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)

ПЛАГИН «ГЕНЕРАТОР ОТВЕРТОК»

ДЛЯ СИСТЕМЫ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ

КОМПАС-3D

Пояснительная записка по дисциплине

«Основы разработки САПР»

Студент гр. 589-3

\_\_\_\_\_\_\_\_\_ Бабушкин А.П.

\_\_\_\_\_\_\_\_\_

Руководитель

преподаватель каф., к.т.н.

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

(оценка) \_\_\_\_\_\_\_\_\_

Томск 2022

**Реферат**

Учебная работа 33 страницы, 12 таблиц, 18 рисунков, 12 источников.

Ключевые слова: КОМПАС-3D, ОТВЕРТКА, СИСТЕМА АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ, БИБЛИОТЕКА.

Целью данной работы является разработка библиотеки «Генератор отверток» для системы автоматизированного проектирования Компас-3D.

В процессе работы должны были пройдены все стадии разработки программного обеспечения: выбор темы, составление технического задания, составление проекта системы, создание первой реализации и её тестирование, создание конечной библиотеки.

Отчет по учебной работе выполнен в текстовом редакторе Microsoft Word 2019.

**Оглавление**

[Введение 4](#_Toc122720374)

[1 ПОСТАНОВКА И АНАЛИЗ ЗАДАЧИ 5](#_Toc122720375)

[1.1 Описание предмета проектирования 5](#_Toc122720376)

[1.2 Описание инструментов и средств реализации 8](#_Toc122720377)

[1.3 Назначение плагина 8](#_Toc122720378)

[2 ОБЗОР АНАЛОГОВ 9](#_Toc122720379)

[2.1 Программа построения 3D моделей по заданным значениям в AutoCAD «Лекало» 9](#_Toc122720380)

[2.2 Плагин «BookGen» для Blender 10](#_Toc122720381)

[3 ОПИСАНИЕ РЕАЛИЗАЦИИ 12](#_Toc122720382)

[4 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЯ 21](#_Toc122720383)

[5 ТЕСТИРОВАНИЕ ПРОГРАММЫ 24](#_Toc122720384)

[5.1 Функциональное тестирование 24](#_Toc122720385)

[5.2 Модульное тестирование 26](#_Toc122720386)

[5.3 Нагрузочное тестирование 27](#_Toc122720387)

[Заключение 31](#_Toc122720388)

[Список использованных источников 32](#_Toc122720389)

# Введение

Автоматизация проектирования 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 проектированием 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.

Практическая реализация методов и идей автоматизированного проектирования происходит в рамках систем автоматизированного проектирования (САПР). В рамках современного «компьютеризированного» общества инженер любой специальности, занимающийся разработкой технических устройств, должен уметь использовать средства автоматизированного проектирования. Это позволяет повысить эффективность проектирования, улучшить его качество, снизить материальные затраты и уменьшить число разработчиков.

Таким образом, целью учебной работы является разработка плагина «Генератор отверток» для системы автоматизированного проектирования Компас-3D. «Компас» – это семейство систем автоматизированного проектирования, универсальная система автоматизированного проектирования, позволяющая в оперативном режиме выпускать чертежи изделий, схемы, спецификации, таблицы, инструкции, расчётно-пояснительные записки, технические условия, текстовые и прочие документы [1].

# 1 ПОСТАНОВКА И АНАЛИЗ ЗАДАЧИ

В рамках учебной дисциплины «Основы разработки САПР» требовалось разработать плагин в соответствии с техническим заданием. На основе заданных параметров программа, взаимодействуя с САПР «Компас-3D», должна строить трёхмерную модель отвертки [2]. Также плагин должен позволять изменять входные параметры получаемой модели.

Изменяемые параметры:

* вид наконечника (плоский, крестовой, треугольный);
* высота наконечника стержня отвертки;
* диаметр рукоятки отвертки;
* длина внешней части стержня;
* длина рукоятки отвертки;
* длина внутренней части стержня;
* наличие и длина закрепляющих крылышек стержня.

# 1.1 Описание предмета проектирования

Предметом проектирования является отвертка. На рисунке 1.1 представлен ее чертеж.

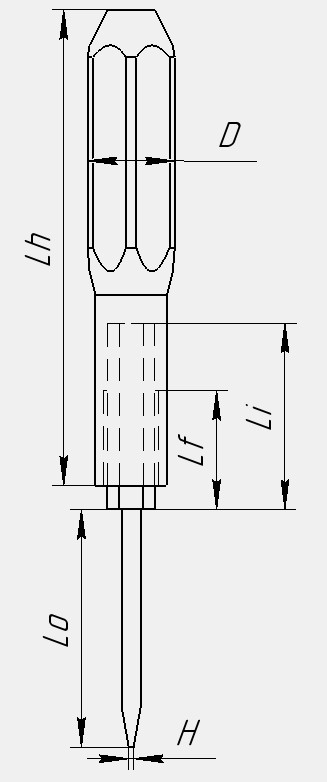


Рисунок 1.1 – чертеж отвертки

На рисунке 1.2 представлена ее 3D-модель.

