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)

## ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

## на разработку плагина моделирования зубчатой шестерни

## для системы «AutoCAD»

Выполнил:

Студент группы 588-1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Березин А.А.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Руководитель:

к.т.н., доцент каф. КСУП

\_\_\_\_\_\_\_\_\_\_\_\_ / Калентьев А. А.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

**Оглавление**

[Введение 3](#_Toc86168387)

[2 Описание САПР 4](#_Toc86168388)

[2.1 Описание программы 4](#_Toc86168389)

[2.2 Описание API 4](#_Toc86168390)

[2.3 Обзор аналогов 5](#_Toc86168391)

[3. Описание предмета проектирования 5](#_Toc86168392)

[4 Описание технических и функциональных аспектов проекта 6](#_Toc86168393)

[4.1 Описание полей, свойств и методов, используемых в проекте 6](#_Toc86168394)

[4.2 Макет пользовательского интерфейса 6](#_Toc86168395)

[Список литературы 8](#_Toc86168396)

## Введение

Область применения систем автоматизированного проектирования (САПР) охватывает сегодня самые различные виды деятельности человека — от расстановки мебели в квартире до проектирования и изготовления интегральных микросхем и современной космической техники. Каждая категория задач технического черчения предъявляет к этим продуктам свои требования, однако наибольшее распространение они получили в машиностроении и архитектуре.

Использование САПР позволяет членам проектных групп одновременно работать над изделием с разных сторон: решать задачи стилевого дизайна, проектирования внешнего вида изделия и параллельной поагрегатной разработки изделия. Одновременно группой специалистов различных профилей, работающих над выпуском нового изделия, выполняются все этапы разработки деталей, узлов и сборок, их технологическая проработка (Concurrent engineering).

Изделие начинают изготавливать еще до того, как будет завершен выпуск всей документации, что приводит к значительному сокращению сроков и повышает качество проектирования. Облегчается автоматизированное управление проектами и предприятием на базе электронного документооборота. Любые изменения в любом элементе изделия незамедлительно становятся доступными как для отдельных конструкторов и технологов, так и для целых отделов и организаций на всех этапах создания изделия — благодаря использованию единой базы данных. Таким образом, САПР сокращает время и трудозатраты на проектирование изделия [1].

## 2 Описание САПР

## 2.1 Описание программы

AutoCAD — это программное обеспечение автоматизированного проектирования (САПР), с помощью которого архитекторы, инженеры и строители создают точные 2D- и 3D-чертежи. Первая версия системы была выпущена в 1982 году. AutoCAD и специализированные приложения на его основе нашли широкое применение в машиностроении, строительстве, архитектуре и других отраслях промышленности.

AutoCAD включает в себя полный набор инструментов для комплексного трёхмерного моделирования (поддерживается твердотельное, поверхностное и полигональное моделирование). AutoCAD позволяет получить высококачественную визуализацию моделей с помощью системы рендеринга mental ray. Также в программе реализовано управление трёхмерной печатью (результат моделирования можно отправить на 3D-принтер) и поддержка облаков точек (позволяет работать с результатами 3D-сканирования) [2].

## 2.2 Описание API

Среда программирования ObjectARX используется для адаптации и расширения функциональных возможностей AutoCAD и продуктов на его основе. Она обеспечивает непосредственный доступ к структурам базы данных AutoCAD, графической системе и определениям встроенных команд.

В состав ObjectARX SDK входит также управляемый API, который часто называют AutoCAD .NET API. Для адаптации и расширения функциональных возможностей AutoCAD и продуктов на его основе может применяться любой язык программирования, поддерживающий .NET. Обеспечивается непосредственный доступ к структурам базы данных AutoCAD, определениям встроенных команд и другим внутренним программным элементам [3].

Все графические объекты, которые отображаются на экране, унаследованы от класса Entity. Класс Entity наследуется от класса DBObject, который является базовым классом для всех объектов, находящихся в базе данных Database (таблица 2.4).

Каждая сущность в базе данных имеет свой уникальный идентификатор, который представлен классом ObjectId.

В таблице 2.1 представлен класс Application, предоставляющий доступ к объекту программы.

Таблица 2.1 – Методы и свойства класса Application.

|  |  |  |
| --- | --- | --- |
| **Название метода** | **Тип** | **Описание** |
| DocumentManager | DocumentManager | Свойство, хранящее менеджер документов. |
| ShowModalWindow(Window) | bool? | Метод для открытия окна |

В таблице 2.2 представлен класс DocumentManager, предоставляющий доступ к текущему документу.

Таблица 2.2 – Методы и свойства класса DocumentManager.

|  |  |  |
| --- | --- | --- |
| **Название метода** | **Тип** | **Описание** |
| MdiActiveDocument | Document | Свойство, хранящее открытый документ. |

В таблице 2.3 представлен класс Document, который хранит сведения об открытом документе AutoCAD.

Таблица 2.3 – Методы и свойства класса Document.

|  |  |  |
| --- | --- | --- |
| **Название метода** | **Тип** | **Описание** |
| Database | Database | Свойство, хранящее базу данных текущего документа. |

В таблице 2.4 представлен класс Database, который позволяет работать с базами данных.

