Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное образовательное учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ

УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра компьютерных систем в управлении и проектировании (КСУП)

**РАЗРАБОТКА ПЛАГИНА «ОТВЁРТКА» ДЛЯ «КОМПАС-3D»**

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

по дисциплине

«Основы разработки САПР» (ОРСАПР)

Выполнил:

студент гр. 581

\_\_\_\_\_\_\_\_\_ Мирошников А.В.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г.

Проверил:

к.т.н., доцент каф. КСУП \_\_\_\_\_\_\_\_\_\_ Калентьев А.А.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024

**РЕФЕРАТ**

Пояснительная записка, 33 страницы, 25 рисунков, 12 таблиц, 12 источников.

Ключевые слова: САПР, КОМПАС-3D, Плагин для САПР, Плагин отвёртки, C#, Windows Forms.

Объектом исследования являются технологии разработки плагинов для САПР.

Предметом исследования является применение технологий разработки плагинов, для автоматизации построения отвёрток разных размеров и параметров в САПР КОМПАС-3D.

Цель работы: создание программы для автоматизации построения отвёртки в САПР КОМПАС-3D.

Для создания использовались Microsoft Visual Studio 2022 (Windows Forms), .NET Framework 4.7.2, NUnit 3.14.0, NUnit3TestAdapter 3.17.0, StyleCop.Analyzers 1.1.118, StyleCop.Analyzers.Unstable 1.2.0.556, ReSharper, Fine Code Coverage, GitHub.

В результате работы было создано приложение Windows Forms, взаимодействующее с САПР КОМПАС-3D.

Областью применения являются предприятия, связанные с моделированием отвёрток.

**СОДЕРЖАНИЕ**

[1 ВВЕДЕНИЕ 4](#_Toc23763)

[2 ПОСТАНОВКА И АНАЛИЗ ЗАДАЧИ 5](#_Toc22434)

[3 ОПИСАНИЕ ПРЕДМЕТА ПРОЕКТИРОВАНИЯ 7](#_Toc31758)

[4 ВЫБОР ИНСТРУМЕНТОВ И СРЕДСТВ РЕАЛИЗАЦИИ 8](#_Toc363)

[5 НАЗНАЧЕНИЕ ПЛАГИНА 9](#_Toc17893)

[6 ОБЗОР АНАЛОГОВ 10](#_Toc13689)

[7 ОПИСАНИЕ РЕАЛИЗАЦИИ 12](#_Toc16292)

[8 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЕЙ 20](#_Toc9887)

[9 ТЕСТИРОВАНИЕ ПЛАГИНА 23](#_Toc28841)

[9.1 Функциональное тестирование 23](#_Toc10064)

[9.2 Модульное тестирование 28](#_Toc20472)

[9.3 Нагрузочное тестирование 30](#_Toc3928)

[10 ЗАКЛЮЧЕНИЕ 32](#_Toc1958)

[11 СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 33](#_Toc5152)

**1 ВВЕДЕНИЕ**

САПР – организационно-техническая система, входящая в структуру проектной организации и осуществляющая проектирование при помощи комплекса средств автоматизированного проектирования[1].

API (Application Programming Interface) — набор правил и протоколов, с помощью которых различные программные приложения могут взаимодействовать друг с другом и обмениваться данными, повышая тем самым функциональность и эффективность работы.[2]

Для разработки плагина для САПР прежде всего необходимо выбрать объект проектирования, подходящую для выбранного объекта САПР, средства разработки плагина (язык программирования и дополнительные средства разработки, выбор может быть основан на наличии API для выбранной САПР на конкретном языке).

Плагин автоматизации построения отвёртки необходим и может быть использован на предприятиях, занимающихся моделированием отвёрток, поскольку он упростит процесс моделирования и снизит нагрузку на моделлеров.

**2 ПОСТАНОВКА И АНАЛИЗ ЗАДАЧИ**

Этапы проведения работ по разработке плагина «Отвёртка» для САПР «Компас 3D» приведены в таблице 2.1.

Таблица 2.1 − Этапы проведения работ по разработке плагина «Отвёртка» для САПР «Компас 3D».

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Этап | Состав работ | Наименование документа | Обозначение | Разработано согласно | Сроки выполнения |
| 1 | Создание технического задания | Техническое задание | - | ГОСТ 34.602-2020 | Не позднее 8.10.2024 |
| 2 | Создание проекта системы | Проект системы | - | ОС ТУСУР 01-2021 | Не позднее 29.10.2024 |
| 3 | Реализация плагина | Программный код | - | RSDN Magazine #1-2004 | Не позднее 10.12.2024 |
| Документ с тремя вариантами дополнительной функциональности плагина для согласования |
| Модульные тесты |

Таблица 2.1 − Продолжение

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Этап | Состав работ | Наименование документа | Обозначение | Разработано согласно | Сроки выполнения |
| 4 | 1. Доработка плагина 2. Создание пояснительной записки | Программный код | - | 1. RSDN Magazine #1-2004 2. ОС ТУСУР 01-2021 | Не позднее 31.12.2024 |
| Модульные тесты |
| Пояснительная записка |

Главной проблемой среди всех этапов оказалась связь второго и третьего этапа. Это обусловлено малыми познаниями в сфере составления проекта системы и допущении множества неточностей, в результате на этапе реализации плагина приходилось вносить некоторые изменения, которые повлекли за собой изменения конечной версии UML-диаграммы классов. Не смотря на моё описание проблемы также известно, что изменения в конечной версии диаграммы классов по сравнению с ней же на этапе проекта системы не являются редкостью. Только в реальных проектах эти изменения чаще вызваны желаниями и потребностями заказчика, чем неопытностью формирования диаграмм разработчиками.

Помимо этого, на этапе выбора предмета проектирования были слабо изучены особенности API КОМПАС-3D. Хоть это и не привело к значительным трудностям в написании кода, данный факт привёл к более высоким затратам по времени, чем планировалось изначально. Вывод один, необходимо более детально изучать выбираемые средства для разработки, во избежание казусов, связанных с непредусмотрительностью разработчика.

# 3 ОПИСАНИЕ ПРЕДМЕТА ПРОЕКТИРОВАНИЯ

Отвёртка − ручной слесарный и столярный монтажный инструмент, предназначенный для завинчивания и отвинчивания крепёжных изделий с резьбой.[3]

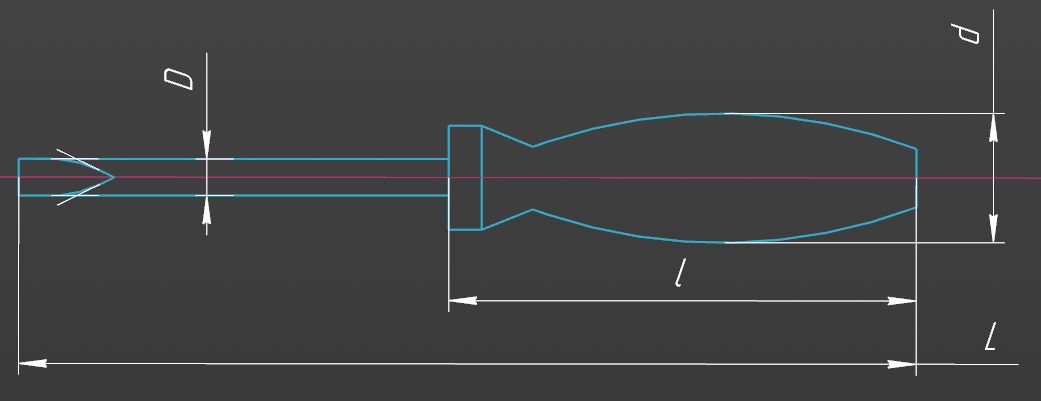


Рисунок 3.1 − Модель отвёртки

***Изменяемые параметры для предмета проектирования*** (также все обозначения показаны на рисунке 3.1):

− Длина ручки отвёртки l (45-150мм);

− Длина наконечника отвёртки L (45-500мм, но не меньше ручки);

− Диаметр наконечника отвёртки D (2/10 (длины ручки+наконечника) +/- 2 мм);

− Диаметр ручки d (1/4 длины ручки +/- 5 мм);

− Форма ручки (шестиугольная призма/цилиндрическая);

− Форма наконечника (крестообразная/плоская).

**4 ВЫБОР ИНСТРУМЕНТОВ И СРЕДСТВ РЕАЛИЗАЦИИ**

При создании плагина использовались следующие инструменты:

− WindowsForms и .NET Framework 4.7.2;

− GitHub;

− ReSharper;

− Fine Code Coverage;

− StyleCop.Analyzers 1.1.118;

− StyleCop.Analyzers.Unstable 1.2.0.556;

− NUnit 3.14.0;

− NUnit3TestAdapter 3.17.0.

