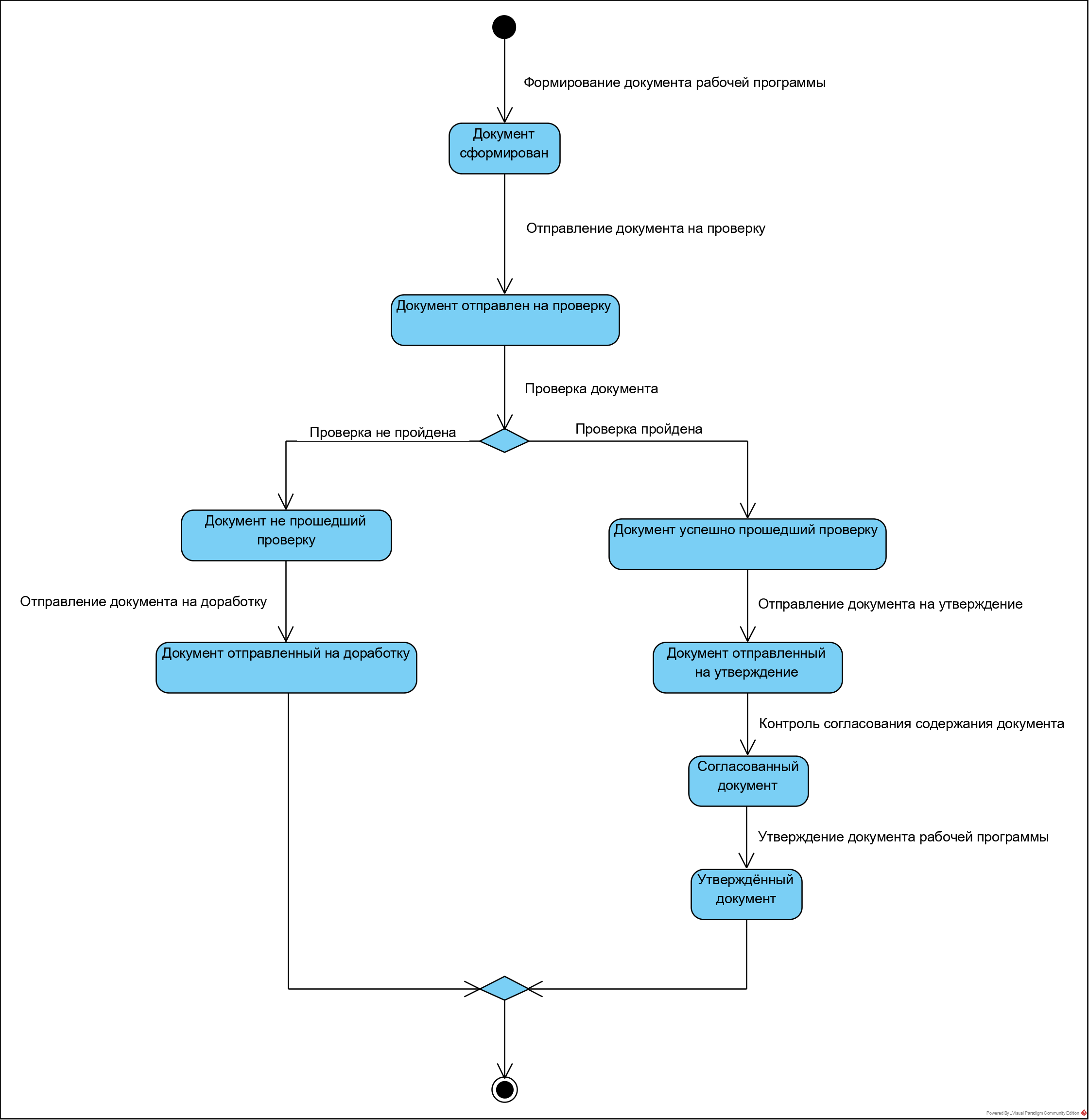


Рисунок 2.4 – Диаграмма состояний (state machine) документа РПД

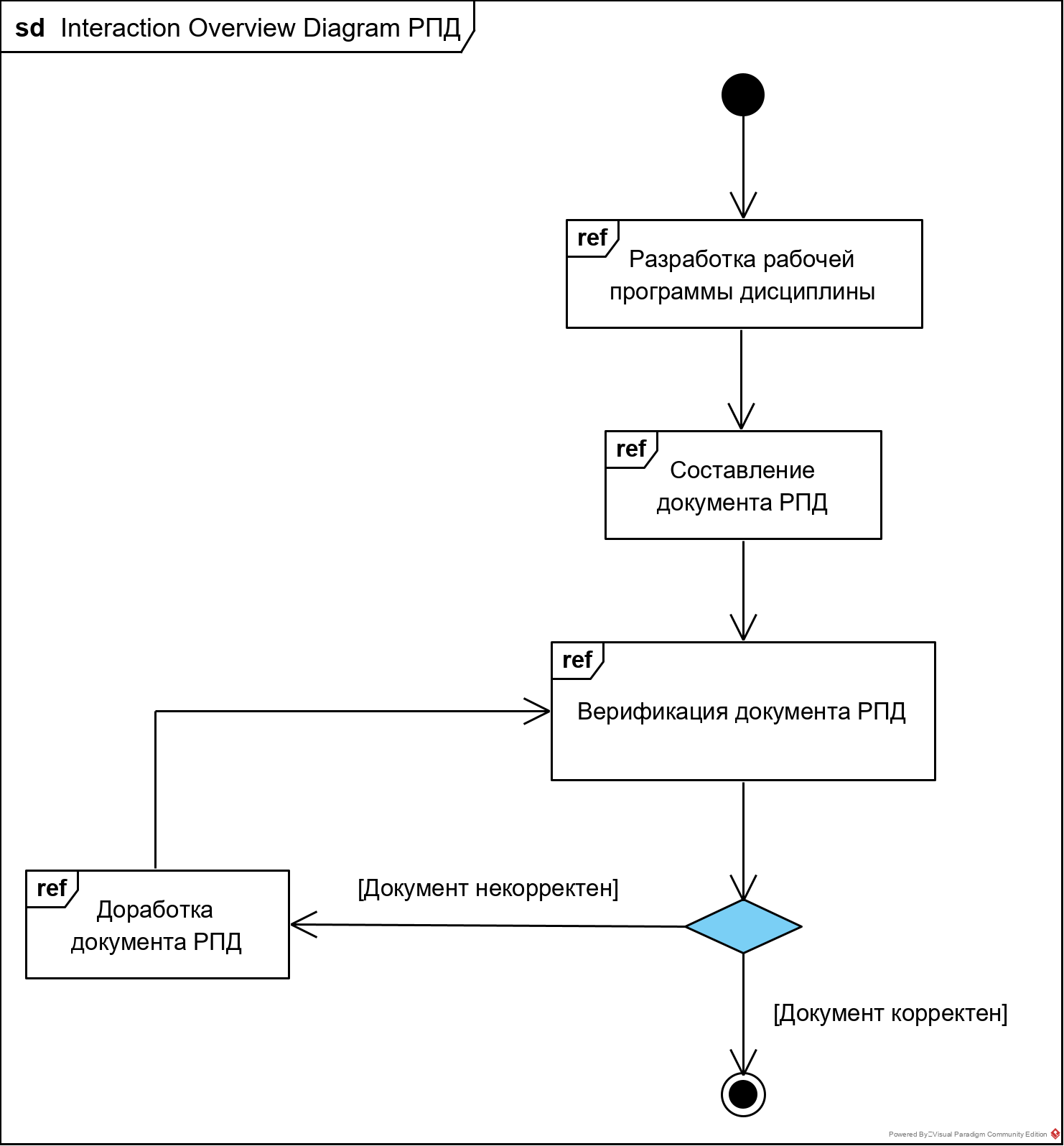


Рисунок 2.5 – Диаграмма обзора взаимодействия (interaction overview) процесса создания РПД

