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

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра компьютерных систем в управлении и проектировании (КСУП)

Разработка плагина «Построение втулки для катушки»

для САПР «Компас 3D»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

по дисциплине

«Основы разработки САПР»

Выполнил:

студент гр. 586–1

\_\_\_\_\_\_\_В.А. Зорин

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_2020 г.

Руководитель:

к.т.н, доцент каф. КСУП

\_\_\_\_\_\_\_ А.А. Калентьев

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_2020 г.

Томск 2020

**Реферат**

Реферат: 31 страница, 24 рисунка, 5 таблиц, 9 использованных источников.

КОМПАС-3D, БИБЛИОТЕКА, ВТУЛКА ДЛЯ КАТУШКИ, РАЗРАБОТКА, ПРОЕКТ, ТЕСТИРОВАНИЕ

Целью данной работы является разработка плагина, который автоматизирует построение втулки в САПР «КОМПАС-3D V 16» при помощи интегрированной среды разработки Visual Studio 2019 Сommunity.

В ходе работы были изучены основы разработки программного обеспечения на примере создания библиотеки для построения детали «Втулка для катушки» для САПР «КОМПАС-3D V 16».

Отчет выполнен в программе Microsoft Word. Библиотека написана на языке программирования C# для платформы .NET Framework 4.7.0. В разработке использовалась система контроля версий Git.

**Содержание**

[Введение 4](#_Toc39487829)

[2 Постановка и анализ задачи 5](#_Toc39487830)

[2.1 Описание САПР 5](#_Toc39487831)

[2.2 Описание API 6](#_Toc39487832)

[2.3 Обзор аналогов 8](#_Toc39487835)

2.4 Описание предмета проектирования……………………………………...10

[3 Описание реализации 13](#_Toc39487836)

[3.1 Диаграмма вариантов использования 13](#_Toc39487837)

[3.2 Диаграмма классов 15](#_Toc39487838)

[3.3 Макет пользовательского интерфейса 17](#_Toc39487839)

[4 Описание программы для пользователя 19](#_Toc39487840)

[5 Тестирование плагина 21](#_Toc39487841)

[5.1 Функциональное тестирование 21](#_Toc39487842)

[5.2 Модульное тестирование 25](#_Toc39487843)

[5.3 Нагрузочное тестирование 27](#_Toc39487844)

[Заключение 30](#_Toc39487845)

[Список использованных источников 31](#_Toc39487846)

**Введение**

В настоящее время проектирование представляет собой автоматизированный процесс. Проектировщику, который занимается разработкой механизма, или устройства, требующего больших расчетов, математических вычислений при построении модели и высокой точности, подходят системы автоматизации проектных решений ­­– САПР.

САПР позволяют сократить временные затраты и уменьшить денежные на разработку макета проекта.

В крупных САПР есть средства для разработки, которые предоставляются с целью дать возможность разработчику расширить функционал данной системы под свои нужны. Это позволяет в короткие сроки реализовать то, что хочет разработчик. Данным средством является API – программируемый интерфейс приложения. Это набор готовых средств: классов, процедур, функций и т.д.

Расширение функционала подразумевает разработку плагина на основе предоставленного API. В данном проекте стоит задача разработки плагина для построения 3D модели втулки в автоматизированном режиме. Плагин – компилируемый программный модуль, который подключается к основной программе, предназначенный для расширения ее возможностей. В данном проекте разработка плагина производилась для САПР «Компас-3D» [1].

**2 Постановка и анализ задачи**

В рамках дисциплины «Основы разработки САПР» требовалось разработать плагин в соответствии с техническим заданием. На основе заданных параметров плагин должен строить 3D модель втулки для катушки, взаимодействуя с САПР «Компас-3D» [1]. Так же в плагине должна быть возможность изменения входных параметров втулки, а именно: диаметры всех окружностей, высоты окружностей, общую высоту втулки, длину и диаметр перемычек, а также их количество.

## 2.1 Описание САПР

КОМПАС-3D — система трехмерного проектирования, ставшая стандартом для тысяч предприятий, благодаря сочетанию простоты освоения и легкости работы с мощными функциональными возможностями твердотельного и поверхностного моделирования. Ключевой особенностью продукта является использование собственного математического ядра С3D и параметрических технологий, разработанных специалистами АСКОН. КОМПАС-3D обеспечивает поддержку наиболее распространенных форматов 3D-моделей (STEP, ACIS, IGES, DWG, DXF), что позволяет организовывать эффективный обмен данными со смежными организациями и заказчиками, использующими любые CAD / CAM / CAE-системы в работе [1].

Система позволяет:

- быстро сгенерировать комплекты конструкторской и технологической документации, необходимые для выпуска изделий (сборочные чертежи, спецификации, деталировки и т.д.);

- передать геометрию изделий во внешние расчетные пакеты;

- передать геометрии в пакеты разработки управляющих программ для оборудования с ЧПУ;

- создать дополнительные изображения изделий (например, для составления каталогов, создания иллюстраций к технической документации и т.д.).

Моделирование изделий в КОМПАС-3D можно вести как: "снизу вверх" (используя готовые компоненты), так и "сверху вниз" (проектируя компоненты в контексте конструкции), опираясь на компоновочный эскиз (например, кинематическую схему), либо смешанным способом. Благодаря такому подходу обеспечивается легкая модификация получаемых моделей.

Система «Компас-3D» предназначена для создания трёхмерных моделей отдельных деталей и сборочных единиц. Параметрическая технология позволяет быстро получать модели типовых изделий на основе проектированного ранее прототипа.

Система «Компас-3D» предоставляет следующие возможности:

* проектирование машиностроительных и приборостроительных изделий;
* классическое твердотельное моделирование;
* создание листовых деталей и обечаек;
* проектирование с применением сложных поверхностей;
* формирование электронной модели изделий;
* решение различных задач в архитектурно-строительном и технологическом проектировании;
* проектирование с помощью технологии MinD.

2.2 Описание API

В КОМПАС на данный момент существуют API двух версий: API 5 и API 7. Обе версии реализуют различные функции системы и взаимно дополняют друг друга. Обе версии программных интерфейсов поддерживаются и развиваются с учетом самих изменений в системе. В основном, для создания полноценных подключаемых модулей достаточно методов и свойств интерфейсов API 5.

Главным интерфейсом API системы КОМПАС является KompasObject. Методы этого интерфейса реализуют наиболее общие функции работы с документами системы, системными настройками, файлами, а также дают возможность получить указатели на другие интерфейсы (интерфейсы динамического массива, работы с математическими функциями, библиотек моделей или фрагментов и различных структур параметров определенного типа).

Ниже в таблице 1.1 представлены свойства и методы интерфейса KompasObject, которые были использованы при разработке плагина.

Таблица 1.1 – Методы и свойства интерфейса KompasObject, используемые при разработке плагина.

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Document3D() | ksDocument | Метод для получения указателя на интерфейс трехмерного графического документа (детали или сборки) |
| Visible | Bool | Свойство видимости приложения |
| Quit() |  | Метод для закрытия активного окна приложения КОМПАС |

В таблице 1.2 представлены методы интерфейса ksEnity, которые были использованы при разработке плагина.

Таблица 1.2 – Методы и свойства интерфейса ksEnity, используемые при разработке плагина.

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Create() | bool | Создать объект в модели |
| Update() | Bool | Изменить свойства объекта |

В таблице 1.3 представлены методы интерфейса ksDocument3D, которые были использованы при разработке плагина.

Таблица 1.3 – Методы и свойства интерфейса ksDocument3D, используемые при разработке плагина.

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Create (boll invisible, bool \_typeDoc) | bool | Создать документ – модель (деталь или сборку) |
| GetPart (int type) | ksPart | Получить указатель на интерфейс компонента в соответствии с заданным типом |

В таблице 1.4 представлены типы объектов документа - модели, которые были использованы при разработке плагина.

Таблица 1.4 – типы объектов документа – модели.

|  |  |  |
| --- | --- | --- |
| Идентификатор объекта | Название объекта | Интерфейс параметров |
| o3d\_planeXOZ | Плоскость XOZ | ksPlaneParam |
| o3d\_planeYOZ | Плоскость YOZ | ksPlaneParam |
| o3d\_planeXOY | Плоскость XOY | ksPlaneParam |
| o3d\_sketch | Эскиз трехмерной операции | ksSketchDefinition |
| o3d\_baseExtrusion | Операция выдавливания | ksBaseExtrusionDefinition |

* 1. **Обзор аналогов**

В качестве аналога будет рассмотрен плагин под названием Fasteners для моделирования метрических болтов и гаек. Плагин применяется при изготовлении составных или сборочных деталей в сфере машиностроении для решения задач проектирования в системе «FreeCAD» [2].

После запуска плагина, при выборе какого-либо винта, плагин автоматически создает винт в исходном положении с размерами по умолчанию. После построения данные можно менять на свои.

С помощью данного плагина пользователь может строить и изменять параметры болтов и гаек разных типов и видов, указывая необходимые для построения параметры, такие как:

- диаметр;

- длина;

- включить/Выключить резьбу.

Для экономии ресурсов процессора по умолчанию резьба не отображается.