Таблица 2.4 – Методы и свойства класса Database.

|  |  |  |
| --- | --- | --- |
| **Название метода** | **Тип** | **Описание** |
| BlockTableId | ObjectId | Свойство, хранящее идентификатор в базе, соответствующий BlockTable |
| TransactionManager | TransactionManager | Свойство, которое предоставляет доступ к TransactionManager для текущей базы данных |

В таблице 2.5 представлен класс TransactionManager, который позволяет работать с транзакциями.

Таблица 2.5 – Методы и свойства класса TransactionManager.

|  |  |  |
| --- | --- | --- |
| **Название метода** | **Тип** | **Описание** |
| StartTransaction() | Transaction | Начало транзакции |

В таблице 2.6 представлен класс Transaction, который используется для чтений данных из баз, и сохранения изменений.

Таблица 2.6 – Методы и свойства класса Transaction.

|  |  |  |
| --- | --- | --- |
| **Название метода** | **Тип** | **Описание** |
| GetObject(ObjectId id,  OpenMode mode) | DBObject | Получение объекта по его идентификатору. OpenMode объявляет права доступа к объекту (для чтения, записи, уведомления) |

Продолжение таблицы 2.6.

|  |  |  |
| --- | --- | --- |
| AddNewlyCreatedDBObject  (DBObject obj, bool add) | void | Добавление нового объекта в транзакцию. Нужно для того, чтобы данные об объекте сохранились |
| Commit() | void | Сохранение данных, отправление коммита. |

В таблице 2.7 представлен класс BlockTableRecord, который используется для записи данных в таблицу блоков.

Таблица 2.7 – Методы и свойства класса BlockTableRecord.

|  |  |  |
| --- | --- | --- |
| **Название метода** | **Тип** | **Описание** |
| AppendEntity(Entity entity) | ObjectId | Добавление сущности в таблицу |

В таблице 2.8 представлен класс Solid3d, который используется для работы с трехмерными фигурами.

Таблица 2.8 – Методы и свойства класса Solid3d.

|  |  |  |
| --- | --- | --- |
| **Название метода** | **Тип** | **Описание** |
| CreateFrustum(double height, double radiusAlongX, double radiusAlongY, double topRadius) | void | Создание цилиндра |
| Extrude(Region region, double height, double taperAngle) | void | Выдавливание 3d объекта из 2d рисунка |
| BooleanOperation(BooleanOperationType operation, Solid3d solid) | void | Выполнение булевых операций между двумя 3d фигурами. К булевым операциям относятся объединение, вычитание и пересечение 3d фигур. |

Продолжение таблицы 2.8.

|  |  |  |
| --- | --- | --- |
| CheckInterference(Solid3d otherSolid) | bool | Проверяет, пересекаются ли фигуры. |

В таблице 2.9 представлен класс Circle, который используется для работы с кругом.

Таблица 2.9 – Методы и свойства класса Circle.

|  |  |  |
| --- | --- | --- |
| **Название метода** | **Тип** | **Описание** |
| Center | Point3d | Центр круга |
| Dispose() | void | Освобождение ресурсов |

В таблице 2.10 представлен класс DBObjectCollection, представляющие коллекцию объектов DBObject.

Таблица 2.10 – Методы и свойства класса DBObjectCollection.

|  |  |  |
| --- | --- | --- |
| **Название метода** | **Тип** | **Описание** |
| Add(DBObject) | int | Добавление объекта DBObject в коллекцию |

В таблице 2.11 представлен класс Region, который используется для работы с двумерной областью.

Таблица 2.11 – Методы и свойства класса Region.

|  |  |  |
| --- | --- | --- |
| **Название метода** | **Тип** | **Описание** |
| Area | double | Площадь поверхности региона |
| Dispose() | void | Освобождение ресурсов |

Продолжение таблицы 2.11.

|  |  |  |
| --- | --- | --- |
| BooleanOperation(  BooleanOperationType operation, Region otherRegion) | void | Выполнение булевых операций между двумя регионами. К булевым операциям относятся объединение, вычитание и пересечение областей |
| CreateFromCurves(  DBObjectCollection curveSegmants) | DBObjectCollection | Создание области из каждого замкнутого цикла, образованного входным массивом объектов. |

В таблице 2.12 представлен класс Polyline, который используется для работы ломаной линией.

Таблица 2.11 – Методы и свойства класса Polyline.

|  |  |  |
| --- | --- | --- |
| **Название метода** | **Тип** | **Описание** |
| AddVertexAt(int index,  Point2d pt, double bulge, double startWidth, double endWidth) | void | Добавление вершин |

## 2.3 Обзор аналогов

Плагин GearCAM позволяет строить 2D-модель шестерни [4]. Особенностью плагина является возможность задавать количество зубцов. Примеры работы плагина приведены на рисунках 2.1 и 2.2.

