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)

БИБЛИОТЕКА ЗВЁЗДНОГО ИСТРЕБИТЕЛЯ T-65 «X-WING» ДЛЯ СИСТЕМЫ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ

КОМПАС-3D

Пояснительная записка по дисциплине

«Основы разработки САПР»

Студент гр. 589-1

\_\_\_\_\_\_\_\_\_ Ю. Н. Кобзарь

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

Руководитель

преподаватель каф., к.т.н.

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

(оценка) \_\_\_\_\_\_\_\_\_

Томск 2022 г

**Реферат**

Учебная работа 36 страниц, 34 рисунка, 11 источников.

Ключевые слова: КОМПАС-3D, ЗВЁЗДНЫЙ ИСТРЕБЛИТЕЛЬ, X-WING, СИСТЕМА АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ, БИБЛИОТЕКА.

Целью данной работы является разработка библиотеки звёздного истребителя T-65 «X-Wing» 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 2016.

**Оглавление**

[Введение 4](#_Toc118097735)

[1 ПОСТАНОВКА И АНАЛИЗ ЗАДАЧИ 5](#_Toc118097736)

[1.1 Описание предмета проектирования 5](#_Toc118097737)

[1.2 Описание инструментов и средств реализации 7](#_Toc118097738)

[1.3 Назначение библиотеки 8](#_Toc118097739)

[2 ОБЗОР АНАЛОГОВ 9](#_Toc118097740)

[2.1 Плагин «Airplane Design Workbench» для FreeCAD 9](#_Toc118097741)

[2.2 Плагин «Rocket Workbench» для FreeCAD 10](#_Toc118097742)

[3 ОПИСАНИЕ РЕАЛИЗАЦИИ 11](#_Toc118097743)

[4 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЯ 22](#_Toc118097744)

[5 ТЕСТИРОВАНИЕ ПРОГРАММЫ 25](#_Toc118097745)

[5.1 Функциональное тестирование 25](#_Toc118097746)

[5.2 Модульное тестирование 30](#_Toc118097747)

[5.3 Нагрузочное тестирование 33](#_Toc118097748)

[Заключение 35](#_Toc118097749)

[Список использованных источников 36](#_Toc118097750)

# 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 библиотеки звёздного истребителя T-65 «X-Wing» 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 библиотеку в соответствии с техническим заданием. На основе заданных параметров библиотека, взаимодействуя с САПР «Компас-3D», должна строить трёхмерную модель звёздного истребителя T-65 «X-Wing» [2]. Также библиотека должна позволять изменять входные параметры звёздного истребителя.

Изменяемые параметры:

* длина корпуса звездолёта;
* ширина крыльев звездолёта;
* длина носовой части корпуса звездолёта;
* длина острия оружейного бластера звездолёта;
* длина турбины ускорителя звездолёта;
* длина сопла ускорителя звездолёта;
* высота установок крыши корпуса.

# 1.1 Описание предмета проектирования

Выбранным предметом для проектирования является звёздный истребитель T-65 «X-Wing» из вселенной Star Wars. Оригинал разрабатываемого объекта представлен на рисунке 1.1.



Рисунок 1.1 – Звёздный истребитель T-65 «X-Wing»

На рисунках 1.2-1.3 представлены 2D-чертежи трёхмерной модели звёздного истребителя T-65 «X-Wing» в видах снизу и спереди.

