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

\_\_\_\_\_\_\_\_\_ Грахович В.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 монитора необходим и может быть использован на предприятиях, занимающихся моделированием мониторов, поскольку он упростит процесс моделирования.

**2 ПОСТАНОВКА И АНАЛИЗ ЗАДАЧИ**

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

Таблица 2.1 − Этапы проведения работ по разработке плагина «Монитор» 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 |

В процессе работы над плагином были внесены некоторые изменения, которые повлекли за собой изменения конечной версии UML-диаграммы классов. Связано это с недостаточными знаниями API Kompas-3D на этапе проектирования. Это значит, что в дальнейшем необходимо более детально подходить к изучению средств разработки и API еще на этапе создания проекта системы.

# 3 ОПИСАНИЕ ПРЕДМЕТА ПРОЕКТИРОВАНИЯ

Монитор – главное периферийное устройство для ЭВМ, предназначенная для вывода изображения, графической информации [4].



Рисунок 2.1 − Модель отвёртки

***Изменяемые параметры для плагина*** (также все обозначения показаны на рисунке 2.1):

**Ширина экрана L** (400–900 мм);

**Высота экрана H**:

* Для **4:3** экранов высота составляет 75% от ширины L, то есть H=0.75⋅L;
* Для **16:10** экранов высота будет 62.5% от ширины L, то есть H=0.625⋅L;
* Для **16:9** экранов высота составляет 56% от ширины L, то есть H=0.56⋅L;
* Для **21:9** экранов высота будет 43% от ширины L, то есть H=0.43⋅L.

**Высота рамки b** (5–30 мм);

**Толщина рамки w** (5–20 мм);

**Глубина экрана g** (0–10 мм; если g=0, то экран и рамка находятся на одной плоскости);

**Высота стойки p** (50–200 мм);

**Ширина стойки m** (30–100 мм);

**Толщина стойки f** (10–50 мм);

**Высота подставки s** (50–150 мм);

**Ширина подставки D** (150–400 мм);

**Толщина подставки z** (10–50 мм);

**Длина рычага j** (0–150мм).

Также в ходе работы были добавлены параметры высоты (20-200мм) и ширины (20-200мм) рычага.

**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;

− UnitTestProject

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]

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 мониторов разных размеров. Используя данный плагин, можно быстро построить монитор по заданным параметрам экрана, корпуса, а также подставки.

**6 ОБЗОР АНАЛОГОВ**

Косвенным аналогом разрабатываемого плагина является **AutoCAD Plant 3D Toolset** [3]. Это специализированный инструмент для проектирования промышленных объектов, который включает инструменты для параметрического проектирования. В то время как он ориентирован на создание инженерных объектов, его функционал позволяет настроить параметрическое проектирование для создания различных объектов по заданным размерам и параметрам, включая экраны или другие объекты с заданными пропорциями. Стоимость **AutoCAD Plant 3D Toolset** включает подписку на AutoCAD с расширенным функционалом, и его цена начинается примерно с **3000 долларов в год.** Интерфейс взаимодействия представлен на рисунке 1.1.



Рисунок 6.1 − Интерфейс приложения «Разъёмные соединения»

# 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[5]. UML диаграмма классов для плагина «Монитор» представлена на рисунке 7.1.