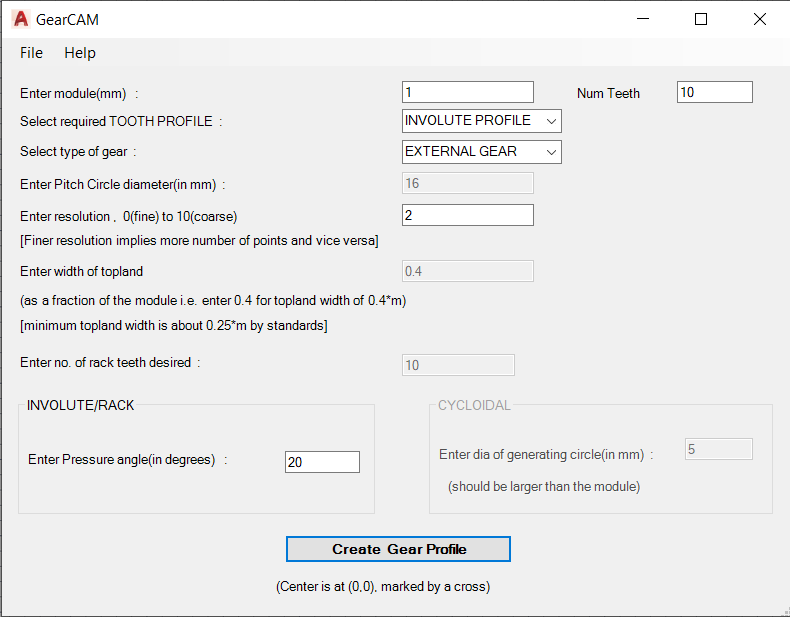


Рисунок 2.1 – Окно задания параметров плагина GearCAM

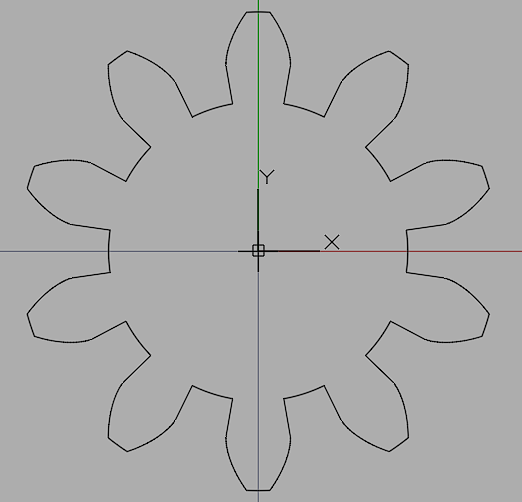


Рисунок 2.2 – Модель, построенная плагином GearCAM

## 3. Описание предмета проектирования

Предметом проектирования является модель зубчатой шестерни. Данная модель имеет 5 основных параметров:

1. Диаметр шестерни D (24≤D≤60) мм;
2. Диаметр отверстия d (4≤d≤D\*1/4) мм;
3. Высота шестерни H (10≤H≤20) мм;
4. Длина зубца A (D\*1/5≤A≤D\*1/2) мм;
5. Ширина зубца B (5≤B≤D\*1/4) мм.

Чертеж модели показан на рисунке 3.1.

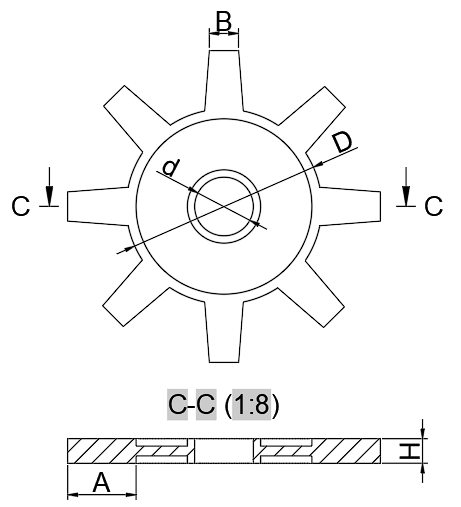


Рисунок 3.1 – Чертеж модели

Также, были добавлены дополнительные параметры для выполнения дополнительного задания:

1. Количество зубцов N (6≤N≤D\*2/B);
2. Форма зуба (рис. 3.2).

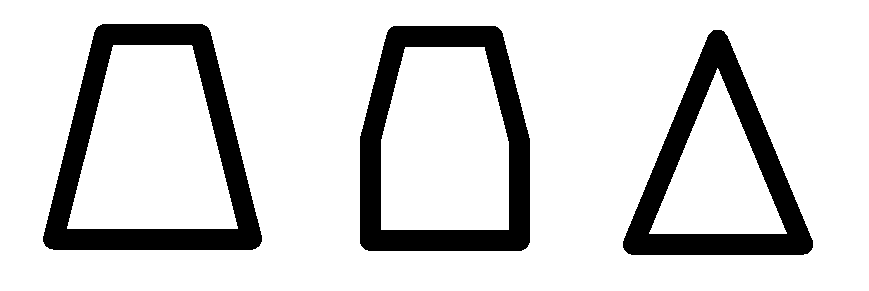


Рисунок 3.2 – Варианты формы зуба

## 4 Проект программы

## 4.1 Диаграмма классов

Диаграмма классов – структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы, их коопераций, атрибутов (полей), методов, интерфейсов и взаимосвязей между ними [5].

Диаграмма классов представлена на рисунке 4.1.

