Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ

УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра компьютерных систем в управлении и проектировании (КСУП)

РАЗРАБОТКА ПЛАГИНА «ВАЛ» НА БАЗЕ СИСТЕМЫ «AUTODESK INVENTOR»

Пояснительная записка к курсовому проекту по дисциплине

«Основы разработки САПР» (ОРСАПР)

|  |  |
| --- | --- |
|  | Выполнил:  Студент гр. 582-2  \_\_\_\_\_\_\_\_\_\_ Ховалкина К.Н.  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_2016 г.  Проверил  м.н.с. ЛИКС, каф. КСУП  \_\_\_\_\_\_\_\_\_\_\_Калентьев А.А  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2016 г. |

Томск 2016

**Реферат**

Курсовой проект 31 с., 9 рис., 3 приложения, 13 источников, 1 таблица. Ключевые слова: САПР, INVENTOR, UML, МОДЕЛИРОВАНИЕ, ВАЛ.

Целью данного проекта является создание приложения для создания трехмерной модели вала в САПР Inventor.

Результатом работы является приложение для построения трехмерной вала в САПР Inventor.

Программный модуль написан в среде MS Visual Studio 2013 Ultimate, язык программирования C# (версия .NET 4.5.1). Отчет по курсовому проекту выполнен в тестовом редакторе Microsoft Word 2010.

Министерство образования и науки РФ

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра компьютерных систем в управлении и проектировании (КСУП)

Утверждаю

Зав. кафедрой КСУП

\_\_\_\_\_\_\_\_\_\_\_\_Шурыгин Ю.А.

"\_\_\_\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_ 2016 г.

ЗАДАНИЕ

на курсовой проект по дисциплине «ОРСАПР»

Студенту гр. 582-2\_\_\_\_\_\_\_\_\_\_Ховалкиной Ксении Николаевне

Тема: разработка плагина «вал» на базе системы «Autodesk Inventor».

Срок сдачи студентом проекта: « » 2016 г.

Исходные данные к проекту:

Разработать программный модуль для автоматического создания трёхмерной модели вала в системе автоматизированного моделирования «Inventor».

Требования к программе:

* модуль должен представлять собой плагин для среды «Inventor»;
* иметь графический интерфейс пользователя с возможностью ввода следующих параметров модели: размеры каждой ступени вала (длина, диаметр), размеры шпоночных пазов(длина, ширина, глубина), и глубину канавки.
* иметь систему для контроля правильности ввода данных;
* при реализации системы использовать язык программирования высокого уровня С# (версия .NET Framework 4.5.1 и выше), в качестве среды разработки Visual Studio 2013;
* должна поддерживаться средой «Inventor» начиная с версии 2015.

Зависимые параметры:

* размеры шпоночных пазов (2,5) не должны превышать размеров ступеней вала (1,4);
* диаметры ступеней валов не должны совпадать;
* ширина и глубина канавки выбираются в соответствии с длиной вала;
* длина ступени вала не должна превышать его длину.

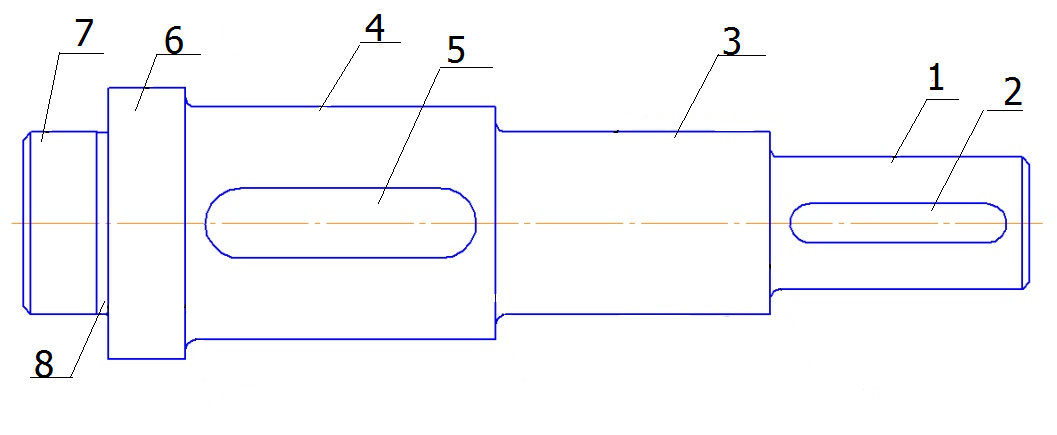


Рисунок 1 – Чертеж вала

Список параметров:

1. Первая ступень вала (длина, диаметр);
2. Шпоночный паз первой ступени;
3. Вторая ступень вала (длина, диаметр);
4. Третья ступень вала (длина, диаметр);
5. Шпоночный паз третьей ступени;
6. Четвертая ступень вала (длина, диаметр);
7. Пятая ступень (длина, диаметр).