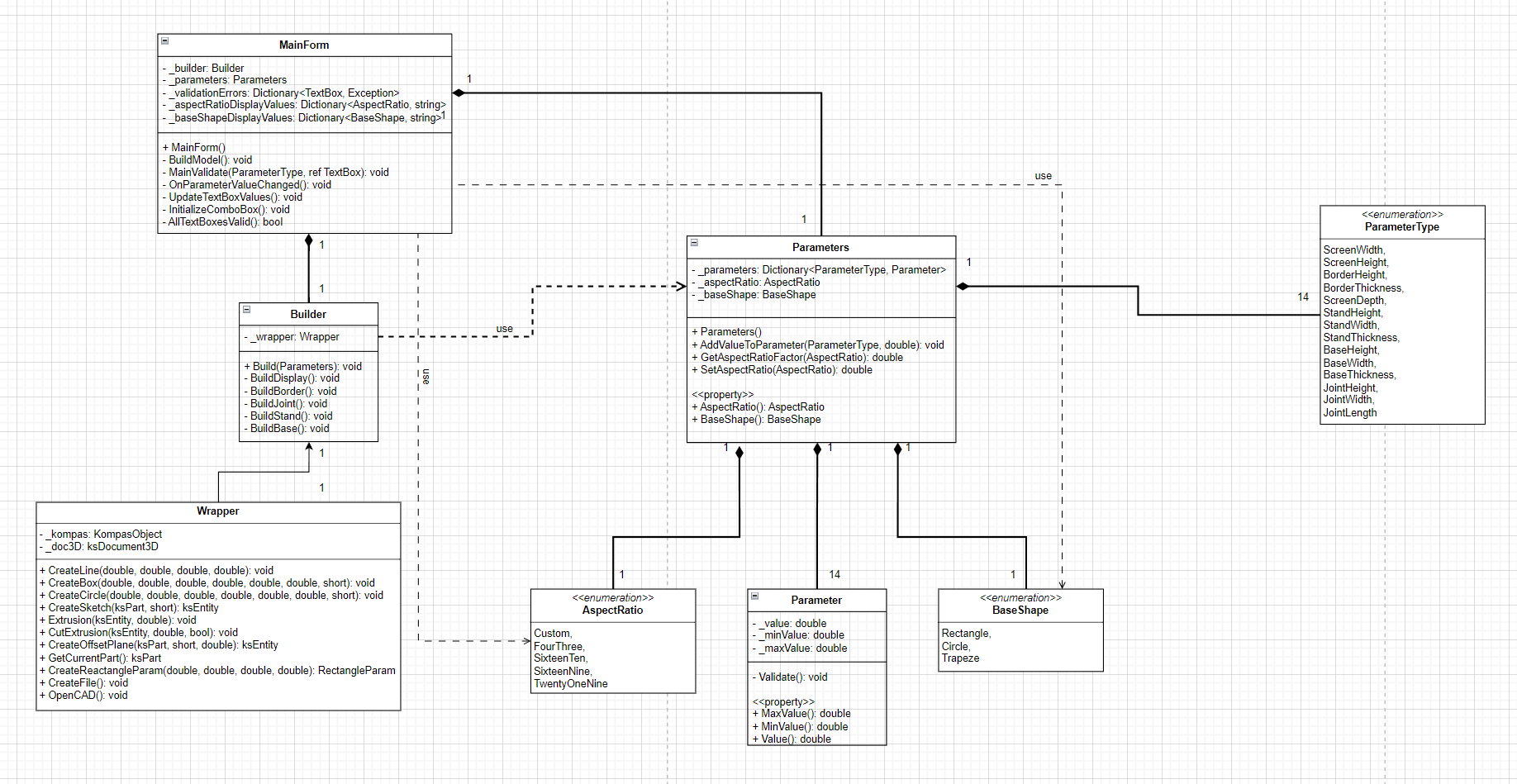


Рисунок 7.1 − UML диаграмма классов после реализации плагина «Монитор»

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 |
| \_validationErrors | Dictionary<TextBox, Exception> | Хранит список ошибок валидации |
| \_aspectRatioDisplayValues | Dictionary<AspectRatio, string> | Хранит отображаемые значения для comboBoxRatio |
| \_baseShapeDisplayValues | Dictionary<BaseShape, string> | Хранит отображаемые значения для comboBoxShape |

Таблица 7.2 − Методы класса MainForm

|  |  |  |
| --- | --- | --- |
| Название | Входные параметры | Описание |
| BuildButton\_Click | object sender, EventArgs e | Запуск построения модели по заданным параметрам |
| MainForm | − | Конструктор MainForm |
| MainValidate | ParameterType, textBoxTemp | Проверка введённых данных по формату |
| textBox\_Leave | object sender, EventArgs e | Обработчик выхода из текстбоксов |
| comboBoxRatio\_SelectedIndexChanged | object sender, EventArgs e | Обработчик изменения значения выпадающего списка для выбора соотношения сторон экрана |
| comboBoxShape\_SelectedIndexChanged | object sender, EventArgs e | Обработчик изменения значения выпадающего списка для выбора формы подставки |
| UpdateTextBoxValues | − | Обновляет визуальное представление textBox’ов |
| InitializeComboBox | − | Обновляет содержимое выпадающих списков |
| AreAllTextBoxesValid | − | Проверяет наличие ошибок в словаре с ошибками |

Таблица 7.3 − Свойства класса Parameters

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_parametersDict | Dictionary<ParameterType, Parameter> | Словарь с параметрами |
| \_aspectRatio | AspectRatio | Перечисление с соотношениями сторон |
| \_baseShape | BaseShape | Перечисление с формами подставки |

Таблица 7.4 − Методы класса Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Выходные параметры | Описание |
| ParametersDict | − | Dictionary<ParameterType, Parameter> | Свойство для \_parametersDict |
| AddValueToParameter | ParameterType, double | − | Устанавливает значение для параметра |
| SetAspectRatio | AspectRatio | − | Задать соотношение сторон |
| GetAspectRatioFactor | AspectRatio | double | Получает коэффициент соотношения сторон |
| BaseShape | BaseShape | BaseShape | Свойство для \_baseShape |

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 |
| BuildDisplay | Parameters | Построение экрана |
| BuildBorder | Parameters | Выдавливание рамки |
| BuildJoint | Parameters | Построение крепления |
| BuildStand | Parameters | Построение стойки |
| BuildBase | Parameters | Построение подставки |

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 |
| \_doc3D | ksDocument | Поле, хранящее в себе документ |

Таблица 7.10 − Методы класса Wrapper

|  |  |  |
| --- | --- | --- |
| Название | Входные параметры | Описание |
| CreateLine | double, double, double, double | Создание линии на плоскости XOZ по координатам |
| CreateBox | double, double, double, double, double, double, short | Создание цилиндра на плоскости short planeType |
| CreateCircle | double, double, double, double, double, short | Создание эскиза (по int выбираем базисную плоскость) |
| CutExtrusion | ksEntity, double, bool | Создание выреза по эскизу |
| Extrusion | ksEntity, double | Выдавливание эскиза на глубину double |
| GetCurrentPart | − | Получить деталь из документа \_doc3D |
| CreateRectangleParam | double, double, double, double | Создание параметров прямоугольника |
| CreateFile | − | 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; добавлено поле и свойство typeOfParameter, а также метод DefineMinMax для определения минимального и максимального значения;

− 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).