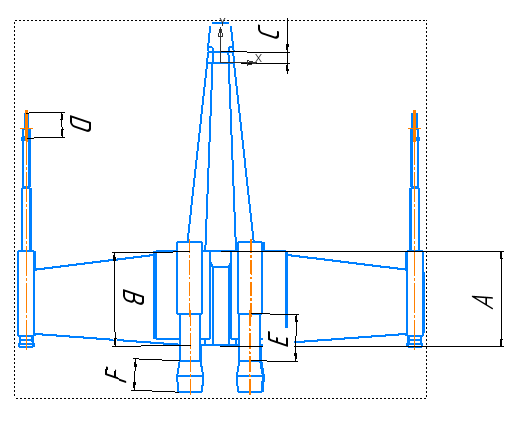


Рисунок 1.2 – Чертёж звёздного истребителя T-65 «X-Wing» в виде снизу

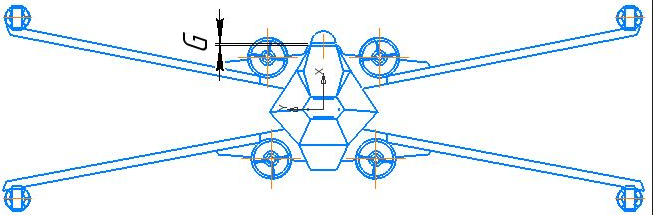


Рисунок 1.3 – Чертёж звёздного истребителя T-65 «X-Wing» в виде спереди

Параметры звёздного истребителя:

* длина корпуса звездолёта: от 300 до 400 мм; параметр представлен на чертеже как A;
* ширина крыльев звездолёта: от 300 до 400 мм; параметр представлен на чертеже как B;
* длина носовой части корпуса звездолёта: от 50 до 100 мм; параметр представлен на чертеже как C;
* длина острия оружейного бластера звездолёта: от 80 до 130 мм; параметр представлен на чертеже как D;
* длина турбины ускорителя звездолёта: от 150 до 250 мм; параметр представлен на чертеже как E;
* длина сопла ускорителя звездолёта: от 50 до 100 мм; параметр представлен на чертеже как F;
* высота установок крыши корпуса звездолёта: от 10 до 20 мм; параметр представлен на чертеже как G.

**Взаимосвязь параметров:**

* ширина крыльев B должна быть не меньше длины корпуса A более, чем на 20 мм, и не больше, чем сама длина корпуса A;
* длина турбины ускорителя E не должна быть более, чем в 4 раза больше, чем длина сопла F;
* длина острия D не должна быть более, чем в 2 раза больше, чем длина носовой части C;
* длина носовой части C должна быть не больше, чем длина острия D.

# 1.2 Описание инструментов и средств реализации

Для создания библиотеки используется среда разработки Visual Studio 2019. Библиотека написана с пользовательским интерфейсом на WindowsForms с использованием .NET Framework 4.7.2. Инструментом для тестирования является библиотеки NUnit 3.13.3 и NUnit3TestAdapter 4.2.1. В качестве системы автоматизированного проектирования выбран «Компас-3D» v.20.

# 1.3 Назначение библиотеки

Программа предназначена для автоматизации моделирования детали «Звёздный истребитель T-65 “X-Wing”».

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 всплывающих подсказок.

При запуске моделирования с некорректными значениями программа выводит сообщение об ошибке и отменяет построение модели.

При правильно введенных значениях результатом работы программы будет созданная по ним модель звёздного истребителя T-65 «X-Wing». Размеры всех параметров, необходимых при построении, рассчитываются автоматически.

# 2 ОБЗОР АНАЛОГОВ

# 2.1 Плагин «Airplane Design Workbench» для FreeCAD

Данный экспериментальный плагин предназначен для проектирования крыльев и объектов самолетов [3][4]. На рисунке 2.1 представлен ввод параметров проектируемого крыла, на рисунке 2.2 представлено крыло, построенное с помощью данного плагина.