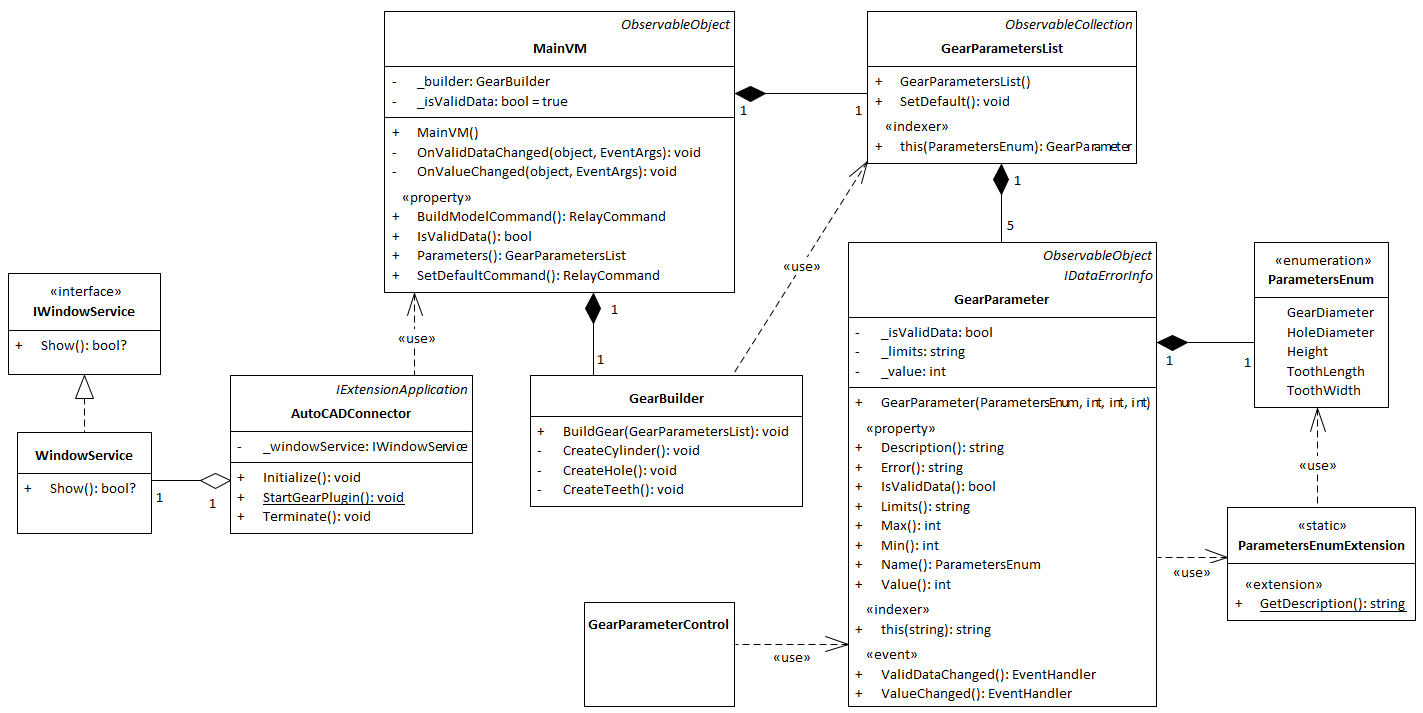


Рисунок 4.1 – UML-диаграмма классов

Класс AutoCADConnector реализует интерфейс IExtensionApplication, необходимый для запуска плагина в программе AutoCAD.

Класс GearParameter хранит данные о каждом параметре модели из перечисления ParametersEnum. Параметр хранит минимальное и максимальное значение параметра, а также текущее значение. Свойство Limits возвращает ограничения для значения в виде строки. Свойство Description возвращает текстовое описание параметра.

Все параметры хранятся в списке GearParametersList, наследуемый от класса ObservableCollection.

Для построения модели используется класс GearBuilder. Метод BuildGear строит 3D объект на основании списка параметров.

Класс MainVM связывает модели и представление через механизм привязки данных.

Класс WindowService – сервис для вызова главного окна.