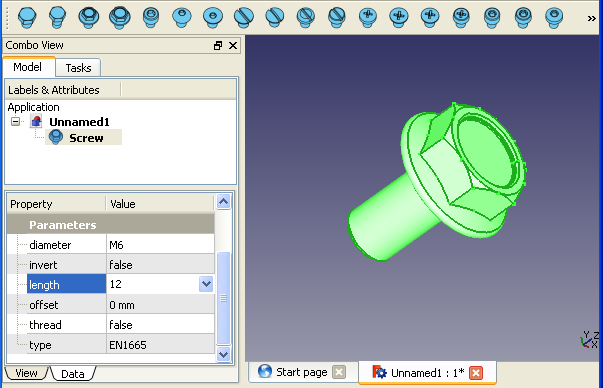


Рисунок 2.1 – Построение болтов

Также можно строить крепежные детали.

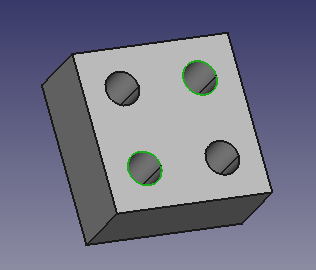


Рисунок 2.2 – Крепежная деталь

Крепежные детали могут быть прикреплены к элементам в других частях. Чтобы прикрепить винт к детали, деталь должна иметь элементы с круговыми краями, например отверстия.

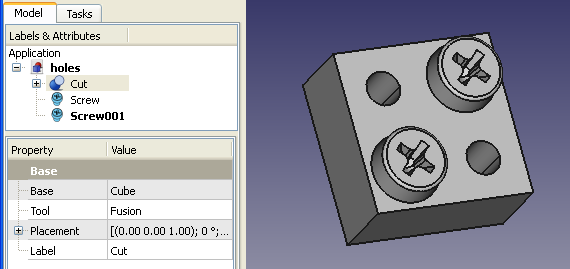


Рисунок 2.3 – Прикрепление винта к крепежной детали

Для данного вида деталей есть возможность настроить:

- свойство смещения;

- свойство инвертирования.

Можно прикреплять винты, болты к деталям, можно делать отверстия, после делать резьбу на болтах и вкручивать болты в отверстия.

## Описание предмета проектирования

Предметом проектирования является модель втулки для катушки. Втулка – плоские или прямоугольные кольца, служащие для прочного и герметичного присоединения их друг к другу, к машинам, аппаратам и ёмкостям, для соединения валов и других вращающихся деталей. Втулка – является телом вращения, которая имеет осевое отверстие, в которое входит сопрягаемая деталь. Данная втулка необходима для катушек, она является важной деталью в строении катушки. Две такие втулки соединяются друг с другом и образуется катушка, на которую наматывают проволоку, леску, нитки и прочее.

Данная втулка должна содержать следующие параметры:

* Внутренний диаметр маленького отверстия D1: от 50мм до 60мм;
* Внешний диаметр маленького отверстия D2: от 80мм до 90мм;
* Внутренний диаметр среднего отверстия D3: от 180мм до 190мм;
* Внешний диаметр среднего отверстия D4: от 195мм до 220мм;
* Внутренний диаметр большого отверстия D5: от 195мм до 220мм;
* Внешний диаметр основания D6: от 230мм до 260мм;
* Длина перемычек L: от 45мм до 55мм;

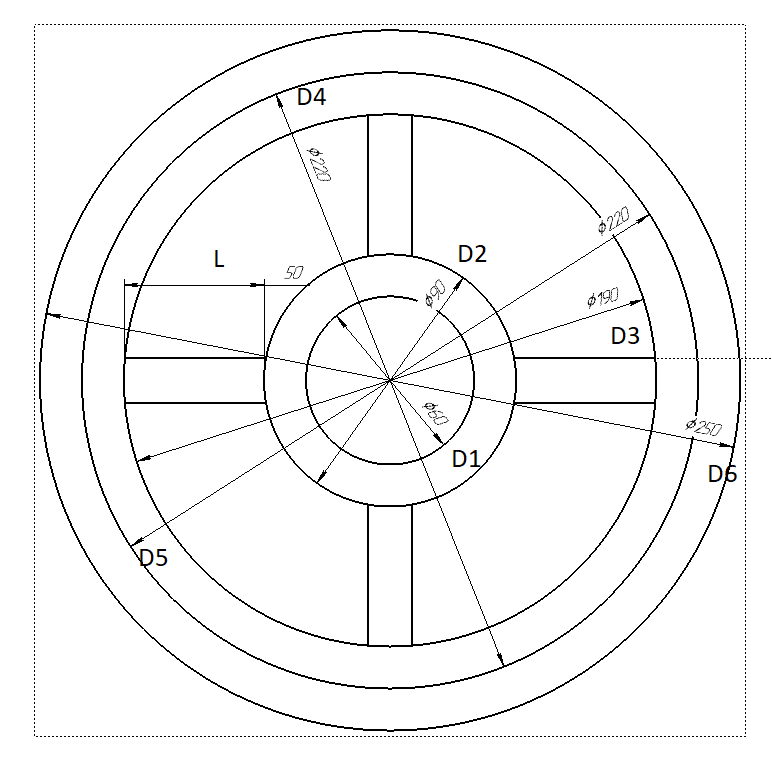


Рисунок 2.4 – Вид детали сверху с обозначенными параметрами

* Высота маленького отверстия H1: от 10мм до 35мм;

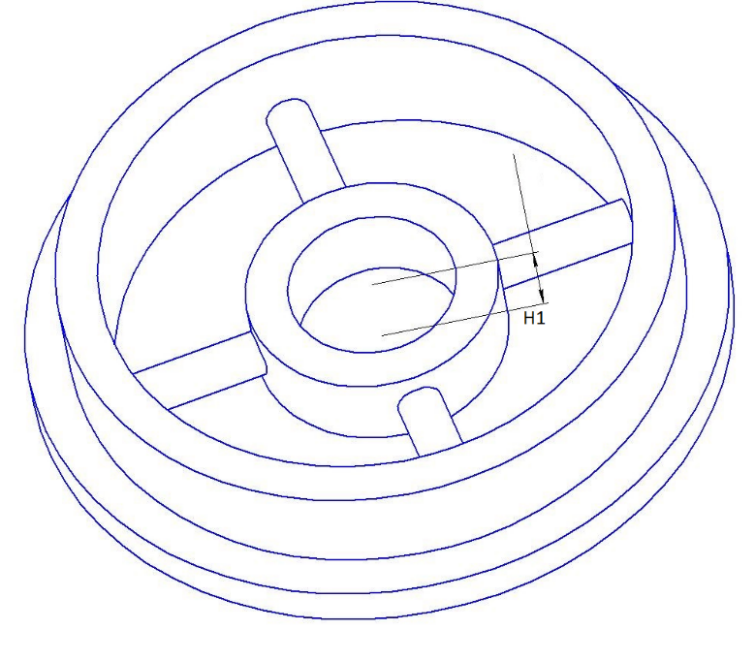


Рисунок 2.5 – Произвольный вид детали с обозначенными параметрами

* Высота среднего отверстия H2: от 10мм до 35мм;
* Высота большого отверстия H3: от 10мм до 35мм;
* Общая высота втулки H4: от 10мм до 35мм;

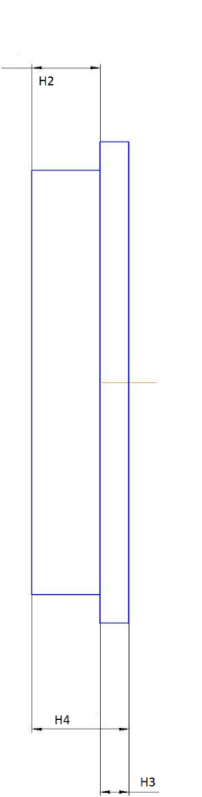


Рисунок 2.6 – Вид детали сбоку с обозначенными параметрами

* Диаметр перемычек D7: от 45мм до 55мм.

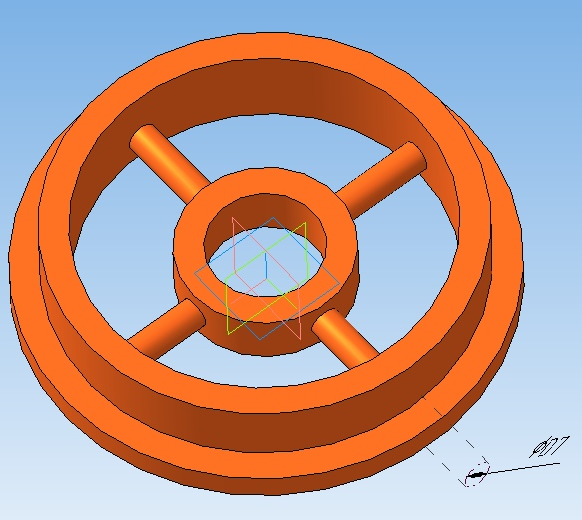


Рисунок 2.7 – Обозначение диаметра перемычек

На рисунке 2.8 представлен вид на 3D модель втулки.