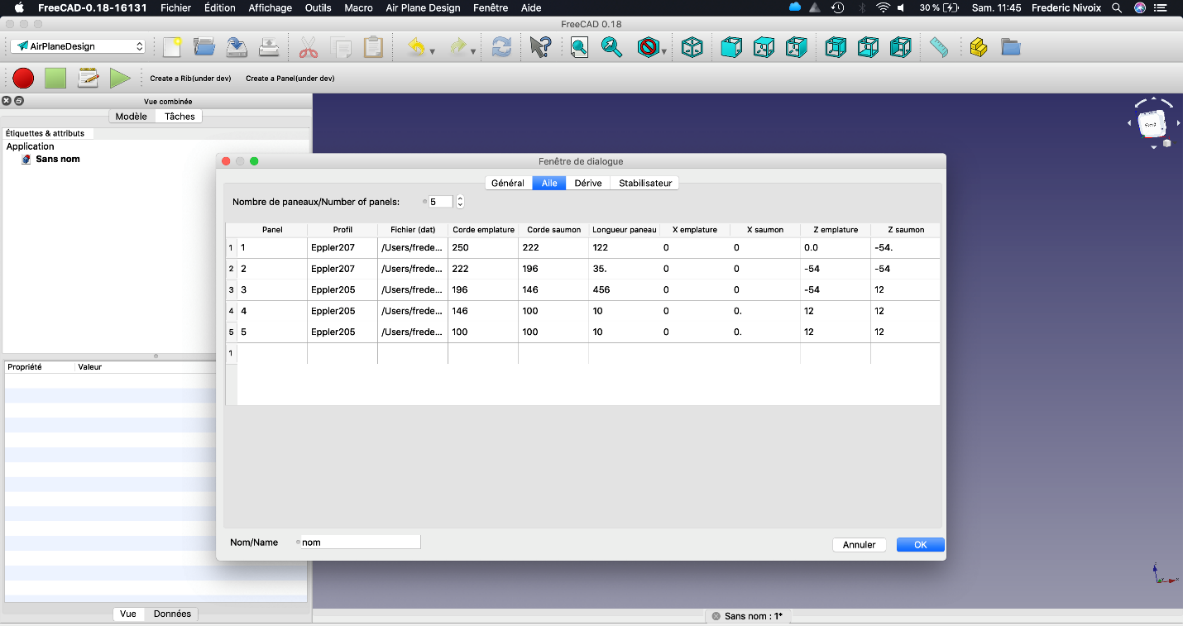


Рисунок 2.1 – Ввод параметров крыла через пользовательский интерфейс

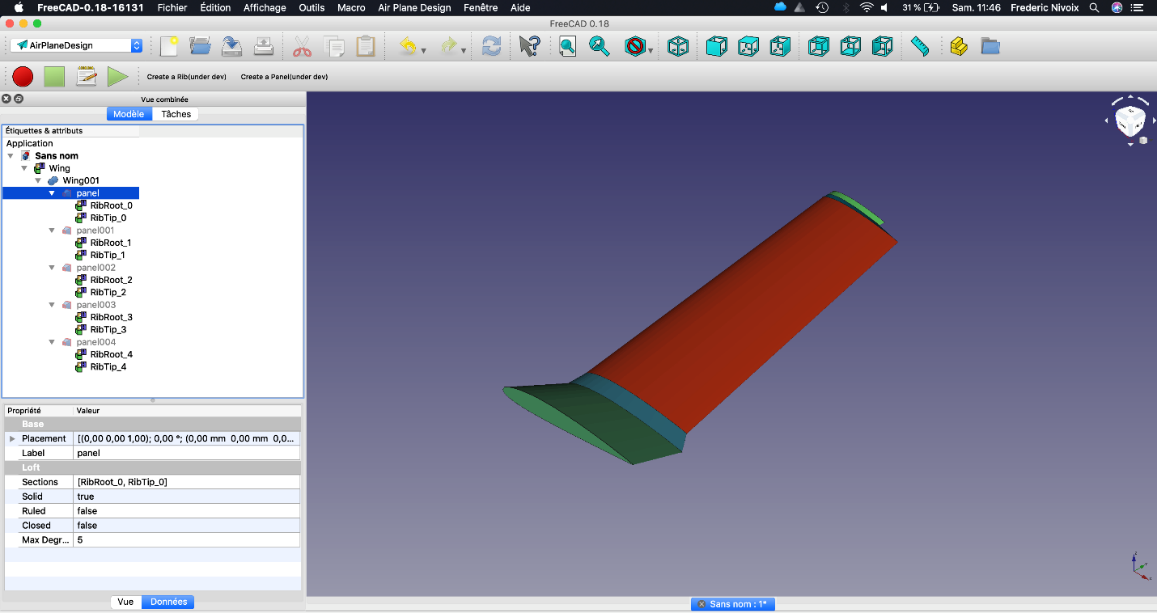


Рисунок 2.2 – Построенное крыло с помощью плагина во FreeCAD

# 2.2 Плагин «Rocket Workbench» для FreeCAD

Rocket Workbench – это набор инструментов для проектирования ракет и их компонентов. Целевая аудитория – производители моделей и любительских ракет всех типов и размеров. В настоящее время инструмент используется в основном для создания отдельных компонентов, а не полных ракет. Детали создаются путем указания параметров в диалоге задачи [3][5].

Преимущество использования Rocket Workbench заключается в том, что детали можно создавать с гораздо меньшим количеством операций [5]:

* носовой обтекатель – создание носового обтекателя;
* переход – создание перехода;
* трубка корпуса – создание полую трубку корпуса;
* центрирующее кольцо – создание центрирующего кольца;
* переборка – создание перегородки;
* плавник – создание плавника;
* руководство по запуску – создание руководства по запуску.

