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

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра компьютерных систем в управлении и проектировании (КСУП)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА  
по дисциплине  
«Основы разработки САПР» (ОРСАПР)

Выполнил:

студент гр. 588-2

\_\_\_\_\_\_\_\_\_\_Е.И. Белов

Руководитель:

к.т.н., доцент каф. КСУП

\_\_\_\_\_\_\_\_\_\_А.А. Калентьев

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2021 г.

Оглавление

[1 Введение 3](#_Toc93321445)

[2 Постановка и анализ задачи 4](#_Toc93321446)

[2.1 Описание предмета проектирования 5](#_Toc93321447)

[2.2 Выбор инструментов и средств реализации 7](#_Toc93321448)

[2.3 Назначение плагина 8](#_Toc93321449)

[3 Обзор аналогов 9](#_Toc93321450)

[4 Описание реализации 10](#_Toc93321451)

[4.1 Диаграмма классов 10](#_Toc93321452)

[5 Описание программы для пользователя 13](#_Toc93321453)

[6 Тестирование программы 15](#_Toc93321454)

[6.1 Функциональное тестирование 15](#_Toc93321455)

[6.2 Модульное тестирование 16](#_Toc93321456)

[6.3 Нагрузочное тестирование 17](#_Toc93321457)

[Заключение 21](#_Toc93321458)

[Список использованной литературы 22](#_Toc93321459)

# Введение

Автоматизация моделирования имеет огромное значение для развития науки, техники и производства в современном обществе. В настоящее время автоматизация – основной способ повышения производительности и эффективности труда инженерно-технических работников, занимающихся моделированием сложных устройств. Использование автоматизации в проектировании позволяет создавать все более сложные технические объекты и гибко реагировать на появление новых решений и технологий в той или иной области техники. Она позволяет значительно повысить точность расчетов, выбрать наилучшие варианты для реализации на основе строгого математического анализа всех или большинства вариантов проекта с оценкой технических, технологических и экономических характеристик производства и эксплуатации проектируемого объекта, значительно повысить качество конструкторской документации, существенно сократить сроки проектирования и передачи конструкторской документации в производство, эффективнее использовать технологическое оборудование с программным управлением [1].

Таким образом, целью данной работы является разработка плагина, автоматизирующего построение модели «Крышка» для системы автоматизированного проектирования Компас3D V20 с помощью интегрированной среды разработки Visual Studio 2019 [2].

Интегрированная среда разработки Visual Studio — это стартовая площадка для написания, отладки и сборки кода, а также последующей публикации приложений. Интегрированная среда разработки (IDE) представляет собой многофункциональную программу, которую можно использовать для различных аспектов разработки программного обеспечения.

# Постановка и анализ задачи

В рамках лабораторных работ в соответствии с технически заданием требовалось разработать плагин, который на основе входных параметров, интегрируя с системой Компас3D V20, строит модель «Крышка». Необходимо чтобы плагин позволял задавать параметры по умолчанию, а также изменять входные параметры забора, такие как:

* диаметр крышки;
* диаметр малого ступенчатого отверстия крышки;
* диаметр большого ступенчатого отверстия крышки;
* диаметр малых отверстий;
* диаметр внешней ступени;
* толщина крышки;
* высота ступени крышки;
* высота внутренней ступени крышки;
* диаметр окружности, на которой лежат малые отверстия;
* количество малых отверстий.

## Описание предмета проектирования

Предметом проектирования является крышка.

Крышка – это деталь тело вращения. Конструкция детали представляет собой взаимное пересечение цилиндрических поверхностей. Крышка предназначена для обеспечения герметизации корпуса редуктора, предотвращения вытекания масла и попадания в корпус редуктора абразивных частиц, а также фиксации наружного кольца подшипника. Вследствие этого крышка должна обладать необходимой герметичностью и прочностью.

Основным предназначением торцевой крышки корпуса является фиксация подшипника в посадочном месте разъемного корпуса [3].

На рисунках 2.1 – 2.3 представлены чертежи крышки.