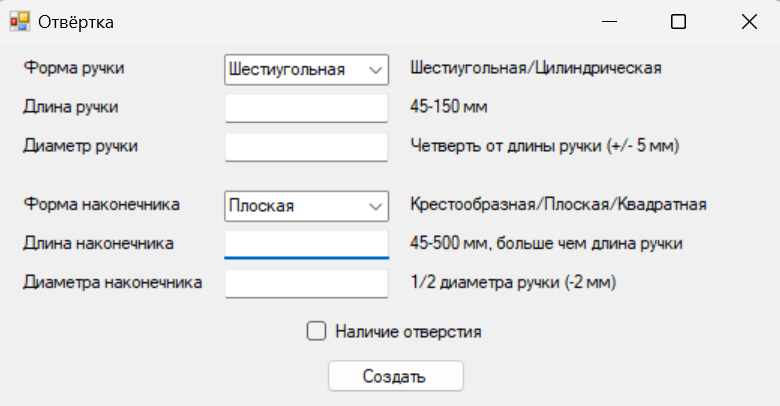


Рисунок 8.1 – Начальная форма в момент запуска приложения

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).

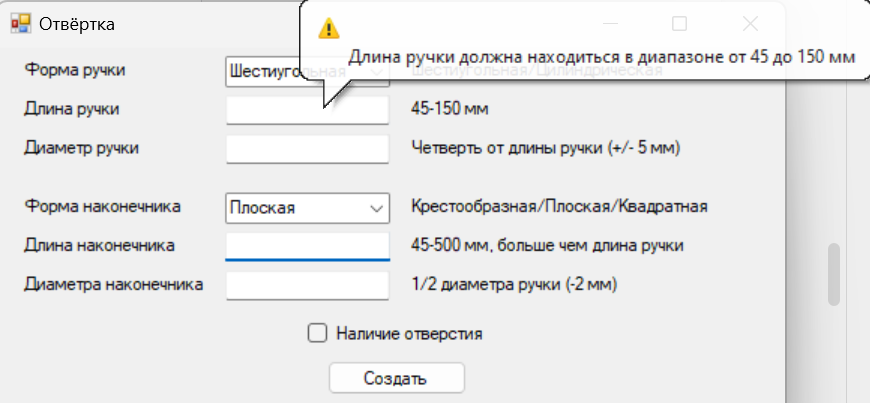


Рисунок 8.2 – Подсказка по заполнению

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

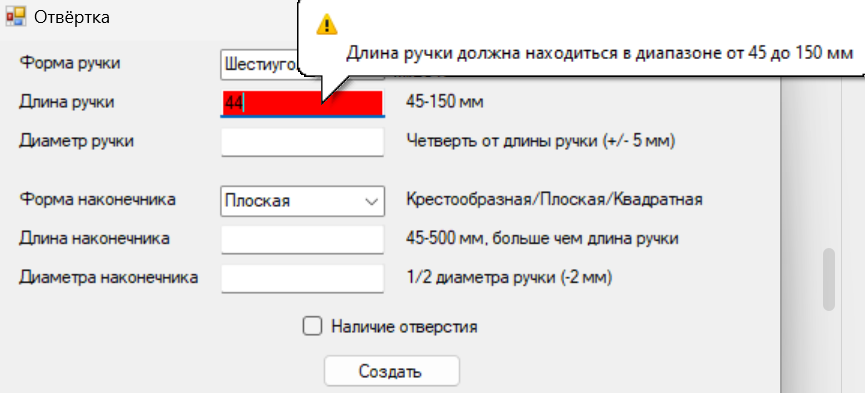


Рисунок 8.3 – Длина ручки выходит за минимальные пределы (меньше 45)

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

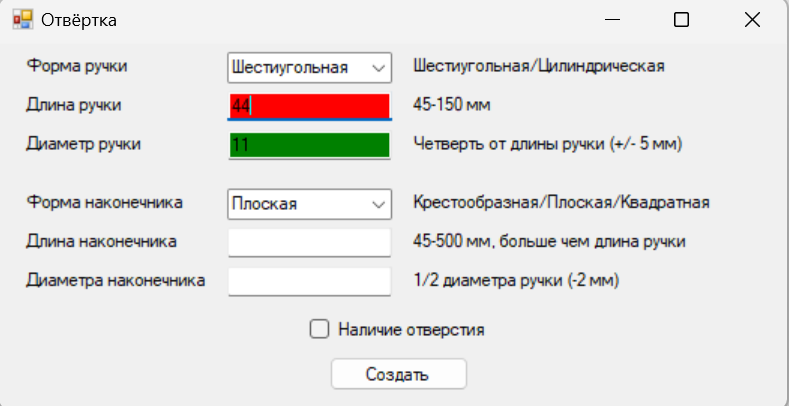


Рисунок 8.4 – Ошибка в собственной валидации длины ручки не влияет на валидацию её диаметра

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.