На рисунке 2.3 показаны примеры объектов, которые можно создавать с помощью данного плагина.

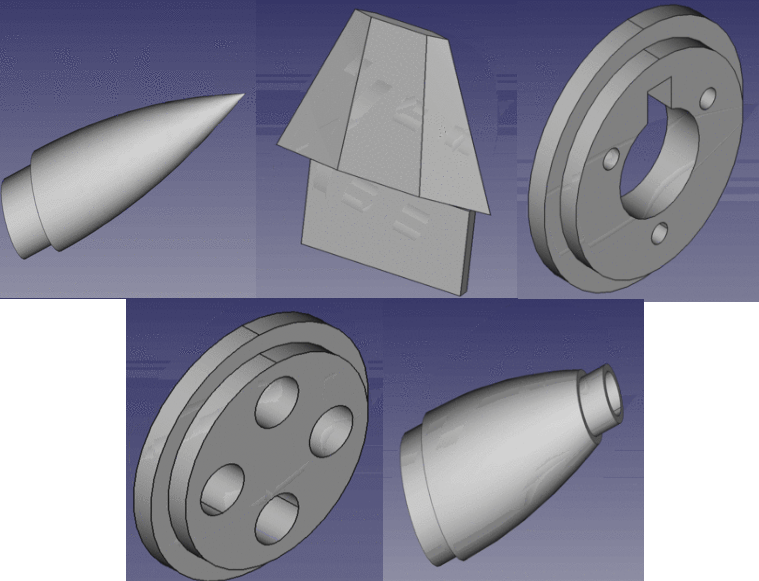


Рисунок 2.3 – Примеры созданных объектов с помощью плагина

# 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 [6].

**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 [7].

В проекте системы с помощью инструмента Enterprise Architect были созданы диаграммы классов в нотации UML. Из-за громоздкости диаграмма всего проекта была поделена на несколько частей. Класс KompasWrapper присутствует на двух частях диаграммы, рисунки 3.1-3.2; класс XWingBuilder присутствует на всех частях диаграммы; на четырёх частях, рисунки 3.2-3.5, присутствуют четыре вспомогательных класса фигур: Point2D, Point3D, Circle, Arc. Эти четыре класса и класс XWingBuilder связаны с классами констант, которые необходимы для построения детали, связи всех этих классов можно наблюдать на рисунках 3.2-3.5. Все диаграммы представлены на рисунках 3.1-3.5. На этих рисунках представлены диаграммы из проекта системы, то есть то, что было до всей реализации.