Системные и аппаратные требования продукта:

* операционная система: MS Windows 7(x32, x64);
* аппаратная часть: оперативная память от 8Гб, процессор Intel Pentium 4 с тактовой частотой 2 ГГц или выше, Intel Xeon, Intel Core, AMD Athlon 64 или AMD Opteron, либо более новый, графический адаптер, поддерживающий Direct3D 9 или Microsoft Direct3D 10 или более поздней версии.

График выполнения работ:

Работы по созданию системы выполняются в три этапа:

* проектирование системы (до 18.10.15);
* реализация проекта, прототип системы (до 29.11.15);
* исправление недочетов, составление проектной документации (до 27.12.15).
* Содержание пояснительной записки:
* титульный лист;
* реферат;
* содержание;
* введение;
* постановка и анализ задачи;
* описание алгоритмов;
* описание реализации;
* описание пользовательского интерфейса;
* тестирование программы;
* заключение;
* список литературы;
* приложение.
* Дата выдачи задания: «\_\_\_\_»\_\_\_сентября\_\_\_\_ 2015 г.

Руководитель

м.н.с. ЛИКС каф. КСУП

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

Задание принял к исполнению

студент гр. 582-2

Ховалкина К.Н. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оглавление

[1 Введение 7](#_Toc450296414)

[2 Постановка и анализ задачи 9](#_Toc450296415)

[3 Обзор программы «Inventor» 10](#_Toc450296416)

[3.1 Возможности программы «Inventor» 11](#_Toc450296417)

[3.2 Описание API системы «Inventor» 12](#_Toc450296418)

[4 Описание реализованного модуля 14](#_Toc450296419)

[4.1 UML диаграммы вариантов использования и классов 14](#_Toc450296420)

[4.2 Описание пользовательского интерфейса 14](#_Toc450296421)

[4.3 Тестирование модуля 17](#_Toc450296422)

[4.3.1 Функциональное тестирование 19](#_Toc450296423)

[4.3.2 Модульное тестирование 21](#_Toc450296424)

[4.3.3 Нагрузочное тестирование 21](#_Toc450296425)

[5 Заключение 23](#_Toc450296426)

[Список использованных источников 24](#_Toc450296427)

[Приложение А – Диаграмма вариантов использования 26](#_Toc450296428)

[Приложение Б – диаграмма классов 27](#_Toc450296429)

[Приложение В – Описание классов 28](#_Toc450296430)

# 1 Введение

Автоматизация проектирования занимает особое место среди информационных технологий. Во-первых, автоматизация проектирования — синтетическая дисциплина, ее составными частями являются многие другие современные информационные технологии. Так, техническое обеспечение систем автоматизированного проектирования (САПР) основано на использовании вычислительных сетей и телекоммуникационных технологий, в САПР используются персональные компьютеры и рабочие станции.

Математическое обеспечение САПР отличается богатством и разнообразием используемых методов вычислительной математики, статистики, математического программирования, дискретной математики, искусственного интеллекта. Программные комплексы САПР относятся к числу наиболее сложных современных программных систем, основанных на операционных системах Unix, Windows 95/NT, языках программирования. С, С++, Java и других, современных CASE технологиях, реляционных и объектно-ориентированных системах управления базами данных (СУБД), стандартах открытых систем и обмена данными в компьютерных средах.

Во-вторых, знание основ автоматизации проектирования и умение работать со средствами САПР требуется практически любому инженеру разработчику. Предприятия, ведущие разработки без САПР или лишь с малой степенью их использования, оказываются неконкурентоспособными, как из-за больших материальных и временных затрат на проектирование, так и из-за невысокого качества проектов. Появление первых программ для автоматизации проектирования за рубежом и в СССР относится к началу 60 х гг. Тогда были созданы программы для решения задач строительной механики, анализа электронных схем, проектирования печатных плат.

Дальнейшее развитие САПР шло по пути создания аппаратных и программных средств машинной графики, повышения вычислительной эффективности программ моделирования и анализа, расширения областей применения САПР, упрощения пользовательского интерфейса, внедрения в САПР элементов искусственного интеллекта.

К настоящему времени создано большое число программно-методических комплексов для САПР с различной степенью специализации и прикладной ориентацией. В результате автоматизация проектирования стала необходимой составной частью подготовки инженеров разных специальностей; инженер, не владеющий знаниями и не умеющий работать в САПР, не может считаться полноценным специалистом[1].

Валом называют деталь (как правило, гладкой или ступенчатой цилиндрической формы), предназначенную для поддержания установленных на ней шкивов, зубчатых колес, звездочек, катков и т. д., и для передачи вращающего момента[2].

В данной работе представлена задача разработки плагина «модели вала ДВС» на базе системы «Autodesk Inventor» с использованием языка программирования C# и интегрированной среды разработки Microsoft Visual Studio 2013.

# Постановка и анализ задачи

В соответствии с техническим заданием требуется реализовать плагин к САПР «INVENTOR», который позволяет осуществлять построение трехмерной модели по введенным пользователем параметрам.