Плагин был создан на технологии Windows Forms, поддерживающей широкий набор функций для разработки приложений, включая элементы управления, графику, привязку данных и ввод пользователя[4], а также .NET Framework 4.7.2, программной платформе основанной на сервероцентрической модели.

GitHub − платформа с возможностями хранения, распространения и совместной работы над написанием кода. Git − система управления версиями, которая интеллектуально отслеживает изменения в файлах.[5]

ReSharper − расширение для Microsoft Visual Studio, помогающее программировать эффективнее. Позволяет исследовать, улучшать, писать и обслуживать код.[6]

Fine Code Coverage − расширение для Microsoft Visual Studio, визуализирующий покрытие кода модульными тестами.[7]

StyleCop − средство для контроля кода, автоматически находящее синтаксические ошибки.[8]

NUnit − фреймфорк для модульного тестирования всех языков .Net.[9]

# 5 НАЗНАЧЕНИЕ ПЛАГИНА

Назначение разрабатываемого плагина обусловлено быстрым моделированием отвёрток разных видов. Благодаря данному расширению, производители отвёрток смогут наглядно рассмотреть спроектированную модель, при необходимости перестроить под необходимые им параметры.

**6 ОБЗОР АНАЛОГОВ**

Первым аналогом является приложении «Разъёмные соединения» [10] для Компас-3D, позволяющее формировать и размещать в сборке набор крепёжных элементов. Данное приложение требует оплаты дополнительной лицензии в размере 46 400 руб (+20% НДС) и позволяет создавать болтовые и винтовые соединения, а также шайбы/гайки для соединения. Данный аналог является прямым для разрабатываемого плагина «Отвёртка». Интерфейс взаимодействия представлен на рисунке 6.1.



Рисунок 6.1 − Интерфейс приложения «Разъёмные соединения»

Вторым аналогом является специализированный модуль к базовому приложению Компас-3D «Валы и механические передачи 3D. Зуборезный инструмент»[11]. Модуль позволяет рассчитать и построить модели червячных фрез для нарезания:

− цилиндрических зубчатых колес с эвольвентным профилем (черновые и чистовые фрезы);

− цилиндрических передач Новикова с двумя линиями зацепления;

− звездочек к приводным роликовым и втулочным цепям;

− червячных колес цилиндрической червячной передачи (черновые и чистовые фрезы);

− шлицевых валов с эвольвентным профилем;

− шлицевых валов с прямобочным профилем.

Лицензия является платной (216 000 руб.). Данный аналог является прямым к плагину «Отвёртка». Пользовательский интерфейс представлен на рисунке 6.2.



Рисунок 6.2 − Интерфейс приложения «Валы и механические передачи 3D. Зуборезный инструмент»

# 7 ОПИСАНИЕ РЕАЛИЗАЦИИ

UML − это стандартный язык визуального моделирования, предназначенный для следующего использования:

− моделирование бизнеса и подобных процессов;

− анализ, проектирование и внедрения программных систем.

UML − это общий язык для бизнес-аналитиков, архитекторов и разработчиков программного обеспечения, используемый для описания, спецификации, проектирования и документирования существующих или новых бизнес-процессов, структуры и поведения артефактов программных систем.[12]

UML диаграмма классов после проектирования для плагина «Отвёртка» представлена на рисунке 7.1.

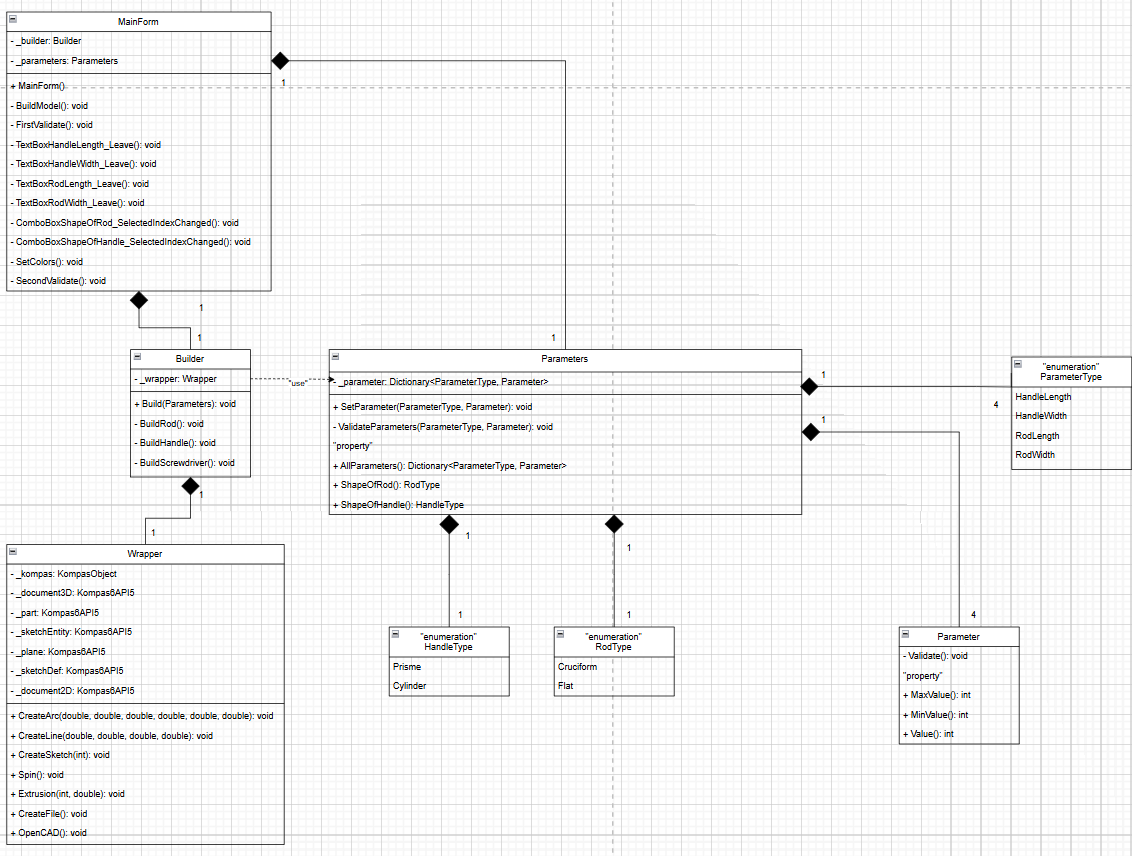


Рисунок 7.1 − UML диаграмма классов после проектирования для плагина «Отвёртка»

UML диаграмма классов после реализации плагина «Отвёртка» представлена на рисунке 7.2.