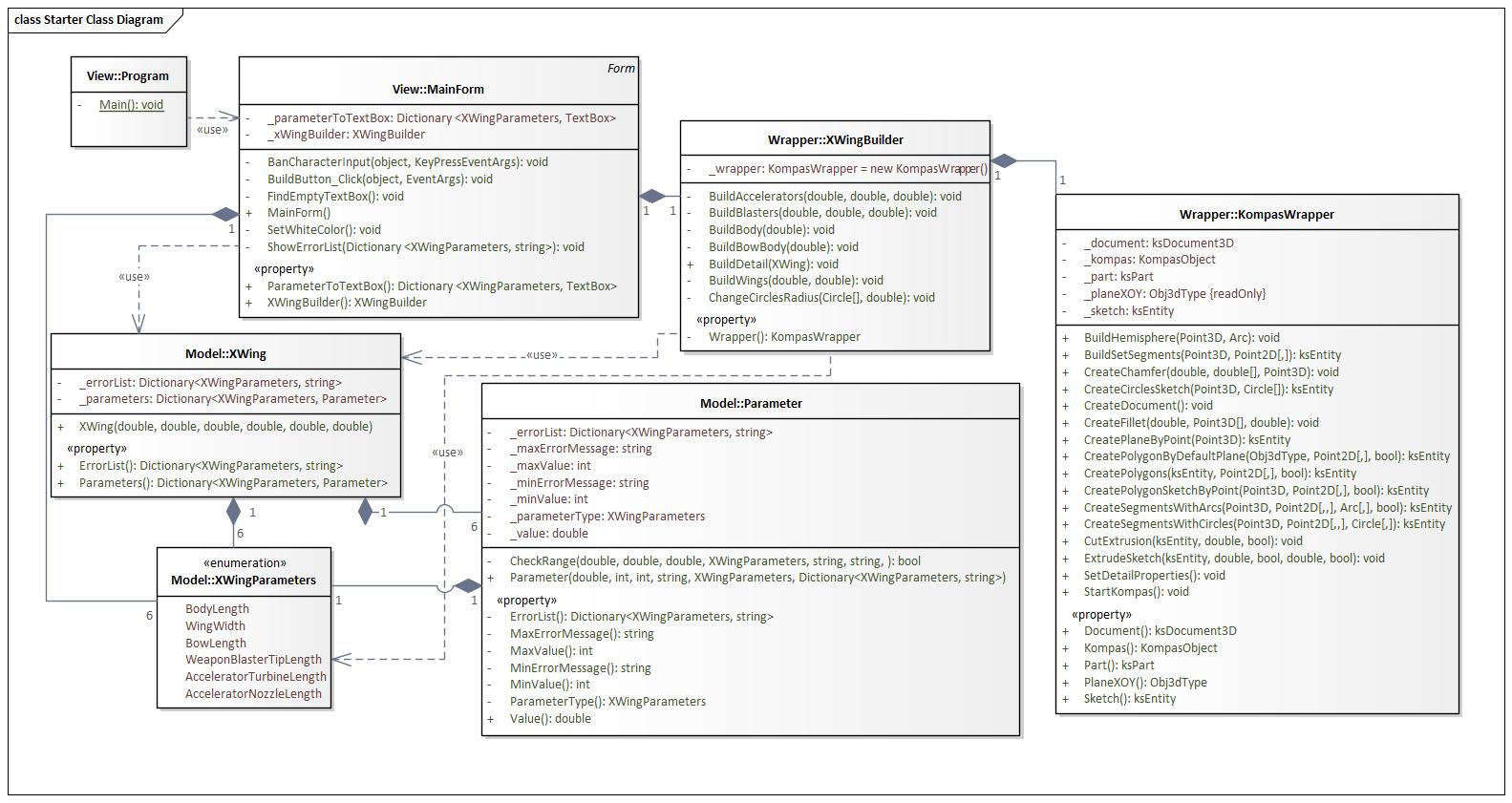


Рисунок 3.1 – Связь основной формы с классами параметров, построителя, обёртки