Рисунок 2.1 – Чертёж крышки: вид снизу

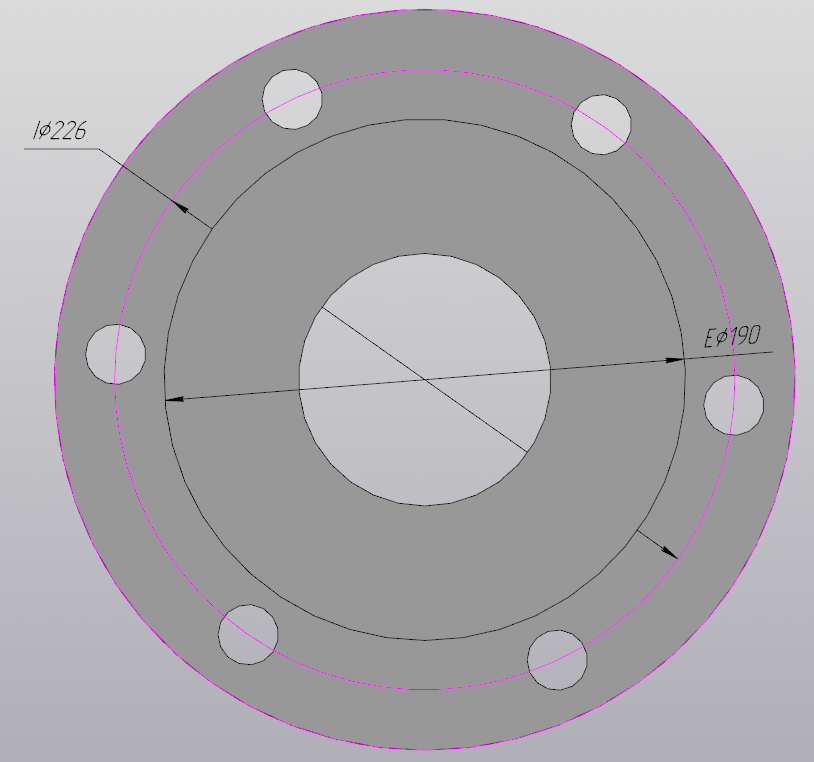


Рисунок 2.2 – Чертёж крышки: вид сверху



Рисунок 2.3 – Чертёж крышки: вид сбоку

Параметры крышки:

* диаметр крышки A (рисунок 2.1): 50 мм – 500мм;
* диаметр малого ступенчатого отверстия крышки B (рисунок 2.1): 15 мм – 330 мм;
* диаметр большого ступенчатого отверстия крышки C (рисунок 2.1): 20 мм – 335 мм;
* диаметр малых отверстий D (рисунок 2.1): 2 мм – 20 мм;
* диаметр внешней ступени E (рисунок 2.2): 35 мм – 350 мм;
* толщина крышки F (рисунок 2.3): 6 мм – 60 мм;
* высота ступени крышки G (рисунок 2.3): 4 мм – 40 мм;
* высота внутренней ступени крышки H (рисунок 2.3): 5 мм – 50 мм;
* диаметр окружности, на которой лежат малые отверстия I (рисунок 2.2): 380 мм – 470 мм;
* количество малых отверстий (рисунок 2.2): 2 – 8 шт.

Зависимые параметры крышки:

* расстояние между внешней ступень крышки (Е) малыми отверстиями (D) должно быть больше 2 мм;
* расстояние между малым ступенчатым отверстием (B) и большим ступенчатым отверстием (C) должно быть больше 5 мм;
* расстояние между малыми отверстиями (D) и диаметром крышки (A) должно быть больше 2 мм;
* расстояние между большим ступенчатым отверстием (C) и внешней ступени крышки (E) должно быть больше 15 мм;
* расстояние между высотой ступенчатой крышки (G) и толщиной крышки (F) должно быть больше 2 мм.

## Выбор инструментов и средств реализации

На основе требований к техническому заданию программа выполнена на языке программирования C# в среде Microsoft Visual Studio 2019 с использованием .NET 4.7.2 Framework, библиотеки для Kompas 3D.

Инструментом тестирования и создания модульных тестов был выбран тестовый фреймворк NUnit 3.12 [4].

Для реализации пользовательского интерфейса использовалась система для построения настольные приложения Windows Forms.

## Назначение плагина

Назначение разрабатываемого плагина обусловлено быстрым моделированием крышки с разными входными параметрами. Благодаря данному расширению, проектировщики могут наглядно рассмотреть спроектированную модель, при необходимости перестроить под необходимые им параметры.