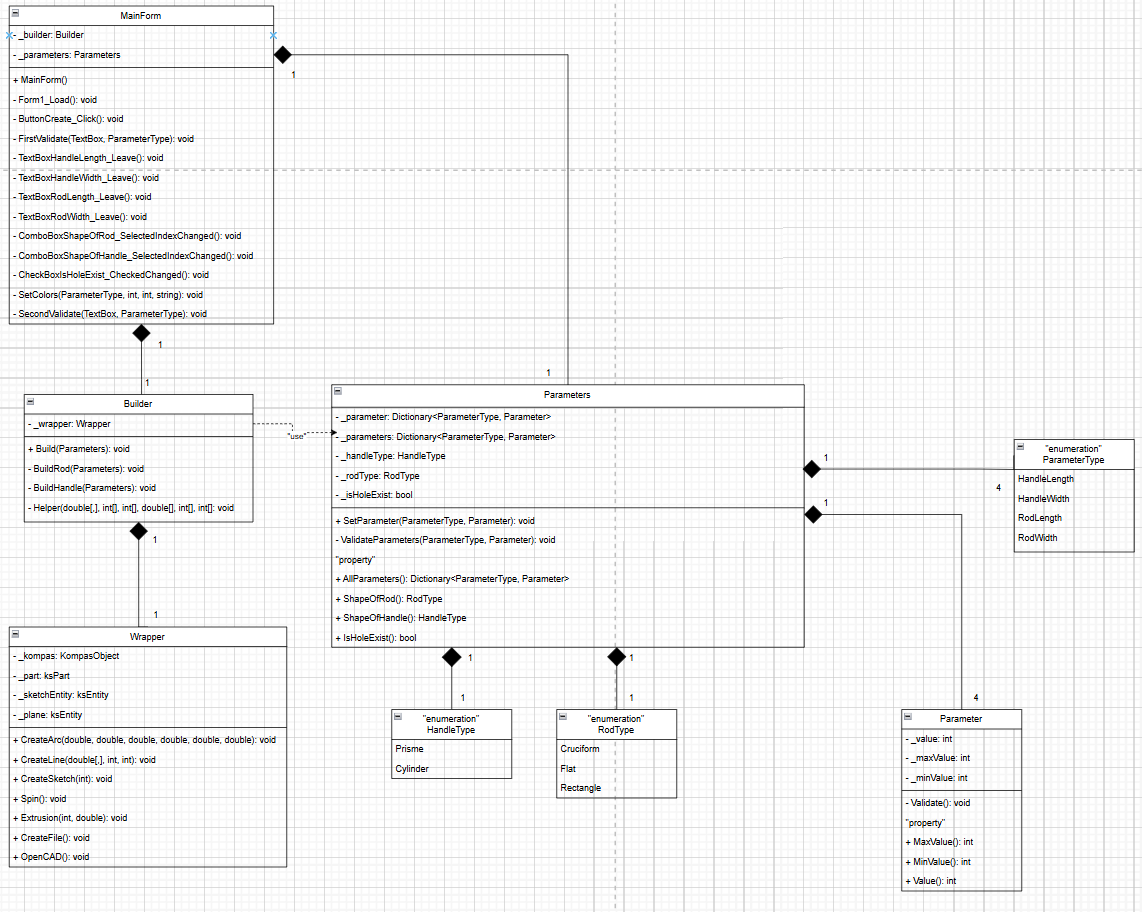


Рисунок 7.2 − UML диаграмма классов после реализации плагина «Отвёртка»

В таблицах ниже представлена информация о свойствах и методах каждого из классов.

Таблица 7.1 − Свойства класса MainForm

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_builder | Builder | Хранит в себе объект построения |
| \_parameters | Parameters | Хранит в себе параметры для объекта построения |

Таблица 7.2 − Методы класса MainForm

|  |  |  |
| --- | --- | --- |
| Название | Входные параметры | Описание |
| ButtonCreate\_Click | − | Запуск построения модели по заданным параметрам |
| MainForm | − | Конструктор MainForm |
| FirstValidate | TextBox, ParameterType | Проверка введённых данных по формату |
| TextBoxHandleLength\_Leave | − | Обработчик выхода из текстбокса длины ручки |
| TextBoxHandleWidth\_Leave | − | Обработчик выхода из текстбокса диаметра ручки |
| TextBoxRodLength\_Leave | − | Обработчик выхода из текстбокса длины наконечника |
| TextBoxRodWidth\_Leave | − | Обработчик выхода из текстбокса ширины наконечника |
| ComboBoxShapeOfRod\_SelectedIndexChanged | − | Обработчик изменения значения ComboBoxShapeOfRod |
| ComboBoxShapeOfHandle\_SelectedIndexChanged | − | Обработчик изменения значения ComboBoxShapeOfHadle |
| CheckBoxIsHoleExist\_CheckedChanged | − | Обработчик изменения CheckBoxIsHoleExist |
| SetColors | ParameterType, int, int, string | Устанавливает цвета для всех текст боксов по результатам проверки, int − выбор цвета для закраски, int − выбор причины закраски, string − передаваемый текст ошибки для установки правильного toolTip |
| SecondValidate | TextBox, ParameterType | Вызов валидации параметров |
| Form1\_Load | − | Инициализация параметров при загрузке формы |

Таблица 7.3 − Свойства класса Parameters

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_parameter | Dictionary<ParameterType, Parameter> | Хранит в себе словарь параметра |
| \_parameters | Dictionary<ParameterType, Parameter> | Хранит в себе словарь всех параметров |
| \_handleType | HandleType | Хранит в себе тип ручки (цилиндрическая/шестиугольная призма) |
| \_rodType | RodType | Хранит в себе тип наконечника (плоский/крестообразный/квадратный) |
| \_isHoleExist | bool | Хранит в себе информацию о наличии отверстия для возможности повесить ручку |

Таблица 7.4 − Методы класса Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Выходные параметры | Описание |
| ValidateParameters | − | − | Валидирует зависимые параметры |
| SetParameter | ParameterType, Parameter | − | Устанавливает параметр |
| AllParameters | − | Dictionary<ParameterType, Parameter> | Свойство для поля \_parameters |
| ShapeOfRod | − | RodType | Устанавливает и возвращает форму наконечника |
| ShapeOfHandle | − | HandleType | Устанавливает и возвращает форму ручки |
| IsHoleExist | − | bool | Устанавливает и возвращает наличие отверстия |

Таблица 7.5 − Свойства класса Builder

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_wrapper | Wrapper | Хранит в себе объект обёртки API |

Таблица 7.6 − Методы класса Builder

|  |  |  |
| --- | --- | --- |
| Название | Входные параметры | Описание |
| Build | Parameters | Построение модели по заданным параметрам |
| BuildRod | Parameters | Построение стержня отвёртки |
| BuildHandle | Parameters | Построение ручки отвёртки |
| Helper | double[,], int[], int[], double[], int[], int[] | Вспомогательный метод, позволяющий избавиться от дублирования кода (повторяет цикл операций: создать эскиз, создать линию, выдавить).  double[,] двумерный массив с координатами точек по которым строится линия, int[] тип выдавливания, int[] тип эскиза (на какой плоскости), double[] глубина выдавливания, int[] начало для сбора точек в двумерном массиве, int[] количество сборов точек (построенных линий) |

Таблица 7.7 − Свойства класса Parameter

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_maxValue | int | Максимально допустимое значение параметра |
| \_minValue | int | Минимально допустимое значение параметра |
| \_value | int | Значение параметра |

Таблица 7.8 − Методы класса Parameter

|  |  |
| --- | --- |
| Название | Описание |
| Value | Свойство для поля \_value |
| MaxValue | Свойство для поля \_maxValue |
| MinValue | Свойство для поля \_minValue |
| Validate | Сравнивает полученное значение с максимальным и минимальным возможными |

Таблица 7.9 − Свойства класса Wrapper

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_kompas | KompasObject | Поле, хранящее в себе экземпляр программы Компас |
| \_part | ksPart | Поле, хранящее в себе основную модель |
| \_sketchEntity | ksEntity | Поле, хранящее в себе текущий эскиз |
| \_plane | ksEntity | Поле, хранящее в себе текущий вид |

Таблица 7.10 − Методы класса Wrapper

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Выходные параметры | Описание |
| CreateArc | double, double, double, double, double, double | − | Создание дуги по трём точкам (double координаты x и y для каждой точки) |
| CreateLine | double[,], int, int | − | Создание линии по двум точкам, double[,] массив точек по которым строяться линии, int позиция с которой необходимо начать считывать строки массива, int количество строк |
| CreateSketch | int | − | Создание эскиза (по int выбираем базисную плоскость) |
| Spin | − | − | Вращение эскиза |
| Extrusion | int, double | − | Выдавливание эскиза (int - тип, double - глубина) |
| CreateFie | − | − | Создание файла |
| OpenCAD | − | − | Открытие Компас3D |

В отличии от диаграммы классов проекта системы диаграмма классов после реализации плагина имеет следующие отличия:

− MainForm: BuildModel заменили на ButtonCreate\_Click; FirstValidate, SecondValidate и SetColors получили входные параметры; добавлен новый метод CheckBoxIsHoleExist\_CheckedChanged;

− Parameters: получил 4 дополнительных поля − \_parameters, \_handleType, \_rodType, \_isHoleExist; ValidateParameters перестал принимать входные аргументы, а SetParameter перестал возвращать Dictionary<ParameterType, parameter>; добавлено свойство IsHoleExist;

− Builder: убрали метод BuildScredriwer и добавили метод Helper, избавляющий разработчика от циклических повторений строк в коде;

− Parameter: были переименованы поля в соответствии с RSDN, а также были созданы свойства для каждого из полей;

− Wrapper: были убраны ненужные поля \_document3D, \_document2D, \_sketchDef; было добавлено необходимое поле \_plane, а также переопределены типы у полей на ksPart у \_part и ksEntity у \_sketchEntity и \_plane; CreateLine изменены входные параметры для избавления от повторений в коде;

− RodType получил новое значение Rectangle.

Большинство из описанных изменений связаны с малым опытом подготовки проекта системы перед написанием кода и упущением деталей, кроме классов Wrapper и Builder, у которых были добавлены изменения в соответствии с объектом моделирования.