Задачей курсового проекта является:

* создание пользовательского интерфейса;
* проверка корректности введённых пользователем параметров;
* построение трехмерной модели вала.

# Обзор программы «****Inventor»****

Autodesk Inventor — система трехмерного твердотельного и поверхностного параметрического проектирования (САПР) компании Autodesk, предназначенная для создания цифровых прототипов промышленных изделий. Инструменты Inventor обеспечивают полный цикл проектирования и создания конструкторской документации:

* 2D/3D-моделирование;
* создание изделий из листового материала и получение их разверток;
* разработка электрических и трубопроводных систем;
* проектирование оснастки для литья пластмассовых изделий;
* динамическое моделирование;
* параметрический расчет напряженно-деформированного состояния деталей и сборок;
* визуализация изделий;

автоматическое получение и обновление конструкторской документации (оформление по ЕСКД)[3].

## 3.1 Возможности программы «****Inventor»****

В Autodesk Inventor можно проектировать[4]:

* **Компоновочные схемы** совмещают отдельные детали и узлы. Пользователи могут проверить возможность сборки объекта, добавить и позиционировать новые части, а также устранить помехи между частями проекта.
* **Литьевые формы и оснастка**. Программа автоматизирует ключевые аспекты процесса проектирования литьевых форм под давлением. Пользователи могут быстро создавать и проверять конструкции форм, а затем экспортировать их в Autodesk Moldflow.
* **Детали из листового материала**. Специальная среда проектирования изделий из листового материала автоматизирует многие аспекты работы. Пользователи могут создавать детали развертки, гнутые профили, формировать фланцы путем 3D-моделирования и вставлять в детали специализированные крепежные элементы.
* **Генератор рам** служит для проектирования каркасов (рам) на основе стандартных профилей. Рамы создаются путем размещения стандартных стальных профилей на каркасе. Формирование конечных условий упрощается благодаря наличию стандартных опций для угловых соединений и соединений встык. Пользователи могут создавать собственные профили и добавлять их в библиотеку.
* **Кабельные и трубопроводные системы**. Среда для создания трубопроводов помогает проектировать их таким образом, чтобы вписать в сложную сборку или ограниченное пространство. Она включает библиотеку стандартных фитингов, труб и шлангов, и обеспечивает создание сборочных чертежей, которые обновляются по мере изменений исходной 3D-модели.

## 3.2 Описание API системы «Inventor»

Общая схема объектной модели AI c рядом основных компонентов представлена на рисунке 3.1.

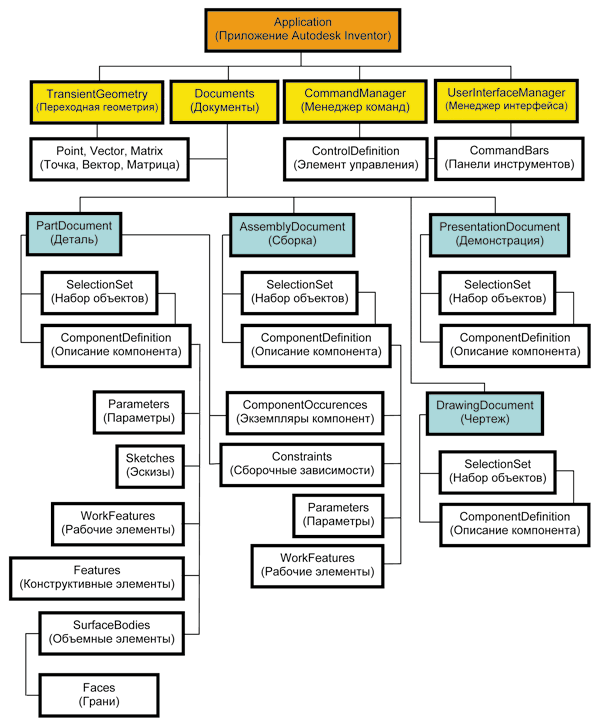


Рисунок 3.1 – Объектная модель Autodesk Inventor (фрагмент)

Проект приложения AI (Application) представляет собой систему специальных документов САПР (Documents), переходной геометрии (TransientGeometry), менеджера команд (CommandManager) и менеджера пользовательского интерфейса (UserInterfaceManager). Основные типы документов AI: сборки (AssemblyDocument), деталь (PartDocument), демонстрация (PresentationDocument), чертеж (DrawingDocument).

AI — среда ассоциативного проектирования, поэтому каждый из типов документов, входящих в проект на всем протяжении разработки, сохраняет взаимосвязь с другими документами, в которых он используется.

Так, изменения в файле детали отражаются в документе сборки, содержащей эту деталь, или файле чертежа, имеющем виды этой детали.