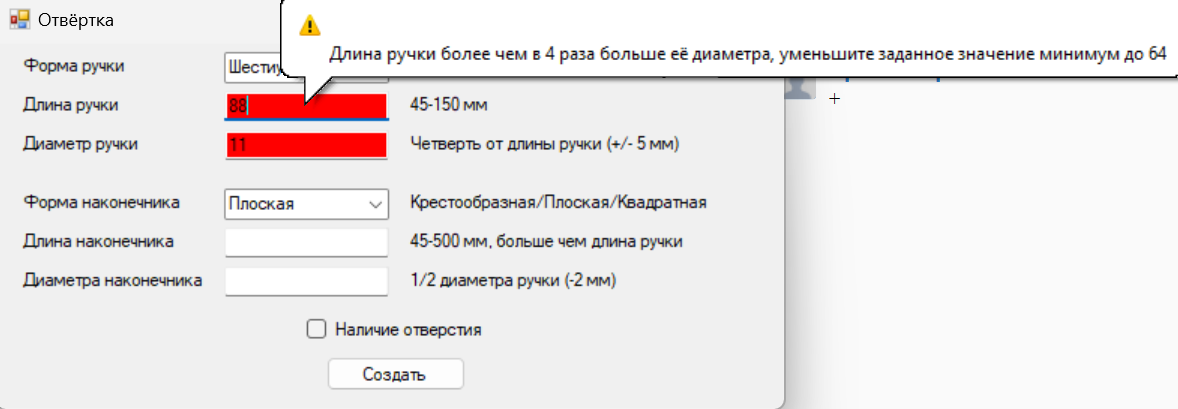


Рисунок 9.1 – Ошибка валидации зависимых параметров длины ручки и диаметра ручки

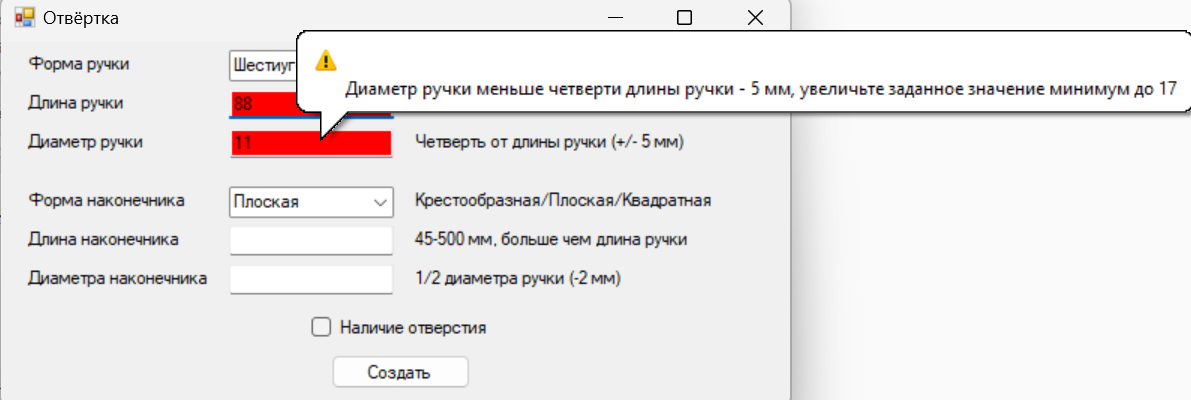


Рисунок 9.2 – Ошибка валидации зависимых параметров длины ручки и диаметра ручки

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.

