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)

Студент гр. 586-2

\_\_\_\_\_\_\_\_\_\_\_\_\_Н.Р. Дууза

«\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2020г.

Руководитель

К.т.н., доцент каф. КСУП

\_\_\_\_\_\_\_\_\_\_А.А. Калентьев

«\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2020г.

2020

**РЕФЕРАТ**

Пояснительная записка 35 с., 27 рис., 18 источников, 1 прил., 9 таблиц.

SOLIDWORKS 2020, ПЛАГИН, ФОРМА ДЛЯ ВЫПЕЧКИ, АВТОМАТИЗИРОВАННОЕ ПРОЕКТИРОВАНИЕ, САПР.

Целью данной работы является разработка плагина, автоматизирующего построение модели «Форма для выпечки» для системы автоматизированного проектирования SOLIDWORKS 2020 [1] с помощью интегрированной среды разработки Visual Studio 2019 Сommunity [2].

В процессе работы изучены основные параметры формы для выпечки и основные функции API [3] системы автоматизированного проектирования SOLIDWORKS 2020.

В результате работы был создан плагин, автоматизирующий построение модели «форма для выпечки».

Отчет по пояснительной записке выполнен в текстовом редакторе Microsoft Word 2016 [4].

**Оглавление**

[1 Введение 4](#_Toc39015152)

[2 Постановка и анализ задачи 5](#_Toc39015153)

[2.1 Описание предмета проектирования 5](#_Toc39015154)

[2.2 Выбор инструментов и средств реализации 6](#_Toc39015155)

[2.3 Назначение плагина 6](#_Toc39015156)

[2.4 Описание аналогов разрабатываемого продукта 7](#_Toc39015157)

[2.4.1 Описание САПР КОМПАС-3D 7](#_Toc39015158)

[2.4.2 Плагин PDF для САПР КОМПАС-3D 7](#_Toc39015159)

[2.4.3 Экспорт из КОМПАС-3D в формат 3D PDF 8](#_Toc39015160)

[3 Описание реализации 9](#_Toc39015161)

[3.1 Описание технических и функциональных аспектов проекта 9](#_Toc39015162)

[3.2 Диаграмма вариантов использования (Use Cases) 9](#_Toc39015163)

[3.3 Диаграмма классов 10](#_Toc39015164)

[4 Описание программы для пользователя 15](#_Toc39015165)

[5 Тестирование программы 17](#_Toc39015166)

[5.1 Функциональное тестирование 17](#_Toc39015167)

[5.2 Модульное тестирование 20](#_Toc39015168)

[5.3 Нагрузочное тестирование 23](#_Toc39015169)

[Заключение 27](#_Toc39015170)

[Список использованных источников 28](#_Toc39015171)

[Приложение А 30](#_Toc39015172)

# 1 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 [5].

Таким образом, целью данной работы является разработка плагина, автоматизирующего построение модели «Форма для выпечки» для системы автоматизированного проектирования SOLIDWORKS 2020 с помощью интегрированной среды разработки Visual Studio 2019 Сommunity.

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.

# 2 Постановка и анализ задачи

В рамках лабораторных работ в соответствии с технически заданием требовалось разработать плагин, который на основе входных параметров, интегрируя с системой SOLIDWORKS 2020, строит модель «Форма для выпечки». Необходимо чтобы плагин позволял создавать новый проект, сохранять текущий построенный проект, очищать, задавать параметры по умолчанию, а также изменять входные параметры формы такие как длина, ширина, высота, количество и высоту стен вдоль длины и ширины фигуры.

# 2.1 Описание предмета проектирования

Форма для выпечки – предмет кухонной утвари для выпекания хлебобулочных изделий, кондитерских изделий (пирогов, кексов, тортов и печенья) и других блюд с использованием духового шкафа, хлебопекарной или кондитерской печи [6].

Параметры формы для выпечки:

* длина формы X (от 200 до 400 мм);
* ширина формы Y (от 200 до 400 мм);
* высота формы Z (от 50 до 150 мм);
* количество стенок по длине X (не больше (X–10) / (SX + 5) мм);
* расстояние между стенками SX по Y (от 20 мм до (X-35) мм);
* количество стенок по длине Y (не больше (Y–10) / (SY + 5) мм);
* расстояние между стенками SY по Y (от 20 мм до (Y-35) мм);
* высота стенок по длине X (от 5 мм до Z – 5 мм);
* высота стенок по ширине Y (от 5 мм до Z – 5 мм).

Пример проектируемого изделия приведен ниже, на рисунке 2.1.

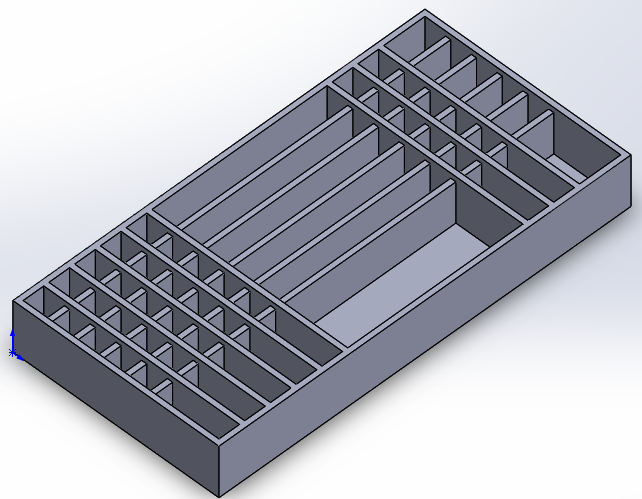


Рисунок 2.1 – Предмет изделия

# 2.2 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 Framework 4.8 [6], библиотеки «SolidWorks.Interop. sldworks» [7] версии 27.3 для основных операций в системе SOLIDWORKS 2020.

Инструментом тестирования и создания модульных тестов был выбран обозреватель тестов расширения для Visual Studio – ReSharper [8] с тестовым фреймворком NUnit [9] версии 3.12, процентное соотношение покрытия библиотеки тестами определено с помощью расширения для Visual Studio – dotCover [10].

Для реализации пользовательского интерфейса использовалась система для построения клиентских приложений Windows – Windows Presentation Foundation (WPF) [11].

# 2.3 Назначение плагина

Назначение разрабатываемого плагина обусловлено быстрым моделированием форм для выпечки разных типов. Благодаря данному расширению, кондитеры могут наглядно рассмотреть спроектированную модель, при необходимости перестроить под необходимые им параметры.

# 2.4 Описание аналогов разрабатываемого продукта

# 2.4.1 Описание САПР КОМПАС-3D

КОМПАС-3D — система трехмерного проектирования, ставшая стандартом для тысяч предприятий. Ключевой особенностью продукта является использование собственного математического ядра С3D и параметрических технологий, разработанных специалистами АСКОН [12].

КОМПАС-3D обеспечивает поддержку наиболее распространенных форматов 3D-моделей (STEP, ACIS, IGES, DWG, DXF), что позволяет организовывать эффективный обмен данными со смежными организациями и заказчиками, использующими любые CAD / CAM / CAE-системы в работе.

Система «Компас-3D» предназначена для создания трёхмерных ассоциативных моделей отдельных деталей и сборочных единиц, содержащих как оригинальные, так и стандартизованные конструктивные элементы.