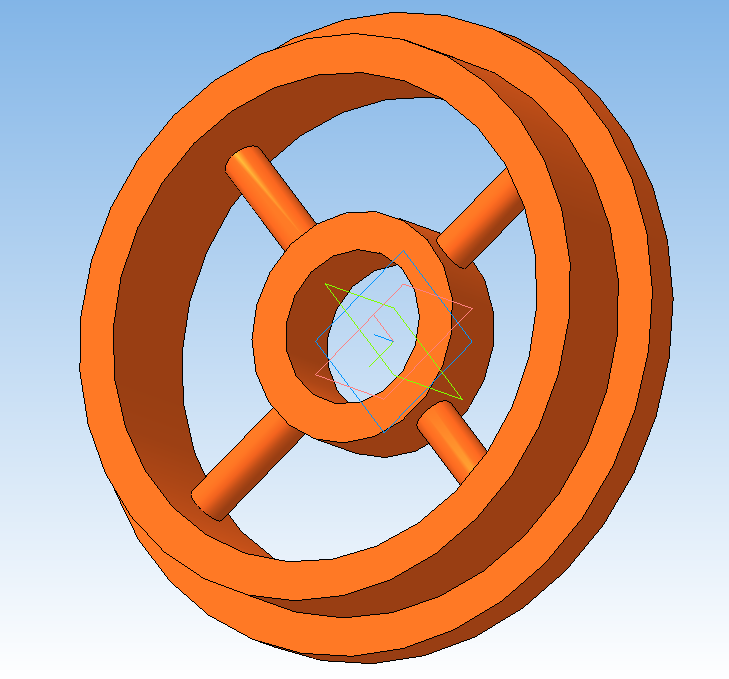


Рисунок 2.8 – Вид на 3D модель втулки

1. **Описание реализации**

Для описания архитектуры пользовательского сценария системы был выбран унифицированный язык моделирования (UML) [3]. На основе UML построены диаграммы вариантов использования и диаграммы классов.

Также одним из важных моментов при проектировании программы является внешний вид программы. Внешний вид программы должен максимально отражать функциональные особенности программы, чтобы пользователь мог за минимальное время освоить программу и пользоваться ею.

В процессе реализации диаграммы классов и вариантов использования были дополнены, так как был увеличен функционал.

## 3.1 Диаграмма вариантов использования

Диаграммы вариантов использования описывают взаимоотношения и зависимости между группами вариантов использования и действующих лиц, участвующими в процессе. Диаграммы вариантов использования предназначены для упрощения взаимодействия с будущими пользователями системы, с клиентами, и особенно пригодятся для определения необходимых характеристик системы. Другими словами, диаграммы вариантов использования говорят о том, что система должна делать, не указывая сами применяемые методы. Любые системы проектируются с учетом того, что в процессе своей работы они будут использоваться людьми и/или взаимодействовать с другими системами. Сущности, с которыми взаимодействует система в процессе своей работы, называются actor, причем каждый actor ожидает, что система будет вести себя строго определенным, предсказуемым образом.

**Actor** – это множество логически связанных ролей, исполняемых при взаимодействии с прецедентами или сущностями (система, подсистема или класс). **Actor** может быть человек или другая система, подсистема или класс, которые представляют нечто вне сущности.

**Прецедент (use - case)** - описание множества последовательных событий, выполняемых системой. Прецедент представляет поведение сущности, описывая взаимодействие между actorи системой. Прецедент не показывает, "как" достигается некоторый результат, а только "что" именно выполняется [3,4].

На рисунке 3.1 представлена диаграмма вариантов использования.

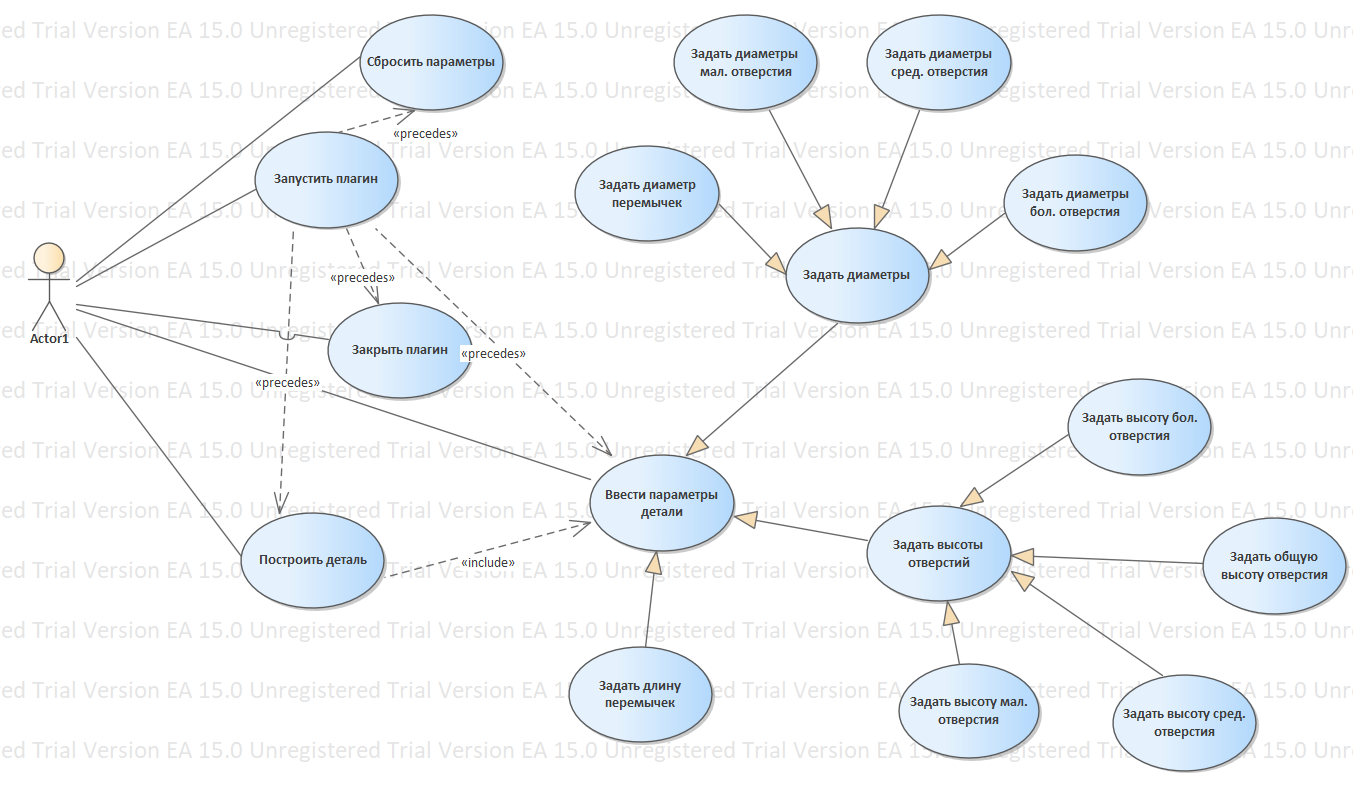


Рисунок 3.1 – Начальная диаграмма вариантов использования

В программу была добавлена дополнительная функциональность в виде возможности включения резьбы и изменения ее типа, поэтому диаграмма вариантов использования приобрела новый вид, представленный на рисунке 3.2.