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

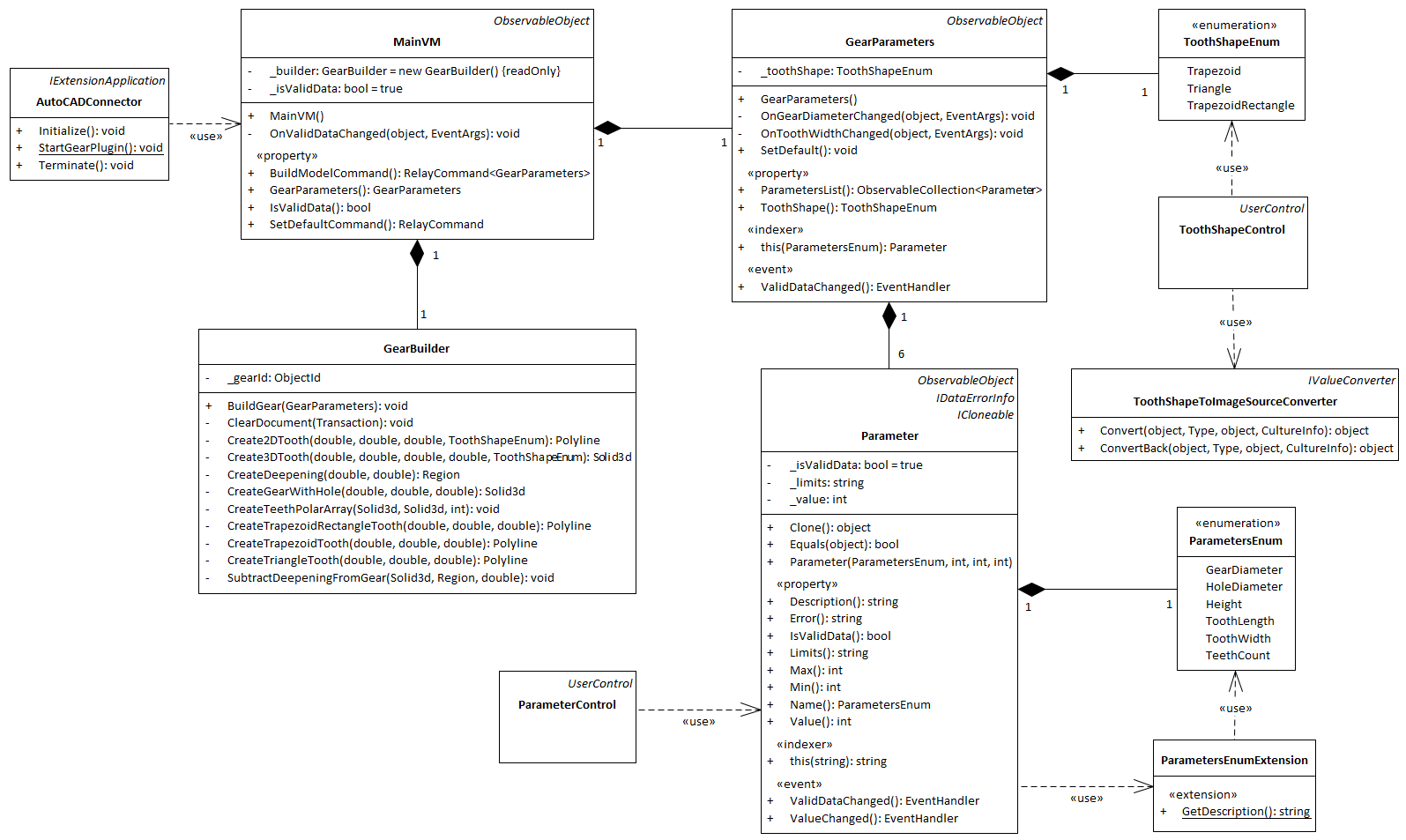


Рисунок 4.2 – Итоговая диаграмма классов

Было добавлено перечисление ToothShapeEnum для создания зубов различной формы. Для выбора формы зуба создан отдельный элемент управления ToothShapeControl, который использует ToothShapeToImageConverter для преобразования значения перечисления в изображение.

В перечисление ParametersEnum добавлен элемент TeethCount для реализации нового параметра «Количество зубьев».

Удален класс WindowService из-за ненадобности. Открытия окна осуществляется в классе AutoCADConnector с помощью специального метода ShowModelessWindow из AutoCAD .Net API.

В класс GearParameters добавлено свойство ToothShape. Для обработки событий изменения свойства наследование класса ObservableCollection заменено на наследование класса ObservableObject.

## 4.2 Макет пользовательского интерфейса

Макет пользовательского интерфейса представлен на рисунке 4.3.

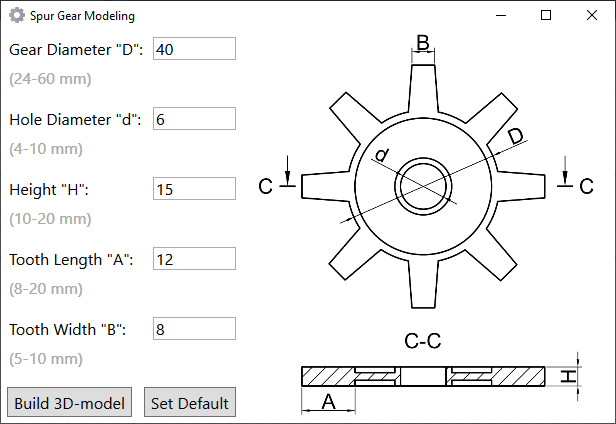


Рисунок 4.3 – Макет пользовательского интерфейса программы

При вводе некорректных значений, поля с ошибками выделяются красным цветом, при наведении на поля показывается описание ошибки. Если введены некорректные данные, кнопка построения модели блокируется. Сообщение показано на рисунке 4.4.

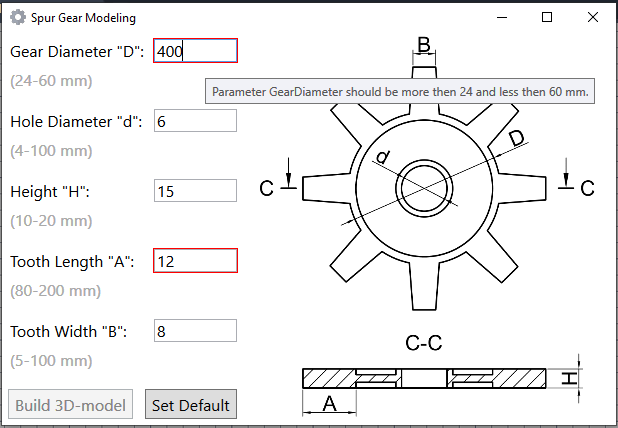


Рисунок 4.4 – Макет пользовательского интерфейса с некорректными данными

На рисунке 4.5 представлен пользовательский интерфейс плагина после добавления дополнительной функциональности. Добавлены поле для задания количества зубьев и выпадающий список для выбора формы зуба. Значения, содержащиеся в выпадающем списке, представлены на рисунке 4.6.