# Обзор аналогов

**Mechanical Desktop Power Pack**

Mechanical Desktop Power Pack – это мощная, динамично развивающаяся инженерная система автоматизации проектирования самых разнообразных объектов – от простейших деталей и узлов до сложных машиностроительных механизмов. [5]

Она позволяет:

* автоматизировать процесс разработки изделия;
* управлять взаимным расположением элементов конструкции с автоматическим обновлением модели и чертежа в процессе внесения в них изменений;
* работать с трёхмерными поверхностями и строить сложные модели с произвольными пространственными формами;
* создавать сложные параметрические твердотельные модели, формообразующие кривые, пространственные NURBES–поверхности и тонкие оболочки;
* генерировать плоские проекции, формировать чертежи изделия;
* управление размерами деталей и узлов и многое другое.

На рисунке 3.1 представлен пример создания крышки в программе Mechanical Desktop Power Pack.



Рисунок 3.1 – Пример интерфейса Mechanical Desktop   
Power Pack и готовой модели

# Описание реализации

Для графического описания абстрактной модели проекта, а также пользовательского взаимодействия (сценарии действия) использован стандарт UML.

UML язык графического описания для объектного моделирования в области разработки программного обеспечения. UML является языком широкого профиля, это – открытый стандарт, использующий графические обозначения для создания абстрактной модели системы, называемой UML –моделью. UML был создан для определения, визуализации, проектирования и документирования, в основном, программных систем. UML не является языком программирования, но на основании UML возможна генерация кода и наоборот [6].

При использовании UML были простроена диаграмма классов.

## Диаграмма классов

Унифицированный язык моделирования (UML) – это семейство графических нотаций, в основе которого лежит единая метамодель. Он помогает в описании и проектировании программных систем, в особенности систем, построенных с использованием объектно–ориентированных (ОО) технологий. [7]

На рисунке 4.1 представлена изначальная диаграмма классов.