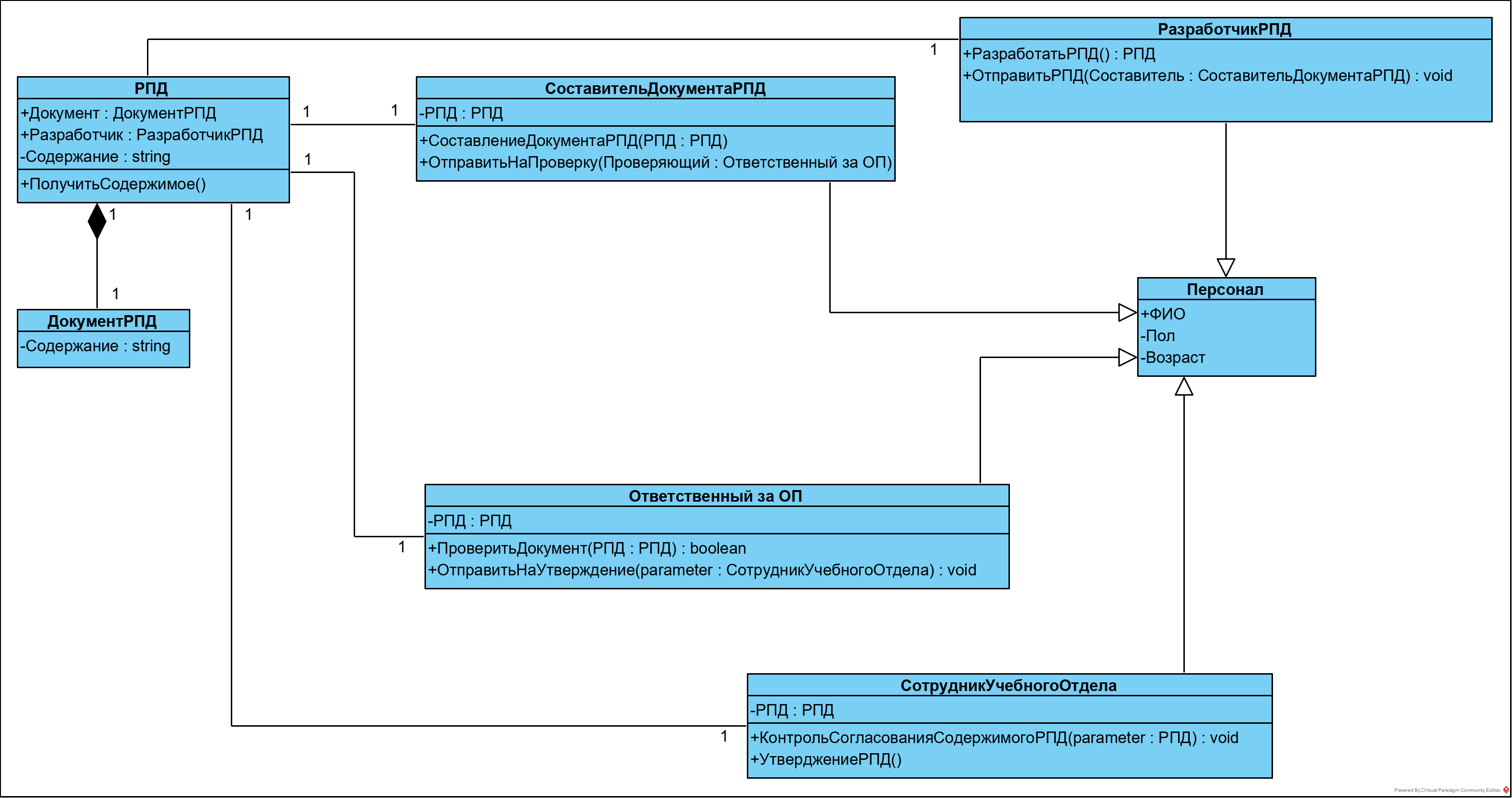


Рисунок 2.6 – Диаграмма классов участников создания РПД

Sub-диаграммы рисунка 2.5 представлены на рисунках 2.7-2.9.