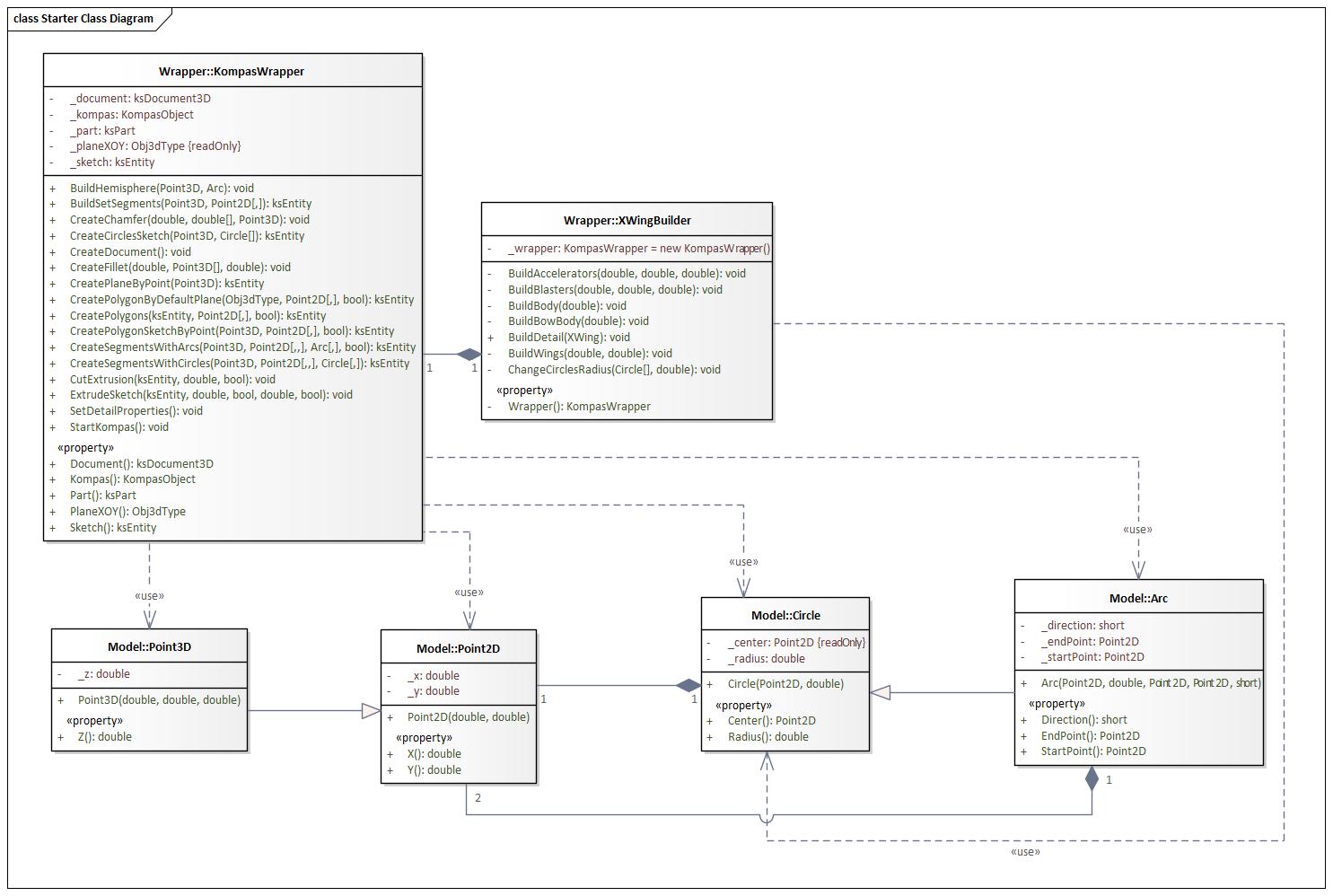
****

Рисунок 3.2 – Связь обёртки с построителем и классами геометрических фигур