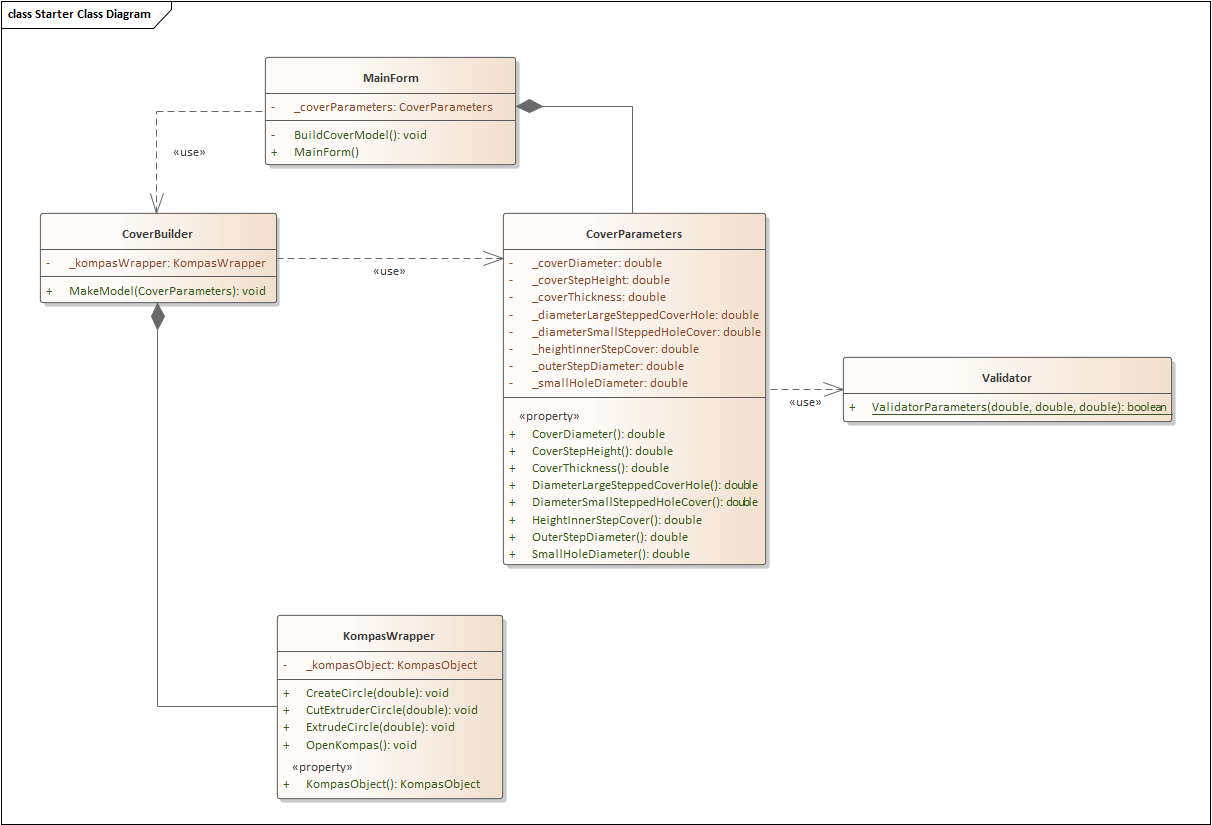


Рисунок 4.1 – Изначальная диаграмма классов

Для реализации был выбран следующий набор классов:

1. MainForm – класс диалогового окна, который обеспечивает взаимодействие между пользователем и программой;
2. CoverParameters – класс, хранящий в себе все параметры проектируемой 3D-модели;
3. Validator – осуществляет проверку зависимых параметров из класса CoverParameters;
4. CoverBuilder – класс, осуществляющий вызов методов API, необходимых для постройки 3D-модели;
5. KompasWrapper – класс для работы с API КОМПАС 3D.

В итоговом проекте созданы классы и методы, которые отображены на итоговой диаграмме классов (рисунок 4.2).