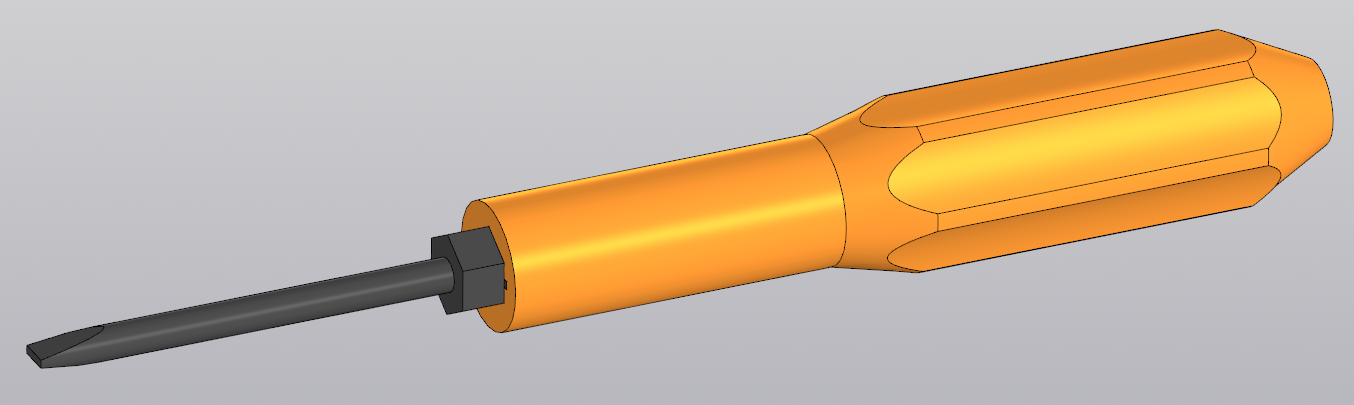
****

Рисунок 1.2 – 3D-модель отвертки

Доступные к изменению параметры отвертки:

* Наконечник – На выбор будет представлено 3 наконечника из списка - плоский, крестовой и треугольный (см. рис 1.3);

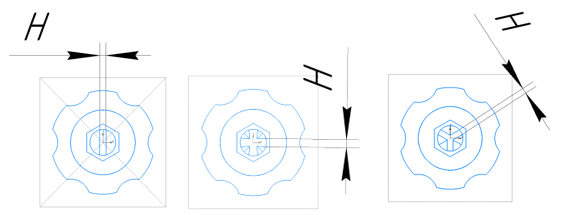


Рисунок 1.3 – чертеж доступных к выбору наконечников

* H – Высота наконечника стержня отвертки (0,1 - 10 мм);
* D – Самая широкая часть рукоятки (1,6 - 240 мм) – Не должна быть меньше, чем H \* 16 и больше, чем H \* 24;
* Lо – Длина внешней части стержня (1 - 400 мм) – Не должна быть меньше, чем H \* 20 и больше, чем H \* 400;
* Lh – Длина рукоятки отвертки (6 - 900 мм) – Не должна быть меньше, чем D \* 3,75 и больше, чем D \* 7,5;
* Li – Длина внутренней части стержня (3 - 540 мм) – Не должна быть меньше, чем Lh \* 0,5 и больше, чем Lh \* 0,6;
* Lf – Длина закрепляющих крылышек отвертки (0,6 - 450 мм) – Не должна быть меньше, чем Lh \* 0,1 и больше, чем Lh \* 0,5. Данное значение необязательно, и, при его отсутствии, крылышки не будут сгенерированы на модели отвертки.

# 1.2 Описание инструментов и средств реализации

Для создания программы используется среда разработки Visual Studio 2019 [3]. Плагин написан с пользовательским интерфейсом на WindowsForms с использованием .NET Framework 4.7.2 [4]. Инструментом для тестирования является библиотеки NUnit 3.13.3 [5] и NUnit3TestAdapter 4.2.1 [5]. В качестве системы автоматизированного проектирования выбран «Компас-3D» v.18.1 [1].

# 1.3 Назначение плагина

Программа предназначена для автоматизации моделирования сборки «Отвертка».

Плагин позволяет пользователю ввести вышеперечисленные значения через графический интерфейс. В программе предусмотрена проверка корректности введенных данных и сообщение пользователю о неправильно заполненных полях с помощью цветового выделения и описания ошибки в нижней части формы.

При вводе некорректных значений программа выводит сообщение об ошибке и не позволяет начать построение модели.

При правильно введенных значениях результатом работы программы будет созданные по ним две модели ручки и стержня отвертки, а также их сборки в единую модель. Размеры всех параметров, необходимых при построении, рассчитываются автоматически.

# 2 ОБЗОР АНАЛОГОВ

2.1 Программа построения 3D моделей по заданным значениям в AutoCAD «Лекало»

Данная программа позволяет создавать следующие 3D модели в AutoCAD посредством ввода размеров с клавиатуры [6]:

* металлопрокат;
* механические соединения;
* механические передачи;
* элементы гидро- и пнемвоприводов;
* построение конструктивных элементов.

На рисунке 2.1 представлен пользовательский интерфейс программы «Лекало» для построения втулки.



Рисунок 2.1 – Пользовательский интерфейс программы «Лекало» для построения втулки

# 2.2 Плагин «BookGen» для Blender

Аналогом приложения для генерации однотипных 3D-моделей можно посчитать плагин BookGen для 3D-редактора Blender. В нем можно создавать объекты, которые будут заполняться случайно сгенерированными книгами. Можно изменять размеры генерируемых книг, материалы страниц и обложки, расстояние между книгами, а также прочие настройки. К каждой числовой настройке можно поменять свойство случайности, для придания набору книг неповторимости.

Концепция данного плагина, во многом, схожа с лабораторным приложением. Единственное глобальное отличие – приложение не генерирует множество изменяемых объектов, а ограничится созданием лишь одного экземпляра изменяемого объекта. Соответственно, значения случайности в будущем приложении не понадобятся. [7]

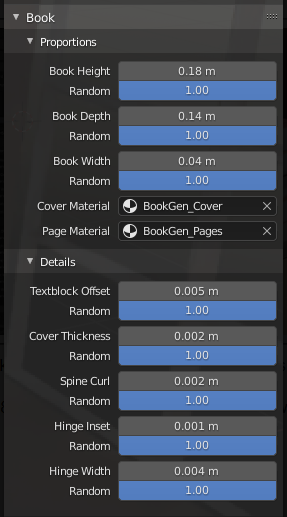


Рисунок 2.2 – Окно настроек объекта дополнения



Рисунок 2.3 – Сцена, сгенерированная при помощи данного плагина

# 3 ОПИСАНИЕ РЕАЛИЗАЦИИ

**UML (Unified Modeling Language)** – это система обозначений, которую можно применять для объектно-ориентированного анализа и проектирования. Его можно использовать для визуализации, спецификации, конструирования и документирования программных систем. Словарь UML включает три вида строительных блоков: диаграммы, сущности, связи [8].