Также часть изменений связана с дополнительной функциональностью разработанной в рамках 5 лабораторной, они затронули MainForm, RodType, Parameters (на UML-диаграмме).

**8 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЕЙ**

При запуске приложения открывается форма для заполнения параметров объекта (рисунок 8.1).

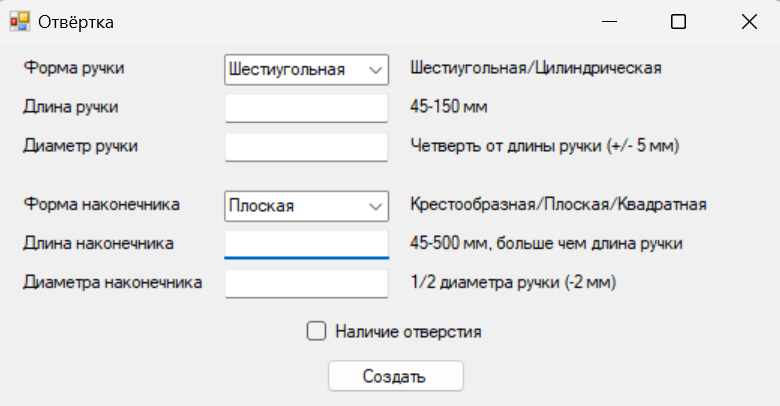


Рисунок 8.1 – Начальная форма в момент запуска приложения

При нажатии на кнопку с незаполненными или неверно заполненными полями не будет происходить ничего, при правильном же заполнении откроется КОМПАС-3D и начнётся построение модели по заданным параметрам.   
 При наведении на незаполненное поле выведется подсказка по заполнению (рисунок 8.2).

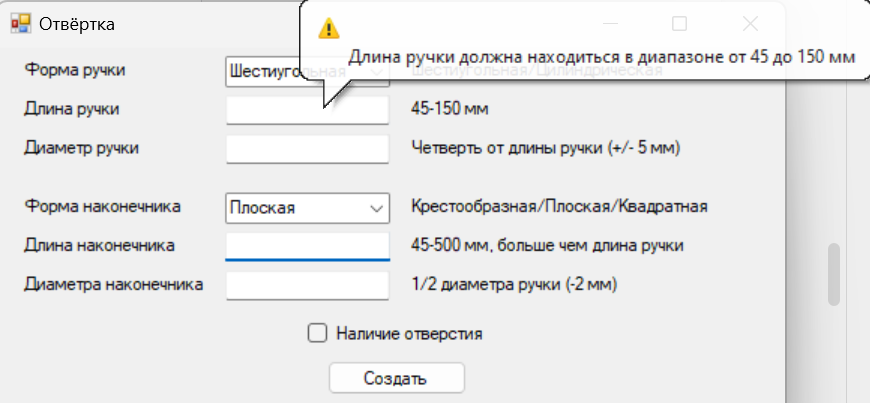


Рисунок 8.2 – Подсказка по заполнению

При неверном заполнении поля (выходе за допустимые пределы) текстбокс будет подсвечен красным, а текст подсказки останется неизменным (рисунок 8.3).

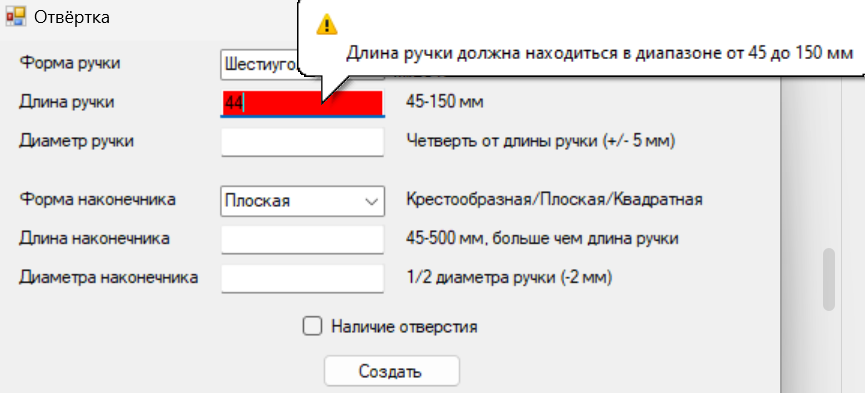


Рисунок 8.3 – Длина ручки выходит за минимальные пределы (меньше 45)

При этом, некорректно заполненные (непрошедшие собственную валидацию) или незаполненные поля не будут мешать возможно корректному заполнению зависимых от них полей (рисунок 8.4).

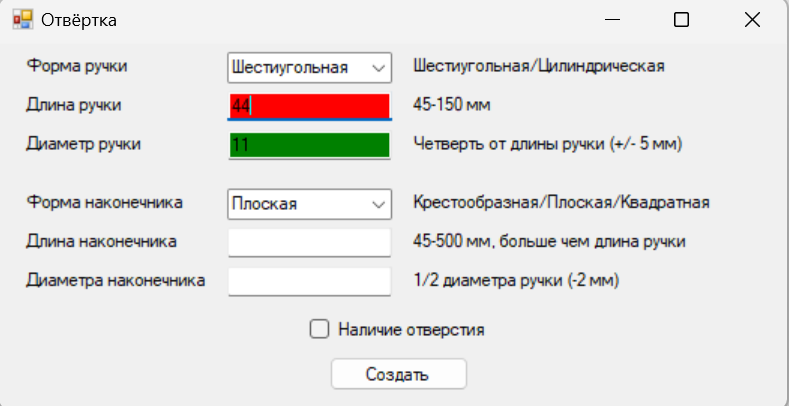


Рисунок 8.4 – Ошибка в собственной валидации длины ручки не влияет на валидацию её диаметра

Значение 11 является допустимым (т.к. 45/4 - 5 минимальное значение для диаметра ручки равняется 7) и подсвечивается зелёным. Помимо этого можно заметить, что при наведении на корректно заполненный текстбокс не выводится никакая подсказка.

Помимо собственной ошибки также может быть вызвана ошибка в зависимых параметрах. Для решения каждой из таких ошибок текст подсказки меняет своё значение и выдаёт рекомендуемые к заполнению параметры для пользователя. Пользователь может изменить значение для любого из зависимых параметров для получения корректных результатов. При наличии ошибок в нескольких зависимых параметрах от одного у пользователя есть два пути решения:

1. Изменить сначала основной параметр, подходящий под хотя бы одну из валидаций, а после изменить оставшийся зависимый от него параметр.
2. Изменить оба зависимых от основного параметра.

К сожалению, система не будет настолько подробно описывать вызванные ошибки, поэтому пользователю придётся самому принимать решение.  
 Помимо этого при появлении в текстбоксе некорректных символов (буквы, символы) – текстбокс очищается и приобретает стандартный цвет.

Также пользователю доступен выбор значений для двух комбобоксов и определение состояния чекбокса, что будет влиять на форму отвёртки.

**9 ТЕСТИРОВАНИЕ ПЛАГИНА**

## 9.1 Функциональное тестирование

Во время использования плагина, плагин обрабатывает ошибки следующим образом.

На рисунках 9.1 и 9.2 представлен результат обработок ошибок системой для зависимых параметров длина и диаметр ручки.