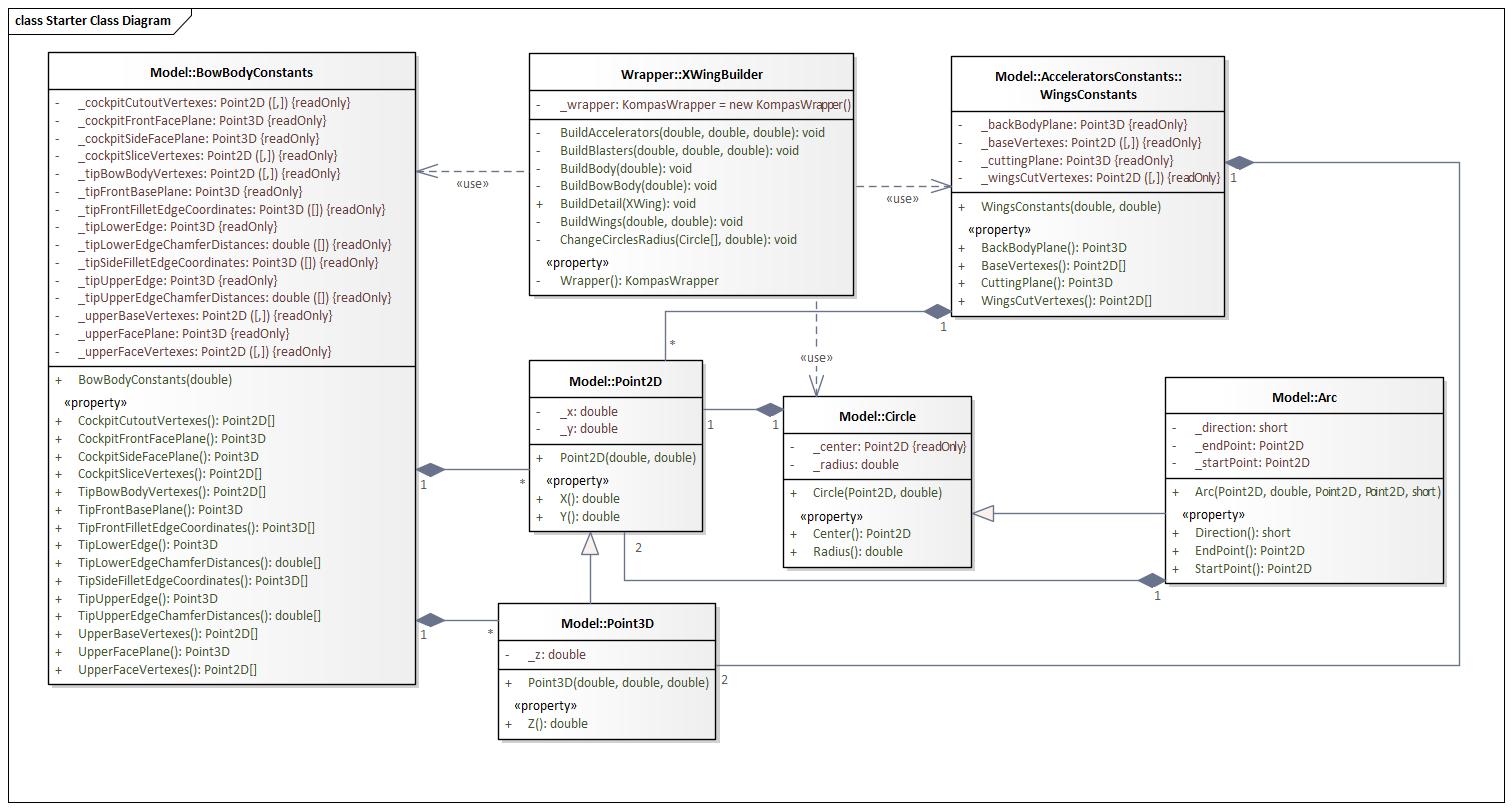
****

Рисунок 3.3 – Связь построителя с константами построения носовой части корпуса и крыльев