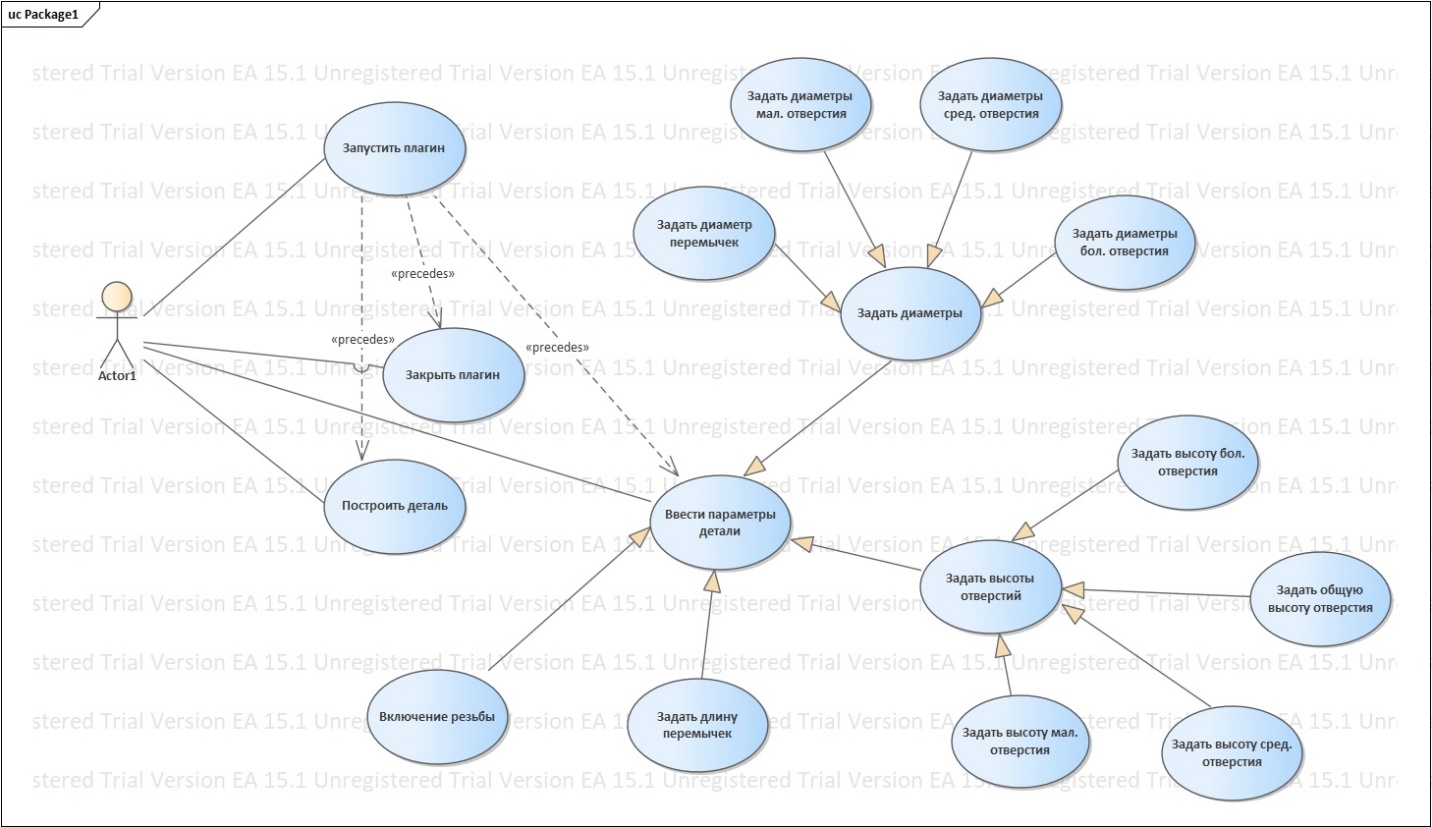


Рисунок 3.2 – Конечная диаграмма вариантов использования

**3.2 Диаграмма классов**

***Диаграмма*** – это графическое представление набора элементов, чаще всего изображенного в виде сущностей. Язык UML включает **13** видов диаграмм, среди которых на первом месте в списке — диаграмма классов.  
Диаграммы классов показывают набор классов, интерфейсов, а также их связи. Диаграммы этого вида чаще всего используются для моделирования объектно-ориентированных систем. Они предназначены для статического представления системы. Большинство элементов UML имеют уникальную и прямую графическую нотацию, которая дает визуальное представление наиболее важных аспектов элемента [3, 4].

**Класс** – это описание набора объектов с одинаковыми атрибутами, операциями, связями и семантикой. Графически класс изображается в виде прямоугольника, разделенного на 3 блока горизонтальными линиями:

* имя класса
* атрибуты (свойства) класса
* операции (методы) класса.

Ниже представлена начальная UML – диаграмма классов (рисунок 3.3).

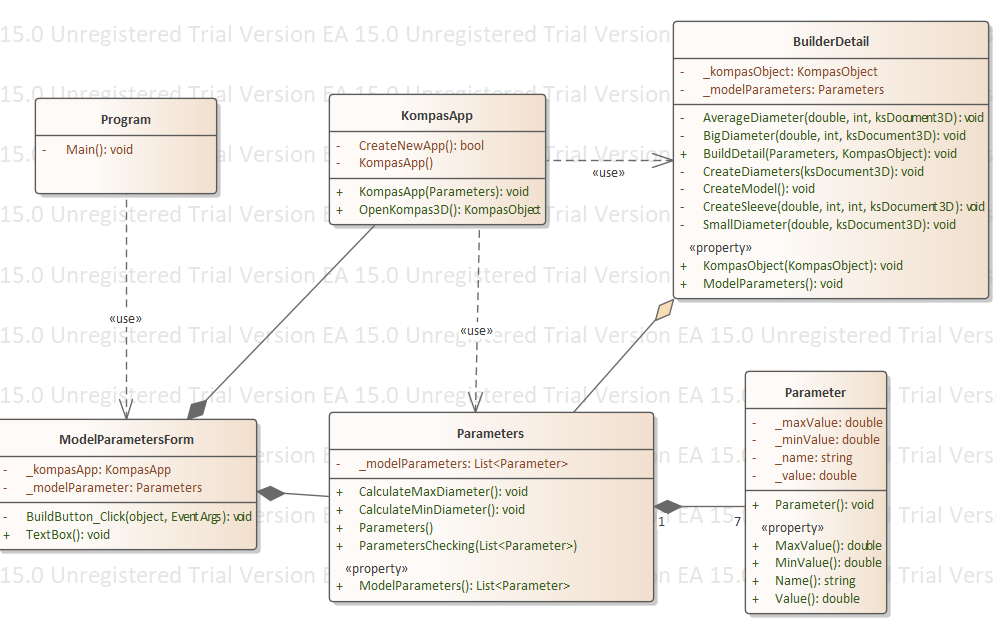


Рисунок 3.3 – Начальная диаграмма классов.

Ввиду внесенных изменений (добавление дополнительной функциональности, добавление дополнительных полей, свойств и методов в ранее созданные классы), диаграмма классов приобрела новый вид, представленный на рисунке 3.4.

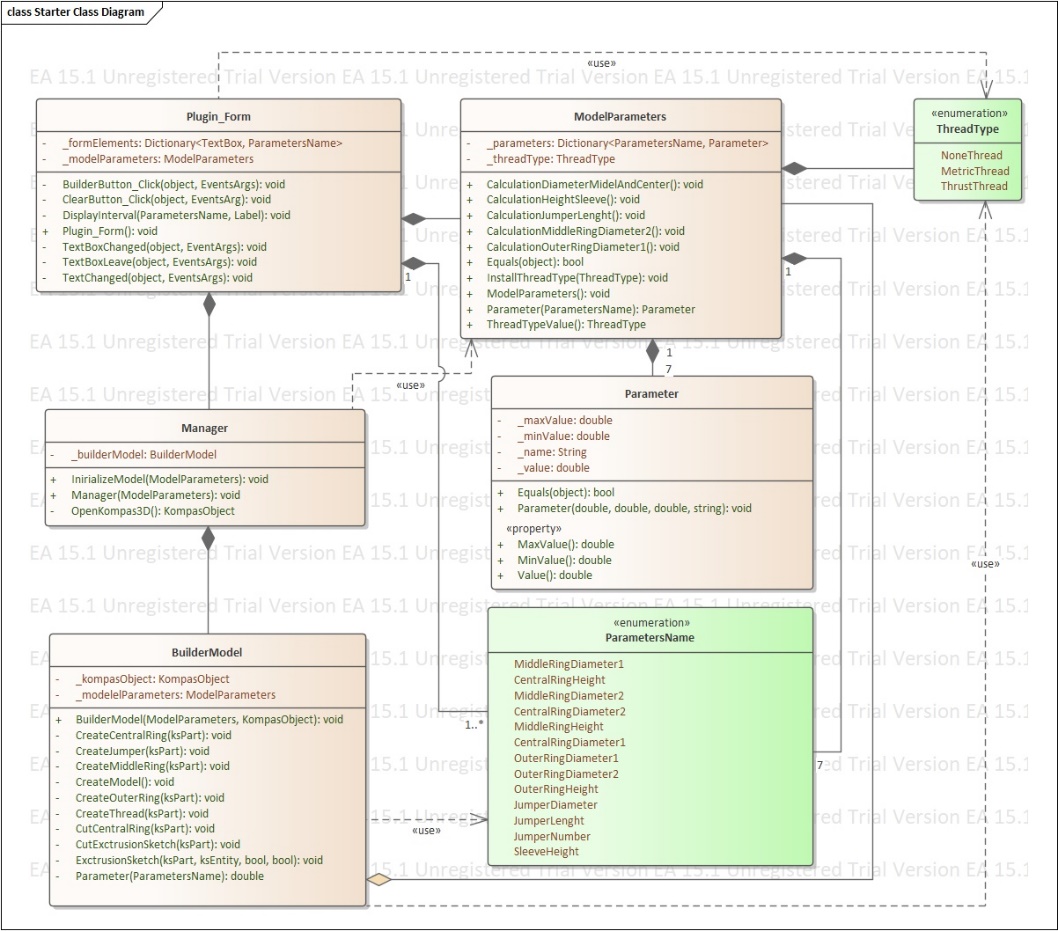


Рисунок 3.4 – Конечная диаграмма классов

**3.3 Макет пользовательского интерфейса**

Пользовательский интерфейс – это набор программных и аппаратных средств, обеспечивающих взаимодействие пользователя с компьютером. [5].

Плагин представляет собой форму с полями для ввода параметров. Запуск построения осуществляется кнопкой «Построить». На рисунке 3.5 представлен начальный макет плагина.

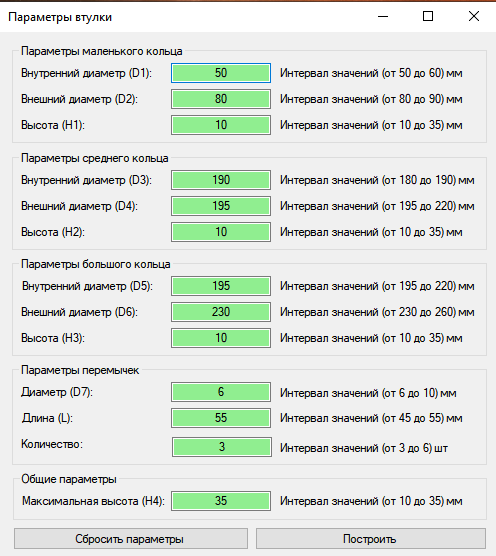
****

Рисунок 3.5 – Начальный макет пользовательского интерфейса

Группа «Параметры маленького кольца» - в данной группе находятся поля для заполнения данных о маленьком отверстии.

Группа «Параметры среднего кольца» - в данной группе находятся поля для заполнения данных о среднем отверстии.

Группа «Параметры большого кольца» - в данной группе находятся поля для заполнения данных о большом отверстии.