Каждый из типов документов позволяет программно формировать и работать с наборами составляющих объектов (SelectionSet) и отдельными компонентами (ComponentDefinition). Для отдельного компонента доступны все характеризующие его объекты и свойства. Например, описание компонента (ComponentDefinition) детали (PartDocument) включает эскизы (плоские и трехмерные контуры), объемные элементы (вращения, выдавливания, сдвига, оболочки и др.) с опорными эскизами, конструктивные элементы (фаски, сопряжения, резьбы и др.), рабочие элементы (вспомогательные плоскости, оси, точки), таблицы параметров, управляющих геометрией. Описание компонента (ComponentDefinition) документа сборки (AssemblyDocument) в качестве экземпляра компонента (ComponentOccurences) может включать документы деталей (PartDocument).

Элементы переходной геометрии (TransientGeometry) — это объекты (точки, векторы, матрицы), не имеющие графического выражения в моделях, но используемые при создании геометрии в документах AI.

Отдельно следует сказать об объектах типа сборочных зависимостей (Constraints), с помощью которых в документе сборки между деталями устанавливаются взаимные ограничения позиционирования и движения, что позволяет моделировать в сборке функциональность объекта-прототипа. Сборочные зависимости (Constraints) ассоциативно доступны как из объектов деталей, так и из применяющих их сборок[5].

# Описание реализованного модуля

## 4.1 UML диаграммы вариантов использования и классов

Приложение представляет собой плагин к САПР «Inventor». Принцип работы с плагином представлен на диаграмме вариантов использования (см. Приложение А, рисунок А.1-А.2) [6].

Отличия диаграммы после реализации заключаются в добавлении нового поведения программы, в связи с дополнениями исходного задания:

– поведение сценария «построить деталь» было расширено сценарием выдавливания текста ступени вала;

­– добавлены сценарии для реализации выдавливания фасок на исходном вале.

Для реализации создания и настройки 3D модели были созданы сущности, представленные на диаграммах классов в приложении Б. Описание классов, методов и их назначение приводится в приложении В.

В результате реализации модуля, архитектура системы, представленная в проекте системы, отличается от архитектуры реализованного модуля (см. Приложение А, рисунок А.1-А.2 и см. Приложение Б, рисунок Б.1-Б.2).

Реализованная программная архитектура отличается от проектной по следующим пунктам:

– класс ValProperties был разбит на два класса, а также было реализовано перечисление. Данная модификация позволила абстрагировать каждый конкретный параметр, тем самым уменьшить дублирование кода;

– благодаря появлению класса Parameter, появилась возможность однозначно привязать изменяемый параметр к пользовательскому элементу управления, что позволило реализовать динамическую проверку и коррекцию ввода пользователя.

## 4.2 Описание пользовательского интерфейса

После запуска плагина открывается основное окно программы (см. рисунок 4.2).



Рисунок 4.2 – Основное окно программы

В главном окне программы можно задать все необходимые параметры. В программе реализован контроль пользовательского ввода. Каждое поле автоматически исправляет ошибки пользователя, в случае некорректного значения устанавливается минимально или максимально допустимое значение параметра.

Благодаря динамической коррекции ошибок ввода параметров, запустить построение модели можно в любой момент работы с плагином. В результате построения будет создана модель вала на основе заданных параметров и различных параметрах системы (см. рисунок 4.3 – 4.4).

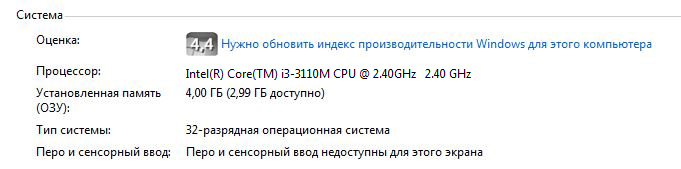
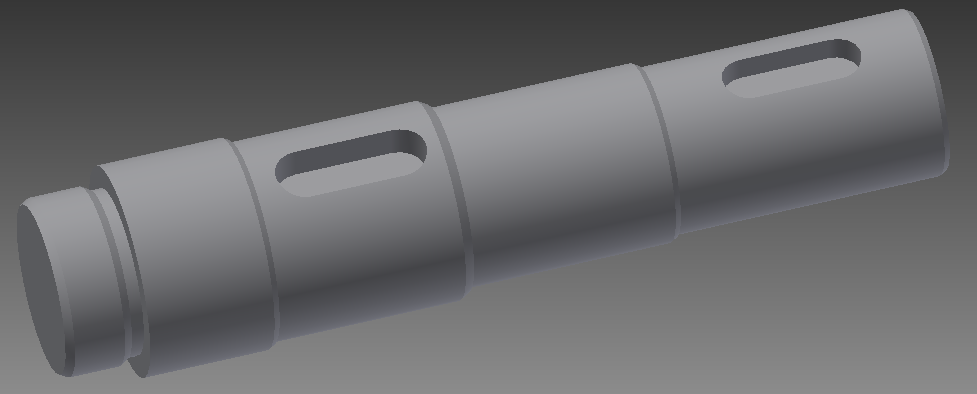