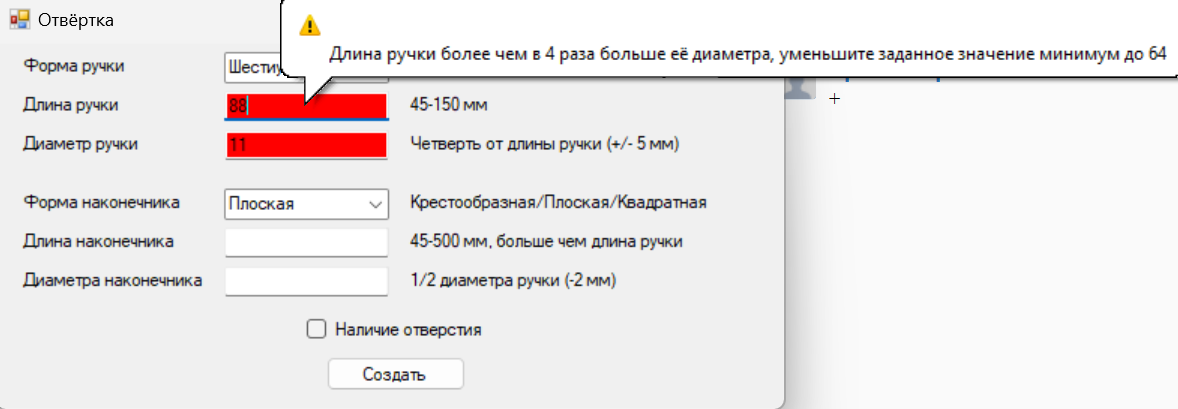


Рисунок 9.1 – Ошибка валидации зависимых параметров длины ручки и диаметра ручки

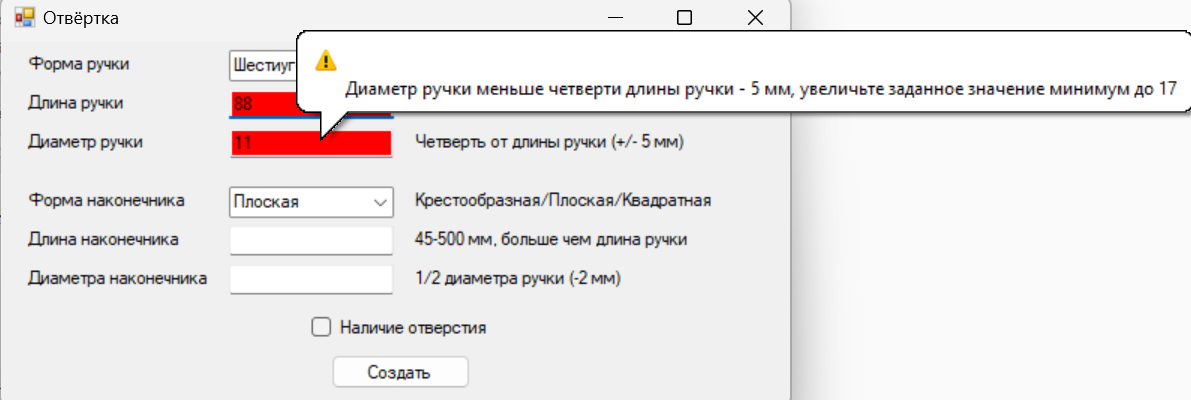


Рисунок 9.2 – Ошибка валидации зависимых параметров длины ручки и диаметра ручки

При ошибке в валидации зависимых параметрах оба зависимых параметра приобретают красный цвет и стандартные подсказки в них изменяются на подсказки для получения корректных значений. Доказательства правильности выведенных подсказок представлены на рисунках 9.3 и 9.4.

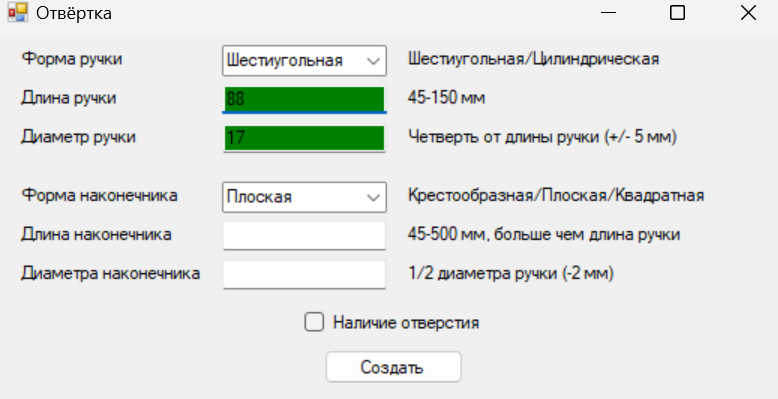


Рисунок 9.3 – Увеличение диаметра ручки до рекомендованных 17 мм

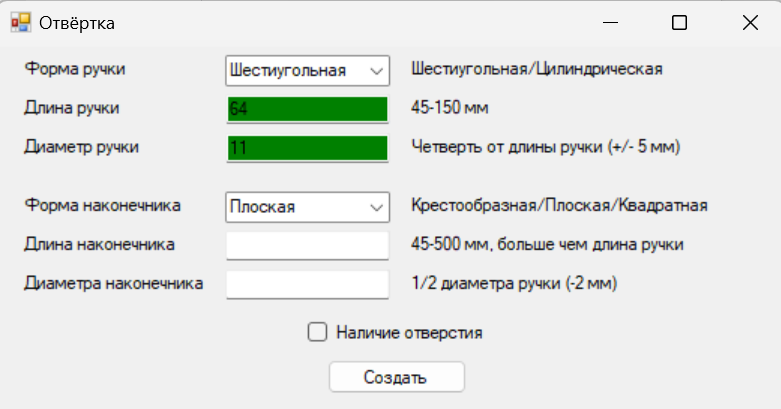


Рисунок 9.4 – Уменьшение длины ручки до рекомендованных 64 мм

Ещё один вариант валидации возможен при некорректности сразу в нескольких связанных параметрах (рисунок 9.5)

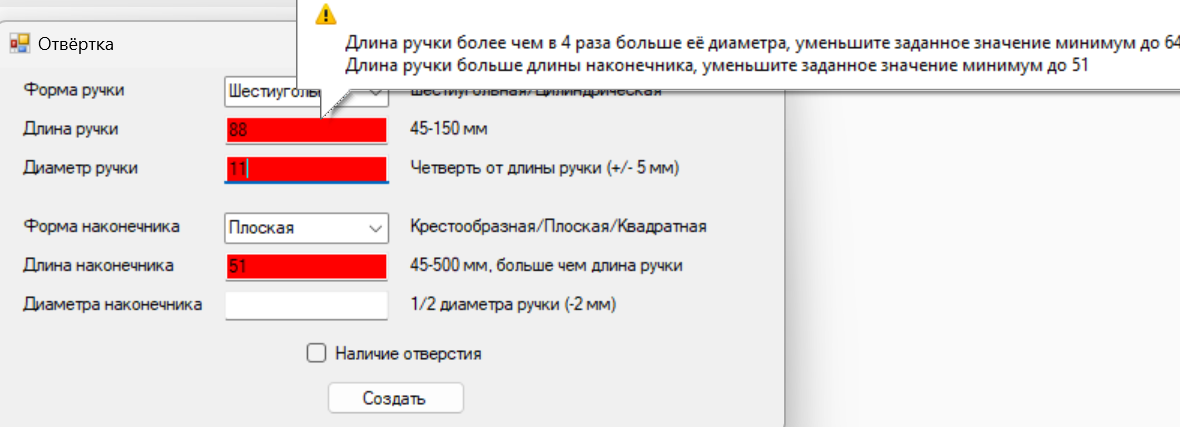


Рисунок 9.5 – Ошибка в валидации длины и диаметра ручки, а также длины ручки и длины наконечник

К сожалению при такой ошибке валидации пользователь должен сам принимать более подходящие ему решения. Но данный вариант не является самым нестандартным, следующий пример (рисунок 9.6) может сильно запутать пользователя.

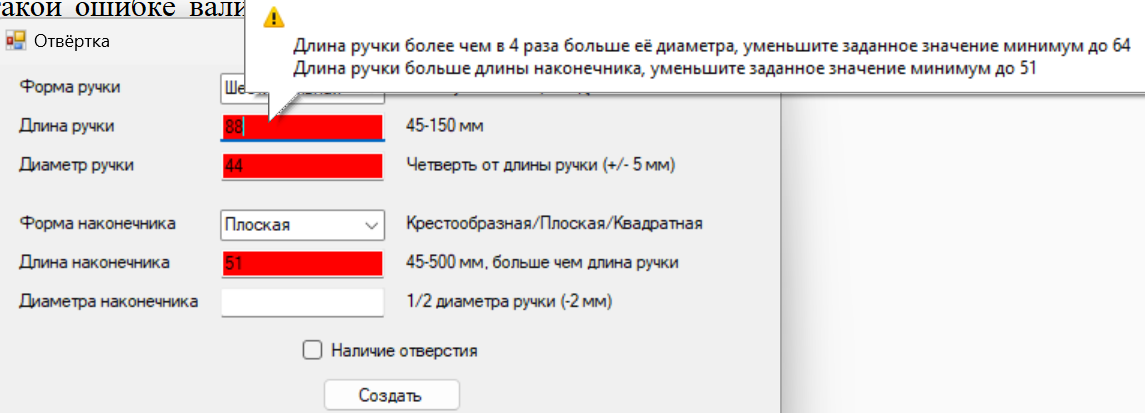


Рисунок 9.6 – Ошибка в валидации длины и диаметра ручки (длина менее чем в 4 раза больше диаметра) и длины ручки с длиной наконечника.

На рисунке 9.7 представлено заполнение формы минимальными параметрами.