Группа «Параметры перемычек» - в данной группе находятся поля для заполнения данных о перемычках.

Группа «Общие параметры» - группа доп. параметров.

Группа «Построить» - Кнопка, отвечающая за построение детали.

Группа «Сбросить параметры» - Кнопка, отвечающая за сброс введенных ранее параметров детали.

На рисунке 3.6 представлен конечный макет пользовательского интерфейса плагина. Добавлена возможность добавления резьбы на маленькое кольцо, а также изменение типа резьбы.

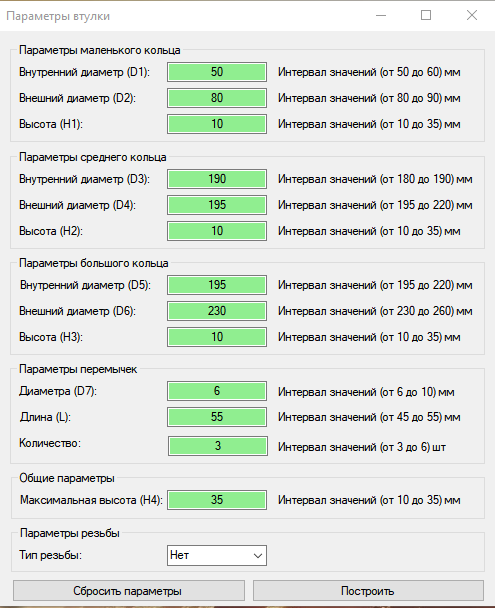


Рисунок 3.6 – Конечный макет пользовательского интерфейса

Группа «Параметры резьбы» - в данной группе находится поле для выбора резьбы или же ее отключение.

1. **Описание программы для пользователя**

При запуске плагина перед пользователем появляется окно, представленное на рисунке 4.1. Для того, чтобы пользователь мог построить втулку для катушки, ему необходимо нажать на кнопку «Построить». Если он хочет изменить параметры для построения, то ему нужно ввести новые данные в поля для ввода и нажать на кнопку «Построить». Также у пользователя есть дополнительная возможность построить резьбу внутри маленького кольца и изменить ее вид.

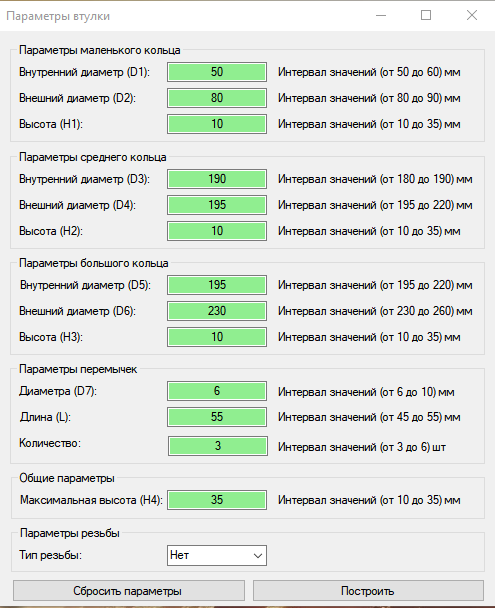


Рисунок 4.1 – Пользовательский интерфейс плагина

Чтобы снизить количество ошибок при вводе параметров, изначально поля заполнены рекомендуемыми значениями. Пользователь может их самостоятельно изменять как ему угодно, но в допустимых пределах, которые указаны рядом с полями для ввода параметров. Если пользователь решит построить деталь по рекомендуемым параметрам, то ему нужно просто запустить плагин и нажать кнопку «Построить». Далее осуществится запуск САПР «Компас-3D», на рабочей области которой построится трехмерная модель детали по рекомендуемым параметрам (рисунок 4.2). Если построение детали выполняется несколько раз, то плагин создает в САПР новый документ.

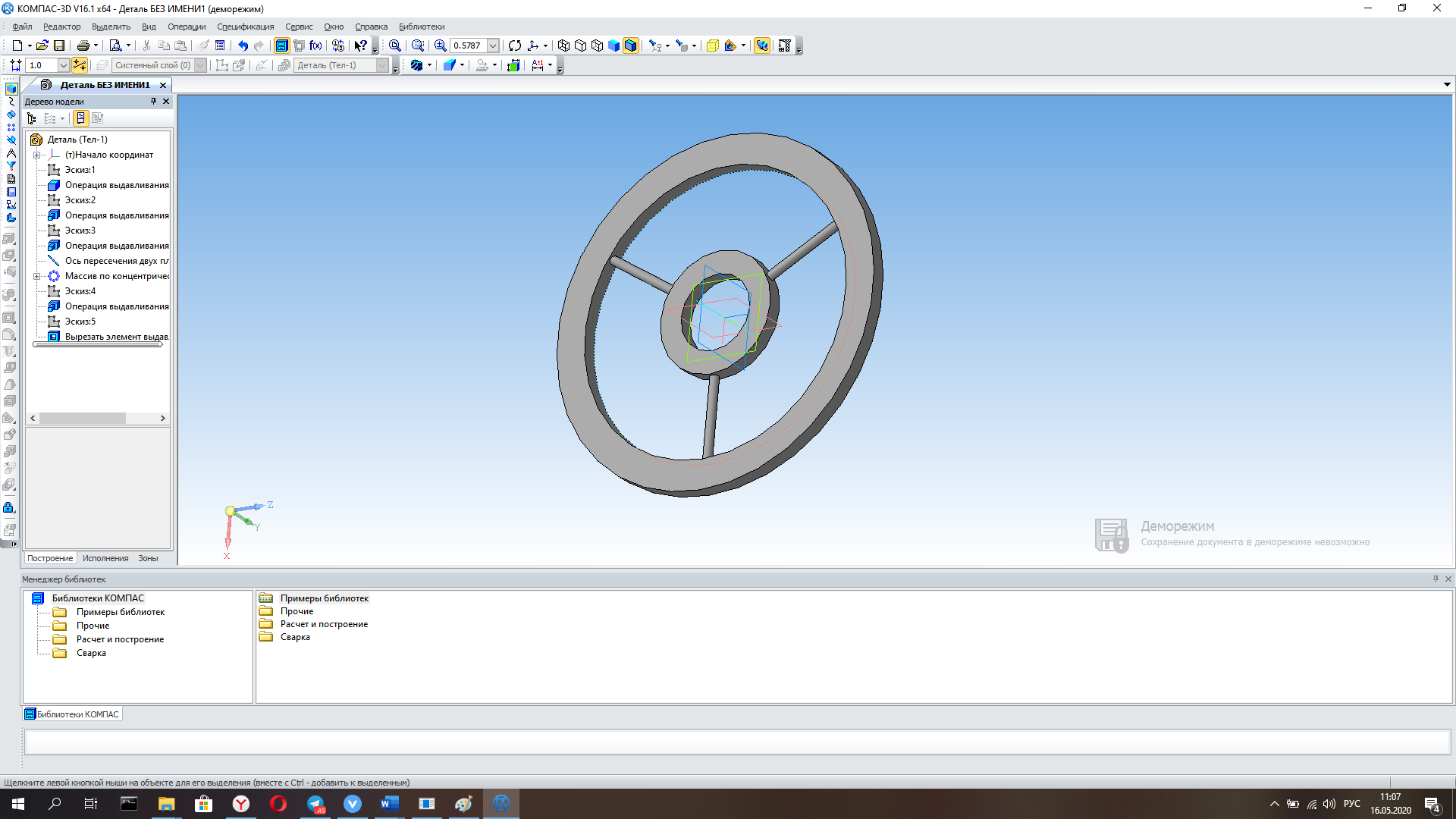


Рисунок 4.2 – Диалоговое окно документа САПР «Компас-3D»

1. **Тестирование плагина**

Тестирование программы позволяет проверить соответствия между реальным и ожидаемым поведением программы [6].

**5.1 Функционально тестирование**

При функциональном тестировании [7] проверяется корректность работы плагина, а именно получение верного результата в виде трехмерной модели.

Произведено функционально тестирование построения втулки для катушки в САПР «Компас-3D» при минимальных и максимальных параметрах, результаты построения приведены на рисунках 5.1 и 5.1. Результат построения модели, соответствующей техническому заданию представлен на рисунке 5.3. На рисунке 5.4 – 5.5 продемонстрирована дополнительная функциональность.