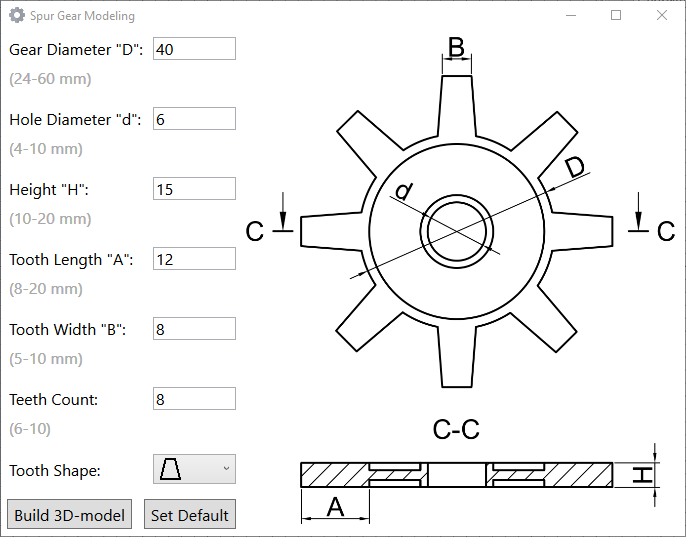


Рисунок 4.5 – Пользовательский интерфейс программы после добавления дополнительной функциональности

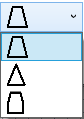


Рисунок 4.6 – Варианты формы зуба

## 5 Тестирование программы

## Функциональное тестирование

При функциональном тестировании проверялось корректность работы плагина «Зубчатая шестерня», а именно, соответствие полученного результата в виде трехмерной модели, с входными параметрами [6]. Проведено тестирование максимальных и минимальных параметров модели.

На рисунке 5.1 представлена шестерня, построенная по заданным значениям по умолчанию (диаметр шестерни 40 мм, диаметр отверстия 6 мм, высота 15 мм, длина зуба 12 мм, ширина зуба 8 мм, количество зубьев 8).

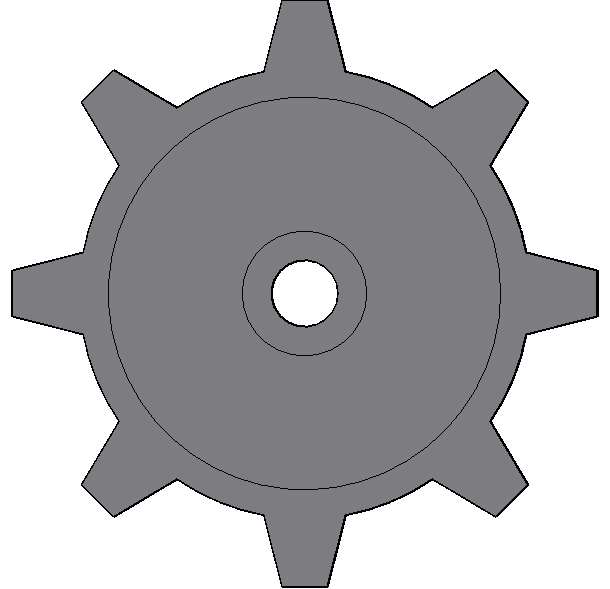


Рисунок 5.1 – Шестерня, построенная по заданным параметрам по умолчанию

На рисунке 5.2 представлена шестерня, построенная по заданным минимальным значениям с зубом в форме трапеции с прямоугольником (диаметр шестерни 24 мм, диаметр отверстия 4 мм, высота 10 мм, длина зуба 4 мм, ширина зуба 5 мм, количество зубьев 6).

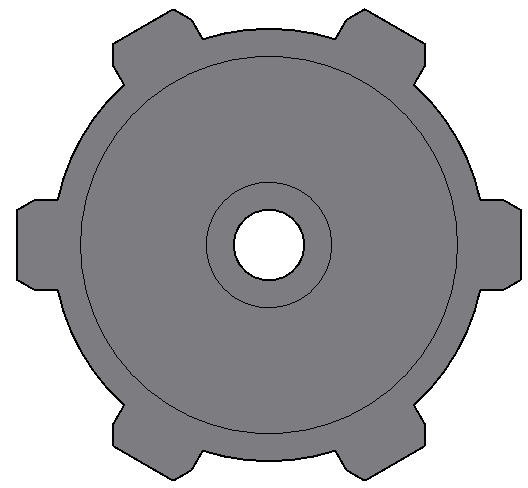


Рисунок 5.2 – Шестерня, построенная по заданным минимальным значениям с зубом в форме трапеции с прямоугольником

На рисунке 5.3 представлена шестерня, построенная по заданным максимальным значениям с треугольной формой зуба (диаметр шестерни 60 мм, диаметр отверстия 15 мм, высота 20 мм, длина зуба 30 мм, ширина зуба 15 мм, количество зубьев 8).