**Sparx Systems Enterprise Architect** – это инструмент визуального моделирования и проектирования, основанный на OMG UML. Платформа поддерживает: проектирование и построение программных комплексов; моделирование бизнес-процессов; и моделирование отраслевых доменов. Он используется предприятиями и организациями не только для моделирования архитектуры своих систем, но и для обработки реализации этих моделей на протяжении всего жизненного цикла разработки приложений [9].

Класс ScrewdriverParameter предназначен для описания каждого изменяемого параметра отвертки и его валидацию, класс ScrewdriverData предназначен для объединения всех параметров воедино, класс ScrewdriverBuilder предназначаен для построения отвертки, класс MainForm предназначен для описания пользовательского интерфейса, перечисление ScrewdriverParametersType необходимо для валидации параметров и корректной связи их с MainForm, класс KompasWrapper предназначен для взаимодействия с САПР «Компас-3D», в нём содержатся методы для связи с САПР и рисования в ней. На рисунке 3.1 представлена UML-диаграмма классов, составленная в графическом редакторе draw.io с использованием технических возможностей Enterprise Architect.

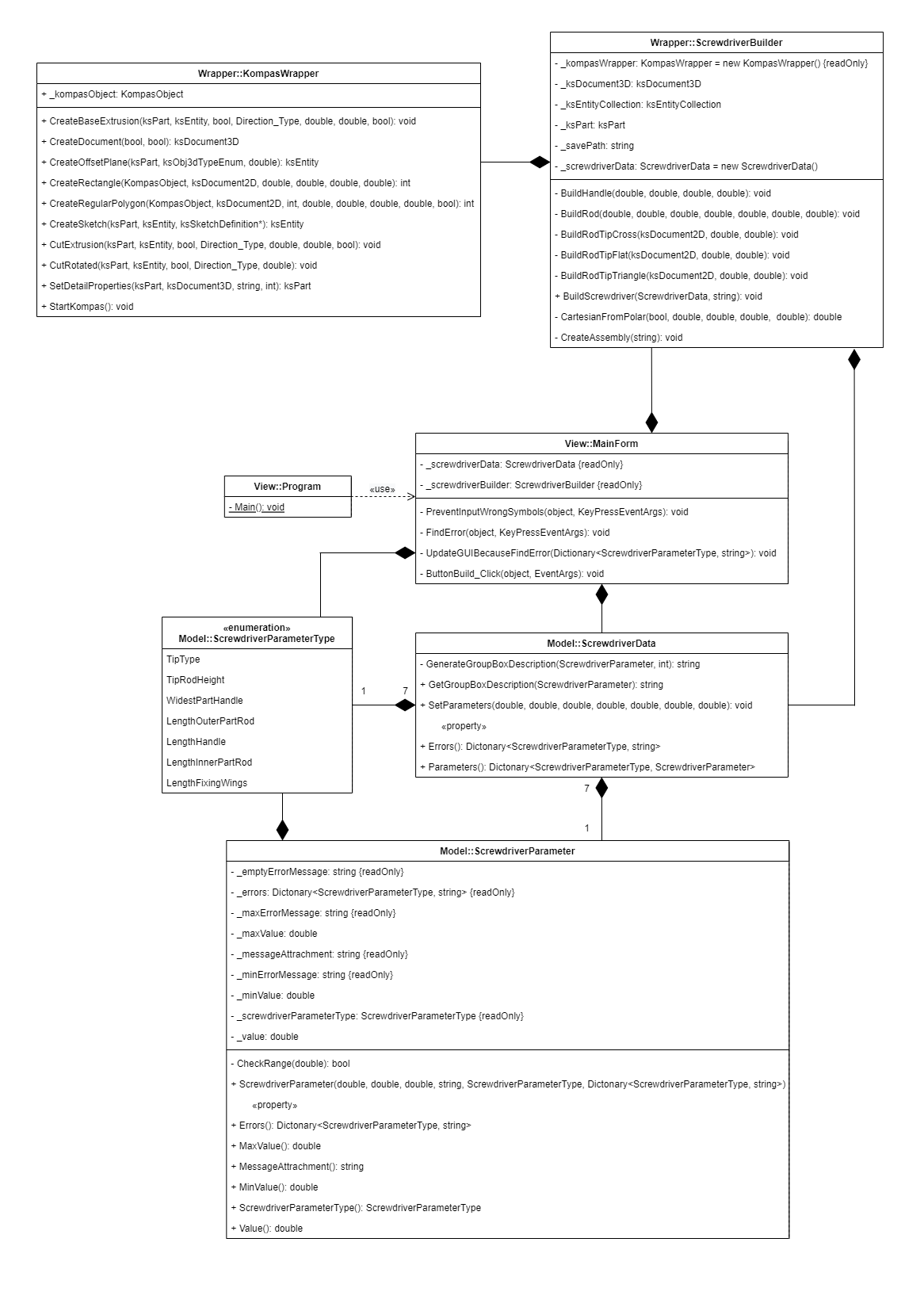


Рисунок 3.1 - UML-диаграмма классов текущей версии программы.