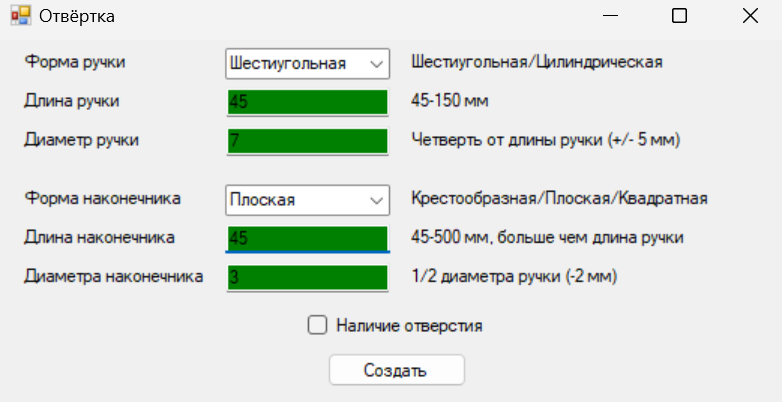


Рисунок 9.7 – Минимальные параметры

На рисунке 9.8 представлен результат построения модели с минимальными параметрами.

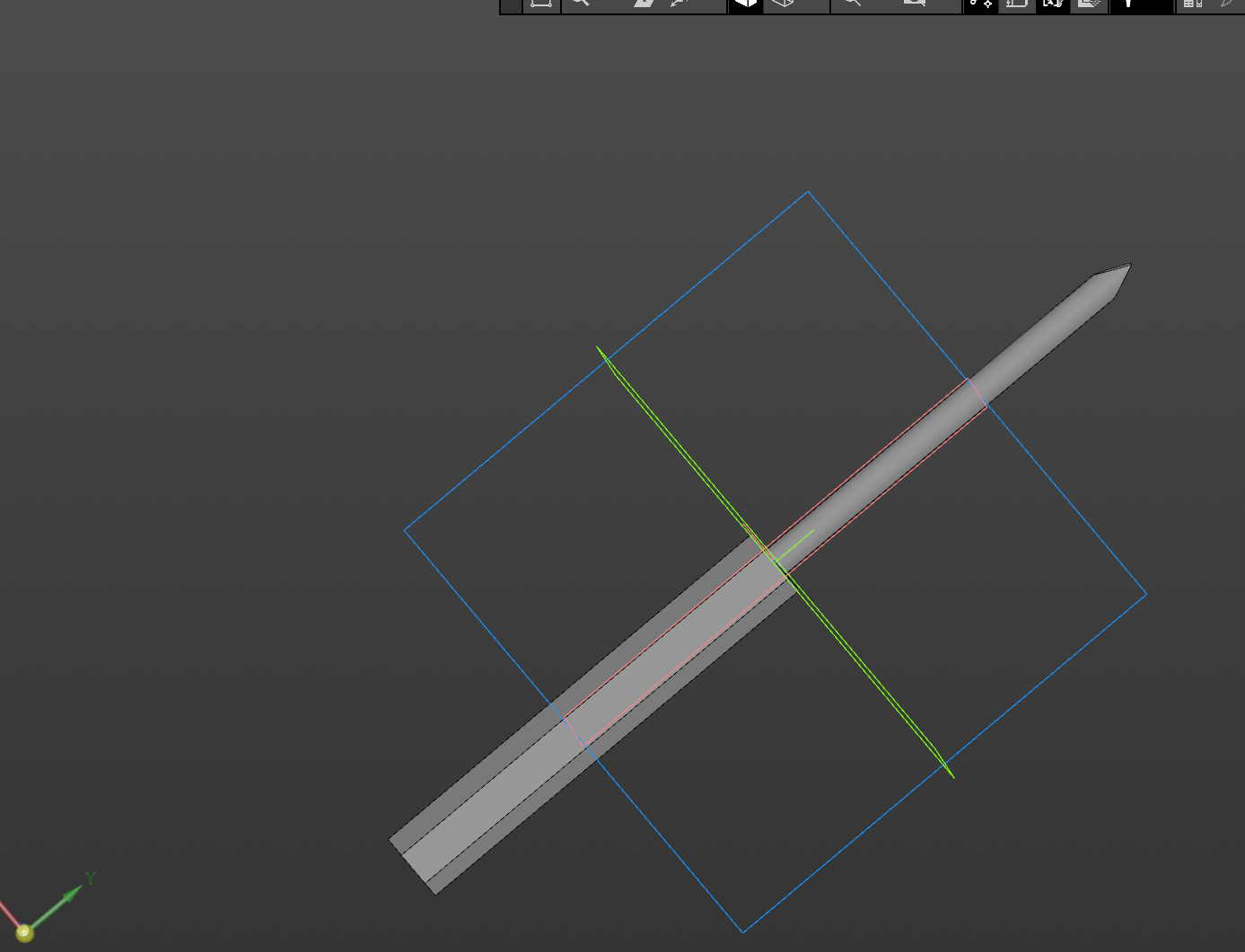


Рисунок 9.8 – Модель по минимальным параметрам

На рисунке 9.9 представлено заполнение формы максимальными параметрами.

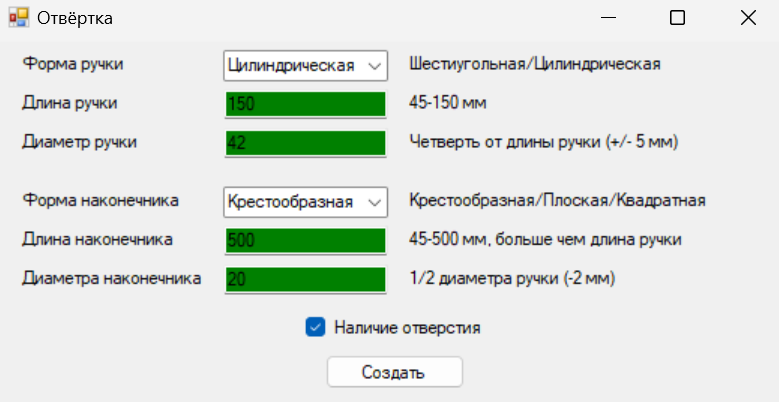


Рисунок 9.9 – Максимальные параметры

На рисунке 9.10 представлен результат построения модели с максимальными параметрами.

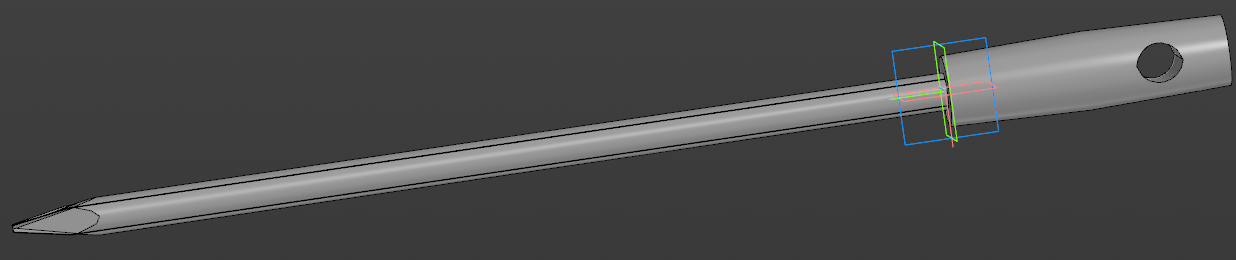


Рисунок 9.10 – Модель по максимальным параметрам

На рисунке 9.11 представлено заполнение формы стандартными параметрами.

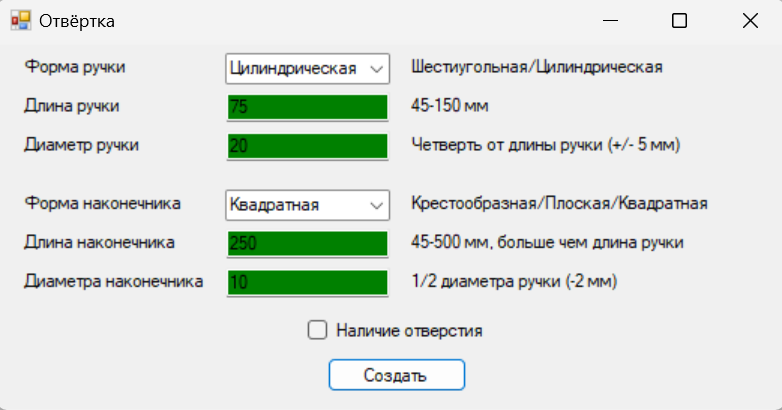


Рисунок 9.11 – Стандартные параметры

На рисунке 9.12 представлен результат построения модели с стандартными параметрами.