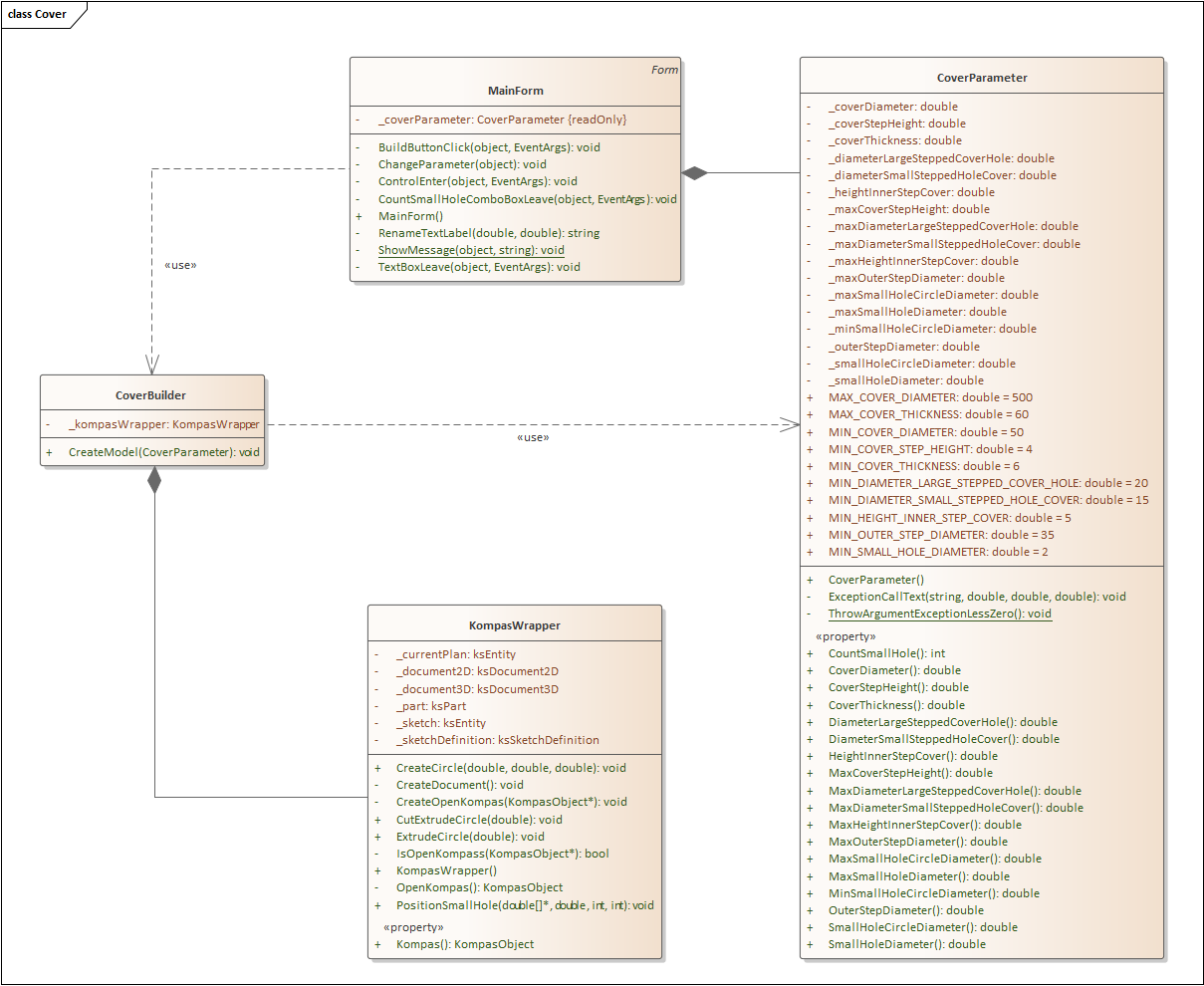


Рисунок 4.2 – Итоговая диаграмма классов

В итоговом варианте был удалён класс Validator, так как проверка зависимых параметров происходит в классе CoverParameters. Так же были добавлены некоторые параметры и методы в классы CoverParameters, KompasWrapper и MainForm.

# Описание программы для пользователя

Макет пользовательского интерфейса (рисунок 5.1) представляет собой форму для ввода параметров крышки. Построение модели осуществляется путем нажатия на кнопку «Build». После ввода некорректных данных (например, введена пустая строка или значение параметра не входит диапазон), на форме будет выделено поле красным цветом с некорректным параметром (рисунок 5.2) и будет показано окно с текстом об ошибке (рисунок 5.3).

Плагин состоит из диалогового окна, которое имеет 9 полей ввода параметров, одну кнопку и одно поле для выбора количества малых отверстий.

Так же плагин имеет параметры по умолчанию, которые показаны на рисунке 5.1.

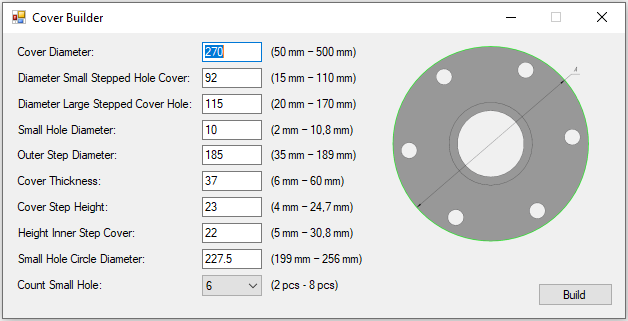


Рисунок 5.1 – Макет пользовательского интерфейса

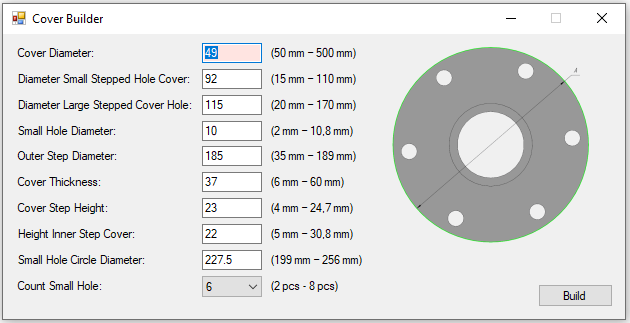


Рисунок 5.2 – Макет пользовательского интерфейса

с некорректными данными

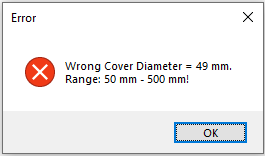


Рисунок 5.3 – Окно ошибки

# Тестирование программы

Тестирование позволяет убедиться в работоспособности программы, выявлять ошибки при изменении какой-либо функциональности.

## Функциональное тестирование

При функциональном тестировании [8] проверялось корректность работы плагина для построения крышки, а именно, соответствие полученного результата в виде трехмерной модели, с входными параметрами.

Проведено тестирование максимальных и минимальных параметров модели.

На рисунке 6.1 представлена проверка размеров модели с минимальными введенными параметрами в САПР «Компас-3D v20».