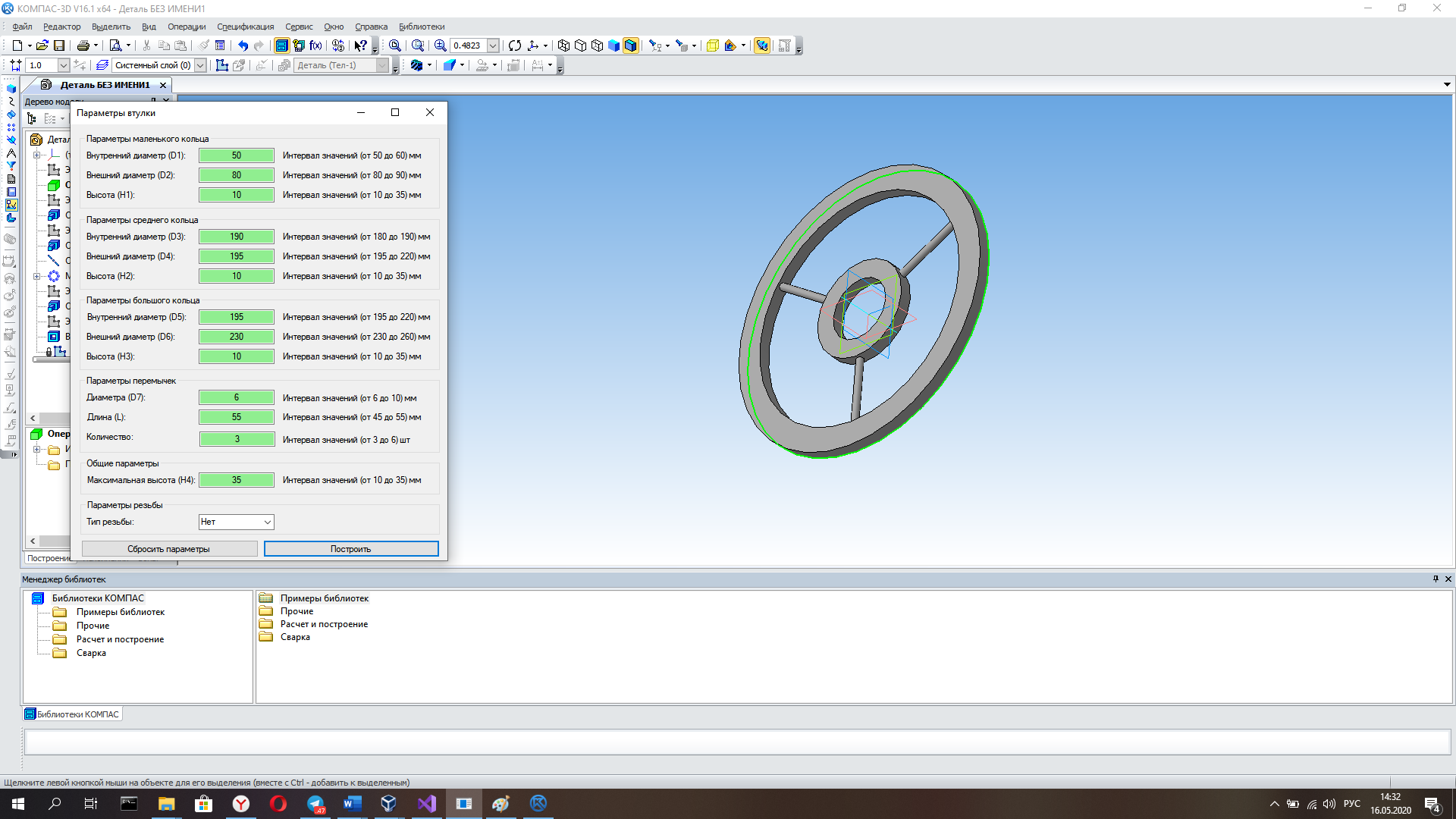


Рисунок 5.1 – Модель втулки для катушки с минимально заданными параметрами

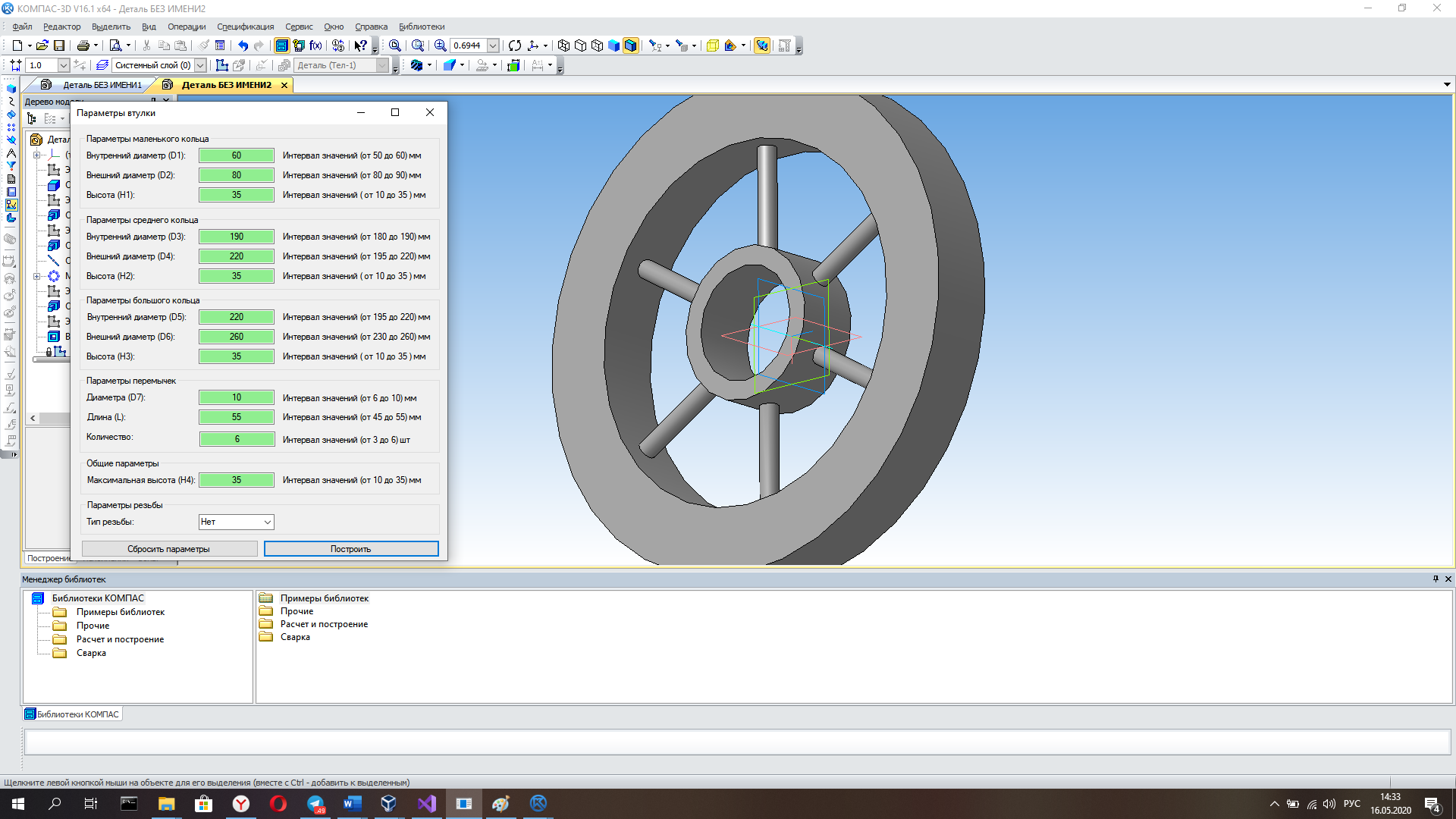


Рисунок 5.2 – Модель втулки для катушки с максимально заданными параметрами

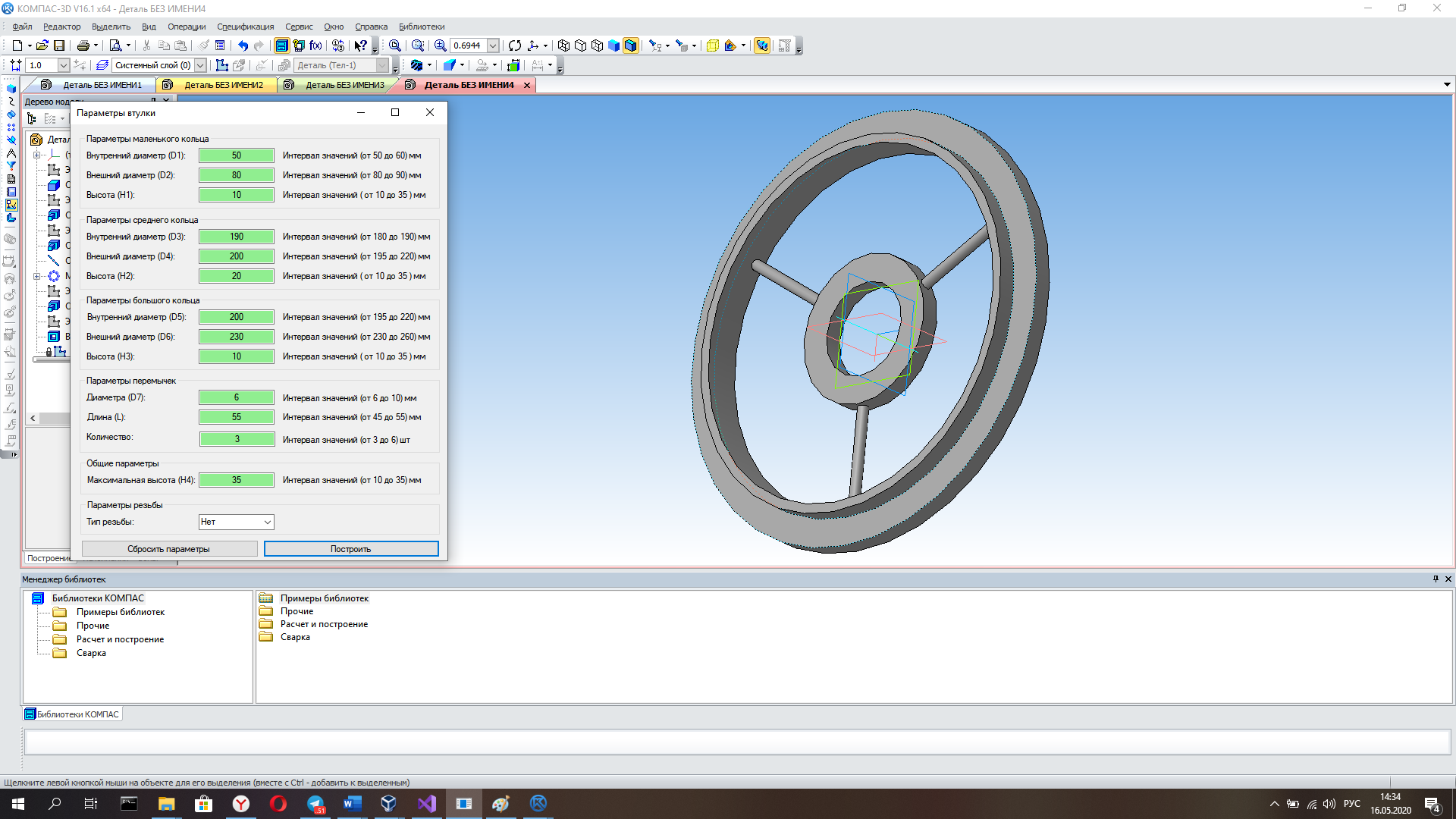


Рисунок 5.3 – Модель втулки для катушки с оптимально заданными параметрами

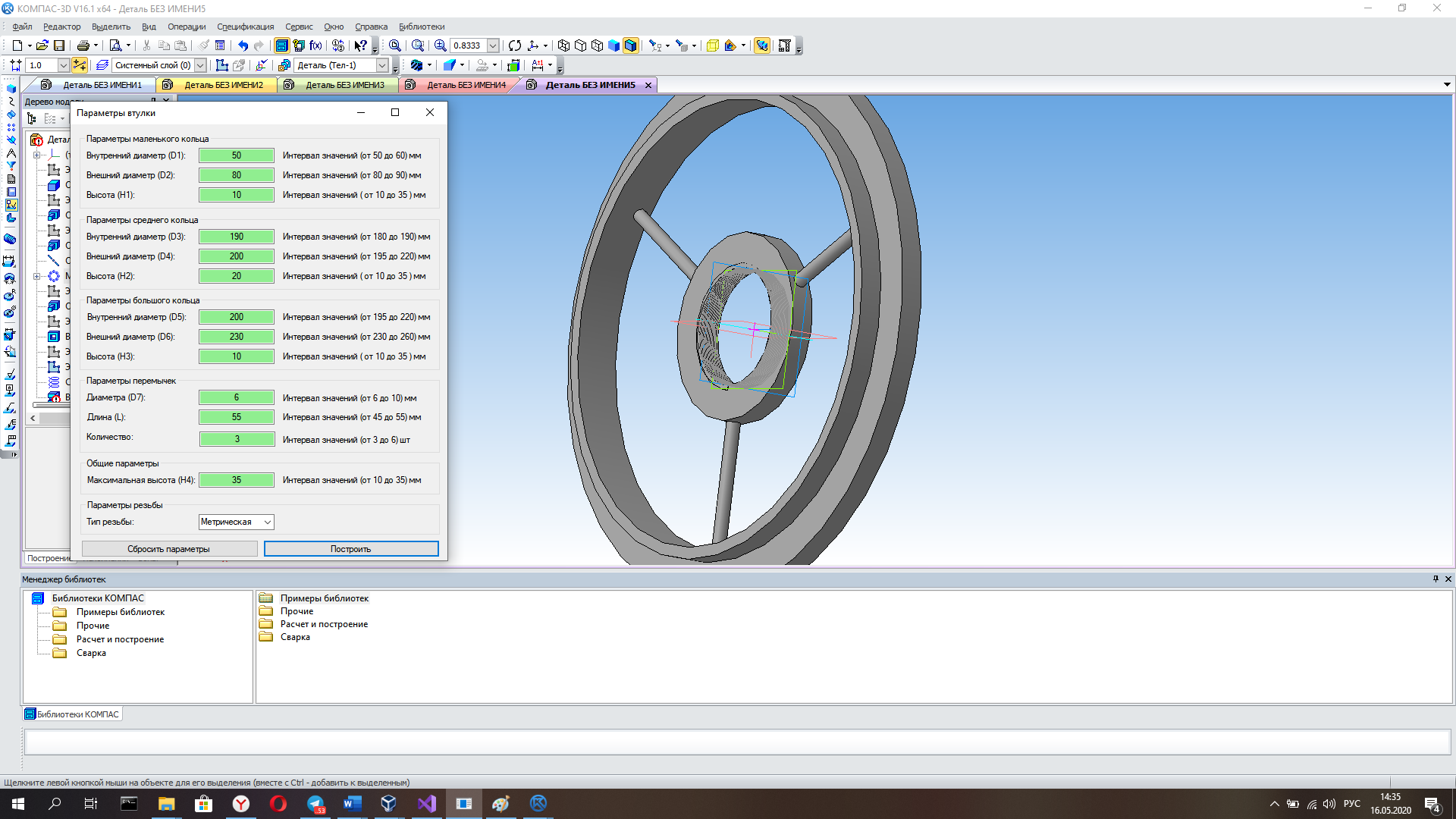


Рисунок 5.4 – Модель втулки с дополнительной функциональностью с метрической резьбой