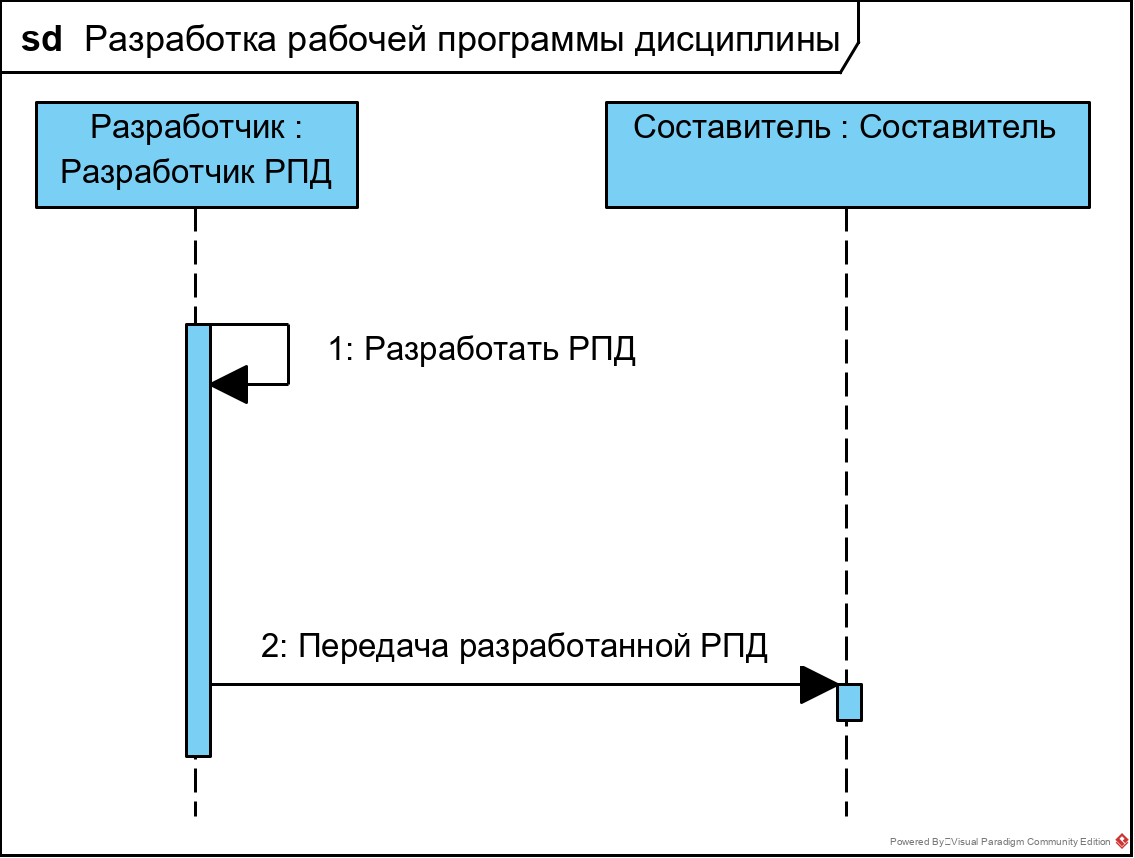


Рисунок 2.7 – Sub-диаграмма процесса разработки РПД

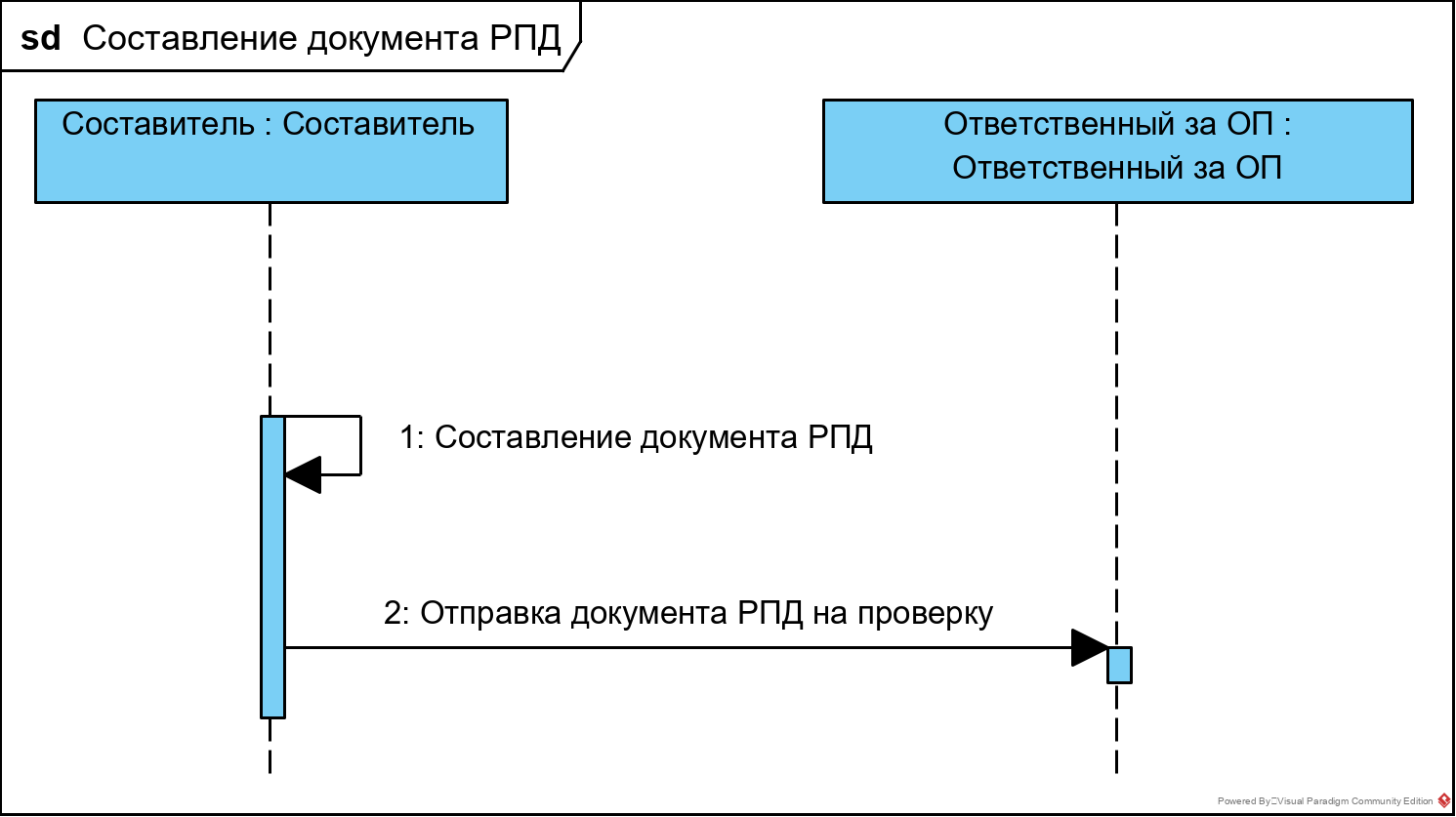


Рисунок 2.8 – Sub-диаграмма процесса составления документа РПД

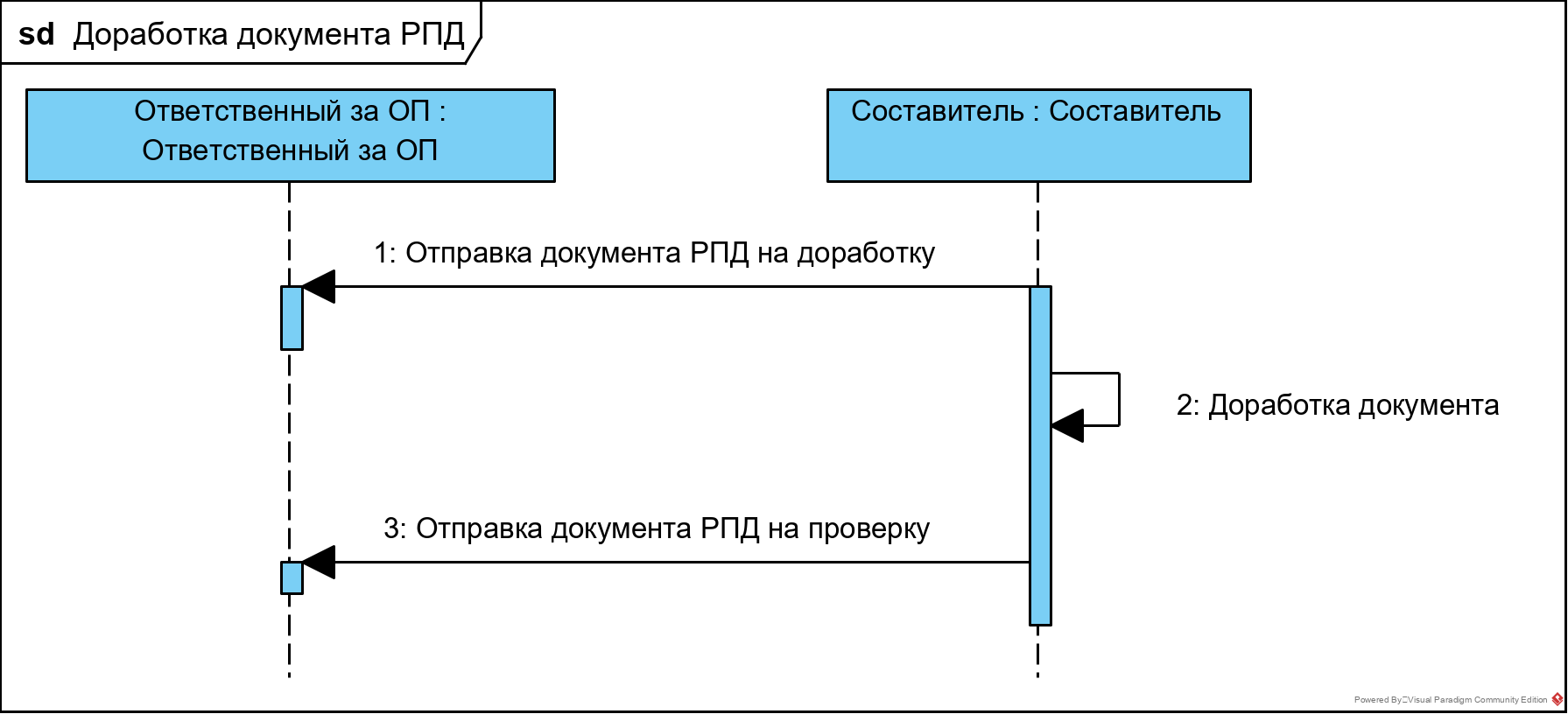


Рисунок 2.9 – Sub-диаграмма процесса доработки документа РПД

Процесс верификации документа РПД был ранее представлен на sequence диаграмме, которая изображена на рисунке 2.2.

## **Выбор и обоснование типа БД информационной системы**

В качестве типа базы данных (БД) был выбран SQLite. SQLite это однопользовательская БД, которая работает с обычными файлами на диске (.db, .sqlite). Она распространяется в виде кроссплатформенной библиотеки, это означает, что она может работать как на Windows, так, например, и на Android. Ещё одним преимуществом SQLite является компактность, она занимает мало места (~300 КБ).

## **Язык программирования**

Приложение будет разрабатываться на языке C#. Тип приложения: десктопное WPF-приложение.

*Общие сведения.*

C# (произносится как «си шарп») — современный объектно-ориентированный и типобезопасный язык программирования. C# позволяет разработчикам создавать разные типы безопасных и надежных приложений, выполняющихся в .NET. C# относится к широко известному семейству языков C, и покажется хорошо знакомым любому, кто работал с C, C++, Java или JavaScript. Это язык со строгой типизацией. Широко применяется в разработке ПО [8].

*Элементы языка.*

Как правило, программный модуль содержит класс, в котором могут быть данные в виде полей и свойств, операции в виде методов. Сам класс помещается в пространство имён. В методах, как правило, используются переменные.

Для объявления пространства имён используется ключевое слово namespace, далее пишется его название и ставятся фигурные скобки. В пространстве имён объявляется класс, пишется ключевое слово class и его название, также ставятся фигурные скобки. Как правило, при создании программного модуля, среда разработки сама создаёт этот шаблон (пространство имён и класс в нём).

Пример:

namespace WpfApp1

{

class Test

{

}

}

Для создания операций программного модуля используются методы. Для того объявить метод, нужно указать его тип. Подробное описание типов приведено в таблице 2.1 [9].

Таблица 2.1 – Типы С#

|  |  |  |  |
| --- | --- | --- | --- |
| Тип | Описание | Размер в байтах | CTS Тип |
| Базовый тип | | | |
| object | Может хранить все что угодно, так как является всеобщим предком | - | System.Object |
| Целые типы | | | |
| byte | Целое число от 0 до 255 | 1 | System.Byte |
| sbyte | Целое число от -128 до 127 | 1 | System.SByte |
| short | Целое со знаком (от -32 768 до 32 767) | 2 | System.Int16 |
| ushort | Целое без знака (от 0 до 65 535) | 2 | System.UInt16 |

Продолжение таблицы 2.1

|  |  |  |  |
| --- | --- | --- | --- |
| Тип | Описание | Размер в байтах | CTS Тип |
| int | Целое со знаком (от -2 147 483 648 до 2 147 483 647) | 4 | System.Int32 |
| uint | Целое число без знака ( от 0 до 4 294 967 295) | 4 | System.UInt32 |
| long | Целое со знаком (от -9 223 372 036 854 775 808 до 9 223 372 036 854 775 807) | 8 | System.Int64 |
| ulong | Целое без знака (от 0 до 0fffffffffffffff) | 8 | System.UInt64 |
| Вещественные типы (с плавающей точкой) | | | |
| float | Число с плавающей точкой одинарной точности. Содержит значения приблизительно от -1.5⋅10-45 до +3.4⋅1038 c 7 значащими цифрами | 4 | System.Single |
| double | Число с плавающей точкой двойной точности. Содержит значения приблизительно от -5.0⋅10- 324 до 1.7⋅10308 c 15-16 значащими цифрами | 8 | System.Double |
| Строковый тип | | | |
| string | Строка из Unicode-символов | Ссылочный тип | System.String |

Продолжение таблицы 2.1

|  |  |  |  |
| --- | --- | --- | --- |
| Тип | Описание | Размер в байтах | CTS Тип |
| Символьный тип | | | |
| char | Символы Unicode | 2 | System.Char |
| Логический тип | | | |
| bool | true или false | 1 | System.Boolean |
| Финансовый тип (вещественный тип с фиксированной точкой) | | | |
| decimal | Число до 28 знаков с фиксированным положением десятичной точки. Обычно используется в финансовых расчетах. Требует суффикса «m» или «М» | 12 | System.Decimal |

В С# наиболее используемыми типами являются int, string, char, double, bool. Вернёмся к методам.

Для объявления метода сначала указывается его тип, затем название, затем скобки, в которых могут быть параметры. У методов может быть тип void, это означает, что метод ничего не возвращает. Также можно указать модификатор доступности (private, protected, public). Потом также указываются фигурные скобки, в качестве тела метода.

В методе, как правило, используются переменные, либо поля/свойства.

Для объявления переменной указывается её тип, название. Только поле объявляется в классе, а не в методе.

Массив объявляется почти также. Только после типа нужно указать квадратные скобки.

Как правило, для доступа к данным используются свойства. Объявляется оно подобно полю, только после названия указывается блок с геттером и сеттером (get, set). У автосвойства геттер и сеттер пустой. Поле для него объявляется неявно.

Пример программного модуля:

namespace WpfApp1

{

class Test

{

public int MyPropertyAuto { get; set; } // автосвойство

private int myVar; // поле

public int MyProperty // свойство, дающее доступ к полю

{

get { return myVar; }

set { myVar = value; }

}

void TestMethod() // метод

{

int digit = 0;// переменная

digit += 5;

int[] mas = new int[2]; // массив

mas[0] = digit;

string str = "Hello ";

string str2 = "world!";

string strres = str + str2;

}

}

}

*Способы структурирования программы.*

Методы можно вызывать в других ключевых методах. В вызываемые методы через параметры можно передавать переменные/поля/свойства, значениями которых будет управлять вызываемый метод (или самими параметрами, если установлен модификатор ref).

Пример:

int TestMethod2(int param)

{

return param += 5;

}

void Method()

{

var var = 0;

var = TestMethod2(var);

}

*Средства обмена данными.*

В случае консольного приложения всё просто. Вывод данных осуществляется с помощью метода Write или WriteLine класса Console. В первом случае просто выводится сообщение на консоль, а во втором случае после вывода сообщения курсор переходит на следующую строку.

Пример:

namespace ConsoleApp1

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Hello world!!!");

int digit = 5;

Console.Write("Число ");

Console.WriteLine(digit.ToString());

Console.WriteLine("Конец");

Console.ReadKey();

}

}

}

Результат вывода представлен на рисунке 2.10.

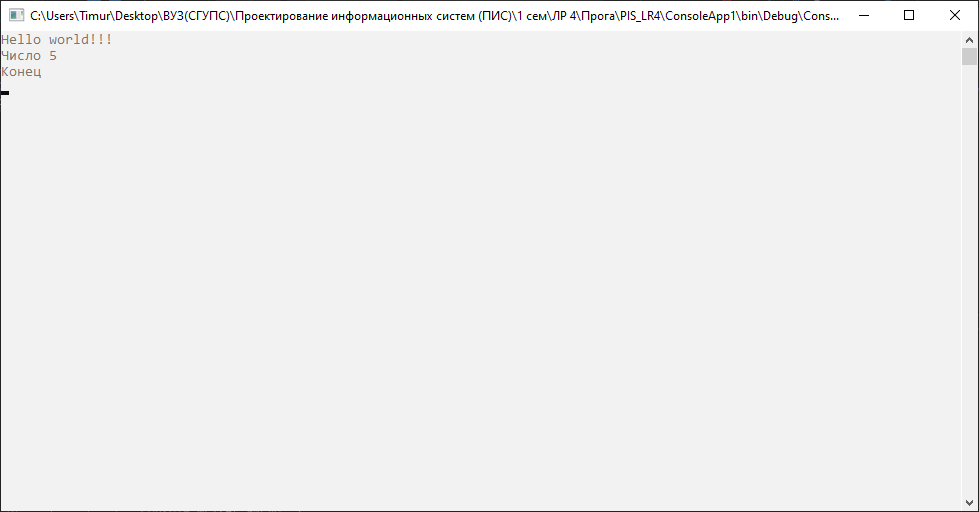


Рисунок 2.10 – Результат вывода

Для ввода ситуация аналогичная. Используется метод Read/Readline (просто вывод, вывод с переносом курсора соответственно).