Таблица 3.1 – Класс ScrewdriverParameter

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Поля** | \_emptyErrorMessage | string | Сгенерированное сообщение ошибки о пустом значении. |
| \_errors | Dictonary<ScrewdriverParameterType, string> | Библиотека ошибок, связанных с этим параметром. |
| \_maxErrorMessage | string | Сгенерированное сообщение ошибки о слишком большом значении параметра. |
| \_maxValue | double | Максимальное значение параметра. |
| \_messageAttrachment | string | Название параметра, присоединяемое к сообщениям об этом параметре. |
| \_minErrorMessage | string | Сгенерированное сообщение ошибки о слишком маленьком значении параметра. |
| \_minValue | double | Минимальное значение параметра. |
| \_screwdriverParameterType | ScrewdriverParameterType | Тип данного параметра. |
| \_value | double | Значение параметра. |
| **Методы** | CheckRange | bool | Проверка входа параметра в рамки значений. |
| ScrewdriverParameter | Конструктор | Создает один из параметров данных отвертки. |
| **Свойства** | Errors | Dictonary<ScrewdriverParameterType, string> | Возвращает библиотеку ошибок параметра. |
| MaxValue | double | Устанавливает или возвращает максимально возможное значение параметра. |
| MessageAttrachment | string | Возвращает имя параметра, прибавляемое к сообщениям. |
| MinValue | double | Устанавливает или возвращает минимально возможное значение параметра. |
| ScrewdriverParameterType | ScrewdriverParameterType | Возвращает тип параметра. |
| Value | double | Возвращает или устанавливает значение параметра. |

Таблица 3.2 – Класс ScrewdriverData

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Методы** | GenerateGroupByBoxDescription | string | Генерация сообщения описания вида данных отвертки. |
| GetGroupBoxDescription | string | Получить описание вида данных отвертки с их границами. |
| SetParameters | void | Устанавливает значения параметров отвертки. |
| **Свойства** | Errors | Dictionary<  ParameterType, string> | Устанавливает и возвращает список ошибок параметров. |
| Parameters | Dictionary<  ParameterType, ScrewdriverParameter> | Устанавливает и возвращает список параметров. |

Таблица 3.3 – Класс MainForm

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Поля** | \_screwdriverData | ScrewdriverData | Объект данных об отвертке. |
| \_screwdriverBuilder | ScrewdriverBuilder | Объект, строящий отвертку в Компас-3D. |
| **Методы** | PreventInputWrongSymbols | void | Предотвращает ввод некорректных символов. |
| FindError | void | Находит и реагирует на ошибки в TextBox. |
| UpdateGUIBecauseFindError | void | Обновляет элементы формы из-за ошибок. |
| ButtonBuild\_Click | void | Реакция на нажатие кнопки «Построить». |

Таблица 3.4 – Перечисление ScrewdriverParameterType

|  |  |
| --- | --- |
| **Название элемента** | **Описание** |
| TipType | Вид наконечника отвертки. |
| TipRodHeight | Высота наконечника стержня отвертки. |
| WidestPartHandle | Самая широкая часть рукоятки. |
| LengthOuterPartRod | Длина внешней части стержня. |
| LengthHandle | Длина рукоятки отвертки. |
| LengthInnerPartRod | Длина внутренней части стержня. |
| LengthFixingWings | Длина закрепляющих крылышек. |

Таблица 3.5 – Класс KompasWrapper

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Поля** | \_kompasObject | KompasObject | Объект Компас API. |
| **Методы** | CreateBaseExtrusion | void | Создать выдавливание на основе эскиза. |
| CreateDocument | ksDocument3D | Создание документа в Компас-3D. |
| CreateOffsetPlane | ksEntity | Метод создания новой плоскости. |
| CreateRectangle | int | Создание прямоугольника. |
| CreateRegularPolygon | int | Создание многоугольника. |
| CreateSketch | ksEntity | Создание скетча. |
| CutExtrusion | void | Вырезание выдавливанием по эскизу. |
| CutRotated | void | Вырезание выдавливанием по эскизу вокруг оси. |
| SetDetailProperties | ksPart | Установка основных свойств детали. |
| StartKompas | void | Запуск экземпляра Компас-3D. |

Таблица 3.6 – Класс ScrewdriverBuilder

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Поля** | \_kompasWrapper | KompassWrapper | Связь с Компас-3D. |
| \_ksDocument3D | ksDocument3D | Класс взаимодействия с документом Компас-3D. |
| \_ksEntityCollection | ksEntityCollection | Класс взаимодействия с коллекциями в Компас-3D. |
| \_ksPart | ksPart | Класс взаимодействия с деталью в Компас-3D. |
| \_savePath | string | Путь сохранения сгенерированных объектов. |
| \_screwdriverData | ScrewdriverData | Объект параметров отвертки. |

Окончание таблицы 3.6

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Методы** | BuildHandle | void | Построение детали ручки отвертки. |
| BuildRod | void | Построение детали стержня отвертки. |
| BuildRodTipCross | void | Построение крестового наконечника отвёртки. |
| BuildRodTipFlat | void | Построение плоского наконечника отвёртки. |
| BuildRodTipTriangle | void | Построение треугольного наконечника отвертки |
| BuildScrewdriver | void | Построение деталей и сборки отвертки. |
| CartesianFromPolar | double | Переводит в декартовые координаты полярные. |
| CreateAssembly | void | Объединяет детали ручки и стержня в сборку отвертки. |

В процессе реализации плагина и его тестирования, в сравнении с первоначальным планом, указанным в проекте системы, были полностью переработаны классы взаимодействия с Компас-API: в них добавились конкретные методы и поля, необходимые для их корректной работы. Также были пересмотрены несколько методов из MainForm, добавлены методы по выводу информации о параметре в ScrewdriverData, добавлены новые, необходимые поля в ScrewdriverParameter и методы для них. Также, были сделаны необходимые для добавления дополнительного функционала изменения - добавлен новый тип параметра, а также его обработка и использование. Далее будут приведены изначальные планы по созданию этой программы, описанные в проекте системы:

Таблица 3.7 – Неактуальное перечисление ScrewdriverParameterType

|  |  |
| --- | --- |
| **Название элемента** | **Описание** |
| TipType | Вид наконечника отвертки. |
| TipRodHeight | Высота наконечника стержня отвертки. |
| WidestPartofHandle | Самая широкая часть рукоятки. |
| LengthofOuterPartofRod | Длина внешней части стержня. |
| LengthHandle | Длина рукоятки отвертки. |
| LengthofInnerPartofRod | Длина внутренней части стержня. |

Таблица 3.8 – Неактуальный класс Parameter

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Поля** | \_error | Dictonary<ScrewdriverParameterType, string> | Библиотека ошибок, связанных с этим параметром. |
| \_maxValue | double | Максимальное значение параметра. |
| \_minValue | double | Минимальное значение параметра. |
| \_parameterType | ScrewdriverParameterType | Тип данного параметра. |
| \_value | double | Значение параметра. |
| **Методы** | Parameter | Конструктор | Создает один из параметров данных отвертки. |