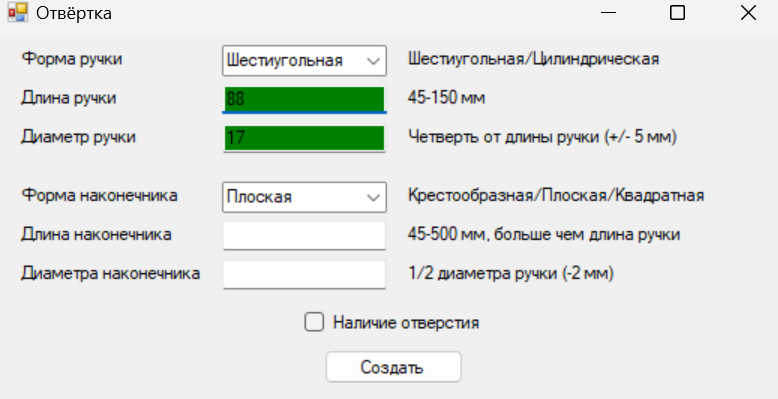


Рисунок 9.3 – Увеличение диаметра ручки до рекомендованных 17 мм

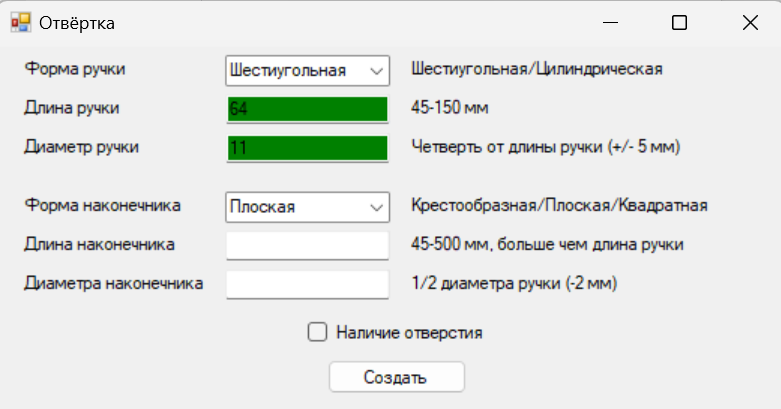


Рисунок 9.4 – Уменьшение длины ручки до рекомендованных 64 мм

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

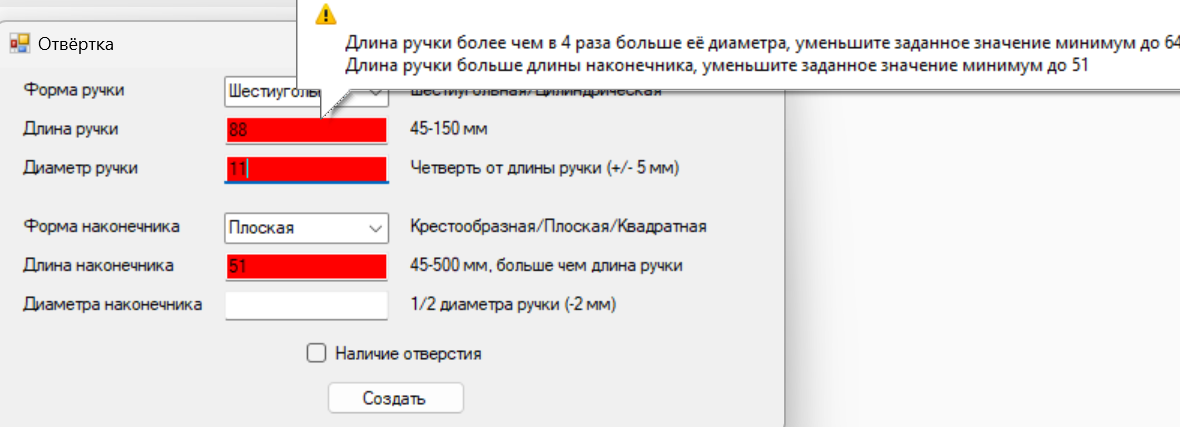


Рисунок 9.5 – Ошибка в валидации длины и диаметра ручки, а также длины ручки и длины наконечник

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.

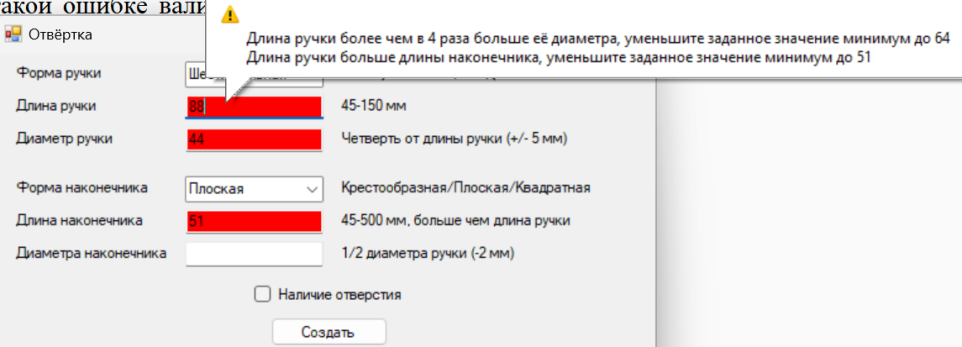


Рисунок 9.6 – Ошибка в валидации длины и диаметра ручки (длина менее чем в 4 раза больше диаметра) и длины ручки с длиной наконечника.

На рисунке 9.7 представлено заполнение формы минимальными параметрами.

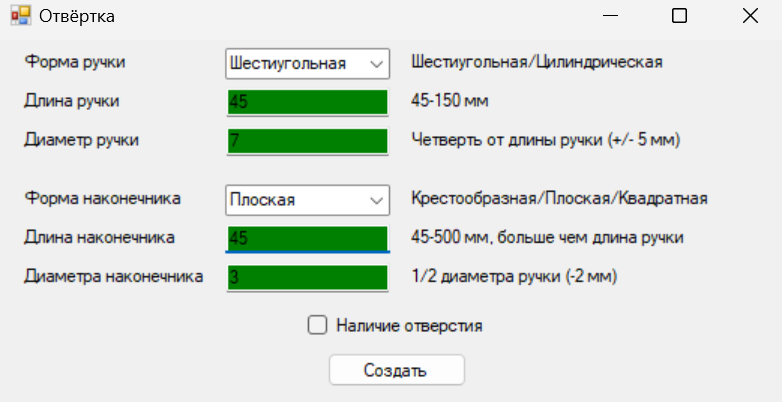


Рисунок 9.7 – Минимальные параметры

На рисунке 9.8 представлен результат построения модели с минимальными параметрами.