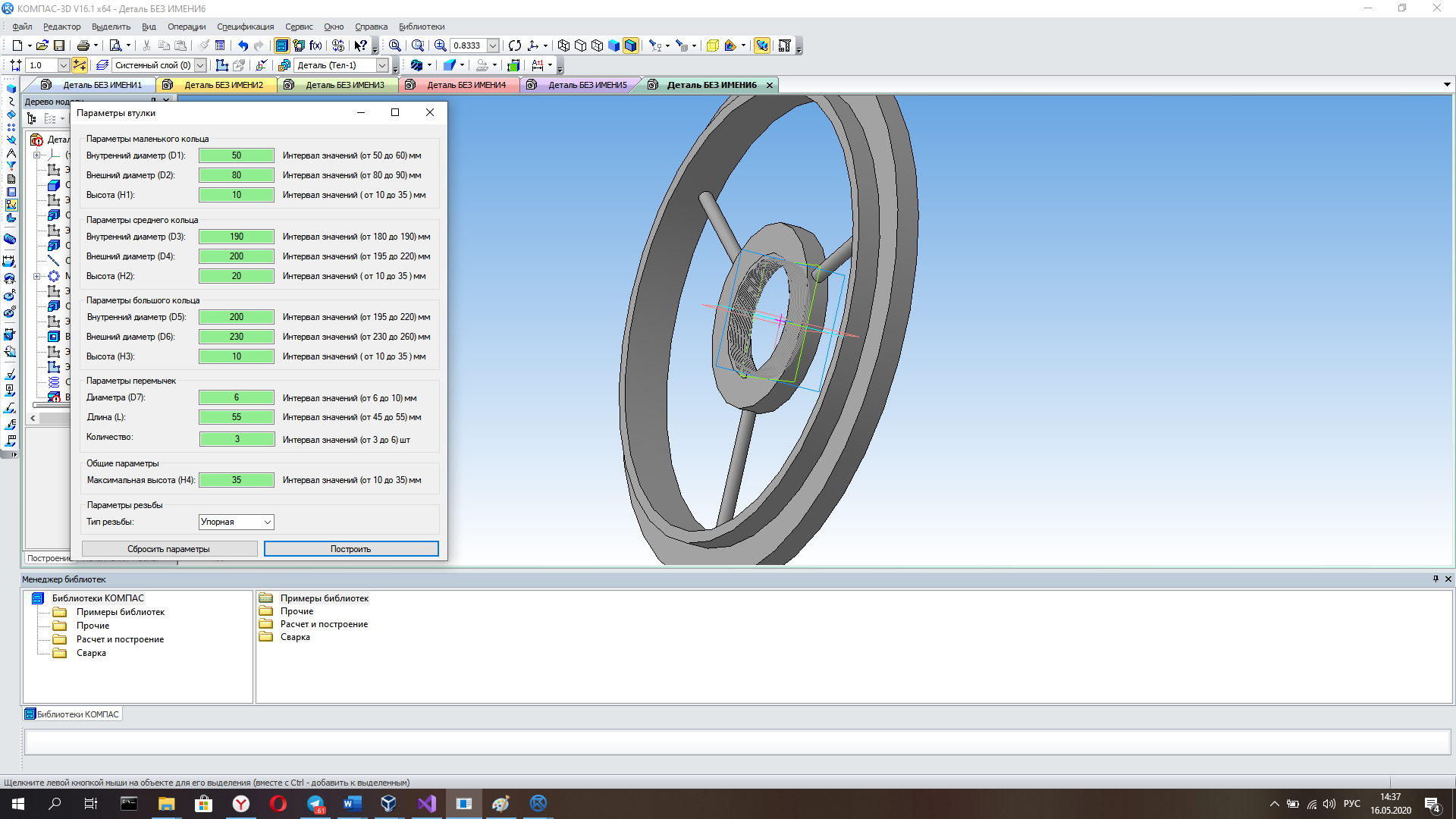


Рисунок 5.5 – Модель втулки с дополнительной функциональностью с упорной резьбой

Если пользователь введет ошибочные параметры, то плагин выделит поле с ошибкой красным цветом. А также, при потере фокуса полем, значение поля пример последнее корректно введенное значение. Допустимый интервал вводимых значений отображается рядом с полями для ввода параметров.