Таблица 3.9 – Неактуальный класс ScrewdriverData

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Поля** | \_error | Dictonary<ScrewdriverParameterType, string> | Библиотека ошибок. |
| \_parameters | Dictonary<ScrewdriverParameterType, Parameter> | Библиотека параметров отвертки |
| **Методы** | Screwdriver | void | Устанавливает значения параметров отвертки. |
| GetParameters | Dictonary<ScrewdriverParameterType, Parameter> | Возвращает библиотеку параметров отвертки. |

Таблица 3.10 – Неактуальный класс MainForm

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Поля** | \_screwdriverData | ScrewdriverData | Объект данных об отвертке. |
| \_screwdriverBuilder | ScrewdriverBuilder | Объект, строящий отвертку в Компас-3D. |
| **Методы** | CheckEmptyTextBox | void | Проверка TextBox на пустоту. |
| ErrorTextUpdate | string | Обновление текста ошибки в форме. |
| Input\_InputChange | void | Реакция на изменение текста в TextBox. |
| ButtonBuild\_Click | void | Реакция на нажатие кнопки «Построить». |

Таблица 3.11 – Неактуальный класс ScrewdriverBuilder

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Поля** | \_wrapper | KompassWrapper | Связь с Компас-3D. |
| **Методы** | BuildRod | void | Построение стержня отверки. |
| BuildHandle | void | Построение рукоятки отвертки. |

Таблица 3.12 – Неактуальный класс KompasWrapper

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Поля** | \_kompas | kompasObject | Объект Компас API. |
|  | \_document | ksDocument3D | Объект документа. |
|  | \_part | ksPart | Объект детали. |
| **Методы** | CreateDocument | void | Создать файл документа. |
| StartKompas | void | Запуск экземпляра Компас-3D. |

Неактуальная диаграмма классов представлена на рисунке 3.2.

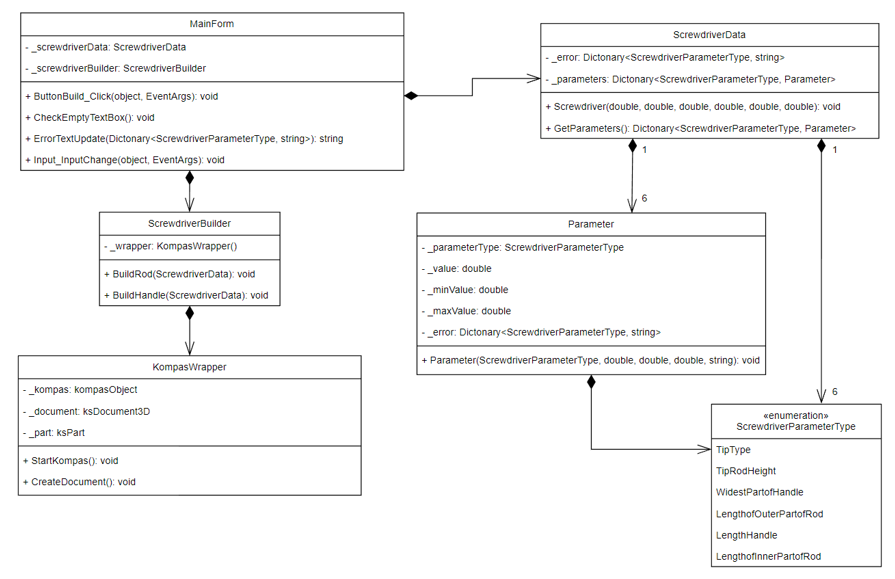


Рисунок 3.2 – Неактуальная диаграмма классов

# 4 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЯ

Пользовательский интерфейс (UI) – интерфейс, обеспечивающий передачу информации между пользователем-человеком и программно-аппаратными компонентами компьютерной системы [10].

Плагин представляет собой пользовательскую форму с полями для ввода соответствующих параметров, форму можно увидеть на рисунке 4.1. Посредством кнопки «Build» осуществляется запуск САПР «Компас-3D», на рабочей области которой построятся две модели деталей (ручки и стержня отвертки), а также модель-сборка из этих деталей. Также, произойдет сохранение этих документов в указанной пользователем папке. Если построение детали выполняется несколько раз, то плагин не запускает несколько копий программы «Компас-3D», а создает в ней новый документ. В случае ввода значений параметров, не входящих в допустимый диапазон, поле для ввода окрашивается в розовый цвет, снизу пишется, в чем пользователь ошибся, а также кнопка «Build» блокируется. Диапазон зависимых параметров изменяется динамически. Доступен выбор варианта наконечника отвертки. Доступно указание, в какую папку сохранится результат работы программы. По умолчанию, программа предлагает сохранить сгенерированные документы во временную папку на диске C. Пример интерфейса программы представлен на рисунке 4.1, пример неправильного ввода представлен на рисунке 4.2.