Пример:

static void Main(string[] args)

{

Input();

Console.WriteLine("Конец");

Console.ReadKey();

}

static void Input()

{

Console.WriteLine("Введите строку");

string str = Console.ReadLine();

Console.Write("Строка ");

Console.WriteLine(str);

}

В случае десктопного приложения (WPF-приложения) ситуация другая. Чтобы что-то ввести, нужно создать объект в разметке XAML окна, в который будет что-то вводится. Как правило, этим объектом служит Textbox. После ввода эта информация может быть обработана в каком-нибудь методе. Обычно для этого используется метод события Click элемента Button. После чего информация выводится в элемент представления. Это также может быть Textbox, однако разработчики чаще всего предпочитают TextBlock, так как. он по умолчанию доступен только для чтения.

Пример разметки XAML:

<Grid>

<StackPanel>

<TextBox x:Name="tbIn"></TextBox>

<CheckBox x:Name="check"></CheckBox>

<Button x:Name="btn" Click="btn\_Click">Click</Button>

<Label>Output</Label>

<TextBlock x:Name="tbOut"></TextBlock>

</StackPanel>

</Grid>

Пример метода обработки данных:

private void btn\_Click(object sender, RoutedEventArgs e)

{

Test test = new Test();

int digit = 5;

digit = test.TestMethod2(digit);

string res = tbIn.Text + " " + check.IsChecked + " " + digit;

tbOut.Text = res;

}

Результат работы программы представлен на рисунке 2.11.

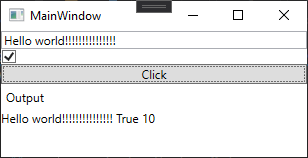


Рисунок 2.11 – Результат работы программы

*Встроенные элементы.*

Для арифметических операций можно использовать класс Math. В нём доступно множество методов для работы с числами [10]:

* Abs(double value): возвращает абсолютное значение для аргумента value;

|  |
| --- |
| double result = Math.Abs(-12.4); // 12.4 |

* Acos(double value): возвращает арккосинус value. Параметр value должен иметь значение от -1 до 1;

|  |
| --- |
| double result = Math.Acos(1); // 0 |

* Asin(double value): возвращает арксинус value. Параметр value должен иметь значение от -1 до 1;
* Atan(double value): возвращает арктангенс value;
* BigMul(int x, int y): возвращает произведение x \* y в виде объекта long;

|  |
| --- |
| double result = Math.BigMul(100, 9340); // 934000 |

* Ceiling(double value): возвращает наименьшее целое число с плавающей точкой, которое не меньше value;

|  |
| --- |
| double result = Math.Ceiling(2.34); // 3 |

* Cos(double d): возвращает косинус угла d;
* Cosh(double d): возвращает гиперболический косинус угла d;
* DivRem(int a, int b, out int result): возвращает результат от деления a/b, а остаток помещается в параметр result;

|  |
| --- |
| int result;  int div = Math.DivRem(14, 5, out result);  //result = 4  // div = 2 |

* Exp(double d): возвращает основание натурального логарифма, возведенное в степень d;
* Floor(decimal d): возвращает наибольшее целое число, которое не больше d;

|  |
| --- |
| double result = Math.Floor(2.56); // 2 |

* IEEERemainder(double a, double b): возвращает остаток от деления a на b;

|  |
| --- |
| double result = Math.IEEERemainder(26, 4); // 2 = 26-24 |

* Log(double d): возвращает натуральный логарифм числа d;
* Log(double a, double newBase): возвращает логарифм числа a по основанию newBase;
* Log10(double d): возвращает десятичный логарифм числа d;
* Max(double a, double b): возвращает максимальное число из a и b;
* Min(double a, double b): возвращает минимальное число из a и b;
* Pow(double a, double b): возвращает число a, возведенное в степень b;
* Round(double d): возвращает число d, округленное до ближайшего целого числа;

|  |
| --- |
| double result1 = Math.Round(20.56); // 21  double result2 = Math.Round(20.46); //20 |

* Round(double a, int b): возвращает число a, округленное до определенного количества знаков после запятой, представленного параметром b;

|  |
| --- |
| double result1 = Math.Round(20.567, 2); // 20,57  double result2 = Math.Round(20.463, 1); //20,5 |

* Sign(double value): возвращает число 1, если число value положительное, и -1, если значение value отрицательное. Если value равно 0, то возвращает 0;

|  |
| --- |
| int result1 = Math.Sign(15); // 1  int result2 = Math.Sign(-5); //-1 |

* Sin(double value): возвращает синус угла value;
* Sinh(double value): возвращает гиперболический синус угла value;
* Sqrt(double value): возвращает квадратный корень числа value;

|  |
| --- |
| * double result1 = Math.Sqrt(16); // 4 |

* Tan(double value): возвращает тангенс угла value;
* Tanh(double value): возвращает гиперболический тангенс угла value;
* Truncate(double value): отбрасывает дробную часть числа value, возвращая лишь целое значение.

|  |
| --- |
| double result = Math.Truncate(16.89); // 16 |

В классе String есть методы для работы со строками:

* *Compare*: сравнивает две строки с учетом текущей культуры (локали) пользователя;
* *CompareOrdinal*: сравнивает две строки без учета локали;
* *Contains*: определяет, содержится ли подстрока в строке;
* *Concat*: соединяет строки;
* *CopyTo*: копирует часть строки, начиная с определенного индекса в массив;
* *EndsWith*: определяет, совпадает ли конец строки с подстрокой;
* *Format*: форматирует строку;
* *IndexOf*: находит индекс первого вхождения символа или подстроки в строке;
* *Insert*: вставляет в строку подстроку;
* *Join*: соединяет элементы массива строк;
* *LastIndexOf*: находит индекс последнего вхождения символа или подстроки в строке;
* *Replace*: замещает в строке символ или подстроку другим символом или подстрокой;
* *Split*: разделяет одну строку на массив строк;
* *Substring*: извлекает из строки подстроку, начиная с указанной позиции;
* *ToLower*: переводит все символы строки в нижний регистр;
* *ToUpper*: переводит все символы строки в верхний регистр;
* *Trim*: удаляет начальные и конечные пробелы из строки.

Для того чтобы преобразовать значение, например, из bool в double, можно воспользоваться классом Convert. Convert содержит следующие методы:

* ToBoolean(value);
* ToByte(value);
* ToChar(value);
* ToDateTime(value);
* ToDecimal(value);
* ToDouble(value);
* ToInt16(value);
* ToInt32(value);
* ToInt64(value);
* ToSByte(value);
* ToSingle(value);
* ToUInt16(value);
* ToUInt32(value);
* ToUInt64(value).

Пример:

public void Method4()

{

bool b = true;

double d = Convert.ToDouble(b);

}

*Средства отладки.*

Как правило, в средах разработки, помимо простого запуска программы, можно поставить точку останова, чтобы посмотреть и проанализировать поведение программы и данные, с которыми она работает. Установленная точка останова приведена на рисунке 2.12.

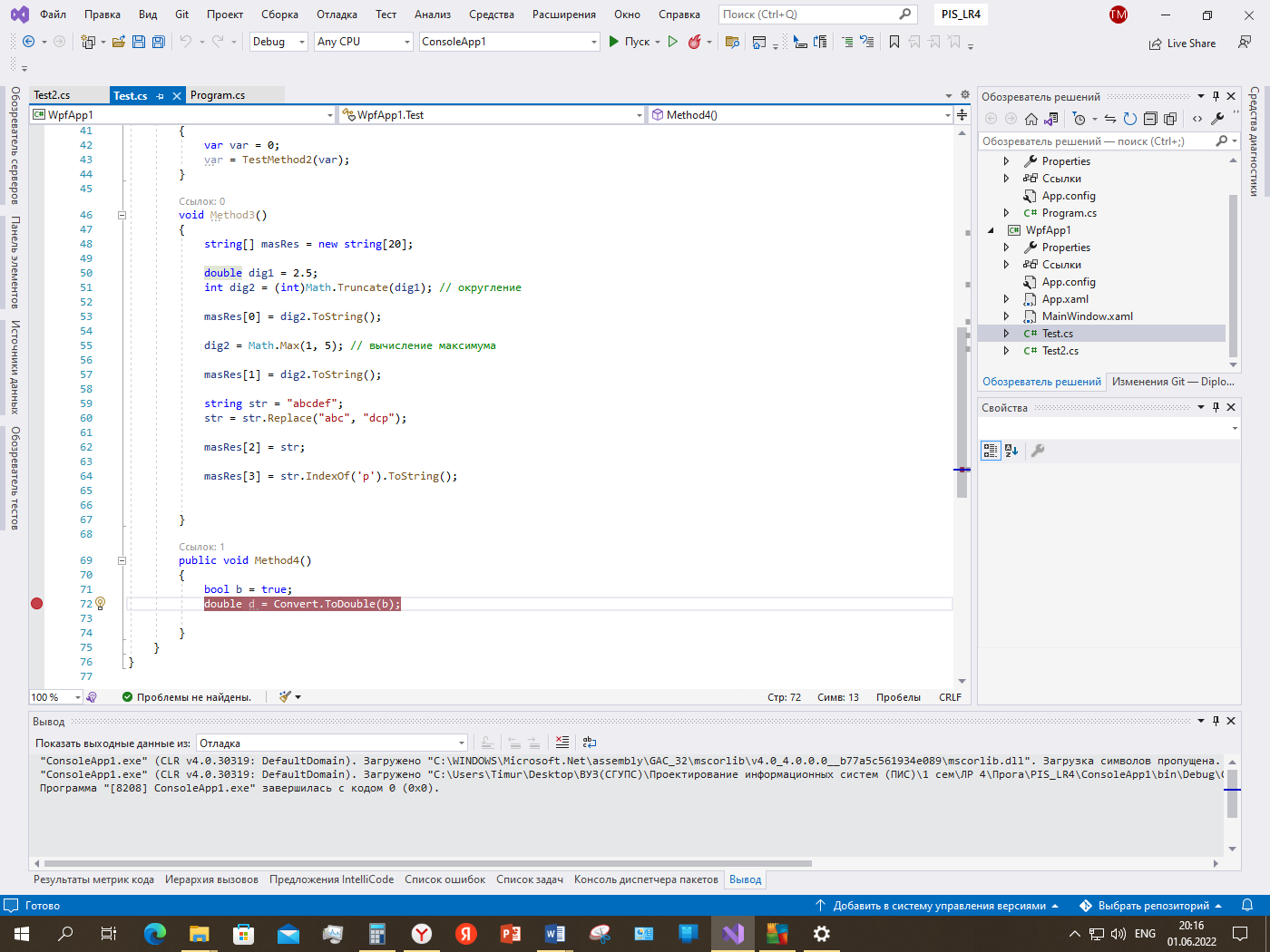


Рисунок 2.12 – Установленная точка останова

В Visual Studio 2019 точку останова можно выключить (не путать с убрать), если в настоящий момент она нам не нужна, но есть вероятность того, что может понадобиться в дальнейшем.