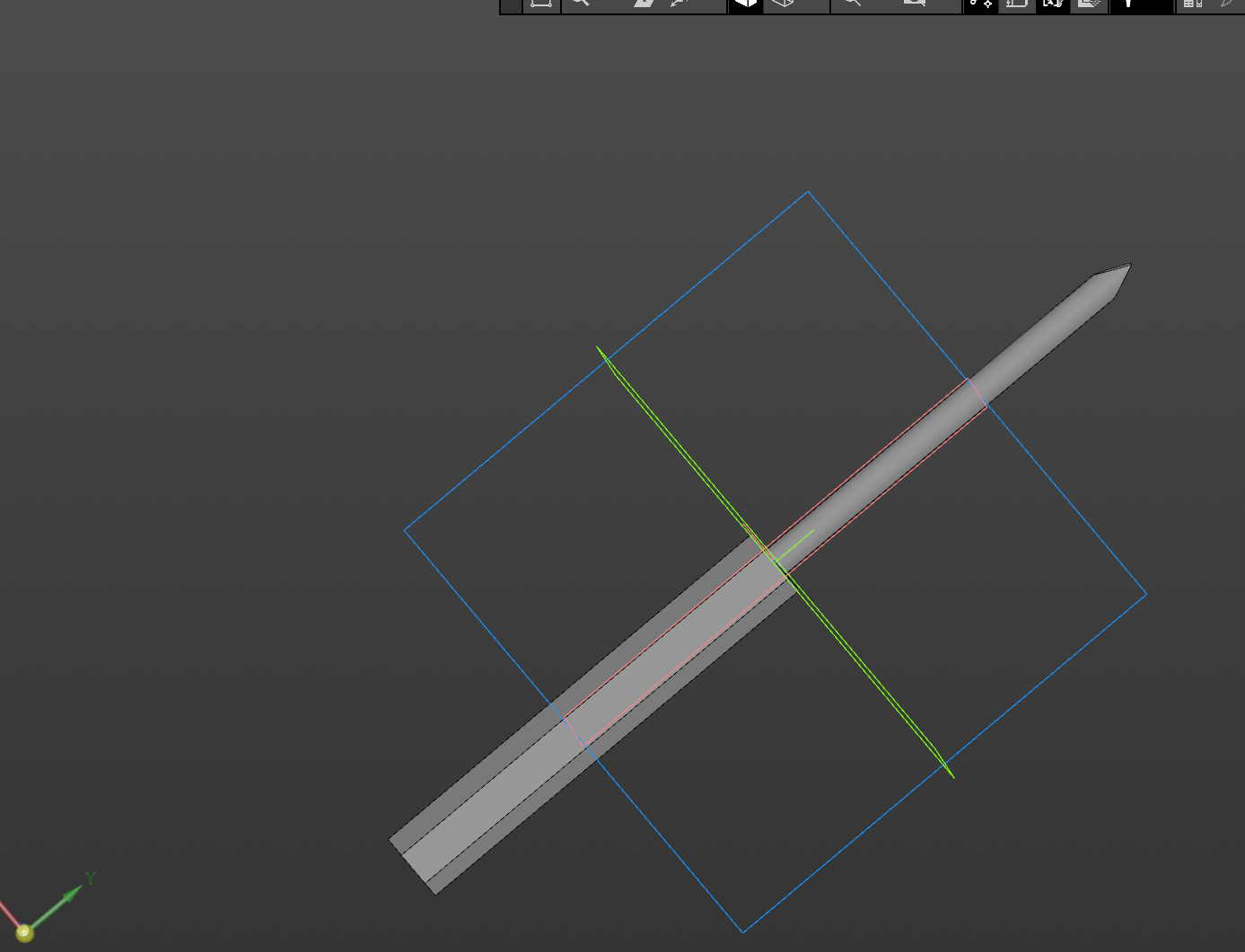


Рисунок 9.8 – Модель по минимальным параметрам

На рисунке 9.9 представлено заполнение формы максимальными параметрами.

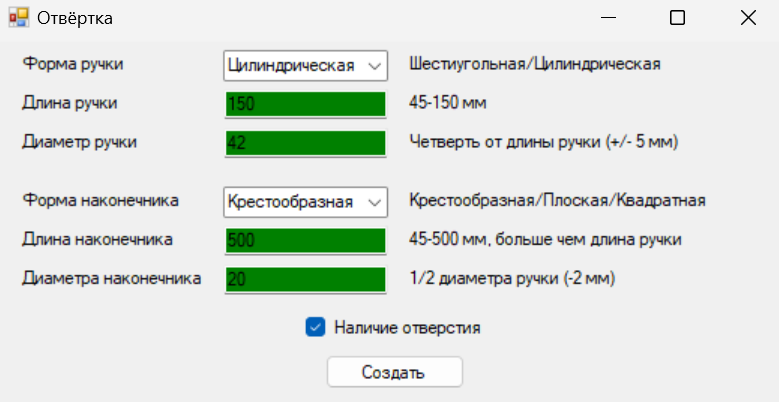


Рисунок 9.9 – Максимальные параметры

На рисунке 9.10 представлен результат построения модели с максимальными параметрами.

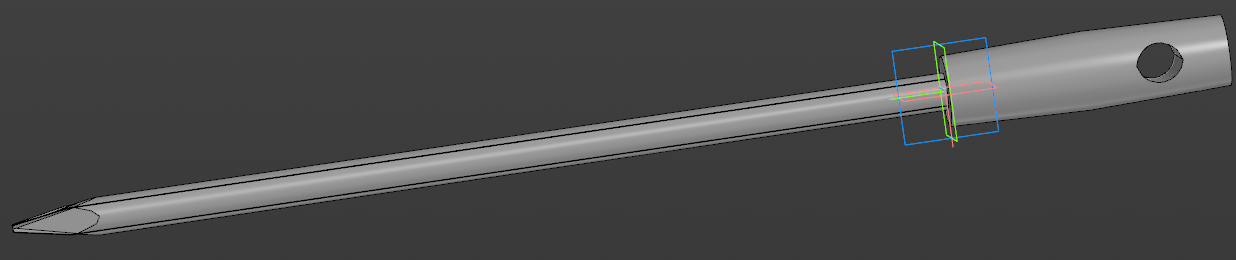


Рисунок 9.10 – Модель по максимальным параметрам

На рисунке 9.11 представлено заполнение формы стандартными параметрами.