Рисунок 4.3 – Модель на основе настроек по умолчанию в 32-х битной системе

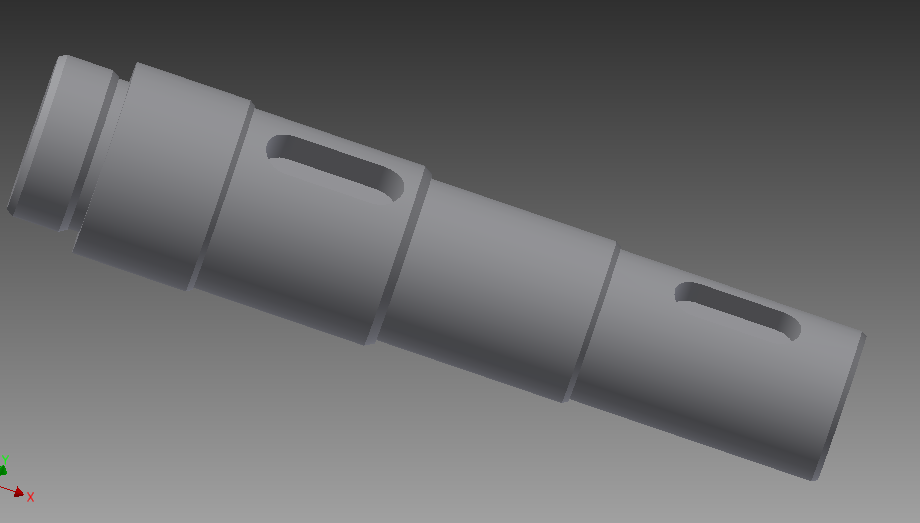
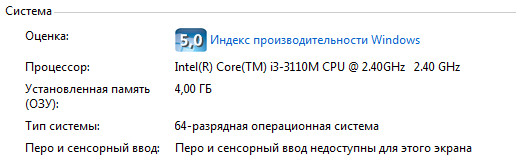
 

Рисунок 4.4 – Модель на основе настроек по умолчанию в 64-х битной системе

## 4.3 Тестирование модуля

Тестирование программного обеспечения — процесс исследования, испытания программного продукта, имеющий две различные цели[7]:

* продемонстрировать разработчикам и заказчикам, что программа соответствует требованиям;
* выявить ситуации, в которых поведение программы является неправильным, нежелательным или не соответствующим спецификации.

Функциональное тестирование рассматривает заранее указанное поведение и основывается на анализе спецификаций функциональности компонента или системы в целом.

Функциональные тесты основываются на функциях, выполняемых системой, и могут проводиться на всех уровнях тестирования (компонентном, встраиваемом, системном, приемочном). Как правило, эти функции описываются в требованиях, функциональных спецификациях или в виде случаев использования системы.

Модульное тестирование, или юнит-тестирование (англ. unit testing) — процесс в программировании, позволяющий проверить на корректность отдельные модули исходного кода программы[8].

Идея состоит в том, чтобы писать тесты для каждой функции или метода. Это позволяет достаточно быстро проверить, не привело ли очередное изменение кода к регрессии, то есть к появлению ошибок в уже оттестированных местах программы, а также облегчает обнаружение и устранение таких ошибок. Цель модульного тестирования — изолировать отдельные части программы и показать, что по отдельности эти части работоспособны.

Нагрузочное тестирование (англ. load testing) — подвид тестирования производительности, сбор показателей и определение производительности и времени отклика программно-технической системы или устройства в ответ на внешний запрос с целью установления соответствия требованиям, предъявляемым к данной системе (устройству)[9].

Для исследования времени отклика системы на высоких или пиковых нагрузках производится стресс-тестирование, при котором создаваемая на систему нагрузка превышает нормальные сценарии её использования. Не существует чёткой границы между нагрузочным и стресс-тестированием, однако эти понятия не стоит смешивать, так как эти виды тестирования отвечают на разные бизнес-вопросы и используют различную методологию. Основная цель нагрузочного тестирования заключается в том, чтобы, создав определённую ожидаемую в системе нагрузку (например, посредством виртуальных пользователей) и, обычно, использовав идентичное программное и аппаратное обеспечение, наблюдать за показателями производительности системы [9].

### 4.3.1 Функциональное тестирование

Проверим построение вала в САПР «Inventor» при минимальных и максимальных параметрах (см. рисунок 4.5-4.6).