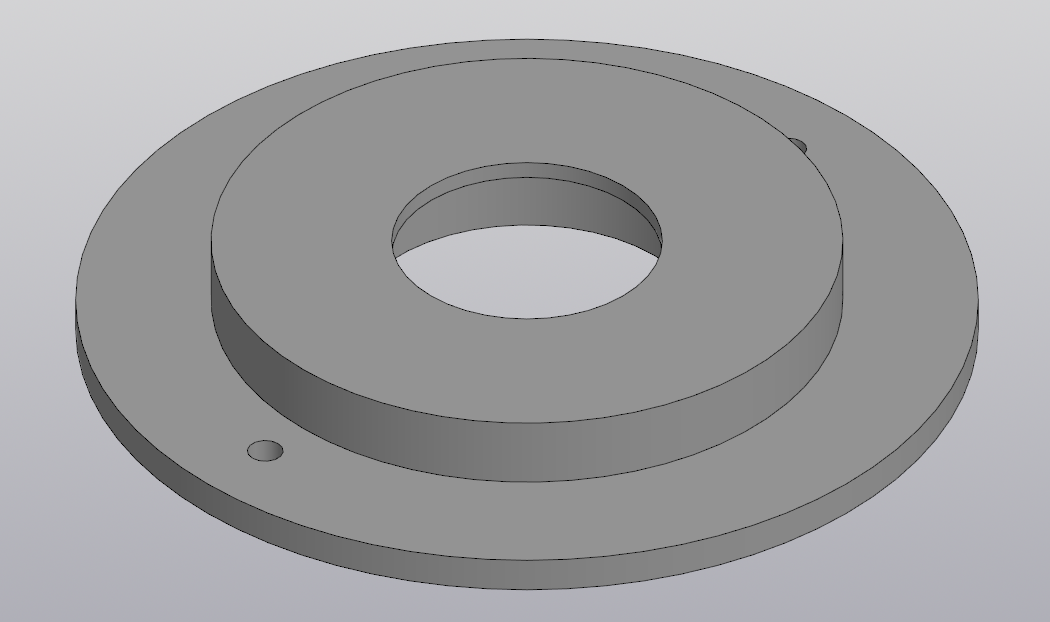


Рисунок 6.1 – Модель крышки с минимальными входными данными

На рисунке 6.2 представлена проверка размеров модели с максимальными введенными параметрами в САПР «Компас-3D v20».

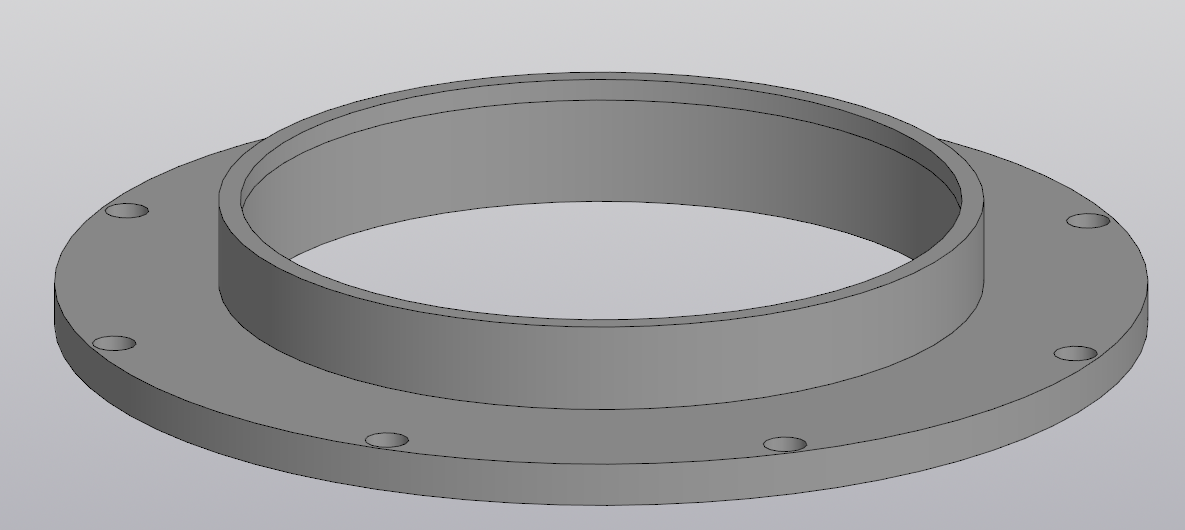


Рисунок 6.2 – Модель крышки с максимальными входными данными

## Модульное тестирование

В целях проверки корректности работы методов и свойств классов при помощи тестового фреймворка NUnit версии 3.13 проведено модульное тестирование, проверялись открытые поля и методы. На рисунке 6.4 представлено тестирование класса CoverParameter. Степень покрытия проектов 100% (рисунок 6.3). Было написано 47 тестов.