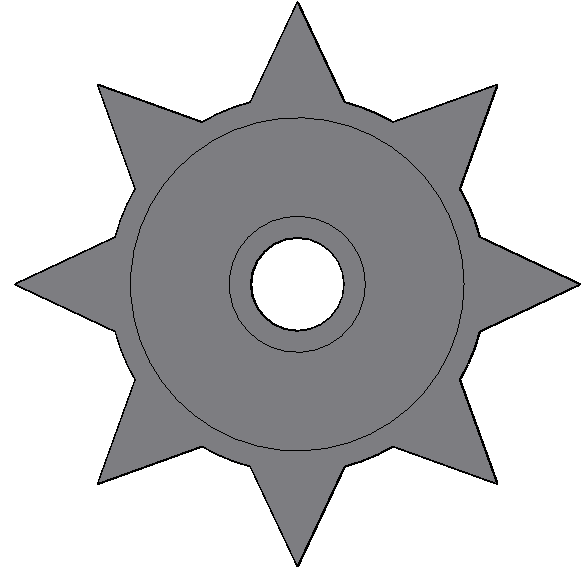


Рисунок 5.3 – Шестерня, построенная по заданным максимальным значениям с треугольной формой зуба

На рисунке 5.4 представлена шестерня, построенная с заданным максимальным количеством зубьев (диаметр шестерни 60 мм, диаметр отверстия 6 мм, высота 15 мм, длина зуба 12 мм, ширина зуба 5 мм, количество зубьев 24).

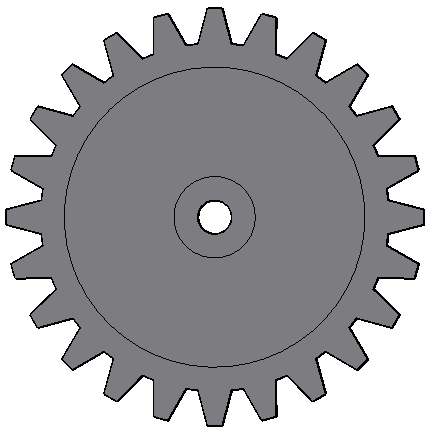
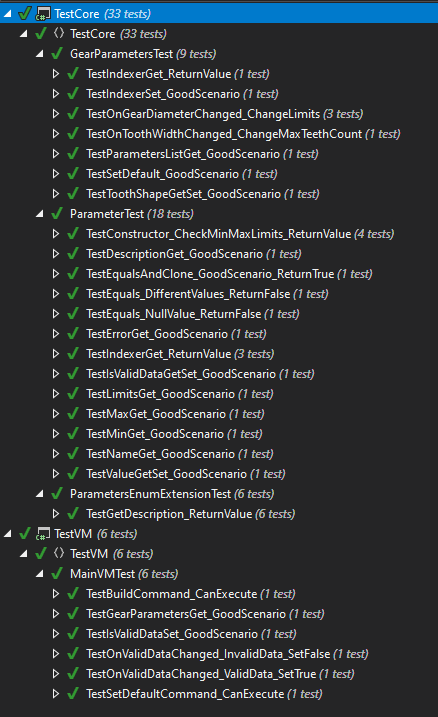


Рисунок 5.4 – Шестерня, построенная с заданным максимальным количеством зубьев

## 5.2 Модульное тестирование

В целях проверки корректности работы методов и свойств классов при помощи тестового фреймворка NUnit версии 3.13 проведено модульное тестирование [7], проверялись открытые поля и методы. На рисунке 5.5 представлено тестирование классов проектов Core и ViewModel. Степень покрытия проектов – сто процентов. Написано тридцать девять тестов.

  
Рисунок 5.5 – Тестирование классов

## 5.3 Нагрузочное тестирование

В целях проверки производительности работы плагина, было проведено нагрузочное тестирование [8]. Тестирование производилось на ПК со следующей конфигурацией:

* ЦП Intel Core i5-8250U 1.6ГГц;
* 12 ГБ ОЗУ;
* графический процессор объемом памяти 4 ГБ.

Для нагрузочного тестирования создан метод с бесконечным циклом построения детали, представленный на рисунке 5.6. Для измерения времени использовался класс Stopwatch.



Рисунок 5.6 – Реализация зацикленного перестроения модели

Первое тестирование заключалось в построении детали с базовыми параметрами. Тестирование проводилось 1 час 12 минут, построено 82101 моделей шестерни. На протяжении тестирования загруженность процессора составляла около 46 процентов. Результаты тестирования представлены на рисунках 5.7 и 5.8. Параметры модели, используемой для тестирования:

* диаметр шестерни 40 мм;
* диаметр отверстия 6 мм;
* высота 15 мм;
* длина зуба 12 мм;
* ширина зуба 8 мм;
* количество зубьев 8.

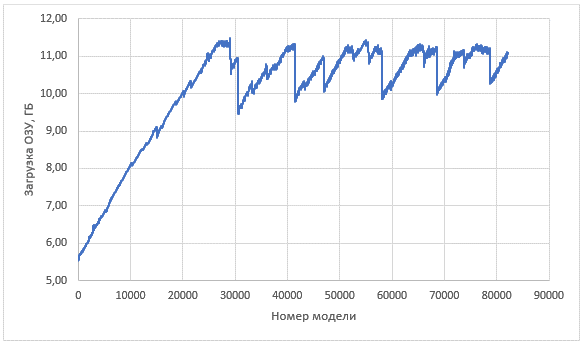


Рисунок 5.7 – График зависимости количества потребляемой оперативной памяти от количества деталей для модели с базовыми параметрами