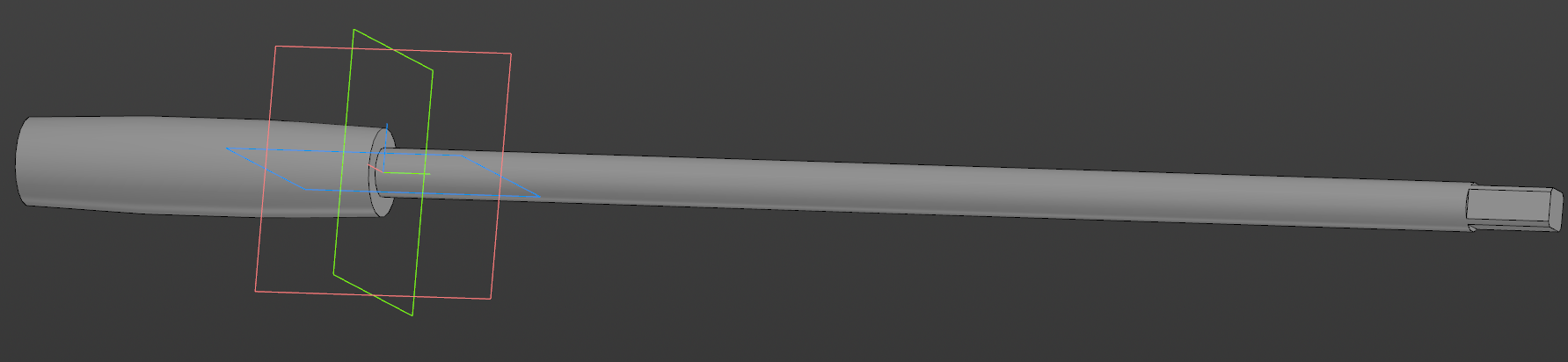


Рисунок 9.12 – Модель по стандартным параметрам

## 9.2 Модульное тестирование

На рисунке 9.13 представлено количество написанных Unit-тестов, а также что их выполнение происходит корректно.

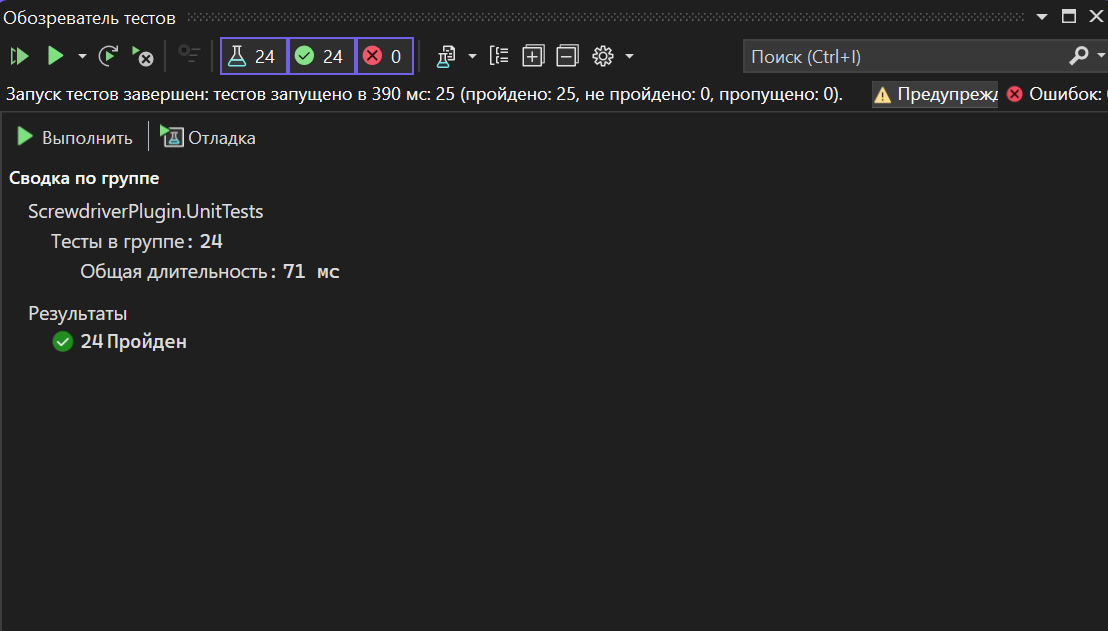


Рисунок 9.13 – Количество написанных Unit-тестов

Необходимо было написать тесты для 2-ух классов: Parameters и Parameter. В таблице 9.1 представлены все написанные тесты и их описание.

Таблица 9.1 – Unit-тесты

|  |  |
| --- | --- |
| Название теста | Описание теста |
| Позитивный тест геттера MaxValue | Проверяет работу get у MaxValue |
| Позитивный тест сеттера MaxValue | Проверяет работу set у MaxValue |
| Позитивный тест геттера MinValue | Проверяет работу get у MinValue |
| Позитивный тест сеттера MinValue | Проверяет работу set у MinValue |
| Позитивный тест геттера Value | Проверяет работу get у Value |
| Позитивный тест сеттера Value | Проверяет работу set у Value |
| Негативный тест Validate | Проверяет вызов исключения при Value<MinValue |
| Проверяет вызов исключения при Value>MaxValue |

Продолжение таблицы 9.1

|  |  |
| --- | --- |
| Название теста | Описание теста |
| Позитивный тест геттера AllParameters | Проверяет работу get у AllParameters |
| Позитивный тест сеттера AllParameters | Проверяет работу set у AllParameters |
| Позитивный тест геттера ShapeOfHandle | Проверяет работу get у ShapeOfHandle |
| Позитивный тест сеттера ShapeOfHandle | Проверяет работу set у ShapeOfHandle |
| Позитивный тест геттера ShapeOfRod | Проверяет работу get у ShapeOfRod |
| Позитивный тест сеттера ShapeOfRod | Проверяет работу set у ShapeOfRod |
| Позитивный тест метода SetParameter | Проверяет работу set для \_parameter |
| Негативный тест ValidateParameters | Проверяет вызов исключения при длине ручке большей более чем в 4 раза диаметра ручки |
| Проверяет вызов исключения при длине ручке большей менее чем в 4 раза диаметра ручки |
| Проверяет вызов исключения при длине ручке большей длины наконечника |
| Проверяет вызов исключения при диаметре ручки более чем в 4 раза меньшем длины ручки |
| Проверяет вызов исключения при диаметре ручки менее чем в 4 раза меньшем длины ручки |
| Проверяет вызов исключения при диаметре ручки менее чем в 2 раза меньшем диаметра наконечника |
| Проверяет вызов исключения при диаметре ручки более чем в 2 раза меньшем диаметра наконечника |
| Проверяет вызов исключения при длине наконечника меньшем длины ручки |
| Проверяет вызов исключения при диаметре наконечника менее чем в 2 раза меньшем диаметра ручки |
| Проверяет вызов исключения при диаметре наконечника более чем в 2 раза меньшем диаметра ручки |

На рисунке 9.14 также представлен скриншот плагина, измеряющего процент покрытия модульными тестами

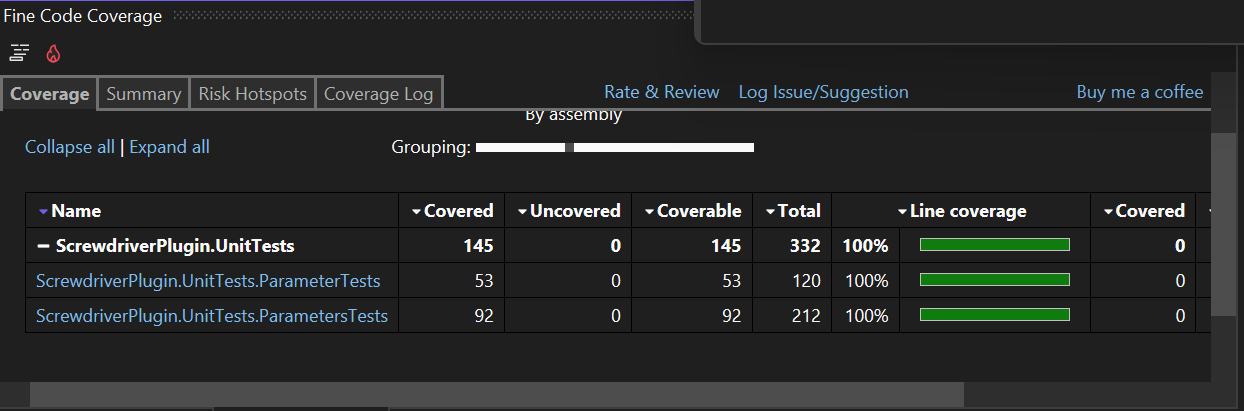


Рисунок 9.14 – Результаты плагина

## 9.3 Нагрузочное тестирование

На рисунке 9.15 представлен график зависимости памяти ОЗУ от построения модели, а на рисунке 9.16 представлен график зависимости времени от построения модели.