Система «Компас-3D» предоставляет следующие возможности:

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

# 2.4.2 Плагин PDF для САПР КОМПАС-3D

Данный плагин позволяет производить экспорт моделей и сборок из КОМПАС-3D в формат PDF формат [13]. Основной особенностью является возможность интерактивного взаимодействия пользователя с сохраненной 3D сценой внутри PDF файла. Например, пользователь может вращать, масштабировать, передвигать детали и сборки внутри 3D PDF файла. Также доступно создание анимации сборки и разборки изделий. В подобных ситуациях традиционным подходом являлся экспорт сборки или детали КОМПАС-3D в промежуточный формат и дальнейшее сохранение в формат 3D PDF. Используемый подход в плагине исключает использование промежуточных файлов для осуществления 3D преобразования, что существенно повышает качество выходной 3D модели в формате PDF. Ключевые возможности:

* сохранение деталей и сборок в формате 3D PDF для интерактивного просмотра при помощи бесплатной программы Adobe Reader;
* создание анимации, имитирующих естественный порядок сборки и разборки создание имитации анимации гибки листовых тел;
* вставка в существующие PDF документы, содержащие основной текст, фоновые картинки, таблицы спецификаций, эмблемы, логотипы;
* пакетный режим для поочередной конвертации всех файлов.

# 2.4.3 Экспорт из КОМПАС-3D в формат 3D PDF

Пользователям КОМПАС-3D стала доступна функция экспорта созданных трехмерных моделей и дальнейшего их использования в создании технической документации. Экспорт происходит в формате 3D PDF [14].

Главной особенностью является то, что пользователь по-прежнему имеет возможность интерактивно взаимодействовать с 3D сценой, находясь внутри файла 3D PDF. То есть пользователь может передвигать детали, вращать их, масштабировать, передвигать сборки внутри самого файла. Пользователь также может создать анимацию сборки и разборки изделия. Этот функционал очень удобен. Он используется при создании презентаций, маркентиговых материалов, при подготовке интерактивных сборочных конструкций. Он значительно упрощает взаимодействие между заказчиками и проектировщиками.

# 3 Описание реализации

# 3.1 Описание технических и функциональных аспектов проекта

Для графического описания абстрактной модели проекта, а также пользовательского взаимодействия (сценарии действия) использован стандарт UML [15].

UML язык графического описания для объектного моделирования в обрасти 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.

При использовании UML были простроены: диаграмма использования и диаграмма классов.

# 3.2 Диаграмма вариантов использования (Use Cases)

Диаграмма вариантов использования (сценариев поведения, прецедентов) является исходным концептуальным представлением системы в процессе ее проектирования и разработки. Данная диаграмма состоит из актеров, вариантов использования и отношений между ними. При построении диаграммы могут использоваться также общие элементы нотации: примечания и механизмы расширения.

Суть данной диаграммы состоит в следующем: проектируемая система представляется в виде множества актеров, взаимодействующих с системой с помощью так называемых вариантов использования. При этом актером (действующим лицом, актантом, актором) называется любой объект, субъект или система, взаимодействующая с моделируемой системой извне. В свою очередь вариант использования – это спецификация сервисов (функций), которые система предоставляет актеру [17].

Ниже, на рисунке 3.1 представлена диаграмма вариантов использования.

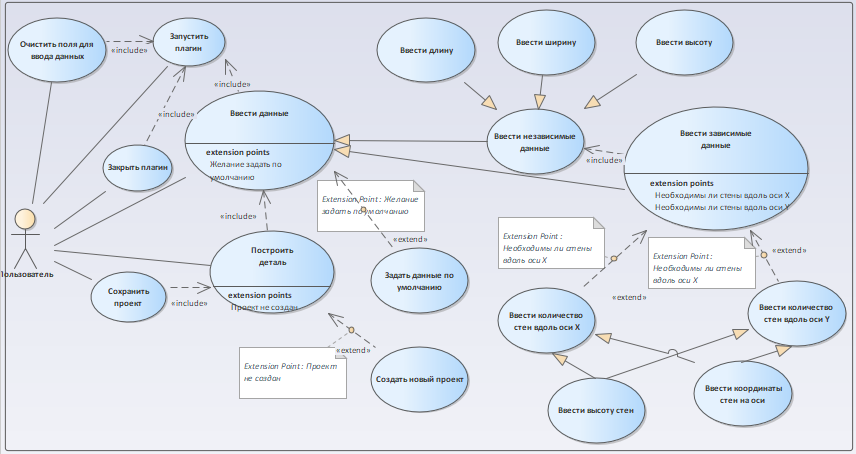


Рисунок 3.1– Диаграмма вариантов использования

# 3.3 Диаграмма классов

В данном проекте для были спроектированы следующие классы.

«Drawer» содержит в себе методы рисования 3D модели в программе «SolidWorks 2020», принимая класс «Figure», содержащий пользовательские настройки фигуры.

«FigureValidator» проверяет входные данные, введенные в графическом интерфейсе. В случае ввода неверных данных класс для проверки данных выбрасывает исключение «FigureExceptoin», содержащее в себе описание ошибки «FigureError».

Интерфейс «ISolidWorksCommander» включает команды для управления программой «SolidWorks 2020», его наследует и реализует «SolidWorksCommander».

«FigureSettings» содержит настройки фигуры, «SolidWorksSettings» настройки программы «SolidWorks 2020».

Классы «Walls» описывает высоту стен и точки вдоль вектора перечисления «Vector», «SizeRange» содержит диапазон размеров векторов.

Взаимодействие пользовательского интерфейса (пакет Drawer3D.ViewWpf) с классами модели (пакет Drawer3D.Model) происходит с помощью библиотеки MVVM Light [18], которая представляет паттерн MVVM соответственно. Класс «ViewModelLocator» хранит в себе модели представлений, «App» является классом и содержит в себе метод обработки необработанных исключений, возникших при выполнении программы.

«MainWindow» является графическим интерфейсом, содержит в себе «ApplicationVm», который имеет команды для создания нового, сохранения текущего проекта, классы «FigureVm», «PointVm» позволяют задать настройки фигуры в графическом представлении.

Ниже, на рисунке 3.2 представлена диаграмма классов модели Drawer3D.Model.