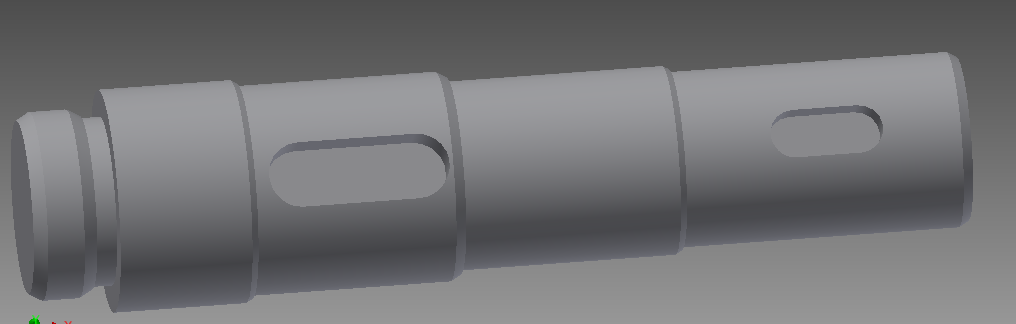
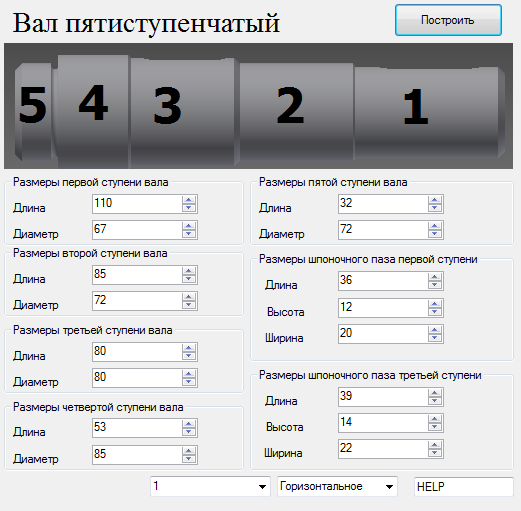
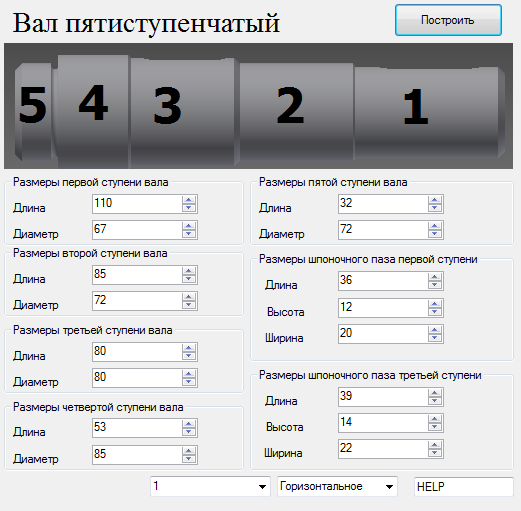


Рисунок 4.5 – Построение вала в САПР «Inventor» с минимальными параметрами



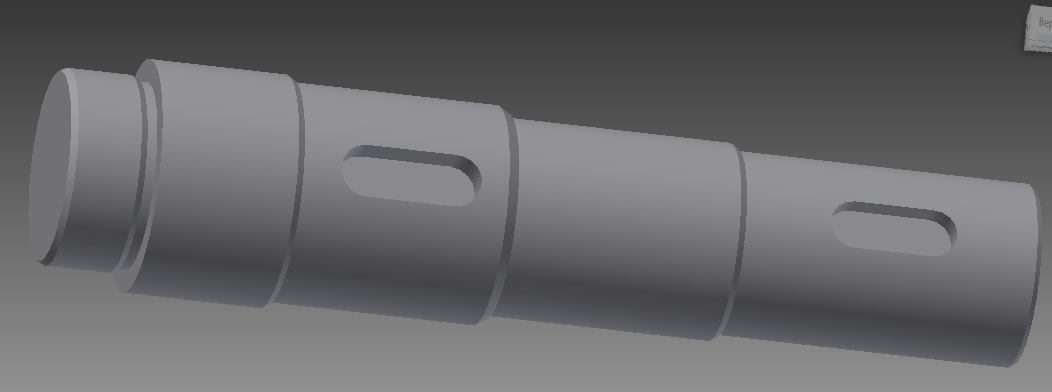


Рисунок 4.6 – Построение вала в САПР «Inventor» с максимальными параметрами

### 4.3.2 Модульное тестирование

Проведено тестирование на основе следующих тестовых сценариев (таблица 4.1).

Таблица 4.1 – Тестовые сценарии

|  |  |
| --- | --- |
| Модуль | Тестовые сценарии |
| Parameter | При установке значения параметра в допустимом диапазоне значение не должно измениться |
| При установке значения меньше минимального параметр должен стать равным минимуму |
| При установке параметра выше максимально возможного, он должен принять значения максимума |
| При установке неверного значения, оно не сохраняется |
| ValProperties | Тестирование метода SetCaption на правильность аргумента, подаваемого на вход. |
| Тестирование метода GetPointX на совпадение значений принимаемых на вход метода. |
| Тестирование метода GetPointZ на совпадение значений принимаемых на вход метода. |

### 4.3.3 Нагрузочное тестирование

На рисунках (4.7-4.8) представлено время построения модели в зависимости от номера вызова:

Рисунок 4.7 – График времени построения модели при 20 запусках

Рисунок 4.8 – График времени построения модели при 50 запусках

Среднее время построения при 20 моделей = 3,42456334

Среднее время построения при 50 моделей = 4,483977172

Периодическое повышение времени построения можно обусловить запуском автоматической очистки мусора. Резкий скачок в середине диаграммы, возможно, связан с очисткой динамической памяти и файла подкачки.