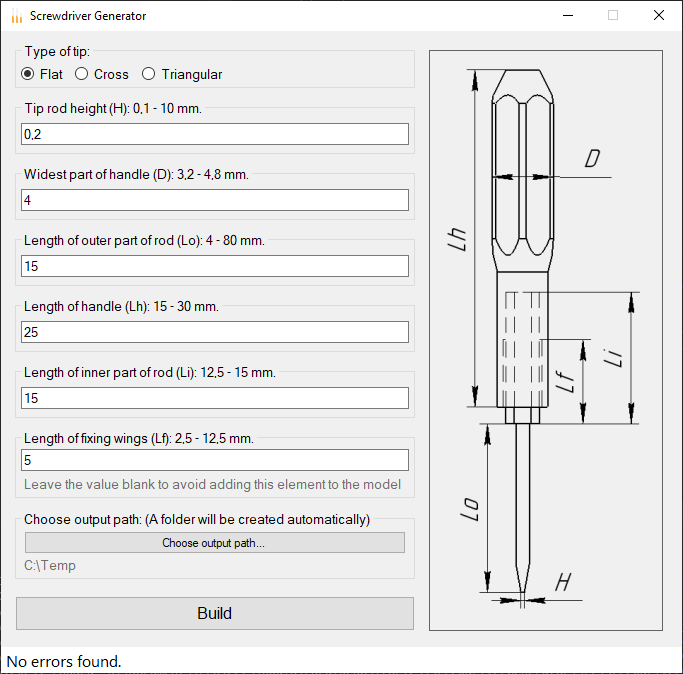


Рисунок 4.1 – Макет пользовательского интерфейса

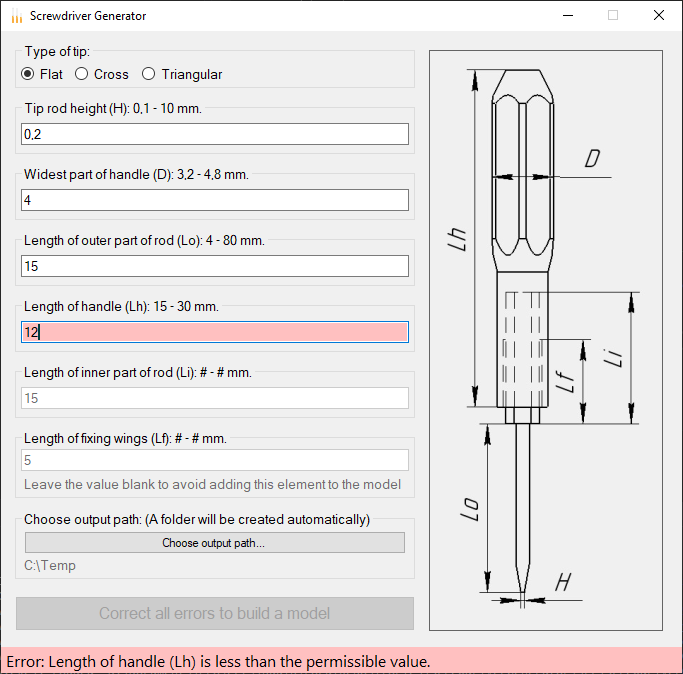


Рисунок 4.2 – Реакция приложения на ввод некорректных значений

Если пользователь ввёл правильные значения, при нажатии кнопки «Build» происходит построение детали. Трёхмерная модель пепельницы, построенная в системе автоматизированного проектирования Компас-3D с помощью разработанной библиотеки представлена на рисунке 4.3

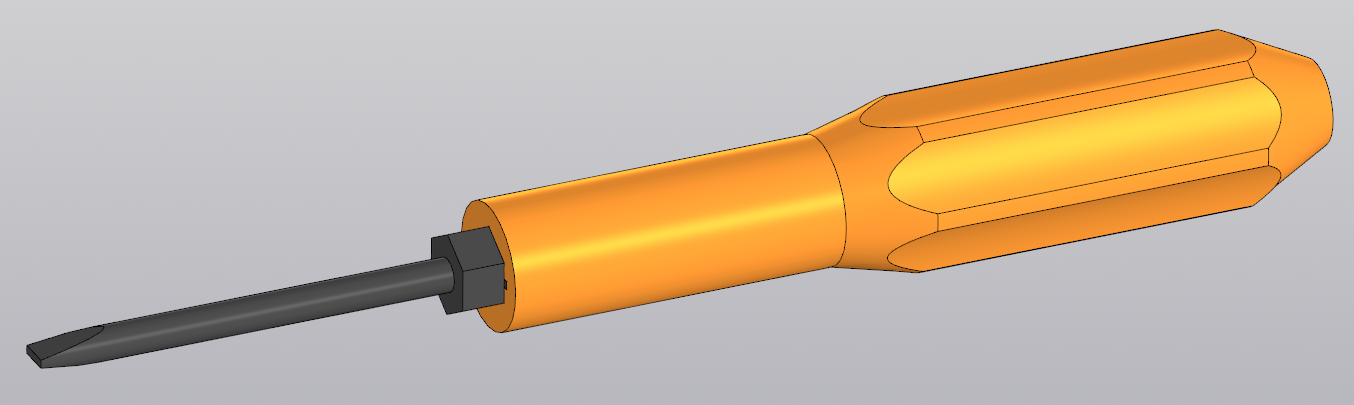
****

Рисунок 4.3 – Трёхмерная модель пепельницы

# 5 ТЕСТИРОВАНИЕ ПРОГРАММЫ

# 5.1 Функциональное тестирование

Функциональное тестирование – это тип тестирования программного обеспечения, при котором тестовые сценарии выполняются тестировщиком вручную без использования автоматизированных инструментов. Целью ручного тестирования является выявление ошибок, проблем и дефектов в программном приложении. Функциональное тестирование программного обеспечения – это самый примитивный метод из всех видов тестирования. Концепции ручного тестирования не требуют знания какого-либо инструмента тестирования [11].

В рамках функционального тестирования будет проверка библиотеки при введённых минимальных, средних и максимальных параметрах.

Тестирование при минимальных параметрах:

* высота наконечника 0,1 мм;
* ширина рукоятки 1,6 мм;
* длина внешней части стержня 2 мм;
* длина рукоятки 6 мм;
* длина внутренней части стержня 3 мм;
* длина фиксирующих крылышек 0,6 мм.

Результат построения при заданных параметрах представлен на рисунках 5.1.