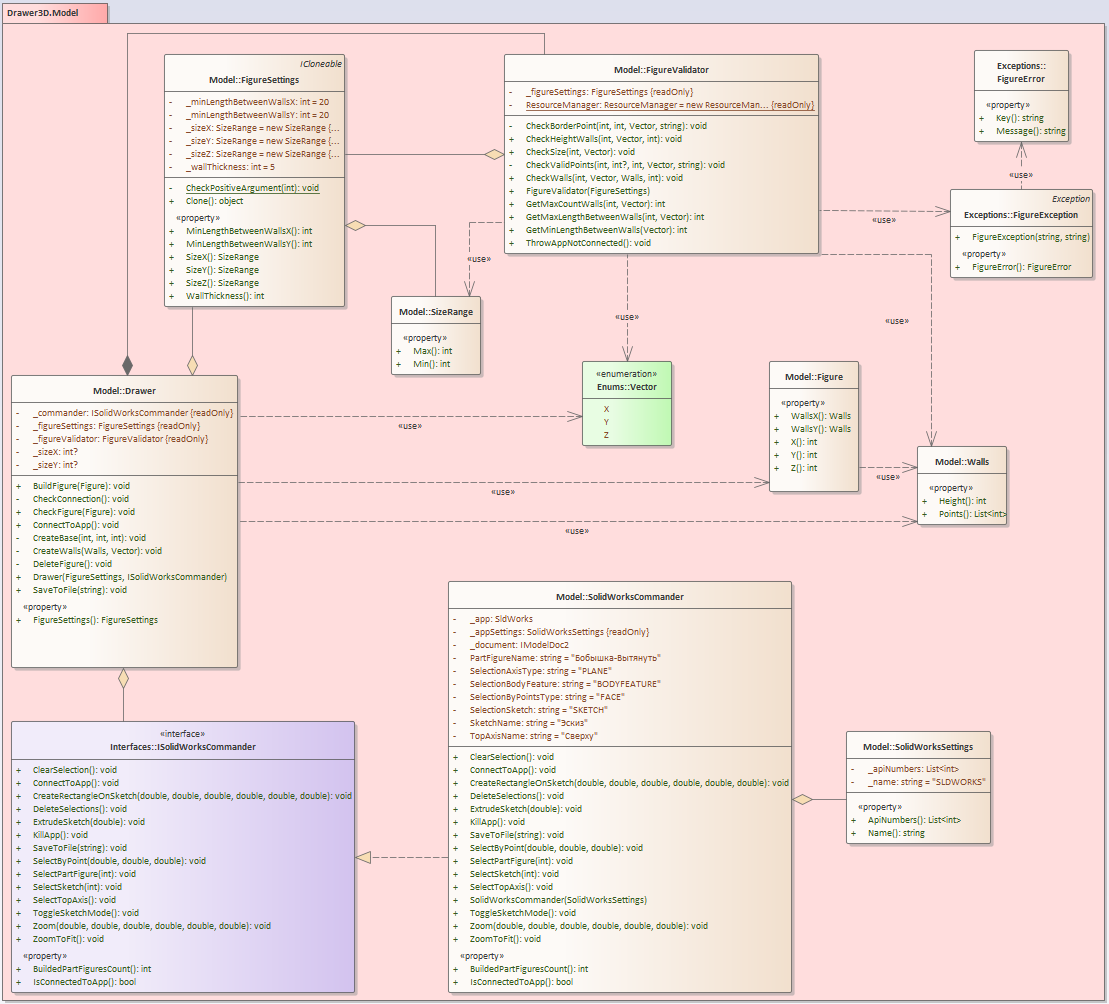
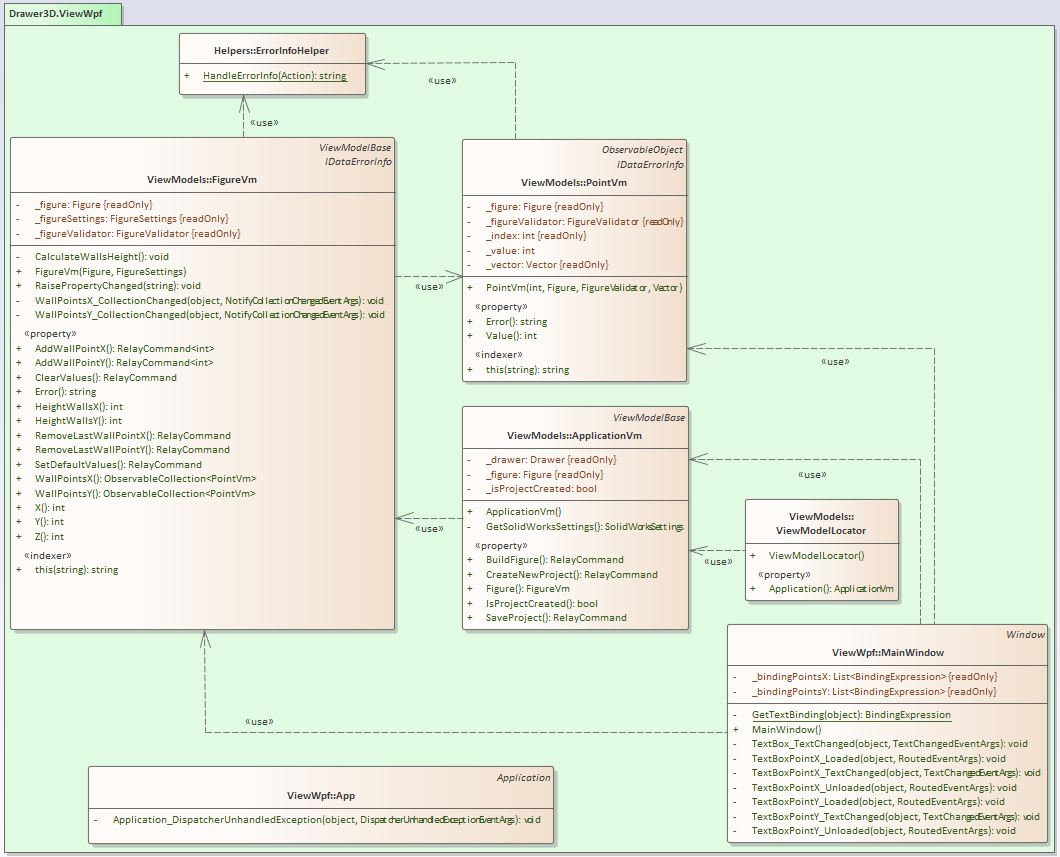
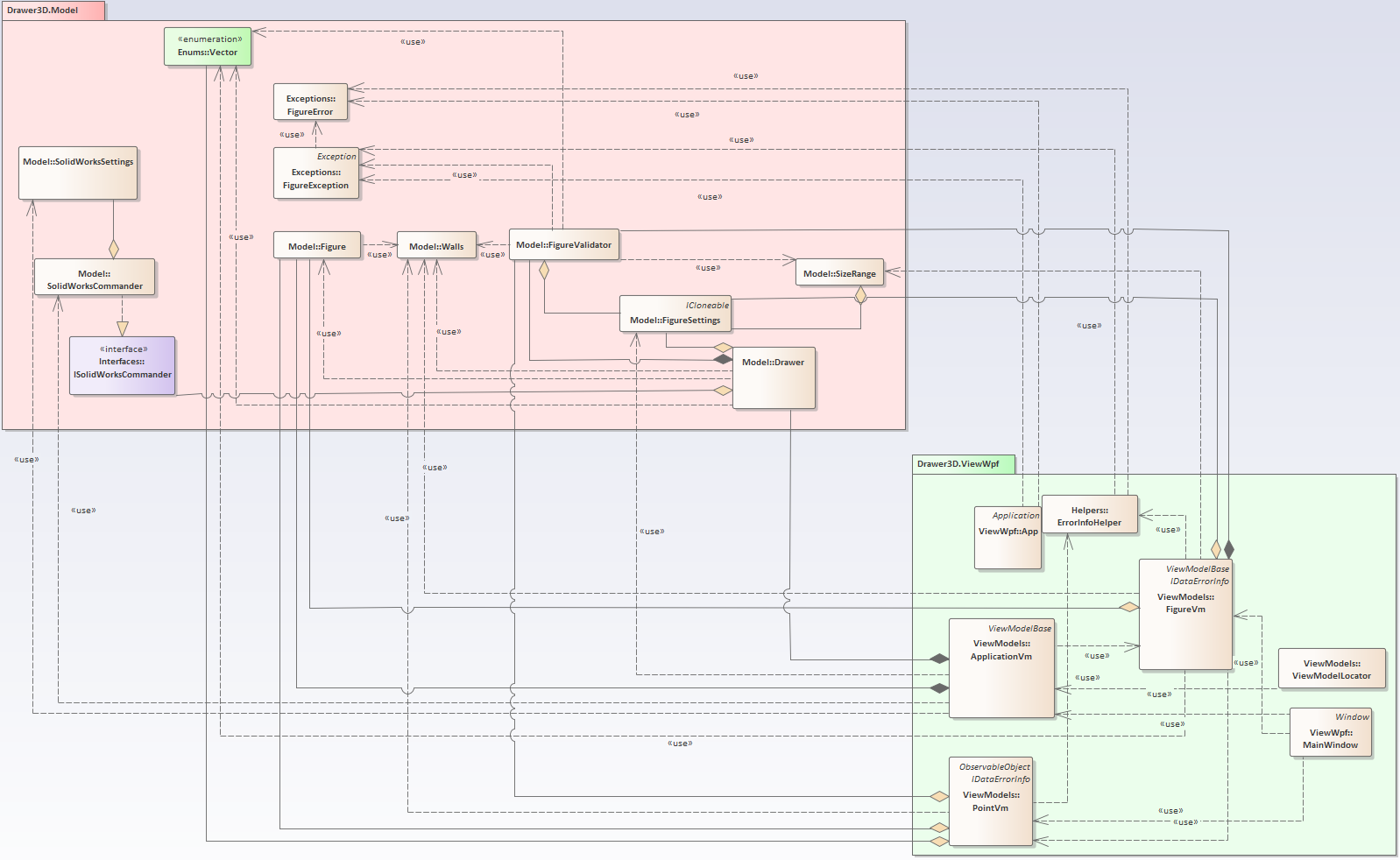


Рисунок 3.2 – Диаграмма классов модели Drawer3D.Model

Ниже, на рисунке 3.3 представлена диаграмма классов проекта, содержащего интерфейс, модель-предствления (view-models), Drawer3D.Model.