При отладке программы снизу будет доступна панель «Контрольные значения». В неё мы можем добавить переменную, поведение которой нас интересует. Также рядом с этой панелью, будут доступны панели «Видимые» и «Локальные». По панели «Локальные» также можно следить за поведением программы. Она показывает значения тех переменных, в методе которых находится выполнение программы (жёлтая стрелка). Процесс отладки представлен на рисунке 2.13.

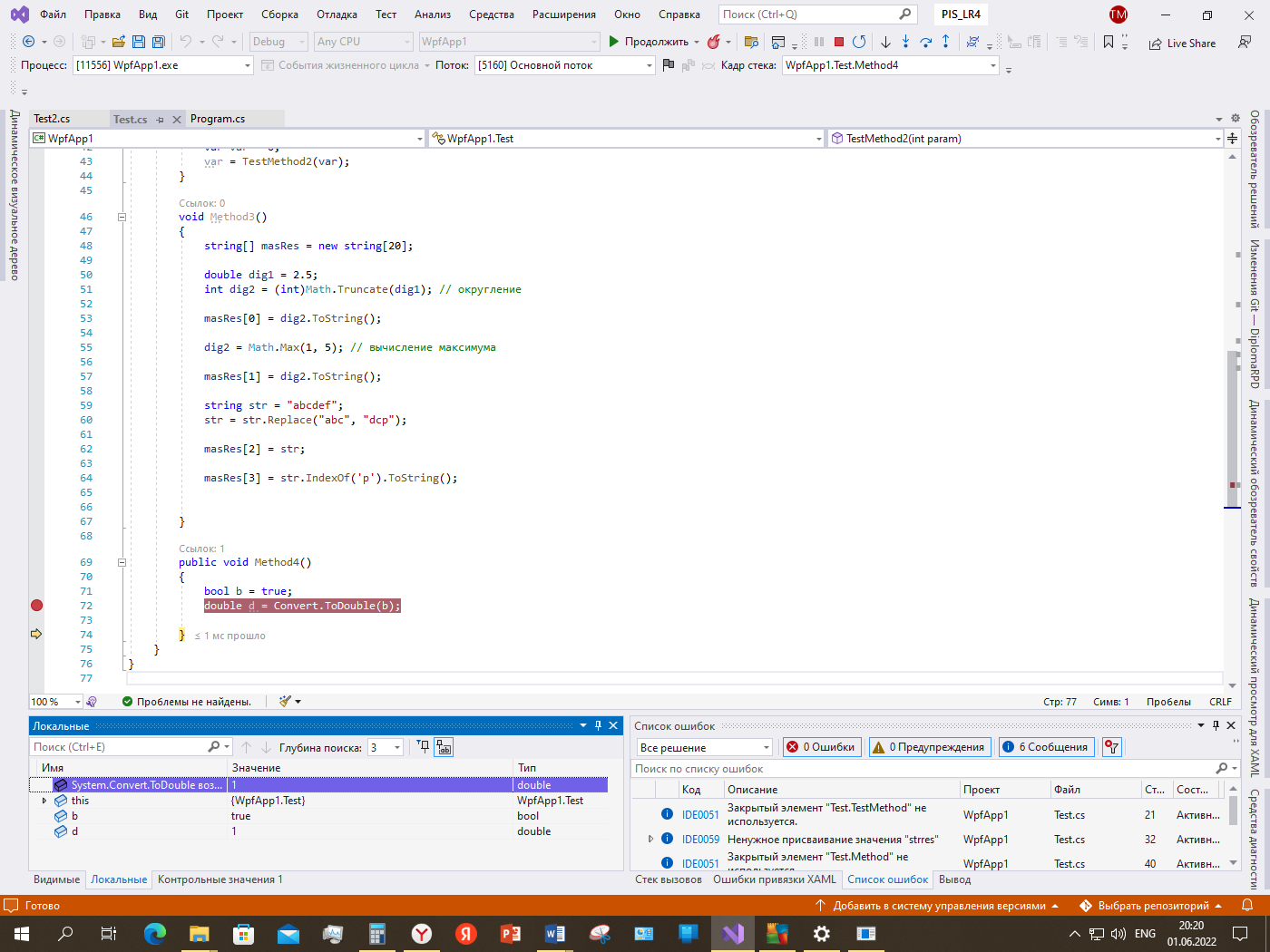


Рисунок 2.13 – Процесс отладки

Также можно узнать значение переменной, наведя на неё курсор, рисунок 2.14.

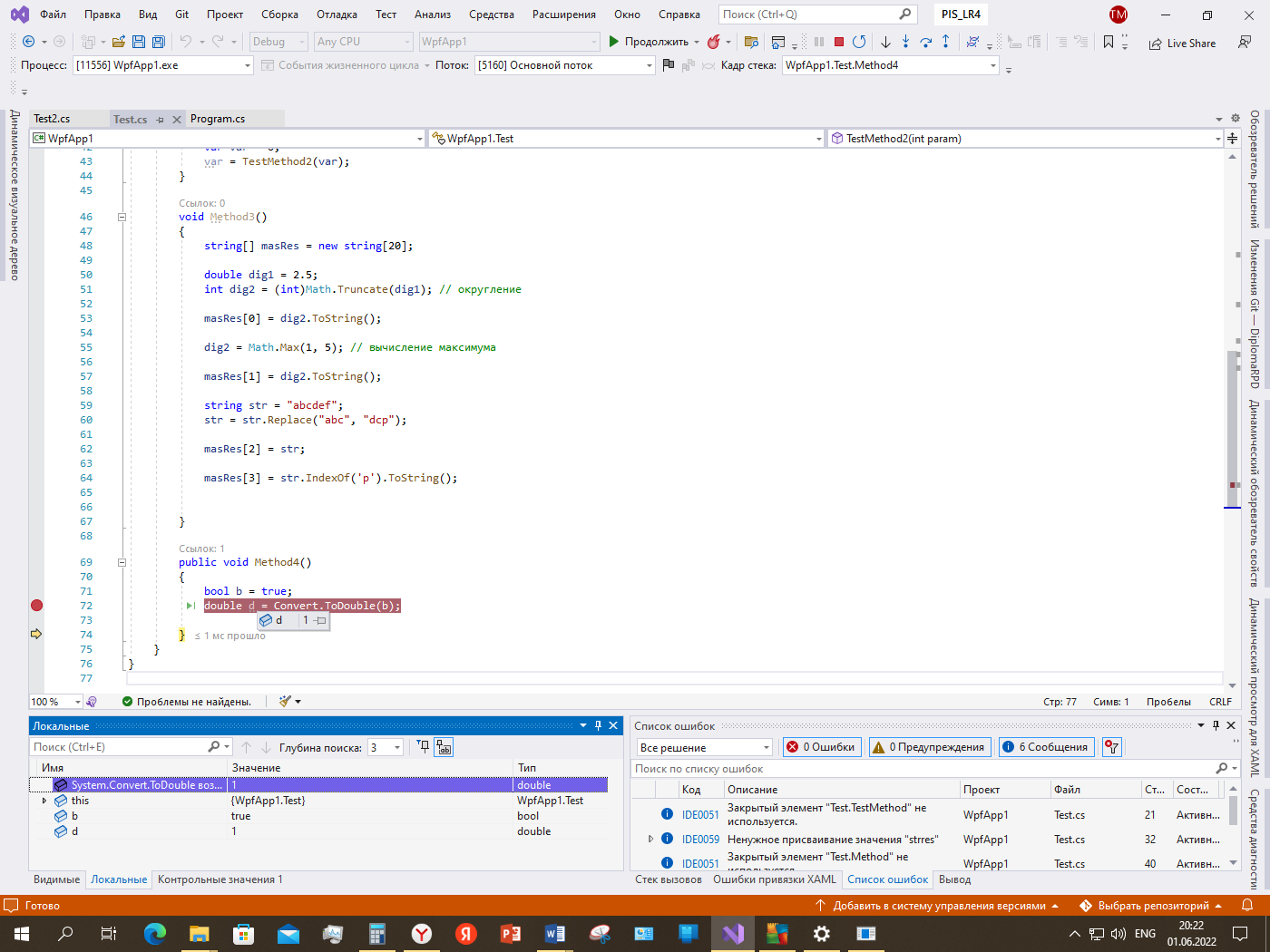


Рисунок 2.14 – Значение переменной

Эти инструменты отладки очень полезны при поиске причин ошибок в работе программы.

## **Среда разработки**

*Выбор среды разработки.*

С учётом выбранного языка программирования и платформы, было принято решение разрабатывать приложение на интегрированной среде разработки Microsoft Visual Studio.

Visual Studio (VS) – это лучшая комплексная среда IDE для разработчиков .NET в Windows. Полноценный набор инструментов и функций для улучшения и усовершенствования каждого этапа разработки программного обеспечения.

Будет использоваться Community версия VS, потому что она бесплатная.

Интерфейс данной IDE представлен на рисунке 2.15.