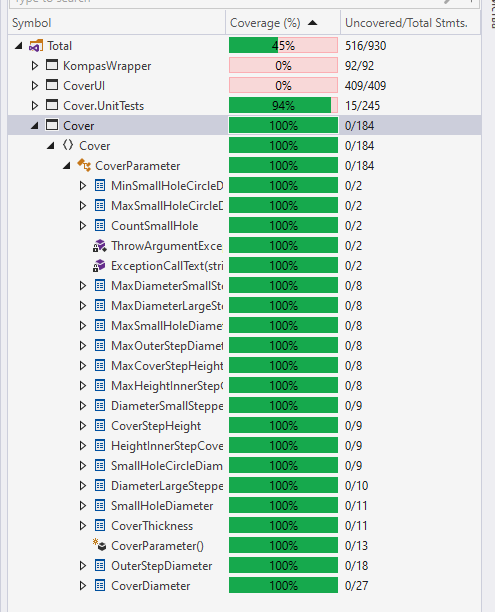


Рисунок 6.3 – Степень покрытия Cover

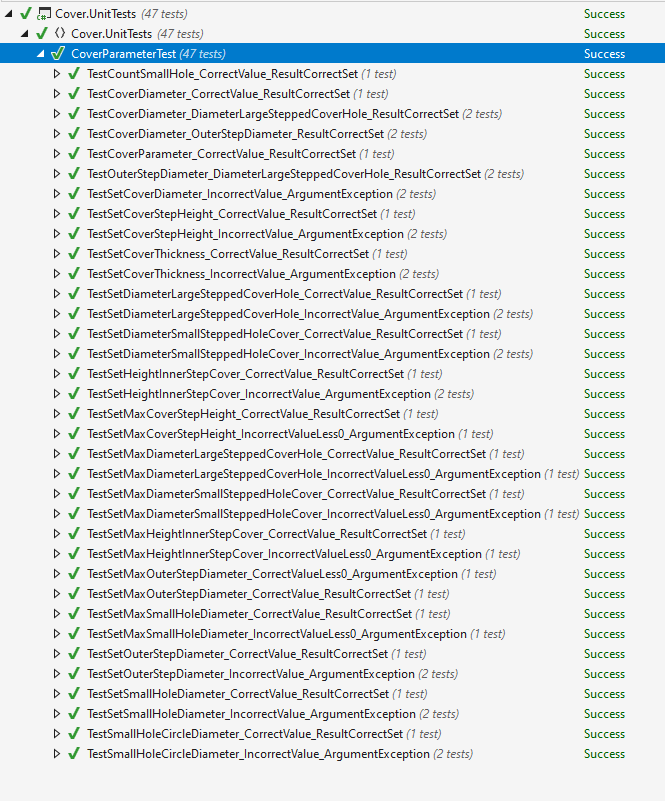


Рисунок 6.4 – Тестирование классов

## Нагрузочное тестирование

В целях проверки производительности работы плагина, было проведено нагрузочное тестирование [9]. Тестирование производилось на ПК со следующей конфигурацией:

* ЦП Intel Core i7-7700HQ 2.80 ГГц;
* ОЗУ объёмом 16 ГБ;
* Графический процессор объёмом 4 ГБ.

Для нагрузочного тестирования создан метод с бесконечным циклом построения детали, представленный на рисунке 6.5. Для измерения времени использовался класс Stopwatch.