  
Рисунок 3.3 – Диаграмма классов проекта Drawer3D.ViewWpf

Взаимодействие всех классов проектов Drawer3D.ViewWpf и Drawer3D.Model представлено ниже, на рисунке 3.4.

  
Рисунок 3.4 – Общая диаграмма классов

В отличии от изначальной диаграммы классов (рисунок 3.5), были внесены следующие изменения:

* «Самописная» реализация паттерна MVVM была заменена библиотекой MVVM Light;
* Был удален класс «RelayCommand» так как в MVVM Light уже имеется класс с подобным и более богатым функционалом;
* В классе «FigureValidator» был добавлен метод «CheckHeightWall», который позволяет проверить высоту стен вдоль векторов;
* В интерфейсе «ISolidWorksCommander» (и в классе «SolidWorksCommander» соответственно) были добавлены методы «Zoom», «ZoomToFit», также в класс было добавлено строковое поле «SelectionByPointsType». Данные изменения позволяют изменить область отображения камеры в САПР;
* Добавлен «ViewModelLocator» так как удобно регистрировать, настраивать превью отображение view-model классов.
* View-model классы теперь наследуются от «ViewModelBase».

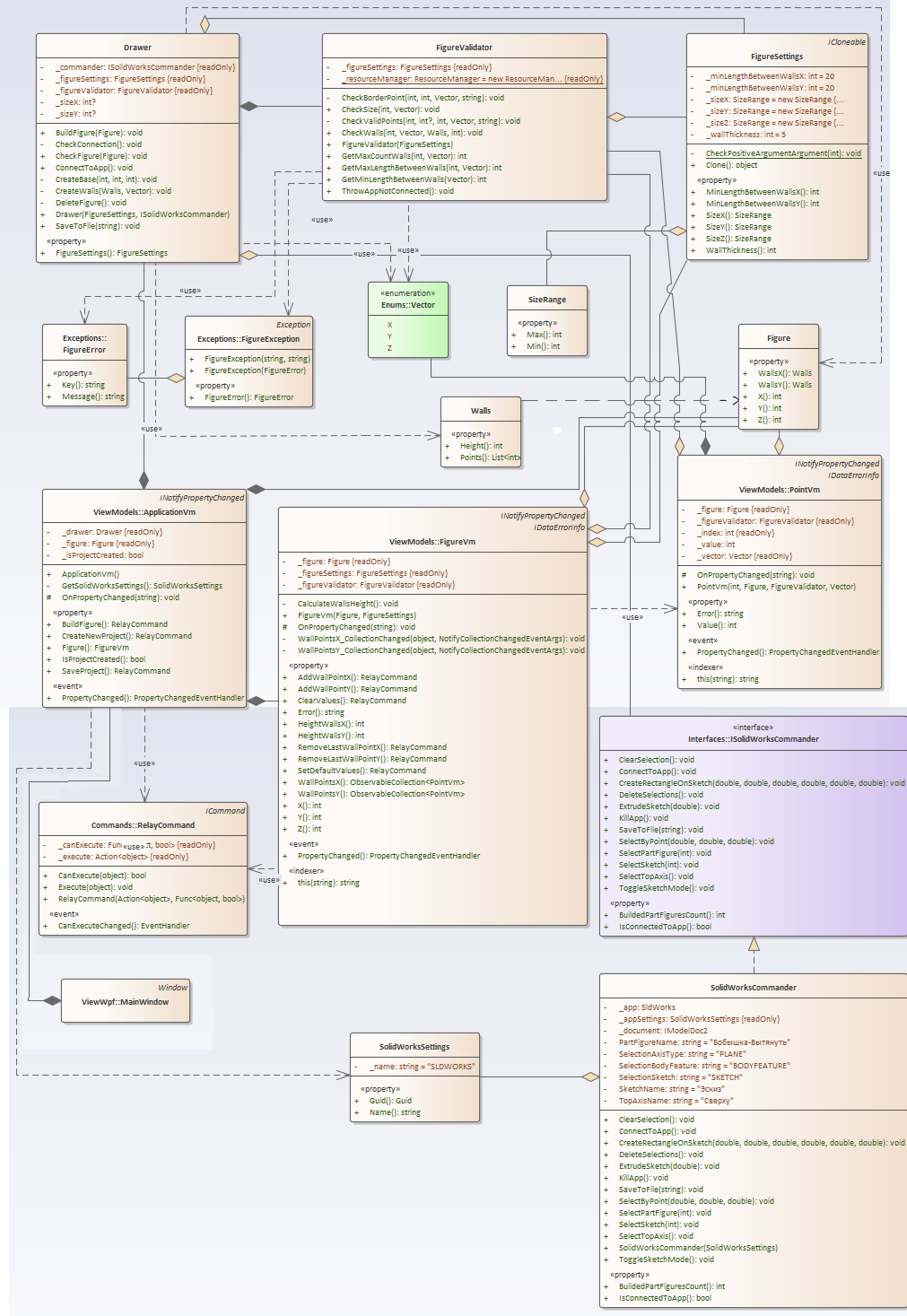
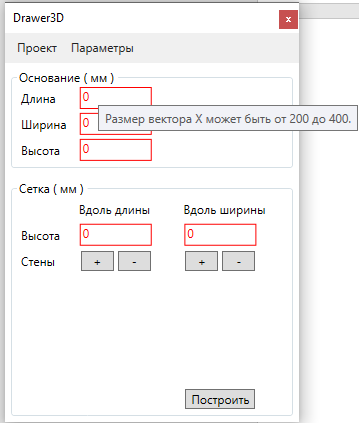


Рисунок 3.5 – Изначальная диаграмма классов

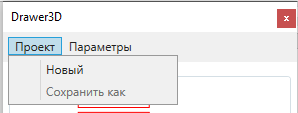
# 4 Описание программы для пользователя

Плагин состоит из диалогового окна, который имеет меню, блоки «Основание», «Сетка». При запуске имеет следующий вид (рисунок 4.1).

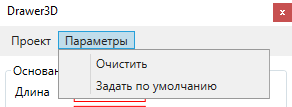
  
Рисунок 4.1 – Начальный вид диалогового окна

При старте плагина поля для ввода данных выделены красным цветом, что означает некорректность введенных данных. Под каждое поле ввода в случае ошибки (выделено красным) можно узнать причину ошибки, наведя курсор мыши на данное поле.

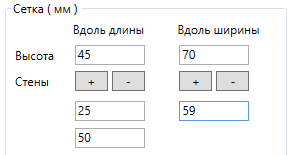
Меню программы постоит из вкладок «Проект» и «Параметры». Во вкладке «Проект» реализован функционал для создания нового и сохранения текущего проекта, уже построенного в САПР SOLIDWORKS 2020.