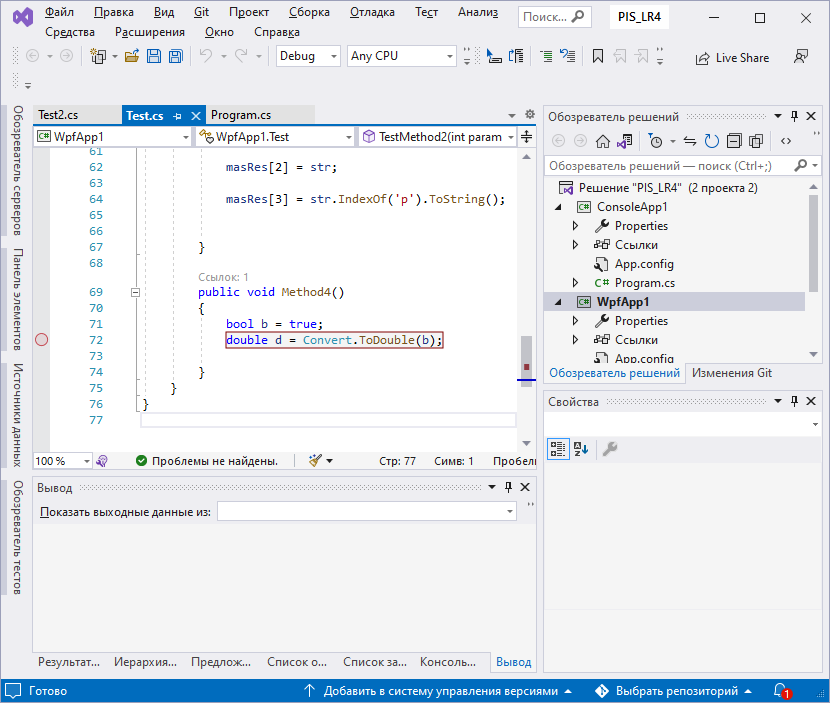


Рисунок 2.15 – Интерфейс Visual Studio 2022

*Обоснование выбора среды разработки.*

Рассмотрим другие среды разработки, которые также можно было бы использовать для разработки.

Самый хороший заменитель VS, SharpDevelop (SD) – свободная среда разработки для C#, Visual Basic .NET, Boo, IronPython, IronRuby, F#, C++. Обычно используется как альтернатива Visual Studio .NET. Это очень хорошая среда разработки для своего времени. Здесь есть визуальный редактор интерфейса, автозавершение текста, подсветка синтаксиса, шаблоны формирования файлов программных модулей (последние два, сугубо по мнению разработчика, даже лучше, чем в VS). Также возможности среды можно расширить дополнительным ПО. Однако эту IDE перестали обновлять с 2016 года. Поэтому изначально на ней, например, нельзя разрабатывать приложения более поздние, чем на .NET Framework 4.5.1. Для этого нужно вручную устанавливать пакеты платформ. Также у SD нет инсталлятора тех самых пакетов и прочих дополнений, что есть у VS.

Сравнение VS с другими IDE представлена в таблице 2.2 [11, 12, 13].

Таблица 2.2 – Сравнение IDE для C#

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| [IDE](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8) | Visual Studio | SharpDevelop | Geany | MonoDevelop |
| [Windows](https://ru.wikipedia.org/wiki/Microsoft_Windows) | + | + | + | + |
| Транслятор | + | + | - | + |
| [Отладчик](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BB%D0%B0%D0%B4%D1%87%D0%B8%D0%BA) | + | + | - | + |
| [Разработка UI](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F) | + | + | - | + |
| [Покрытие кода](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BA%D1%80%D1%8B%D1%82%D0%B8%D0%B5_%D0%BA%D0%BE%D0%B4%D0%B0) | + | + | - | + |
| [Автодополнение](https://ru.wikipedia.org/wiki/%D0%90%D0%B2%D1%82%D0%BE%D0%B4%D0%BE%D0%BF%D0%BE%D0%BB%D0%BD%D0%B5%D0%BD%D0%B8%D0%B5) | + | + | + | + |
| [Браузер классов](https://ru.wikipedia.org/w/index.php?title=%D0%91%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80_%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%BE%D0%B2&action=edit&redlink=1) | + | + | + | + |
| Подсветка синтаксиса | + | + | + | + |

Продолжение таблицы 2.2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| [IDE](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8) | Visual Studio | SharpDevelop | Geany | MonoDevelop |
| Поддержка дополнений | + | + | + | + |
| Менеджер дополнений | + | - | - | - |
| Сворачивание кода | + | + | + | + |
| Поддержка разработчиков в настоящее время | + | - | + | - |

Из приведённой выше таблицы, также можно сделать вывод, что VS является наиболее оптимальной IDE для разработки ПО (по крайней мере, в данном случае).

## **Система управления базами данных (СУБД)**

SQLite имеет встраиваемую СУБД, то есть не требует отдельной установки.

Для разработчиков, использующих SQLite, существует утилита DB Browser for SQLite, с помощью которой можно самому проводить манипуляции над БД [14, 15].

Интерфейс данной СУБД представлен на рисунке 2.16.

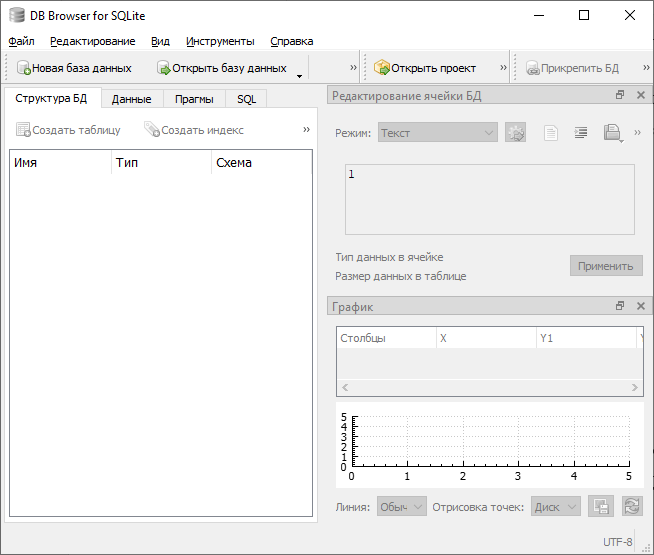


Рисунок 2.16 – Интерфейс DB Browser for SQLite

## **Модель данных (ER-модель)**

В ходе анализа предметной области была построена ER-диаграмма будущей БД.

ER-диаграмма БД представлена на рисунке 2.17.

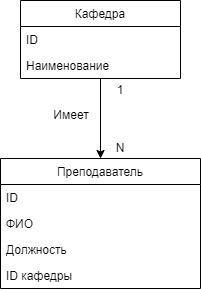


Рисунок 2.17 – ER-модель БД

На диаграмме присутствует всего одна связь: преподаватель закреплён на конкретной кафедре.

## **Средства и методы тестирования**

В качестве метода тестирования было выбрано модульное тестирование. Тесты будут писаться после написания кода, это называется тестирование белого ящика, когда мы знаем всё о том, что тестируем.

В качестве средств тестирования будут использоваться специальные проекты с юнит-тестами, которые будут покрывать код самих приложений.

1. **Выводы по второму разделу**

По итогу второго раздела была спроектирована информационная система. Были смоделированы бизнес-процессы предметной области. Был сделан и обоснован выбор типа БД информационной системы.

Рассмотрен язык программирования, который будет применяться для разработки, и его особенности. Помимо этого, был сделан и обоснован выбор среды разработки, в которой будут разрабатываться приложения. Была выбрана и рассмотрена СУБД для разработки и использования базы данных.

После этого была спроектирована ER-диаграмма, которая представляет сущности и их связи в базе данных.

Также были выбраны средства и методы тестирования.

1. **Создание приложения**
2. **Развёртывание, установка, настройка генератора**

Установка программы происходит через её установщик. Нужно запустить его и указать такие данные, как: расположение, где будет находиться программа; название папки программы в меню «Пуск»; необходимость создания ярлыков.

После вышеописанных действий, программа сможет полноценно функционировать и взаимодействовать с БД.

1. **Описание генератора**

Внешний вид генератора представлен на рисунках 3.1-3.4.