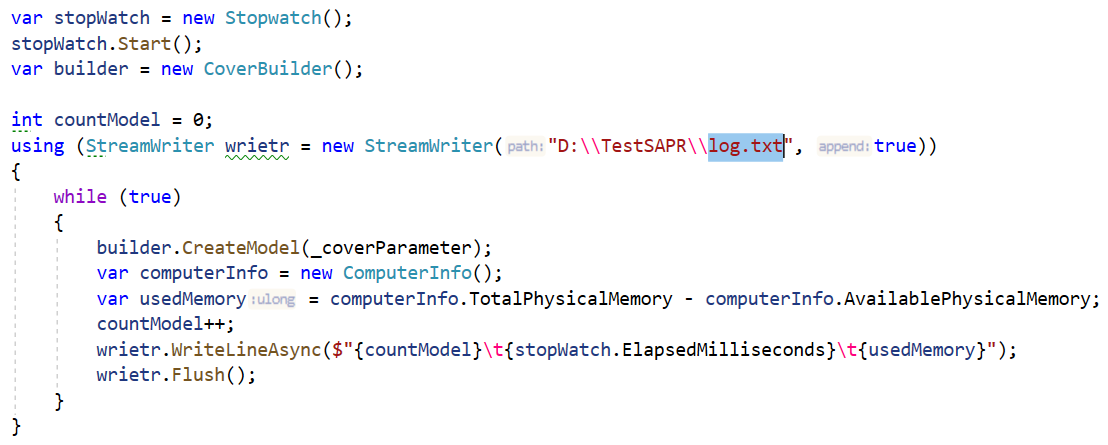


Рисунок 6.5 – Код для стресс-теста

Во время тестирования было построено 171 модель крышки за 138 минут. Модели были построены с параметрами по умолчанию. На протяжении тестирования загруженность процессора была в районе 30%, с временными скачками до 50%.

На рисунках 6.6 и 6.7 показаны результаты тестирования программы.

Рисунок 6.6 – График зависимости количества потребляемой оперативной памяти от количества деталей для модели с

параметрами по умолчанию

Рисунок 6.7 – График зависимости времени построения одной детали от количества деталей для модели с параметрами по умолчанию

Исходя из приведенных графиков на рисунках 6.5 и 6.6, можно увидеть, что чем больше построено моделей, то тем больше занимается оперативной памяти и тем медленнее стоится модель. Оперативная память заполняется пока есть место, как только место начинает заканчиваться система пытается очистить её. От этого и получается задержки в построении детали. Из этого следует, что построение множества моделей не имеет смысла.

# Заключение

В результате выполнения работы в рамках курса «Основы разработки систем автоматизированного проектирования» была разработана программа-плагин для системы «КОМПАС-3D v20», выполняющая построение модели крышки.

Плагин выполнен как отдельное приложение, которое подключается к САПР Компас 3D и с помощью его API строит модель. Заявленная функциональность реализована полностью, включая дополнительную функциональность. Тестирование программы проведено в три этапа: функциональное, модульное и нагрузочное.

# Список использованной литературы

1. Автоматизация вычислительных процедур в прикладных задачах инженерного проектирования [Электронный ресурс]. – URL: https://scienceforum.ru/2014/article/2014000201 (дата обращения: 10.12.2021).
2. Visual Studio [Электронный ресурс]. – URL: https://visualstudio.microsoft.com/ru/ (дата обращения: 10.12.2021).
3. Разработка технологического процесса детали Крышка. [Электронный ресурс] – Режим доступа: <https://vmasshtabe.ru/mashinostroenie-i-mehanika/tm/razrabotka-tehnologicheskogo-protsessa-detali-kryishka.html> (дата обращения 30.10.2021);
4. NUnit [Электронный ресурс]. – URL: https://nunit.org/ (дата обращения: 10.12.2021).
5. Кудрявцев Е.М. Mechanical Desktop Power Pack Основы работы в системе. Изд: ДМК Пресс, 2012 г., 535 с.: ил. (Серия «Проектирование»).
6. UML. [Электронный ресурс]. – Режим доступа: http://www.uml.org/ (дата обращения 10.12.2021).
7. Фаулер М. UML. Основы. Краткое руководство по стандартному языку объектного моделирования. Изд: Символ–Плюс, 2011, с.192   
   (3-е издание);
8. Функциональное тестирование [Электронный ресурс]. – URL: https://daglab.ru/funkcionalnoe-testirovanie-programmnogo-obespechenija/ (дата обращения: 10.12.2021).
9. Нагрузочное тестирование: с чего начать и куда смотреть [Электронный ресурс]. – URL: https://habr.com/ru/company/jugru/blog/329174/ (дата обращения: 10.12.2021).