  
Рисунок 4.1 – Функционал вкладки «Проект»

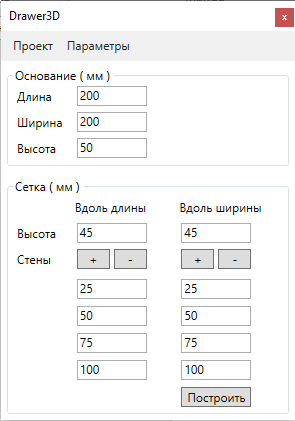
Введенные параметры в блоки «Основание», «Сетка» можно очистить или задать по умолчанию с помощью функционала, который реализован во вкладке «Параметры» (рисунок 4.2).

  
Рисунок 4.2 – Функционал вкладки «Параметры»

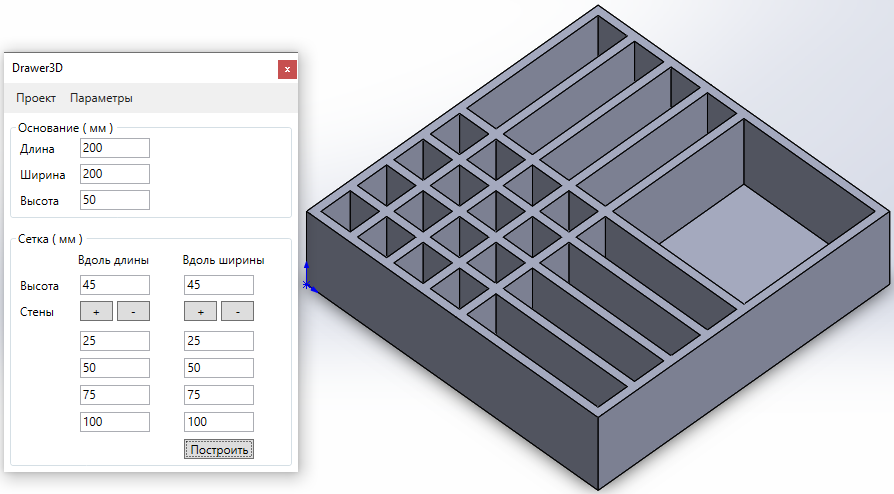
В блоке «Основание» необходимо ввести длину, ширину и высоту проектируемой модели «Форма для выпечки». В случае если необходимо добавить стены вдоль длины, глубины, то необходимо ввести их высоту, а также изменить количеств стен, нажав на кнопки «+», «–» (рисунок 4.3).

  
Рисунок 4.3 – Изменение количества стен вдоль длины и ширины

Нажав, на «Параметры – Задать по умолчанию», форма с полями для ввода данных примет следующий вид (рисунок 4.4).

  
Рисунок 4.4 – Диалоговое окно плагина

После ввода необходимых параметров, построить делать в САПР SOLIDWORKS 2020 можно с помощью кнопки «Построить», заранее убедившись, что у плагина создан проект. Например, фигура, построенная по заданным параметрам по умолчанию, выглядит следующим образом (рисунок 4.5).

  
Рисунок 4.5 – Модель, построенная по заданным параметрам по умолчанию

# 5 Тестирование программы

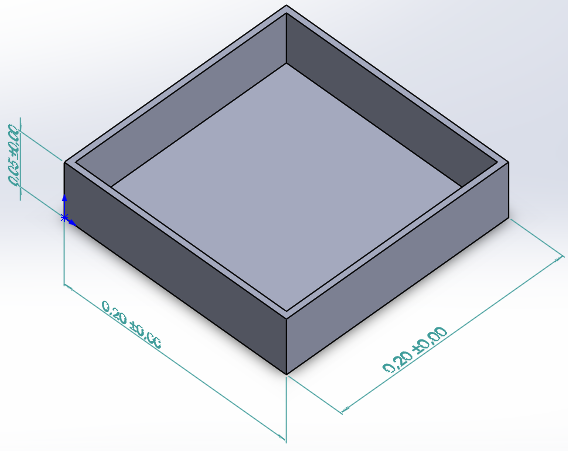
Тестирование позволяет убедиться в работоспособности программы, выявлять ошибки при изменении какого-либо функционала.

# 5.1 Функциональное тестирование

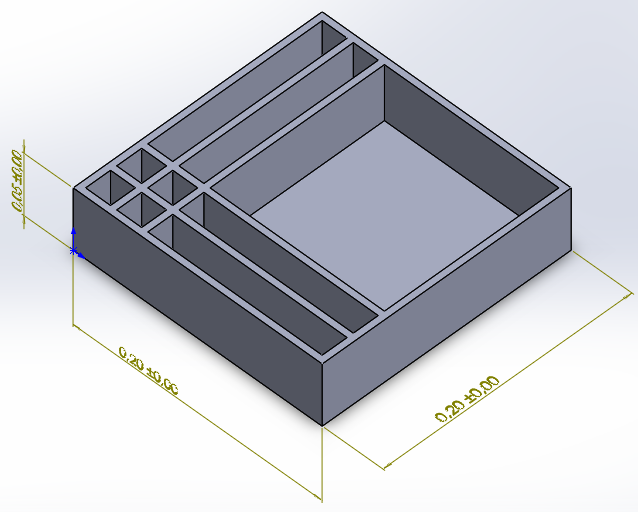
При функциональном тестировании проверялось корректность работы плагина «Форма для выпечки», а именно, соответствие полученного результата в виде трехмерной модели, с входными параметрами.

Проведено тестирование максимальных и минимальных параметров модели.

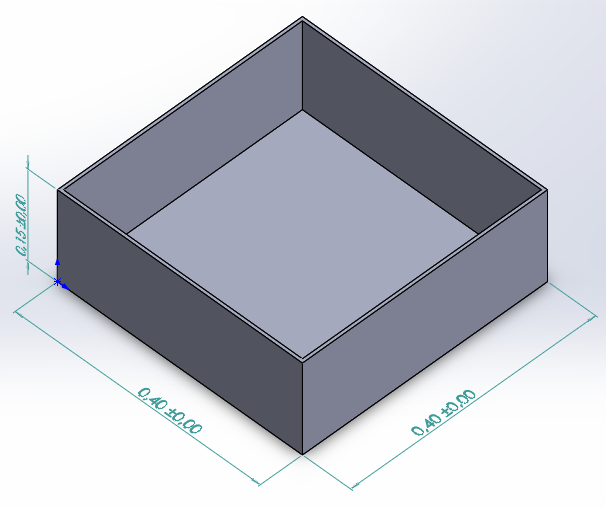
Ниже на рисунке 5.1 представлена проверка размеров модели с минимальным введенными параметрами без стен (длина 200 мм, ширина 200 мм, высота 50 мм).

  
Рисунок 5.1 – Модель с минимальными веденными параметрами без стен

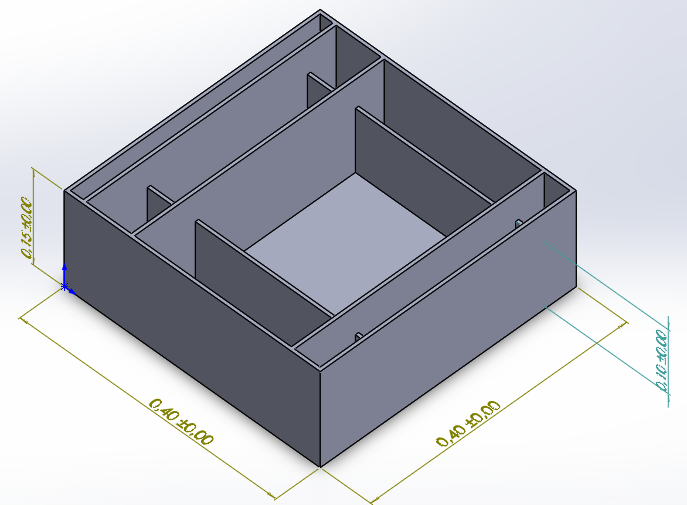
Ниже на рисунке 5.2 представлена проверка размеров модели с минимальным введенными параметрами с четырьмя стенами (длина 200 мм, ширина 200 мм, высота 50 мм, высота стен 45 мм, вдоль двух сторон стены построены по точкам 25, 50).