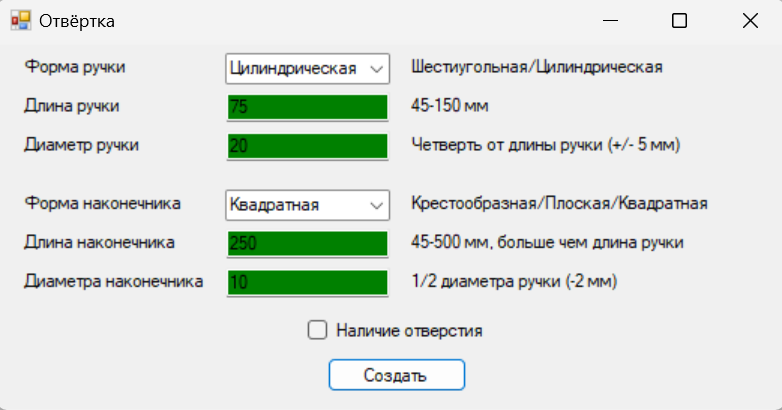


Рисунок 9.11 – Стандартные параметры

На рисунке 9.12 представлен результат построения модели с стандартными параметрами.

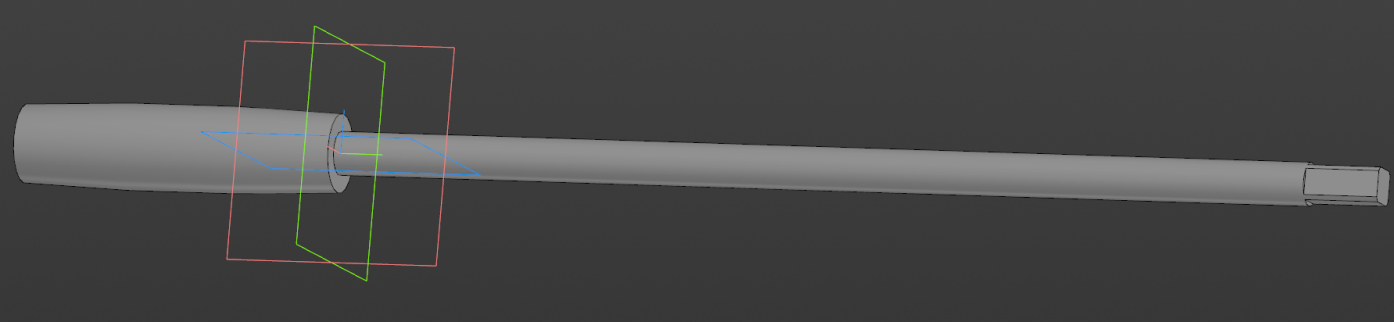


Рисунок 9.12 – Модель по стандартным параметрам

## 9.2 Модульное тестирование

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

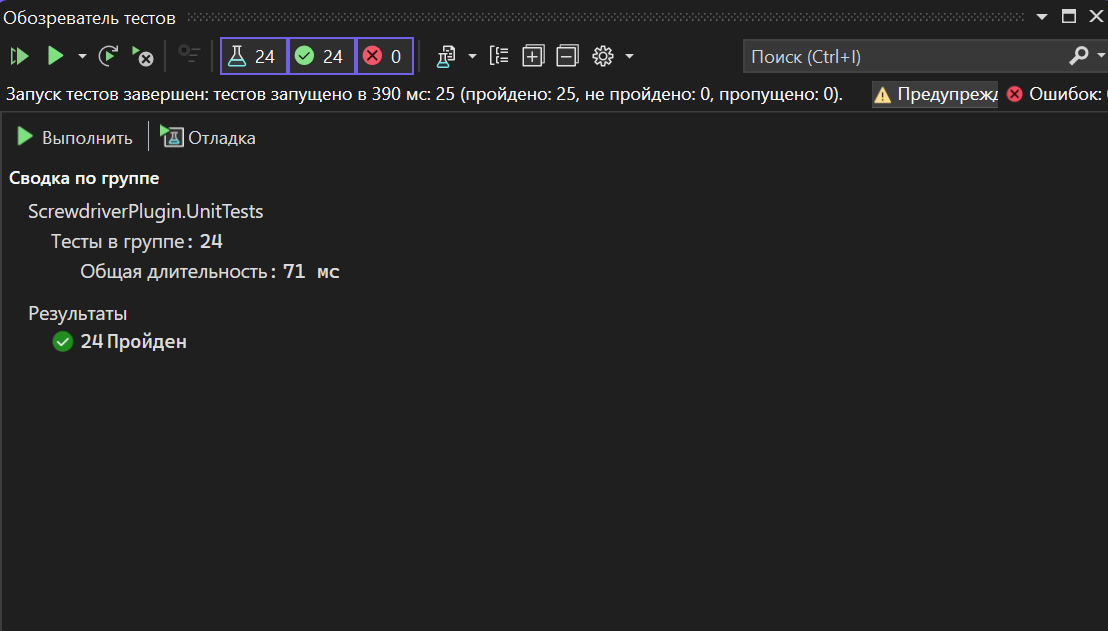


Рисунок 9.13 – Количество написанных Unit-тестов

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