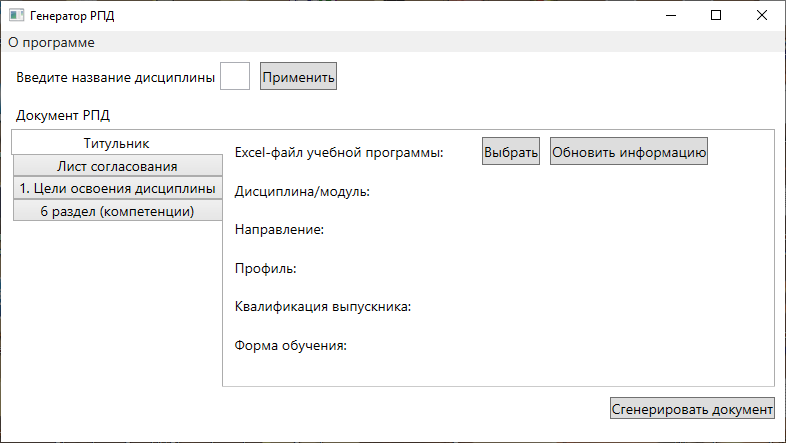


Рисунок 3.1 – Внешний вид генератора

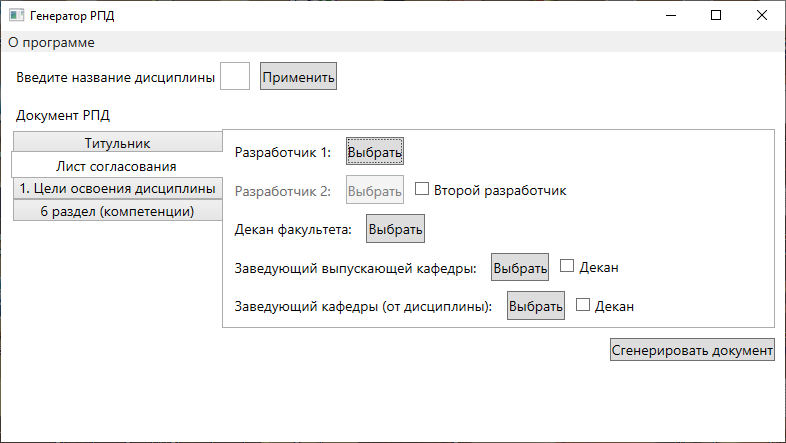


Рисунок 3.2 – Внешний вид генератора

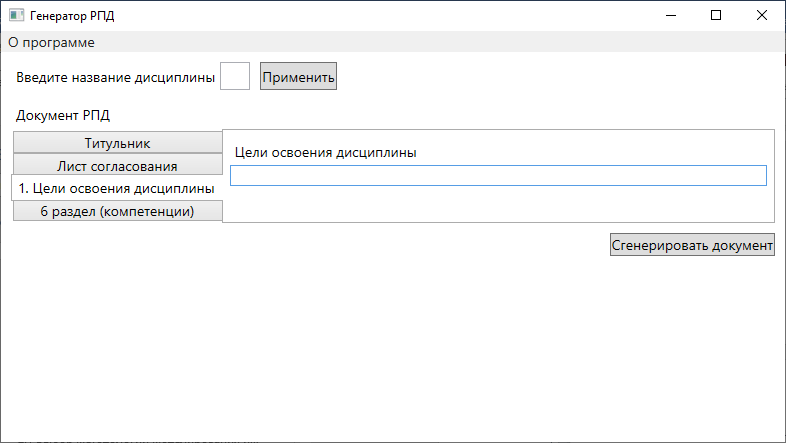


Рисунок 3.3 – Внешний вид генератора

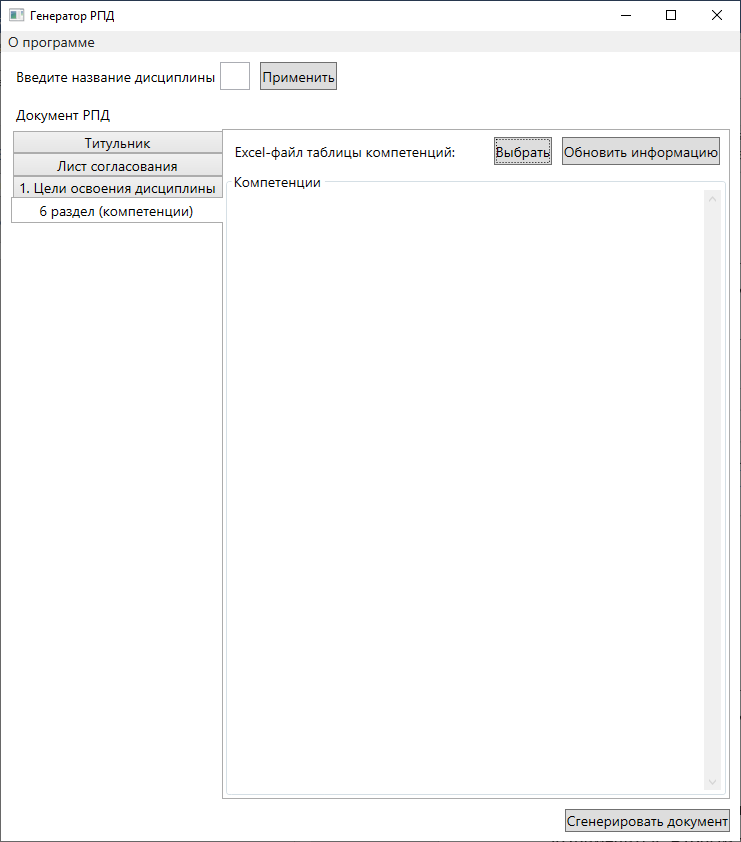


Рисунок 3.4 – Внешний вид генератора

Сначала сверху нужно ввести название дисциплины и нажать кнопку «Применить». Вторым шагом будет выбор файла УП с помощью стандартного диалогового окна OpenFileDialog. Диалоговое окно OpenFileDialog представлено на рисунке 3.5.

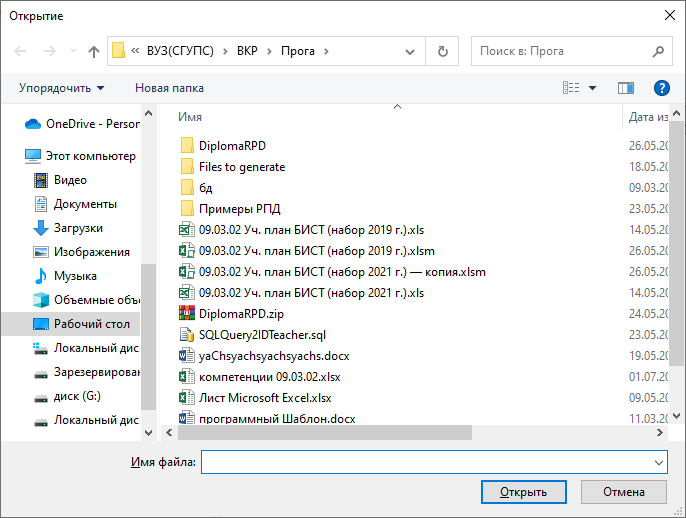


Рисунок 3.5 – Диалоговое окно выбора файла

Третьим шагом нужно выбрать разработчиков РПД и заведующих кафедр (выпускающей и от дисциплины) во вкладке «Лист согласования», в отдельном окне. Окно выбора разработчика представлено на рисунке 3.6.

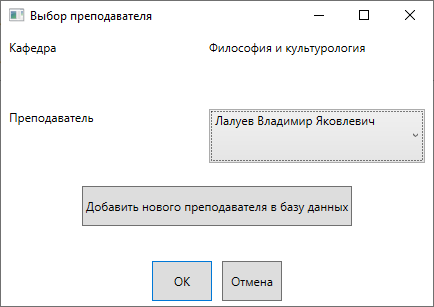


Рисунок 3.6 – Окно выбора преподавателя

Четвёртым шагом нужно ввести данные в раздел «Цели освоения дисциплины». Пятым предпоследним шагом нужно выбрать файл компетенций и их индикаторов из того же стандартного диалогового окна выбора файлов, рисунок 3.5, после чего заполнить результаты обучения по каждому индикатору. Последним шестым шагом будет нажатие на кнопку «Сгенерировать документ» и выбор расположения, где будет сохранён сгенерированный документ РПД, с помощью диалогового окна выбора расположения. Окно выбора расположения сгенерированного документа представлено на рисунке 3.7.

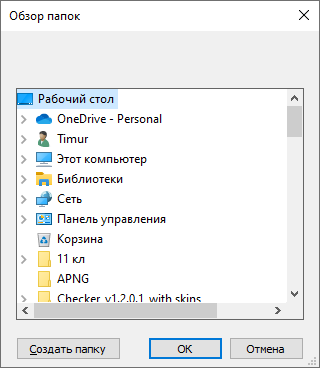


Рисунок 3.7 – Окно выбора расположения

В дальнейшем, если нужно сформировать документ дисциплины из той же специальности, то во втором и пятом шагах вместо выбора одного и того же файла, можно будет просто обновить данные при помощи кнопки «Обновить информацию».

При генерации документа он сохраняется в файл формата Word (.docx), после этого об успешности результата генерации будет уведомлён пользователь. Так что, после генерации документа РПД, его можно будет отредактировать непосредственно в Word.

1. **Руководство администратора (программиста)**

*Назначение и условия применения программы.*

Программное обеспечение «Генератор документов РПД» предназначено для автоматизации процесса формирования документов рабочих программ дисциплин.

СУБД SQLite в составе данной информационной системы позволяет хранить данные преподавателей и кафедр, и обеспечивать их загрузку при выборе разработчиков РПД.

Система включает в себя следующие функциональные возможности:

* генерация документов РПД;
* добавление, удаление и редактирование данных преподавателя;
* добавление, удаление и редактирование данных кафедры.

Для функционирования системы необходимо техническое обеспечение со следующими характеристиками.

Минимальные требования:

* процессор с тактовой частотой 1 ГГц с архитектурой x86;
* оперативная память с 4 ГБ (6 ГБ для Windows 10);
* жесткий диск с 50 МБ свободного места;
* видеокарта со 1 ГБ видеопамяти;
* монитор типа LED;
* операционная система – Windows 7.

Рекомендуемые требования:

* процессор с тактовой частотой 2 ГГц и выше с архитектурой x86;
* оперативная память с 6 ГБ и больше;
* жесткий диск с 100 МБ свободного места и больше;
* видеокарта со 2 ГБ видеопамяти;
* монитор типа LED;
* операционная система – Windows 10.

Перечень программных средств, необходимых для дальнейшего редактирования документов после генерации:

* Microsoft Word – текстовый процессор, предназначенный для создания, просмотра, редактирования и форматирования текстов статей, деловых бумаг, а также иных документов, с локальным применением простейших форм таблично-матричных алгоритмов (входит в состав Microsoft Office);
* Microsoft Excel – программа для работы с электронными таблицами, предоставляющая возможности экономико-статистических расчетов и графические инструменты (входит в состав Microsoft Office).

*Характеристики программы.*

Система имеет диалоговый режим функционирования (взаимодействие с приложением через интерфейс).

Количество одновременно работающих в системе пользователей: 1 человек.

Время реакции системы на действия пользователя не превышает 10 секунд.

Для обеспечения надежности системы используется:

* контроль входной и выходной информации;
* обработка ошибок, возникающих в ходе работы системы.

*Обращение к программе.*

Для запуска системы необходимо открыть .exe файл самой программы (через ярлык или напрямую). После запуска приложения, на экране отображается главное окно, в котором есть разделы для заполнения данных будущего документа РПД. Сверху имеется меню, в котором можно посмотреть информацию о программе.

Выход из приложения возможен при нажатии на кнопку «Закрыть», ещё можно воспользоваться сочетанием клавиш Alt+F4.

*Входные и выходные данные.*

Входные данные:

Состав данных, которые вводит пользователь:

* наименование дисциплины;
* выбранный файл УП;
* лист согласования (выбор первого разработчика, второй разработчика, декана факультета, заведующего выпускающей кафедры, заведующего кафедры от дисциплины);
* цели освоения дисциплины;
* выбранный файл компетенций;
* результаты обучения по каждому индикатору.

Выходными данными является сгенерированный документ рабочей программы дисциплины.