  
Рисунок 5.2 – Модель с минимальными веденными параметрами со стенами

Ниже на рисунке 5.3 представлена проверка размеров модели с максимальными введенными параметрами без стен (длина 400 мм, ширина 400 мм, высота 150 мм).

  
Рисунок 5.3 – Модель с минимальными веденными параметрами без стен

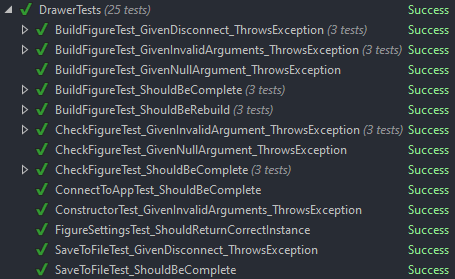
Ниже на рисунке 5.4 представлена проверка размеров модели с максимальными введенными параметрами с шестью стенами (длина 400 мм, ширина 400 мм, высота 150, высота стен вдоль длины 150 мм, высота стен вдоль ширины 100 мм, вдоль двух сторон стены построены по точкам 25, 100, 300).

  
Рисунок 5.4 – Модель с максимальными веденными параметрами со стенами

# 5.2 Модульное тестирование

В целях проверки корректности работы методов и свойств классов при помощи обозревателя тестов расширения для Visual Studio – ReSharper с тестовым фреймворком NUnit версии 3.12 проведено модульное тестирование [11], проверялись открытые поля и методы.

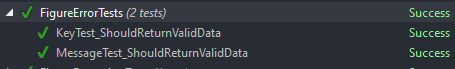
Ниже, на рисунке 5.5 представлен результат тестирования класса Drawer с помощью вспомогательного класса DrawerTests (описание полей и методов находится в приложении А.1).

  
Рисунок 5.5 – Результат тестирования класса Drawer

Ниже, на рисунке 5.6 представлен результат тестирования класса EnumerableExtensions с помощью вспомогательного класса EnumerableExtensionsTests (описание полей и методов находится в приложении А.2).

  
Рисунок 5.6 – Результат тестирования класса EnumerableExtensions

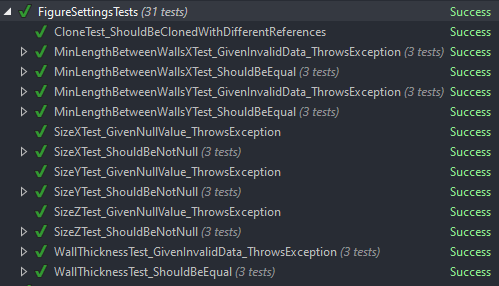
Ниже, на рисунке 5.7 представлен результат тестирования класса FigureError с помощью вспомогательного класса FigureErrorTests (описание полей и методов находится в приложении А.3).

  
Рисунок 5.7 – Результат тестирования класса FigureError

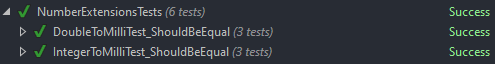
Ниже, на рисунке 5.8 представлен результат тестирования класса FigureException с помощью вспомогательного класса FigureExceptionTests (описание полей и методов находится в приложении А.4).

  
Рисунок 5.8 – Результат тестирования класса FigureException

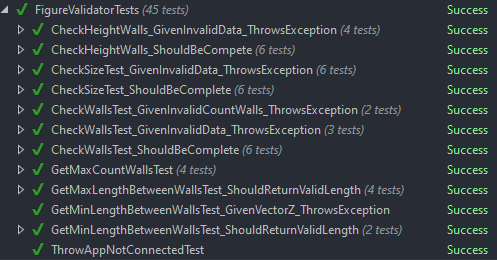
Ниже, на рисунке 5.9 представлен результат тестирования класса FigureSettings с помощью вспомогательного класса FigureSettingsTests (описание полей и методов находится в приложении А.5).

  
Рисунок 5.9 – Результат тестирования класса FigureSettings

Ниже, на рисунке 5.10 представлен результат тестирования класса NumberExtensions с помощью вспомогательного класса NumberExtensionsTests (описание полей и методов находится в приложении А.6).

  
Рисунок 5.10 – Результат тестирования класса NumberExtensions

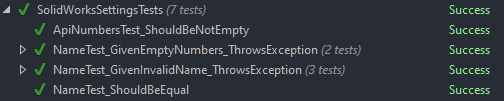
Ниже, на рисунке 5.11 представлен результат тестирования класса FigureValidator с помощью вспомогательного класса FigureValidatorTests (описание полей и методов находится в приложении А.7).

  
Рисунок 5.11 – Результат тестирования класса FigureValidator

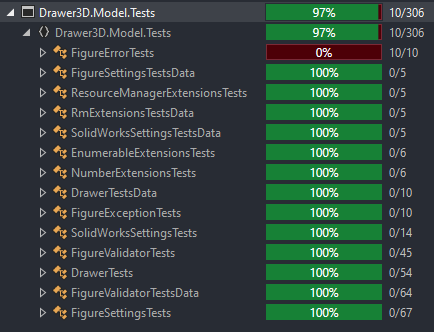
Ниже, на рисунке 5.12 представлен результат тестирования класса ResourceManagerExtensions с помощью вспомогательного класса ResourceManagerExtensionsTests (описание полей и методов находится в приложении А.8).

  
Рисунок 5.12 – Результат тестирования класса ResourceManagerExtensions

Ниже, на рисунке 5.12 представлен результат тестирования класса SolidWorksSettings с помощью вспомогательного класса SolidWorksSettingsTests (описание полей и методов находится в приложении А.9).

  
Рисунок 5.12 – Результат тестирования класса SolidWorksSettings

Ниже, на рисунке 5.13 представлено процентное соотношение покрытия библиотеки Drawer3D.Model тестами, оно определено с помощью расширения для Visual Studio – dotCover.

  
Рисунок 5.13 – Процентное соотношение покрытия библиотеки Drawer3D.Model тестами

# 5.3 Нагрузочное тестирование

В целях проверки производительности работы плагина, было проведено нагрузочное тестирование [16]. Тестирование производилось на ПК со следующей конфигурацией:

Процессор: Intel Core i7-4710HQ

Графическая система: Intel HD Graphics 4600

ОЗУ: 16 GB DDR 1333 MHz

Стоит отметить, что графическая система данного ПК является устарелой, маломощной для моделирования, работы в САПР.

На рисунке 5.14 для проведения нагрузочного тестирования был добавлен секундомер («Stopwatch»), который засекал время от начала построения, с каждым успешным построением-перестроением фигуры («\_drawer.BuildFigure(\_figure)») производилась запись результатов в текстовый файл «log.txt».