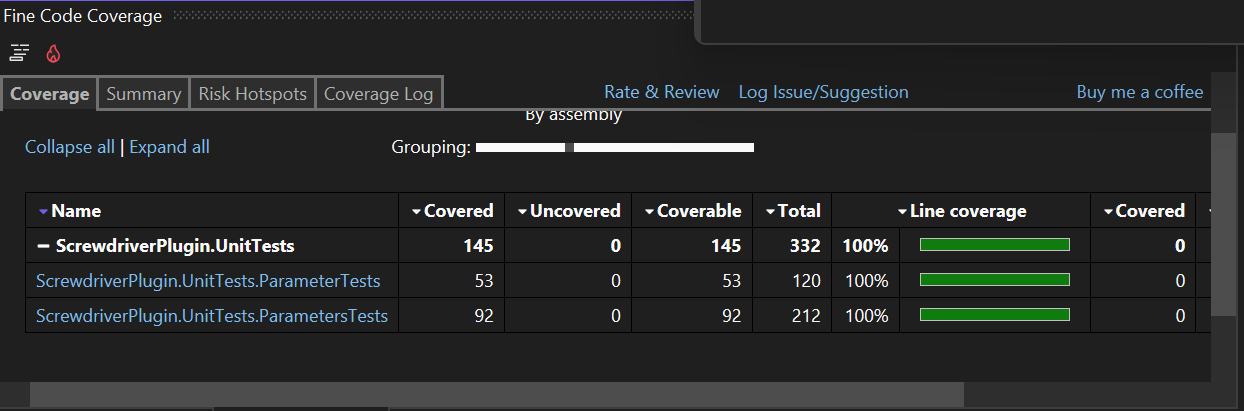


Рисунок 9.14 – Результаты плагина

## 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.

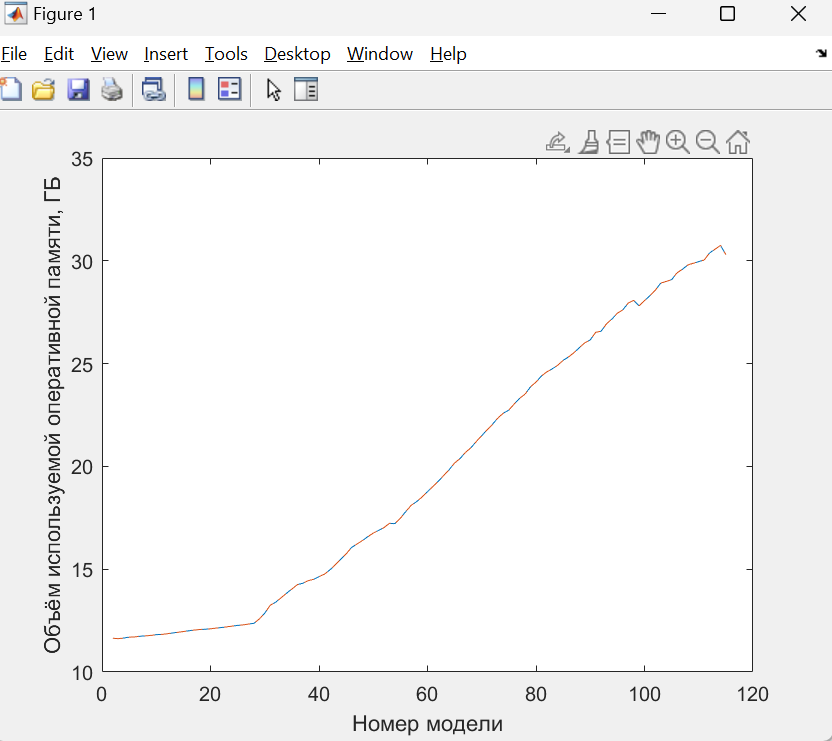


Рисунок 9.15 – График зависимости памяти ОЗУ от количества построенных моделей

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).

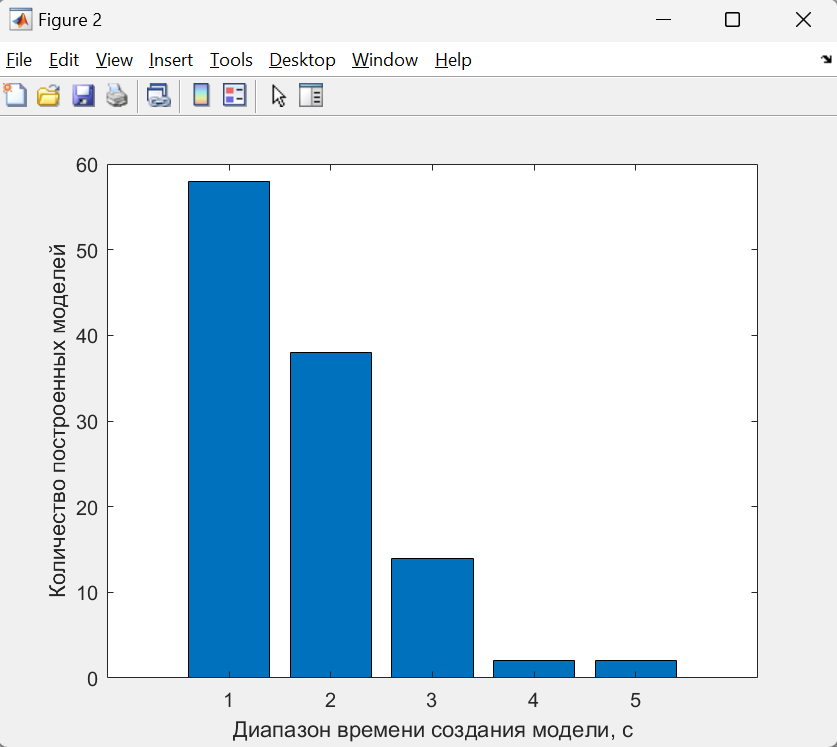


Рисунок 9.16 – График гистограммы построения модели

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)