# 5 Заключение

В ходе выполнения курсового проекта были изучены основные этапы проектирования программного продукта, была изучена предметная область объекта проектирования, API САПР Inventor. Реализовано приложение «Построение модели пятиступенчатого вала» для системы Inventor. В приложении предусмотрена смена параметров вала, а также динамическая корректировка ошибок ввода.

# Список использованных источников

1. Разработка приложений для Autodesk Inventor. [Электронный ресурс]. – Режим доступа: http://www.autodesk.ru/adsk/servlet/index?id=22740406&siteID=871736 (дата обращения: 29.09.15)
2. API Inventor. [Электронный ресурс]. – Режим доступа: http://www.sapr.ru/article.aspx?id=20656&iid=941 (дата обращения: 29.09.15)
3. Справка: Генератор компонентов вала. [Электронный ресурс]. – Режим доступа: ttp://help.autodesk.com/view/INVNTOR/2015/RUS/?guid=GUID-9BDDFABF-0A38-4A10-AFDC-4D5450A51DD6 (дата обращения: 02.11.15)
4. Autodesk Inventor — Википедия. [Электронный ресурс].– Режим доступа: http://ru.wikipedia.org/wiki/Autodesk\_Inventor (дата обращения: 02.11.15)
5. Технологический процесс на изготовление детали – ступенчатый вал. [Электронный ресурс].– Режим доступа: http://5fan.ru/wievjob.php?id=42954 (дата обращения: 02.11.15)
6. UML tools for software development and modelling – Enterprise Architect [Электронный ресурс]. – Режим доступа: UML modeling tool. <http://www.sparxsystems.com.au/> (дата обращения: 10.10.15)
7. Современные методы и средства тестирования web-приложений [Электронный ресурс]. – Режим доступа: http://studopedia.su/20\_44894\_sovremennie-metodi-i-sredstva-testirovaniya-web-prilozheniy.html (дата обращения: 02.11.15)
8. Юнит-тестирование для чайников / Хабрахабр [Электронный ресурс]. – Режим доступа: https://habrahabr.ru/post/169381/ (дата обращения: 02.11.15)
9. Quality Assurance Patterns - Нагрузочное тестирование (Load testing) [Электронный ресурс]. – Режим доступа: http://qa-patterns.com.ua/articles/23 (дата обращения: 02.11.15)
10. Проектирование программного обеспечения/ Хабрахабр. [Электронный ресурс]. – Режим доступа: <http://habrahabr.ru/post/74330> (дата обращения: 02.11.15)
11. Фаулер М. UML. Основы, 3-е издание. – Пер. с англ. – СПб: Символ-Плюс, 2004. – 192с., ил.
12. ГОСТ 10948-64 - Радиусы закруглений и фаски. М., 1964 – 4 с.
13. ГОСТ 23360-78 - Основные нормы взаимозаменяемости. Соединения шпоночные с призматическими шпонками. Размеры шпонок и сечений пазов. Допуски и посадки. М., 1978. – 19с.

# Приложение А – Диаграмма вариантов использования

Диаграммы вариантов использования программного модуля.



Рисунок А.1 – Диаграмма вариантов использования «Работа с плагином»



Рисунок А.2 – Диаграмма вариантов использования «Работа с плагином» после реализации

# Приложение Б – диаграмма классов

Диаграммы классов программного модуля



Рисунок Б.1 – Диаграмма классов программного модуля до реализации



Рисунок Б.2 – Диаграмма классов программного модуля после реализации

# Приложение В – Описание классов

Описание классов, методов и их назначения. В таблицах приложения для обозначения модификаторов доступа приняты следующие условные знаки:

«−» − обозначение private (открытого) поля;

«+» − обозначение public (открытого) поля.

Таблица В.1 – Описание полей и методов класса ValForm

|  |  |
| --- | --- |
| Класс\поля\методы | Описание |
| +ValForm() | Конструктор класса ValForm, для инициализации компонентов и параметров |
| –InitParameters():void | Метод инициализация параметров |
| –Form\_Load(object, EventArgs):void | Метод загрузка формы |
| –BuildVal\_Click(object, EventArgs):void | Метод, обрабатывающий функции построения вала |
| –\_inventorApi: InventorApi | Интерфейс API Inventor |
| –\_valModel: ValModel | Модель вала |
| –\_valProperties: ValProperties = new ValProperties() {readOnly} | Параметры модели вала |