Диалоговое окно с введенным некорректным значением и отображенным интервалом параметров изображено на рисунке 5.6.

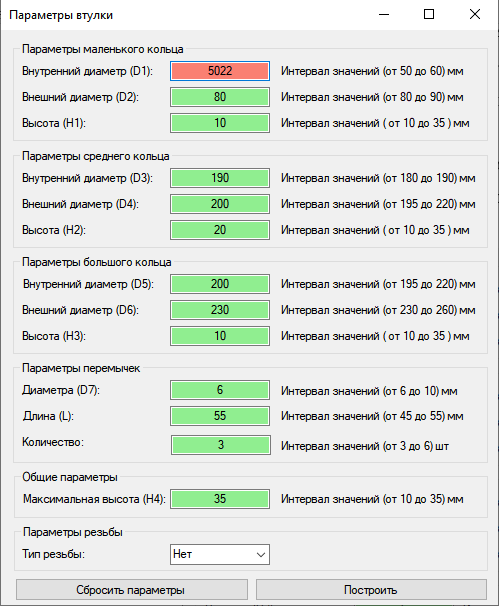
****

Рисунок 5.6 – Ошибочно введенный параметр

**5.2 Модульное тестирование**

В целях проверки корректности работы методов и свойств классов при помощи обозревателя тестов Visual Studio было проведено модульное тестирование [8].

Для проверки открытых членов классов ModelParameters и Parameter в модульном тестировании использовались тестовые методы, которые описаны в таблице 5.1.

Таблица 5.1 – Название и описание используемых в тестировании методов классов ModelParameters и Parameter

|  |  |
| --- | --- |
| Название тестового метода | Описание |
| Test\_CalculationJumperLenght | Метод для проверки корректности расчета длин перемычки |
| Test\_CalculationMiddleRingD2 | Метод для проверки корректности расчета среднего кольца |
| Test\_CalculationOuterRingD1 | Метод для проверки корректности расчета большого кольца |
| Test\_ModelParameters | Метод для проверки корректности работы конструктора |
| Test\_Parameter | Метод для проверки корректности возвращаемых методом Parameter данных |
| Test\_ToString | Метод для проверки корректности преобразования элемента перечисления в строку |

Продолжение таблицы 5.1

|  |  |
| --- | --- |
| Test\_CalculationDMidelAndCenter | Метод проверки корректности расчета модели и центра |
| Test\_CalculationHeightSleeve | Тестирование расчета высоты втулки |
| Test\_MaxValue\_Get\_CorrectValue | Метод для проверки корректности возврата максимального значения |
| Test\_MaxValue\_Set\_CorrectValue | Метод для проверки корректности записи максимального значения |
| Test\_MinValue\_Get\_CorrectValue | Метод для проверки корректности возврата минимального значения |
| Test\_MinValue\_Set\_CorrectValue | Метод для проверки корректности записи минимального значения |
| Test\_Parameter\_Designer | Метод для проверки корректности работы конструктора |
| Test\_Value\_Get\_CorrectValue | Метод для проверки корректности возврата текущего значения |
| Test\_Value\_Set\_CorrectValue | Метод для проверки корректности записи текущего значения |
| TestLastModTimeSet\_ArgumentException | Метод для проверки вызова исключения при записи некорректных значения |

Результаты успешного прохождения всех модульных тестов приведены на рисунке 5.7.

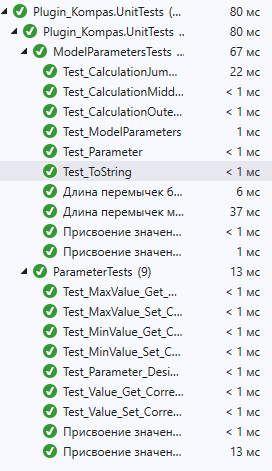
****

Рисунок 5.7 – Результат выполнения модульных тестов

* 1. **Нагрузочное тестирование**

В целях проверки производительности работы плагина, было проведено нагрузочное тестирование [9]. Нагрузочное тестирование – это тестирование производительности, определение производительности и времени отклика устройства в ответ на внешний запрос с целью установления соответствия требованиям, которые были предъявлены к данному устройству.

Аппаратная конфигурация ПК, на котором проводилось нагрузочное тестирование:

* процессор: AMD Ryzen 7 3750H 2.30 ГГц;
* графический контроллер: GeForce GTX 1650;
* оперативная память: 16 ГБ.
* система: Windows 10 x64.

Нагрузочное тестирование проводилось со следующими параметрами втулки:

* внутренний диаметр маленького кольца: 50мм;
* внешний диаметр маленького кольца: 80мм;
* высота маленького кольца: 10мм;
* внутренний диаметр среднего кольца: 190мм;
* внешний диаметр среднего кольца: 195мм;
* высота среднего кольца: 10мм;
* внутренний диаметр большого кольца: 195мм;
* внешний диаметр среднего кольца: 230мм;
* высота большого кольца: 10мм;
* диаметр перемычек: 6мм;
* длина перемычек: 55мм;
* количество перемычек: 3
* максимальная высота втулки: 35мм;
* тип резьбы: метрическая;

После построения 150 детали программа завершилась со сбоем из-за нехватки памяти.

На рисунке 5.8 представлен график потребляемой оперативной памяти относительно количества построенных деталей.

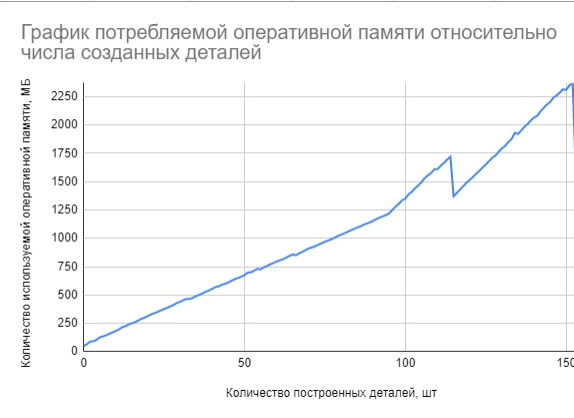
****

Рисунок 5.8 – График зависимости загрузки программой центрального процессора от количества построенных деталей

График зависимости загрузки программой центрального процессора от количества построенных деталей представлен на рисунке 5.9.

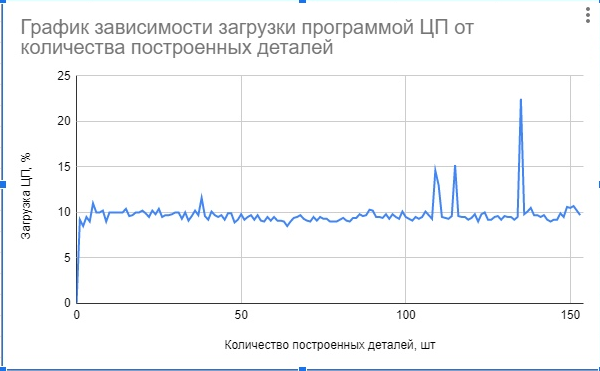


Рисунок 5.9 – График зависимости загрузки программой центрального процессора от количества построенных деталей

**Заключение**

В ходе выполнения данного проекта были изучены основные этапы разработки ПО на примере создания библиотеки для построения детали «Втулка для катушки» для САПР «КОМПАС-3D V 16».

В разработанной библиотеке предусмотрена проверка вводимых данных на корректность, построение детали по верно заданным параметрам, подсветка неверно заполненных полей и сброс на верные значения.

Проведены модульные, функциональные и нагрузочные тесты разработанной библиотеки.

**Список используемых источников**

1. Компас 3d [Электронный ресурс]. – URL: <https://kompas.ru/kompas-3d/about/#about> (Дата обращения: 24.02.2020).
2. Плагин для построения болтов и крепежных деталей “Fasteners” [Электронный ресурс]. – URL: [http://theseger.com/projects/2015/06/fasten ers-workbench-for-freecad/](http://theseger.com/projects/2015/06/fasten%20ers-workbench-for-freecad/) (Дата обращения: 24.02.2020).
3. UML. Основы / Фаулер, М. – 3-е изд., пер. с англ. – СПб: Символ-Плюс, 2004. – 192 с.
4. Новые технологии в программировании: учебное пособие / А.А. Калентьев, Д. В. Гарайс, А. Е. Горяинов. – Томск, 2014. − ­ 176 стр.
5. Интерфейс пользователя – Национальная библиотека им. Н. Э. Баумана [Электронный ресурс] – Режим доступа: https://ru.bmstu.wiki/Интерфейс\_пользователя (дата обращения: 16.05.2020)
6. Про Тестинг – Тестирование программного обеспечения [Электронный ресурс] – Режим доступа: http://www.protesting.ru/testing/ (дата обращения: 16.05.2020)
7. Функциональное тестирование [Электронный ресурс]. – URL: <https://daglab.ru/funkcionalnoe-testirovanie-programmnogo-obespechenija/> (дата обращения: 16.05.2020);
8. Модульное тестирование [Электронный ресурс]. – URL: <http://espressocode.top/unit-testing-software-testing/> (дата обращения: 16.05.2020);
9. Нагрузочное тестирование [Электронный ресурс]. – URL: <https://daglab.ru/nagruzochnoe-testirovanie-programmnogo-obespechenija/> (дата обращения: 16.05.2020);