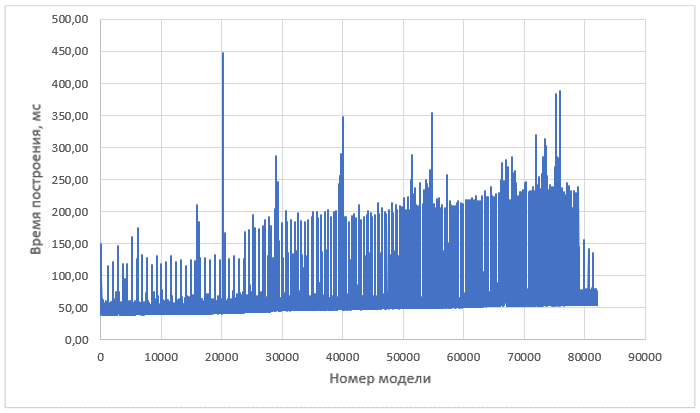


Рисунок 5.8 – График зависимости времени построения одной детали

от количества деталей для модели с базовыми параметрами

Второе тестирование заключалось в построении детали с максимальными параметрами. Тестирование проводилось 59 минут, построено 35082 моделей шестерни. На протяжении тестирования загруженность процессора составляла около 50 процентов. Результаты тестирования представлены на рисунках 5.9 и 5.10. Параметры модели, используемой для тестирования:

* диаметр шестерни 60 мм;
* диаметр отверстия 15 мм;
* высота 20 мм;
* длина зуба 12 мм;
* ширина зуба 5 мм;
* количество зубьев 24.

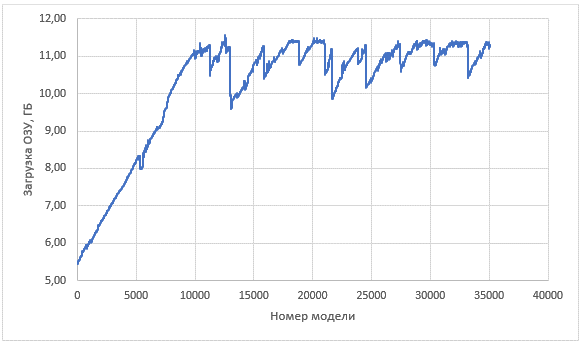


Рисунок 5.9 – График зависимости количества потребляемой оперативной памяти от количества деталей для модели с максимальными параметрами

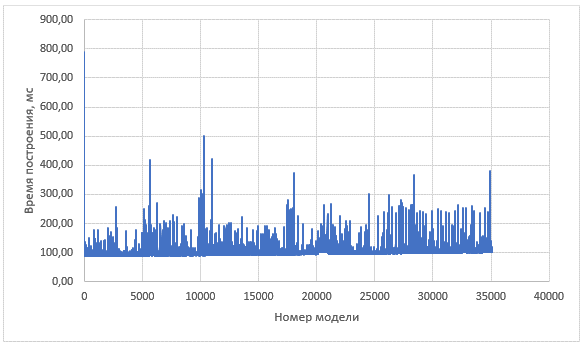


Рисунок 5.10 – График зависимости времени построения одной детали

от количества деталей для модели с максимальными параметрами

Исходя из графиков, представленных на рисунке 5.7 и 5.9 можно сделать вывод, что использование оперативной памяти, затрачиваемое программой, линейно увеличивается до окончания свободного места. Затем память частично очищается, и процесс повторяется. Также, память для моделей с максимальными параметрами заканчивается быстрее.

Исходя из графика на рисунке 5.8 можно сделать вывод, что время построения для модели с параметрами по умолчанию увеличивается с каждым построением. При этом время построения модели с максимальными значениями практически не изменяется.

## Заключение

В ходе выполнения лабораторных работ были изучены предметная область проектирования, предмет проектирования, аналоги предмета проектирования, API, функциональное и нагрузочное тестирование и на основании полученных данных были спроектированы UML диаграммы классов, разработан плагин для создания 3D моделей «Зубчатая шестерня» в САПР AutoCAD 2022 и проведено функциональное и нагрузочное тестирование плагина.

## Список литературы

1. Актуальность применения САПР в машиностроении [Электронный ресурс]. URL: https://sapr.ru/article/7837 (дата обращения: 23.10.2021).
2. AutoCAD – Википедия [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/AutoCAD (дата обращения: 23.10.2021).
3. Разработка приложений для AutoCAD [Электронный ресурс]. URL: https://www.autodesk.ru/autodesk-developer-network/software-platform-russian/develop-autocad (дата обращения: 24.10.2021).
4. GearCAM | AutoCAD Mechanical | Autodesk App Store [Электронный ресурс]. URL: https://apps.autodesk.com/AMECH/ru/Detail/Index?id=6292197326232010119&appLang=en&os=Win32\_64 (дата обращения: 27.10.2021).
5. Мартин Фаулер. UML. Основы. Краткое руководство по стандартному языку объектного моделирования. Изд: Символ-Плюс,2011, с.192 (3-е издание).
6. Функциональное тестирование [Электронный ресурс]. – URL: https://daglab.ru/funkcionalnoe-testirovanie-programmnogo-obespechenija/ (дата обращения: 22.12.2021).
7. Юнит-тестирование для чайников [Электронный ресурс]. – URL: https://habr.com/ru/post/169381/ (дата обращения: 22.12.2021).
8. Нагрузочное тестирование: с чего начать и куда смотреть [Электронный ресурс]. – URL: https://habr.com/ru/company/jugru/blog/329174/ (дата обращения: 22.12.2021).