Таблица В.2 – Описание полей и методов класса InventorApi

|  |  |
| --- | --- |
| Класс\поля\методы | Описание |
| +InventorApi() | Класс, в котором содержатся используемые методы Inventor API |
| +MakeNewWorkingPlane(int n, double offset) | Метод, создающий новую рабочую плоскость |
| +DrawRectangle(double pointOneX, double pointOneY, double pointTwoX, double pointTwoY) | Метод, рисующий прямоугольник на активном скетче |
| –\_invApp: Application | Ссылка на приложение Inventor |
| –\_currentSketch: PlanarScetch | Текущий эскиз |

Окончание таблицы В.2

|  |  |
| --- | --- |
| +DrawCircle(double centerPointX, double centerPointY, double diameter) | Метод, рисующий круг на активном скетче |
| +CutExtrudeRectangle(double pointOneX, double pointOneY, double pointTwoX, double pointTwoY, double distance) | Метод, рисующий прямоугольник и выдавливающий его с вычитанием |
| +Extrude(double distance, PartFeatureExtentDirectionEnum extrudeDirection = PartFeatureExtentDirectionEnum.kPositiveExtentDirection) | Метод, выдавливающий текущий эскиз |
| +CutExtrudeCircle(double centerPointX, double centerPointY, double diameter, double distance) | Метод, рисующий круг и выдавливающий его с вычитанием |
| –\_partDef: PartComponentDefinition | Описание документа |
| –\_pertDoc: PartDocument | Создание документа в приложении |
| –\_transGeometry: TransientGeometry | Геометрия приложения |

Таблица В.3– Описание полей и методов класса ValProperties

|  |  |
| --- | --- |
| Класс\поля\методы | Описание |
| +ValProperties | Класс, хранящий в себе все параметры модели, так же осуществляет проверку зависимых параметров |
| –ParameterChangedValue(object sender, EventArgs e) | Событие, вызываемое при изменении какого-либо параметра модели |
| +GetParameter(ParameterType parameterType) | Возвращает указанный параметр |
| –SetParameters () | Данное свойство проводит проверку зависимых друг от друга параметров и динамически их изменяет |
| –\_parametrs: Dictionery<ParametrType, Parametr> | Словарь параметров вала |

Таблица В.4 – Описание полей и методов класса Parameter

|  |  |
| --- | --- |
| Класс\поля\методы | Описание |
| +Parameter | Класс предназначен для хранения и обработки параметра вала |
| +Validate(): void | Данный метод меняет значение параметра на корректное в случае выхода его за пределы допустимого диапазона |
| +ParameterChanged(): EventHandler | Данное событие вызывается в случае, если значение параметра было изменено |
| +Value(): double | Данное свойство позволяет установить или запросить текущее значение параметра |
| +Min(): double | Данное свойство позволяет установить или запросить минимальное значение параметра |
| +Max(): double | Данное свойство позволяет установить или запросить максимальное значение параметра |
| –\_max: double | Максимально возможное значение параметра |
| –\_min: double | Минимальное возможное значение параметра |
| –\_value: double | Значение параметра |

Таблица В.5– Описание полей и методов класса ValModel

|  |  |
| --- | --- |
| Класс\поля\методы | Описание |
| +ValModel(ValProperties, InventorApi) | Класс для построения модели вала |
| +Build(): double | Метод, строящий модель вала |
| –BuildShaft(): void | Метод, строящий ступени вала |
| –BuildKeyway(): void | Метод, строящий шпоночные пазы |
| –\_api: InventorApi | Интерфейс API Inventor |
| –\_valPropertiea: ValProperties | Параметры модели вала |

Таблица В.6– Описание перечислителей перечисления ParametrType

|  |  |
| --- | --- |
| Список перечислителей | Описание |
| ShaftDiameter1Stage | Диаметр первой ступени |
| ShaftDiameter2Stage | Диаметр второй ступени |
| ShaftDiameter3Stage | Диаметр третий ступени |
| ShaftDiameter4Stage | Диаметр четвертой ступени |

Окончание таблицы В.6

|  |  |
| --- | --- |
| ShaftDiameter5Stage | Диаметр пятой ступени |
| ShaftLength1Stage | Длина первой ступени |
| ShaftLength2Stage | Длина второй ступени |
| ShaftLength3Stage | Длина третий ступени |
| ShaftLength4Stage | Длина четвертой ступени |
| ShaftLength5Stage | Длина пятой ступени |
| GrooveHeight | Глубина канавки |
| WidthKeyway1Stage | Ширина шпоночного паза первой ступени |
| WidthKeyway3Stage | Ширина шпоночного паза третьей ступени |
| LengthKeyway1Stage | Длина шпоночного паза первой ступени |
| LengthKeyway3Stage | Длина шпоночного паза третьей ступени |
| HeightKeyway1Stage | Высота шпоночного паза первой ступени |
| HeightKeyway3Stage | Высота шпоночного паза третьей ступени |
| RadiusKeyway1Stage | Радиус шпоночного паза первой ступени |
| RadiusKeyway3Stage | Радиус шпоночного паза третьей ступени |