*Сообщения.*

Сообщение «Дисциплина не может быть пустой.». Попытка выбора файла УП или файла компетенций без указанного названия дисциплины. Необходимо ввести название дисциплины и нажать кнопку «Принять».

Сообщение «Дисциплина не найдена.». Дисциплина с текущем названием не найдена. Необходимо либо ввести правильное название дисциплины, либо проверить файлы УП и файл компетенции на наличие текущей дисциплины.

Сообщение «Сначала нужно выбрать файл учебного плана.». Попытка обновить информацию, либо выбрать разработчика РПД, когда файл УП не выбран. Необходимо выбрать файл УП.

Сообщение «Сначала нужно выбрать файл таблицы компетенций.». Попытка обновить информацию, когда файл таблицы компетенций не выбран. Необходимо выбрать файл таблицы компетенций.

Сообщение «Документ сгенерирован». Подтверждение успешного окончания генерации файла документа РПД.

1. **Руководство пользователя (ответственного за ОП)**

*Назначение программы.*

Программное обеспечение «Генератор документов РПД» предназначено для автоматизации процесса формирования документов рабочих программ дисциплин.

СУБД SQLite в составе данной информационной системы позволяет хранить данные преподавателей и кафедр, и обеспечивать их загрузку при выборе разработчиков РПД.

Система включает в себя следующие функциональные возможности:

* генерация документов РПД;
* добавление, удаление и редактирование данных преподавателя;
* добавление, удаление и редактирование данных кафедры.

*Условия выполнения программы.*

Для дальнейшей работы с документами после генерации необходимо следующее программное обеспечение:

* Microsoft Word (входит в состав Microsoft Office);
* Microsoft Excel (входит в состав Microsoft Office).

Работа с ПО «Генератор документов РПД» доступна любым пользователям, работающим за рабочими местами.

Пользователь ПО «Генератор документов РПД» должен иметь опыт работы с ОС Microsoft Windows, навык работы с Microsoft Word и Microsoft Excel, а также обладать следующими знаниями:

* знать соответствующую предметную область;
* знать и иметь навыки работы с аналитическими приложениями.

*Выполнение программы.*

Для запуска системы необходимо открыть .exe файл самой программы (через ярлык или напрямую). После запуска приложения, на экране отображается главное окно, в котором есть разделы для заполнения данных будущего документа РПД. Сверху имеется меню, в котором можно посмотреть информацию о программе.

Выход из приложения возможен при нажатии на кнопку «Закрыть», а также можно воспользоваться сочетанием клавиш Alt+F4.

Работа с запущенной системой состоит из выполнения следующих этапов:

1. Ввести название дисциплины и нажать кнопку «Применить».
2. Выбрать файл учебного плана.
3. Выбрать разработчиков рабочей программы.
4. Заполнить раздел «Цели освоения дисциплины».
5. Выбрать файл компетенций и заполнить результаты обучения по каждому индикатору.
6. Нажать кнопку «Сгенерировать документ» и выбрать расположение для сохранения документа.

*Сообщения оператору.*

Сообщение «Дисциплина не может быть пустой.». Попытка выбора файла УП или файла компетенций без указанного названия дисциплины. Необходимо ввести название дисциплины и нажать кнопку «Принять».

Сообщение «Дисциплина не найдена.». Дисциплина с текущем названием не найдена. Необходимо либо ввести правильное название дисциплины, либо проверить файлы УП и файл компетенции на наличие текущей дисциплины.

Сообщение «Сначала нужно выбрать файл учебного плана.». Попытка обновить информацию, либо выбрать разработчика РПД, когда файл УП не выбран. Необходимо выбрать файл УП.

Сообщение «Сначала нужно выбрать файл таблицы компетенций.». Попытка обновить информацию, когда файл таблицы компетенций не выбран. Необходимо выбрать файл таблицы компетенций.

Сообщение «Документ сгенерирован». Подтверждение успешного окончания генерации файла документа РПД.

1. **Условия обеспечения работоспособности генератора**

*Численность и квалификация персонала системы.*

Для работы системы необходим пользователь, работающий в системе.

*Информационное обеспечение системы.*

Состав данных по разделам, которые вводит пользователь:

* наименование дисциплины;
* выбранный файл УП;
* лист согласования (выбор первого разработчика, второй разработчика, декана факультета, заведующего выпускающей кафедры, заведующего кафедры от дисциплины);
* цели освоения дисциплины;
* выбранный файл компетенций;
* результаты обучения по каждому индикатору.

*Программное обеспечение системы.*

Перечень программных средств, необходимых для дальнейшего редактирования документов после генерации:

* Microsoft Word – текстовый процессор, предназначенный для создания, просмотра, редактирования и форматирования текстов статей, деловых бумаг, а также иных документов, с локальным применением простейших форм таблично-матричных алгоритмов (входит в состав Microsoft Office);
* Microsoft Excel – программа для работы с электронными таблицами, предоставляющая возможности экономико-статистических расчетов и графические инструмент (входит в состав Microsoft Office).

*Техническое обеспечение системы.*

Для функционирования системы необходимо техническое обеспечение со следующими характеристиками.

Минимальные требования:

* процессор с тактовой частотой 1 ГГц с архитектурой x86;
* оперативная память с 4 ГБ (6 ГБ для Windows 10);
* жесткий диск с 50 МБ свободного места;
* видеокарта со 1 ГБ видеопамяти;
* монитор типа LED;
* операционная система – Windows 7.

1. **Выводы по третьему разделу**

По итогу, в третьем разделе была описана реализация генератора документов РПД. Были рассмотрены развёртывание, установка и настройка генератора. Были описаны функциональные особенности генератора. Также были разработаны руководства администратора и пользователя. В заключении были рассмотрены условия обеспечения работоспособности генератора.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы было разработано приложение, которое формирует документы рабочих программ дисциплин.

Также были решены следующие задачи:

* проведено исследование предметной области;
* проведено моделирование бизнес-процессов предметной области;
* выбраны современные инструментальные средства для проектирования и разработки программного обеспечения;
* проведена разработка, тестирование и апробация приложения;
* в соответствии с нормативными документами подготовлена программная документация на информационную систему.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Разработка, согласование и утверждение РПД (февраль 2016) [Электронный ресурс] — URL: https://spb.hse.ru/data/2016/03/25/1127998395/Разработка, согласование и утверждение РПД (февраль 2016).pdf (дата обращения 7.05.2022).
2. III.1. Примерный учебный план \ КонсультантПлюс – Википедия [Электронный ресурс] — URL: http://www.consultant.ru/document/cons\_doc\_LAW\_282289/17d054a0c965b970c64ac1f4cc1d9586fb08c533/ (дата обращения 15.05.2022).
3. Сайт «Юрайт» [Электронный ресурс] — URL: https://urait.ru/book/gosudarstvennye-finansy-469965 (дата обращения 8.05.2022).
4. Сайт ТУСУРа [Электронный ресурс] — URL: https://workprogram.tusur.ru/docs#workprogram-developers (дата обращения 8.05.2022).
5. Малышев Е.Н. Проектирование информационных систем (Раздел 5. Индустриальное проектирование информационных систем. Объектно-ориентированная Case-технология проектирования информационных систем): учебное пособие / Е.Н. Малышев – Кемерово: КемГУ, 2009. 70 с (дата обращения 11.05.2022).
6. [Стасышин В.М.](http://biblioclub.ru/index.php?page=author&id=78209) Проектирование информационных систем и баз данных: учебное пособие / В.М. [Стасышин](http://biblioclub.ru/index.php?page=author&id=78209) - Новосибирск: [НГТУ](http://biblioclub.ru/index.php?page=publisher&pub_id=4951), 2012 . –100 c. - ISBN: 978-5-7782-2121-5 (дата обращения 11.05.2022).
7. [Золотов С.Ю.](http://biblioclub.ru/index.php?page=author&id=53080) Проектирование информационных систем: учебное пособие С.Ю. [Золотов](http://biblioclub.ru/index.php?page=author&id=53080) - Томск: [Эль Контент](http://biblioclub.ru/index.php?page=publisher&pub_id=14982), 2013 – 88 с. ISBN: 978-5-4332-0083-8 (дата обращения 11.05.2022).
8. Подборка статей «Анализ и проектирование систем». [Электронный ресурс]. - URL: https://habrahabr.ru/hub/analysis\_design/ (дата обращения 11.05.2022).
9. Подборка статей «Проектирование и рефакторинг». [Электронный ресурс]. - URL: https://habrahabr.ru/hub/refactoring/ (дата обращения 11.05.2022).
10. Куликов, С. Тестирование программного обеспечения. Базовый курс. EPAM Systems, 2017. – 296 с. : ил (дата обращения 11.05.2022).
11. Сравнение интегрированных сред разработки [Электронный ресурс] — URL: https://ru.wikipedia.org/wiki/Сравнение\_интегрированных\_сред\_разработки#C# (дата обращения 11.05.2022).
12. Geany [Электронный ресурс] — URL: https://ru.wikipedia.org/wiki/Geany (дата обращения 11.05.2022).
13. MonoDevelop [Электронный ресурс] — URL: https://ru.wikipedia.org/wiki/MonoDevelop (дата обращения 11.05.2022).
14. [Пржиялковский В.В.](http://biblioclub.ru/index.php?page=author&id=81762) Введение в Oracle SQL В.В. [Пржиялковский](http://biblioclub.ru/index.php?page=author&id=81762) - М.: [Интернет-Университет Информационных Технологий](http://biblioclub.ru/index.php?page=publisher&pub_id=16091), 2011 – 254 с (дата обращения 11.05.2022).
15. [Букатов А.А.](http://biblioclub.ru/index.php?page=author&id=83996) Методы и средства интеграции независимых баз данных в распределенных телекоммуникационных сетях: монография - А.А. [Букатов](http://biblioclub.ru/index.php?page=author&id=83996) , А.В. [Пыхалов](http://biblioclub.ru/index.php?page=author&id=84149) - Ростов-н/Д: [Издательство Южного федерального университета](http://biblioclub.ru/index.php?page=publisher&pub_id=16475), 2013 – 160 с. - ISBN: 978-5-9275-1189-1 (дата обращения 11.05.2022).