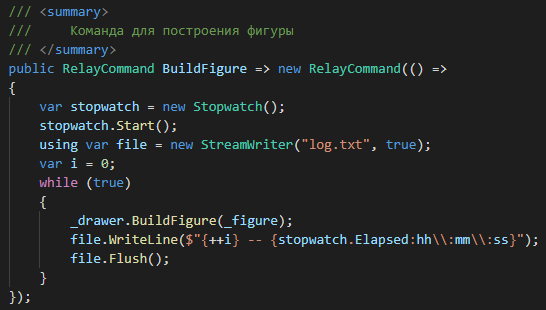


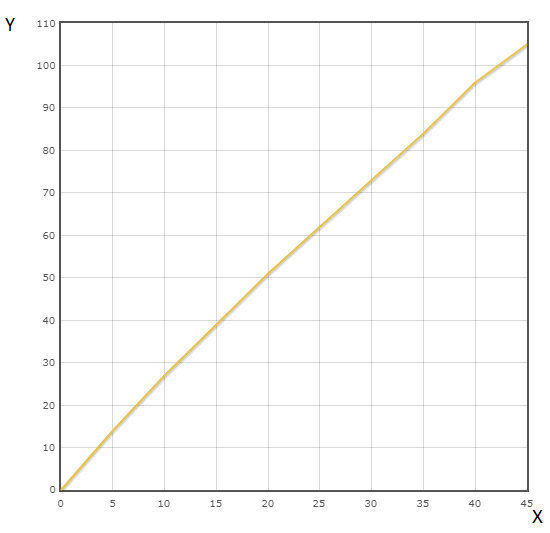
Рисунок 5.14 – Зацикливание перестроения фигуры

На представленных графиках в текущей главе ось «X» – время в минутах, ось «Y» – количество построенных-перестроенных деталей. На протяжении всех тестов (продолжительностью в 45 минут каждый) общая загруженность процессора была в пределах 20 процентов, потребление ОЗУ плагином прямолинейное от 17мб до 29мб (от 0 до 45 минут соответственно).

На рисунке 5.15 представлено тестирование зацикленного перестроения фигуры со следующими максимальными параметрами:

* Длина 400мм;
* Ширина 400мм;
* Высота 150мм;
* Стены вдоль длины по точкам (25, 50, 75, 100, 125, 150, 175, 200, 225, 250, 275, 300, 325, 350), высота стен 145мм;
* Стены вдоль ширины по точкам (25, 50, 75, 100, 125, 150, 175, 200, 225, 250, 275, 300, 325, 350), высота стен 145мм.

Заметно, что на графике (рисунок 5.15) на 40 минуте производительность слегка упала.

  
Рисунок 5.15 – График зависимости времени от количества построенных фигур с максимальными параметрами

На рисунке 5.16 представлено тестирование зацикленного перестроения фигуры со следующими минимальными параметрами:

* Длина 200мм;
* Ширина 200мм;
* Высота 50мм;
* Отсутствуют стены вдоль длины и ширины.

На графике (рисунок 5.16) не заметны падения производительности, зависимость количества построенных фигур от времени прямолинейное.

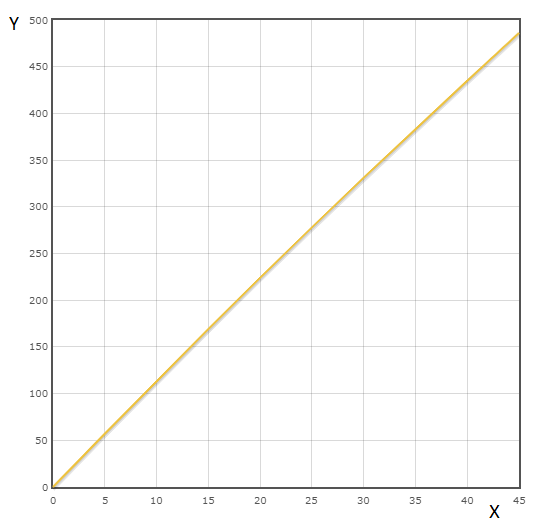


Рисунок 5.16 – График зависимости времени от количества построенных фигур с минимальными параметрами

На протяжении 45 минут работы плагина с САПР ошибок не обнаружено, программы не сломались.

# Заключение

В ходе выполнения лабораторных работ были изучены предметная область проектирования, предмет проектирования, аналоги предмета проектирования, API и на основании полученных данных были спроектированы UML диаграммы классов, разработан плагин для создания 3D моделей «Форма для выпечки» в САПР SOLIDWORKS 2020.

# Список использованных источников

1. SOLIDWORKS 2020 [Электронный ресурс]. – URL: <https://www.solidworks.com/> (дата обращения: 20.04.2020);
2. Visual Studio [Электронный ресурс]. – URL: <https://visualstudio.microsoft.com/ru/> (дата обращения: 20.04.2020);
3. Что такое API? Хабр [Электронный ресурс]. – URL: <https://habr.com/ru/post/464261/> (дата обращения: 20.04.2020);
4. Microsoft Word 2016 [Электронный ресурс]. – URL: <https://officeproduct.info/> (дата обращения: 20.04.2020);
5. Автоматизация вычислительных процедур в прикладных задачах инженерного проектирования [Электронный ресурс]. – URL: https://scienceforum.ru/2014/article/2014000201 (дата обращения: 20.04.2020);
6. Форма для выпечки [Электронный ресурс]. – URL: <https://dombutik.ru/utvar-dlya-gotovki/konditerskii-tsekh.html/nid/69986> (дата обращения: 20.04.2020);
7. SolidWorks.Interop.sldworks Namespace. SOLIDWORKS API Help. [Электронный ресурс]. – URL: https://help.solidworks .com/2019/English/api/sldworksapi/SolidWorks.Interop.sldworks~SolidWorks.Interop.sldworks\_namespace.html (дата обращения: 20.04.2020);
8. ReSharper. [Электронный ресурс]. – URL: <https://www.jetbrains.com/ru-ru/resharper/> (дата обращения: 20.04.2020);
9. NUnit [Электронный ресурс]. – URL: <https://nunit.org/> (дата обращения: 20.04.2020);
10. dotCover. Инструмент для запуска юнит-тестов .NET и оценки покрытия кода [Электронный ресурс]. – URL: <https://www.jetbrains.com/ru-ru/dotcover/> (дата обращения: 20.04.2020);
11. Начало работы с WPF [Электронный ресурс]. – URL: <https://docs.microsoft.com/ru-ru/visualstudio/designers/getting-started-with-wpf?view=vs-2019> (дата обращения: 20.04.2020);
12. КОМПАС-3D: О программе. Официальный сайт САПР КОМПАС [Электронный ресурс]. – URL: <http://kompas.ru/kompas-3d/about/> (дата обращения: 20.04.2020);
13. Плагин PDF [Электронный ресурс]. – URL: <http://gkmsoft.ru/allcatalog/pdf2dkompas_plugin/> (дата обращения: 20.04.2020);
14. 3D PDF [Электронный ресурс]. – URL: <http://sapr-journal.ru/novosti/eksport-iz-kompas-3d-v-formate-3d-pdf/> (дата обращения: 20.04.2020);
15. UML [Электронный ресурс]. – URL: <http://www.uml.org/> (дата обращения: 13.04.2020);
16. Нагрузочное тестирование: с чего начать и куда смотреть [Электронный ресурс]. – URL: <https://habr.com/ru/company/jugru/blog/329174/> (дата обращения: 20.04.2020);
17. Назначение и состав диаграммы вариантов использования [Электронный ресурс]. – URL: [https://sites.google.com/site/anisimovkhv /learning/pris/lecture/tema12/tema12\_2](https://sites.google.com/site/anisimovkhv%20/learning/pris/lecture/tema12/tema12_2) (дата обращения: 28.04.2020);
18. MVVM Light Toolkit [Электронный ресурс]. – URL: <http://www.mvvmlight.net/> (дата обращения: 28.04.2020).

# Приложение А

(Справочное)

Все таблицы, представленные ниже, являются описанием вспомогательных классов для проведения модульного тестирования. Каждый метод в классе является тестовым случаем. В каждой таблице описаны поля, методы класса и их описание.