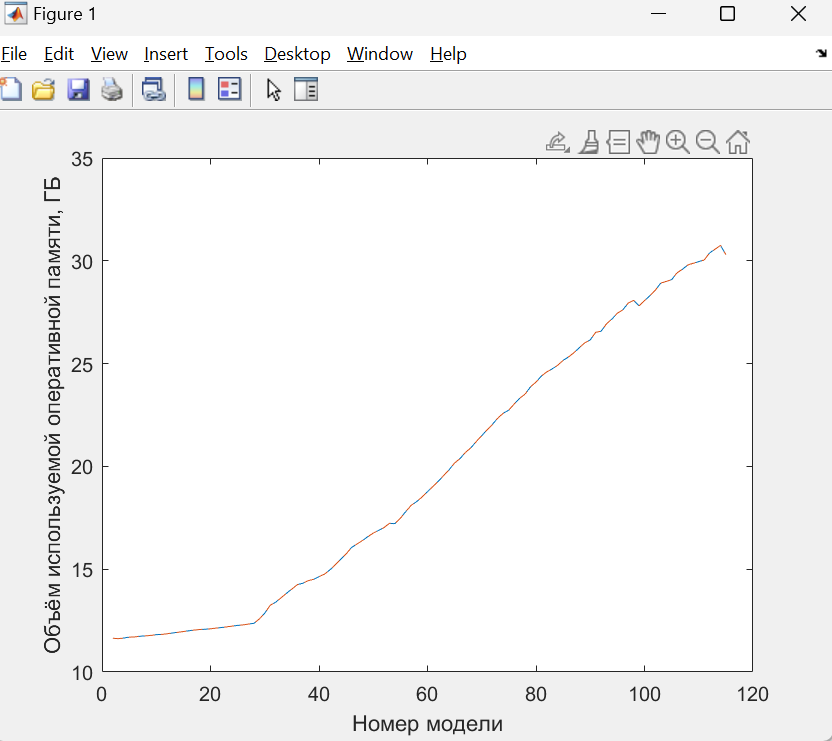


Рисунок 9.15 – График зависимости памяти ОЗУ от количества построенных моделей

Из графика 9.15 можно сделать вывод, что начиная с 30 модели происходит линейный рост в объёме используемой оперативной памяти. Моё предположение связано с тем, что до 30 модели «Компас» заранее резервирует определённое количество памяти для работы, а при достижении её пределов начинает динамически расширять её объёмы (по мере необходимости).

При достижении порогового значения ~32 ГБ происходит экстренное завершение работы КОМПАС-3D и остановка работы программы. Чтобы занять 32 ГБ памяти пришлось построить всего 115 моделей. Это не связано со сложностью строимой модели, поскольку при попытке открыть 115 пустых файлов деталей в компасе он уже занимает 30 ГБ оперативной памяти (при условии других фоновых задач, погрешность 2ГБ зависит скорее от этого).

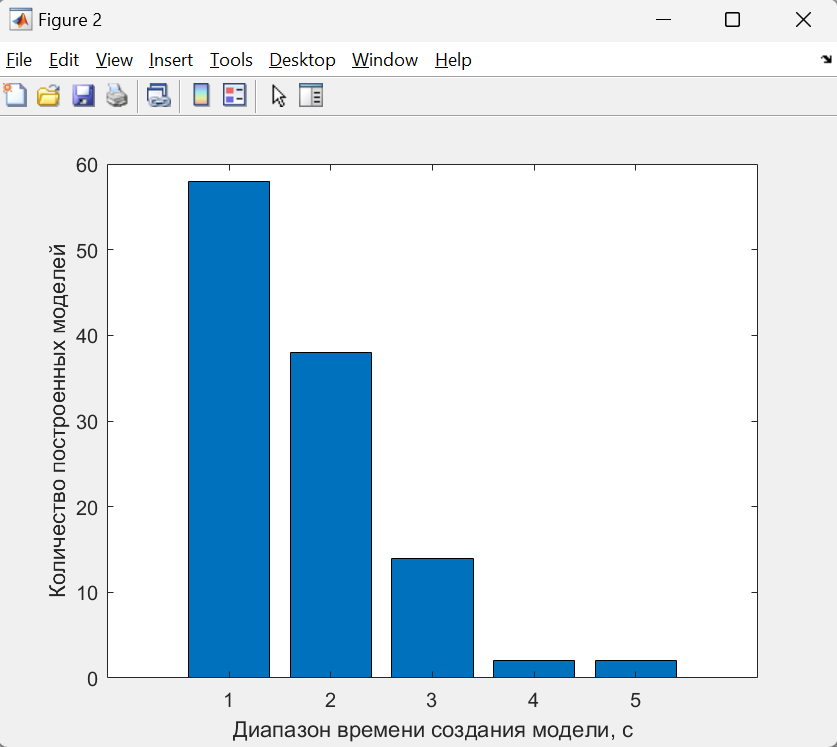


Рисунок 9.16 – График гистограммы построения модели

По графику можно сделать вывод, что основное время построения модели от 0 до 3 с. Скорее всего, это связанно с простотой модели, из-за чего даже при значительных количествах моделей нагрузки на оперативную память и процессор недостаточно для замедления построения моделей. Оставшиеся модели лежащие в диапазоне от 4 до 5 с можно связать с загруженностью ОС другими задачами, не связанными с выполнением моделирования.

**10 ЗАКЛЮЧЕНИЕ**

В результате выполнения лабораторных работ был разработан плагин для КОМПАС-3D, способный самостоятельно строить отвёртку по заданным пользователем параметрам. Каждая из лабораторных работы была направлена на достижение данного результата, выбор объекта моделирования и САПР повлиял на изучение материалов связанных с ними, техническое задание позволило скорректировать курс направленности лабораторных работ, проект системы заставлял продумывать разные мелочи во избежание дальнейших серьёзных изменений в коде, ну и само написание кода, затрагивающее обращение с API, а также взаимодействие с формой пользователя. Из неожиданных результатов была изучена провальность подхода при валидации обоих параметров, что привело к изменению траектории мышления и правильному результату. Также немалое удивление произвело использование различных средств для стандартизации кода, например StyleCops. Код действительно становится более читаемым и перевариваемым для дальнейшей разработки.

**11 СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. ГОСТ 23501.101-87 «Системы автоматизированного проектирования. Основные положения» (дата обращения 13.12.2024)
2. API [Электронный ресурс]. − Режим доступа <https://itglobal.com/ru-ru/company/glossary/api/> (дата обращения 28.09.2024)
3. ГОСТ 17199-88 «Отвёртки слесарно-монтажные» (дата обращения 20.09.2024)
4. Windows Forms [Электронный ресурс]. − Режим доступа <https://learn.microsoft.com/ru-ru/dotnet/desktop/winforms/overview/?view=netdesktop-9.0> (дата обращения 13.12.2024)
5. Github [Электронный ресурс]. − Режим доступа <https://docs.github.com/ru/get-started/start-your-journey/about-github-and-git> (13.12.2024)
6. ReSharper [Электронный ресурс]. − Режим доступа <https://www.jetbrains.com/ru-ru/resharper/> (13.12.2024)
7. Fine Code Coverage [Электронный ресурс]. − Режим доступа <https://marketplace.visualstudio.com/items?itemName=FortuneNgwenya.FineCodeCoverage> (13.12.2024)
8. StyleCop [Электронный ресурс]. − Режим доступа <https://andrey.moveax.ru/post/net-standard-using-style-cop> (13.12.2024)
9. NUnit [Электронный ресурс]. − Режим доступа <https://nunit.org/> (13.12.2024)
10. Разъёмные соединения [Электронный ресурс]. − Режим доступа <https://kompas.ru/kompas-3d/application/machinery/threaded-connection/> (дата обращения 05.10.2024)
11. Валы и механические передачи 3D. [Электронный ресурс]. − Режим доступа <https://kompas.ru/kompas-3d/application/machinery/gear-cutting/> (дата обращения 05.10.2024)
12. UML [Электронный ресурс]. − Режим доступа <https://www.uml-diagrams.org/> (дата обращения 07.10.2024)