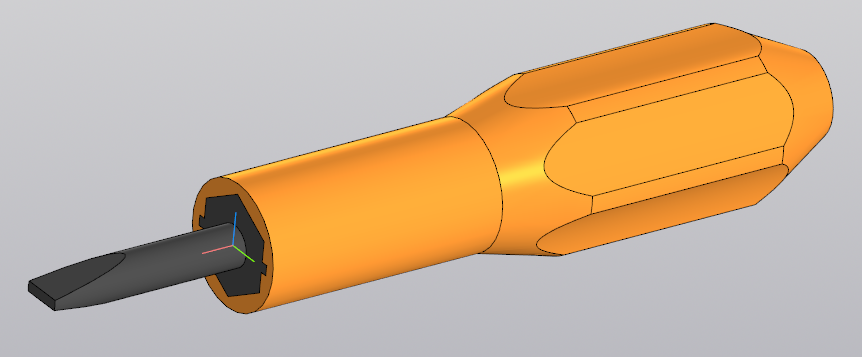


Рисунок 5.1 – Параметрический вид минимальной отвертки

Тестирование при средних параметрах:

* высота наконечника 0,2 мм;
* ширина рукоятки 4 мм;
* длина внешней части стержня 15 мм;
* длина рукоятки 25 мм;
* длина внутренней части стержня 15 мм;
* длина фиксирующих крылышек 5 мм.

Результат построения при заданных параметрах представлен на рисунках 5.2.

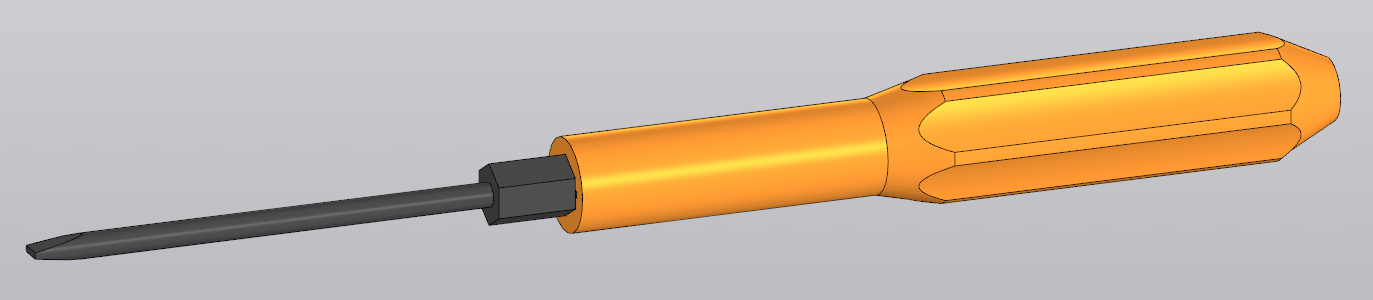


Рисунок 5.2 – Параметрический вид средней отвертки

Тестирование при максимальных параметрах:

* высота наконечника 10 мм;
* ширина рукоятки 240 мм;
* длина внешней части стержня 4000 мм;
* длина рукоятки 1800 мм;
* длина внутренней части стержня 1080 мм;
* длина фиксирующих крылышек 900 мм.

Результат построения при заданных параметрах представлен на рисунках 5.3.

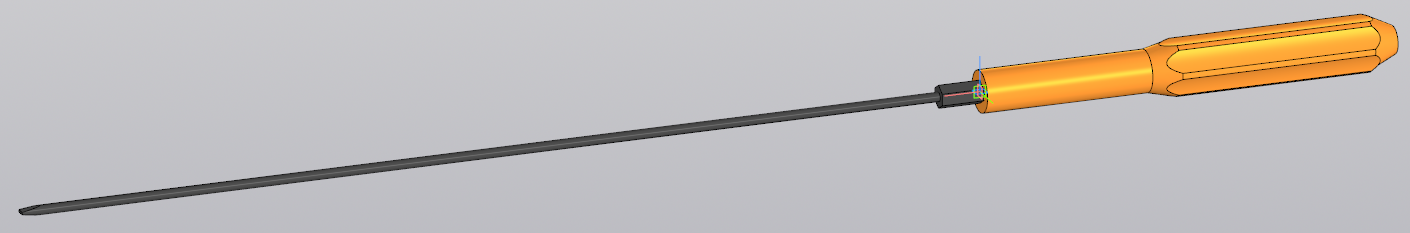


Рисунок 5.3 – Параметрический вид максимальной отвертки

# 5.2 Модульное тестирование

**Модульное тестирование (Unit Testing)**– это тип тестирования программного обеспечения, при котором тестируются отдельные модули или компоненты программного обеспечения. Его цель заключается в том, чтобы проверить, что каждая единица программного кода работает должным образом. Данный вид тестирование выполняется разработчиками на этапе кодирования приложения. Модульные тесты изолируют часть кода и проверяют его работоспособность. Единицей для измерения может служить отдельная функция, метод, процедура, модуль или объект [13].

Для тестирования используется библиотеки NUnit 3.13.3 [5] и NUnit3TestAdapter 4.2.1 [5]. Тестами покрыта вся логика программы. Всего было написано 27 тестов. На рисунке 5.4 представлено тестирование логики.

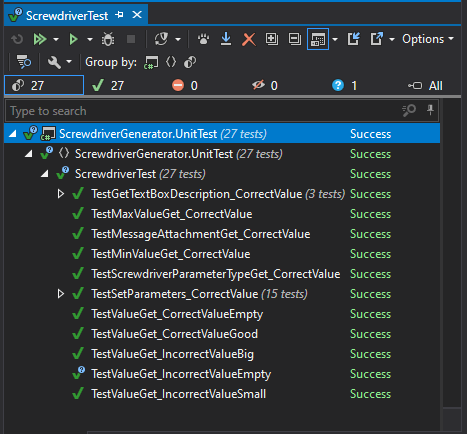


Рисунок 5.4 – Тестирование логики программы

# 5.3 Нагрузочное тестирование

Нагрузочное тестирование – это 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) [11].

Нагрузочное тестирование проводилось на ноутбуке HP Pavilion Gaming Laptop 15-ec0xxx со следующей конфигурацией:

* процессор AMD Ryzen 5 3550H with Radeon Vega Mobile Gfx, 2100 МГц, ядер: 4, логических процессоров: 8;
* установленная оперативная память (RAM) 16,0 ГБ;

Для тестирования будет зациклено построение модели со следующими параметрами:

* высота наконечника 0,2 мм;
* ширина рукоятки 4 мм;
* длина внешней части стержня 15 мм;
* длина рукоятки 25 мм;
* длина внутренней части стержня 15 мм;
* длина фиксирующих крылышек 5 мм.