Таблица А.1 – Класс DrawerTests

|  |  |
| --- | --- |
| Наименование | Описание |
| \_commander: ISolidWorksCommander | API Команды к программе SOLIDWORKS |
| \_drawer: Drawer | Построитель-рисовальщик фигуры |
| InitializeEachTest() | Инициализировать каждый тест |
| ConstructorTest\_GivenInvalidArguments  \_ThrowsException() | Негативный тест конструктора |
| FigureSettingsTest  \_ShouldReturnCorrectInstance() | Проверка свойства FigureSettings |
| CheckFigureTest\_ CheckFigureTest\_  ShouldBeComplete(Figure figure) | Проверка метода CheckFigure |
| CheckFigureTest\_GivenNullArgument  \_ThrowsException() | Негативная проверка метода CheckFigure (null аргумент) |
| CheckFigureTest\_GivenInvalidArgument  \_ThrowsException(Figure figure) | Негативная проверка метода CheckFigure (недопустимый аргумент) |
| BuildFigureTest\_ShouldBeComplete  (Figure figure) | Проверка метода BuildFigure |
| BuildFigureTest\_ShouldBeRebuild  (Figure figure) | Проверка метода BuildFigure (должен перестроить) |
| BuildFigureTest\_GivenNullArgument  \_ThrowsException() | Негативная проверка метода BuildFigure (null аргумент) |

Продолжение таблицы А.1

|  |  |
| --- | --- |
| Наименование | Описание |
| BuildFigureTest\_GivenDisconnect  \_ThrowsException(Figure figure) | Негативная проверка метода BuildFigure (нет подключения к САПР) |
| BuildFigureTest\_GivenInvalidArguments  \_ThrowsException(Figure figure) | Негативная проверка метода BuildFigure (недопустимый аргумент) |
| ConnectToAppTest\_ShouldBeComplete() | Проверка метода ConnectToApp |

Таблица А.2 – Класс EnumerableExtensionsTests

|  |  |
| --- | --- |
| Наименование | Описание |
| IsNullOrEmptyTest\_  ShouldReturnValidResponse() | Проверка расширения IsNullOrEmpty (должен вернуть валидный ответ) |

Таблица А.3 – Класс FigureErrorTests

|  |  |
| --- | --- |
| Наименование | Описание |
| KeyTest\_ShouldReturnValidData() | Проверка свойства Key |
| MessageTest\_ShouldReturnValidData() | Проверка свойства Message |

Таблица А.4 – Класс FigureExceptionTests

|  |  |
| --- | --- |
| Наименование | Описание |
| FigureErrorTest\_  PropertiesShouldReturnValidData() | Проверка всех свойств (key, message) |

Таблица А.5 – Класс FigureSettingsTests

|  |  |
| --- | --- |
| Наименование | Описание |
| \_figureSettings: FigureSettings | Настройки фигуры |

Продолжение таблицы А.5

|  |  |
| --- | --- |
| Наименование | Описание |
| SizeXTest\_ShouldBeNotNull  (SizeRange sizeRange) | Проверка свойства SizeX |
| SizeXTest\_GivenNullValue  \_ThrowsException() | Негативная проверка свойства SizeX |
| SizeYTest\_ShouldBeNotNull  (SizeRange sizeRange) | Проверка свойства SizeY |
| SizeYTest\_GivenNullValue  \_ThrowsException() | Негативная проверка свойства SizeY |
| SizeZTest\_ShouldBeNotNull  (SizeRange sizeRange) | Проверка свойства SizeZ |
| SizeZTest\_GivenNullValue  \_ThrowsException() | Негативная проверка свойства SizeZ |
| WallThicknessTest\_ShouldBeEqual  (int wallThickness) | Проверка свойства WallThickness |
| WallThicknessTest\_GivenInvalidData  \_ThrowsException(int wallThickness) | Негативная проверка свойства WallThickness |
| MinLengthBetweenWallsXTest  \_ShouldBeEqual(int length) | Проверка свойства MinLengthBetweenWallsX |
| MinLengthBetweenWallsXTest  \_GivenInvalidData\_ThrowsException(int length) | Негативная проверка свойства MinLengthBetweenWallsX |
| MinLengthBetweenWallsYTest  \_ShouldBeEqual(int length) | Проверка свойства MinLengthBetweenWallsY |
| MinLengthBetweenWallsYTest  \_GivenInvalidData\_ThrowsException(int length) | Негативная проверка свойства MinLengthBetweenWallsY |
| InitializeEachTest() | Инициал-ть перед каждым тестом |

Продолжение таблицы А.5

|  |  |
| --- | --- |
| Наименование | Описание |
| CloneTest\_ShouldBeCloned  WithDifferentReferences() | Проверка клонирования FigureSettings |

Таблица А.6 – NumberExtensionsTests

|  |  |
| --- | --- |
| Наименование | Описание |
| IntegerToMilliTest\_ShouldBeEqual  (double expected, int actual) | Проверка расширения ToMilli (int) |
| DoubleToMilliTest\_ShouldBeEqual  (double expected, double actual) | Проверка расширения ToMilli (double) |

Таблица А.7 – FigureValidatorTests

|  |  |
| --- | --- |
| Наименование | Описание |
| \_figureValidator: FigureValidator | Валидатор фигуры |
| InitializeEachTest() | Инициализировать перед каждым тестом |
| ThrowAppNotConnectedTest() | Метод ThrowAppNotConnected должен выбросить исключение FigureException |
| GetMinLengthBetweenWallsTest  \_ShouldReturnValidLength(Vector vector) | Проверка метода GetMinLengthBetweenWalls |
| GetMinLengthBetweenWallsTes  t\_GivenVectorZ\_ThrowsException() | Негативная проверка метода GetMinLengthBetweenWalls (передан параметр Z) |
| CheckSizeTest\_ShouldBeComplete  (int size, Vector vector) | Проверка метода CheckSize |

Продолжение таблицы А.7

|  |  |
| --- | --- |
| Наименование | Описание |
| CheckSizeTest\_GivenInvalidData  \_ThrowsException  (int size, Vector vector) | Негативная проверка метода CheckSize (недопустимые параметры) |
| GetMaxLengthBetweenWallsTest  \_ShouldReturnValidLength  (int size, Vector vector) | Проверка метода GetMaxLengthBetweenWalls |
| GetMaxCountWallsTest  (int size, Vector vector) | Проверка метода GetMaxCountWalls |
| CheckWallsTest\_ShouldBeComplete  (int size, Vector vector,  Walls walls, int sizeVectorZ) | Проверка метода CheckWalls |
| CheckWallsTest\_GivenInvalidData  \_ThrowsException  (int size, Vector vector, Walls walls, int sizeVectorZ) | Негативная проверка метода CheckWalls |
| CheckWallsTest\_GivenInvalidCountWalls  \_ThrowsException(int size, Vector vector, Walls walls, int sizeVectorZ) | Негативная проверка метода CheckWalls (недопустимое значение количества стен) |
| CheckHeightWalls\_ShouldBeCompete(int height, Vector vector, int sizeVectorZ) | Проверка метода CheckHeightWalls |
| CheckHeightWalls\_GivenInvalidData  \_ThrowsException  (int height, Vector vector,int sizeVectorZ) | Негативная проверка метода CheckHeightWalls (недопустимый аргумент) |

Таблица А.8 – ResourceManagerExtensionsTests

|  |  |
| --- | --- |
| Наименование | Описание |
| GetFormattedStringTest\_GivenInvalidKey  \_ThrowsException(string key) | Негативная проверка расширения GetFormattedString (передан недопустимый ключ |

Таблица А.9 – SolidWorksSettingsTests

|  |  |
| --- | --- |
| Наименование | Описание |
| NameTest\_ShouldBeEqual() | Проверка свойства Name |
| NameTest\_GivenInvalidName  \_ThrowsException(string name) | Негативная проверка свойства Name (передан недопустимый агрумент) |
| ApiNumbersTest\_ShouldBeNotEmpty() | Проверка свойства ApiNumbers (должен быть не пустой) |
| NameTest\_GivenEmptyNumbers  \_ThrowsException(List<int> numbers) | Негативная проверка свойства ApiNumbers |