В результате тестирования было построено 17 сборок, т.е, 51 модель. По полученным измерениям был рассчитан средний результат, по которому были построены графики. На рисунке 5.5 показана зависимость количества используемой оперативной памяти в гигабайтах от количества построенных деталей.

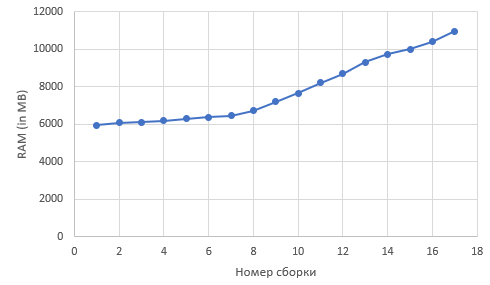


Рисунок 5.5 – Зависимость оперативной памяти от количества сборок

Из графика видно, что, на протяжении первых восьми сборок, используемая оперативная память практически не увеличивается. Связано это с тем, что Компас-3D, при запуске, резервирует себе некоторую часть оперативной памяти. Далее, зарезервированная память закончилась, и Компас-3D начал потихоньку забирать в свое пользование новую оперативную память, линейно увеличивая ее потребление. В конце концов, Компас-3D просто не смог создать новую модель, ссылаясь на недостаток оперативной памяти. [12]

На рисунке 5.6 показана зависимость потраченного времени от количества построенных сборок.

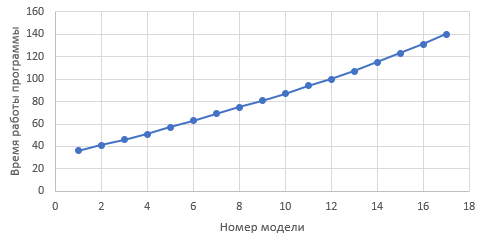


Рисунок 5.6 – Зависимость времени от количества построенных деталей

Из графика видно, что зависимость количества необходимого времени от числа деталей является линейной. Однако, между началом работы программы и началом создания первой сборки прошло больше времени, чем между началом создания остальных сборок, так как программе требовалось время на запуск Компас-3D. В некоторых тестах скорость создания моделей незначительно замедлялась к концу свободной оперативной памяти. Это происходило из-за большего количества действий, необходимых для выделения оперативной памяти.

На рисунке 5.7 показана зависимость использования процессора от количества построенных сборок.

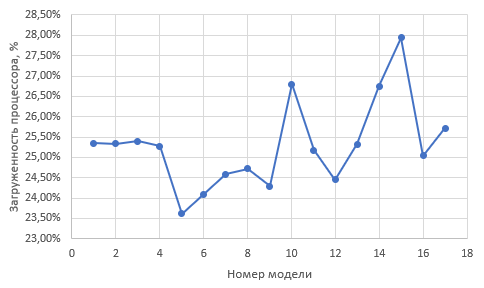


Рисунок 5.7 – Зависимость времени от количества построенных деталей

Из графика видно, что, несмотря на его кажущуюся хаотичность, загруженность процессора была практически неизменна, и держалась на отметке около 25,5%. Это происходило из-за того, что процессор никак не взаимодействовал с уже созданными сборками и деталями, а, значит, их количество практически никак не могло повлиять на его загруженность.

# Заключение

В результате выполнения данной учебной работы были пройдены все стадии разработки программного обеспечения: выбор темы, составление технического задания, составление проекта системы, создание первой реализации и её тестирование, создание конечной программы. Был изучен API приложения Компас-3D. Также были найдены аналоги разрабатываемого плагина. Были спроектированы UML диаграммы классов и было проведено функциональное, модульное и нагрузочное тестирования созданной программы.

# Список использованных источников

1. Официальный сайт системы автоматизированного проектирования «Компас-3D» [Электронный ресурс] – Режим доступа: https://kompas.ru/kompas-3d/about/ (дата обращения: 30.09.2022)
2. Компас (САПР) [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Компас\_(САПР) (дата обращения: 30.09.2022)
3. Заметки о выпуске Visual Studio 2019 версии 16.10 2019 [Электронный ресурс] – Режим доступа: https://learn.microsoft.com/ru-ru/visualstudio/releases/2019/release-notes-v16.10 (дата обращения: 19.11.2022)
4. Платформа .NET Framework 4.7.2 для Windows [Электронный ресурс] – Режим доступа: https: //support.microsoft.com/ru-ru/topic/автономный-установщик-microsoft-платформа-net-framework-4-7-2-для-windows-05a72734-2127-a15d-50cf-daf56d5faec2 (дата обращения: 19.11.2022)
5. Фреймворк NUnit [Электронный ресурс] – Режим доступа: https://nunit.org/ (дата обращения: 19.11.2022)
6. Программа автоматического построения 3D моделей и разверток по заданным значениям в AutoCAD «Лекало». Расчет и построение механических передач [Электронный ресурс] – Режим доступа: https://www.2d-3d.ru/3d-galereia/autocad/811-programma-dlya-autocad-lekalo.html (дата обращения: 07.10.2022)
7. BookGen // Oliver Weissbarth Software Engineer, 3D-Graphics Enthusiast URL: https://www.oweissbarth.de/software/bookgen/ (дата обращения: 13.10.2022).
8. UML-диаграммы классов [Электронный ресурс] – Режим доступа: http://www.uml.org/ (дата обращения: 06.10.2022)
9. Enterprise Architect (software) [Электронный ресурс] – Режим доступа: https://sparxsystems.com/ (дата обращения: 06.10.2022)
10. Интерфейс пользователя [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Интерфейс\_пользователя (дата обращения: 15.12.2022)
11. Куликов С. Тестирование программного обеспечения. Базовый курс. 3-е издание, 301 с
12. Оперативная память [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Оперативная\_память (дата обращения: 15.12.2022)