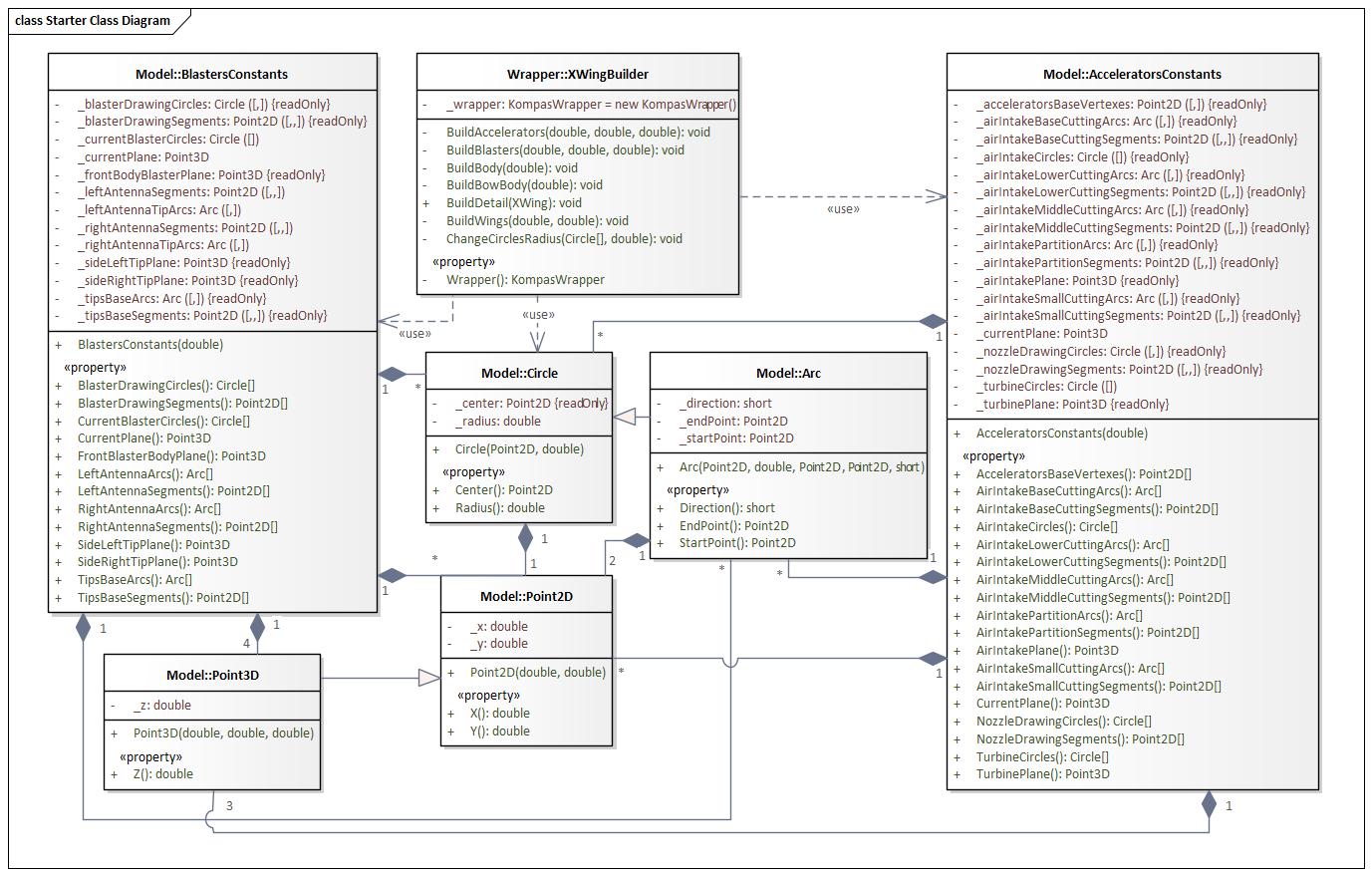


Рисунок 3.4 – Связи построителя с константами построения бластеров и ускорителей

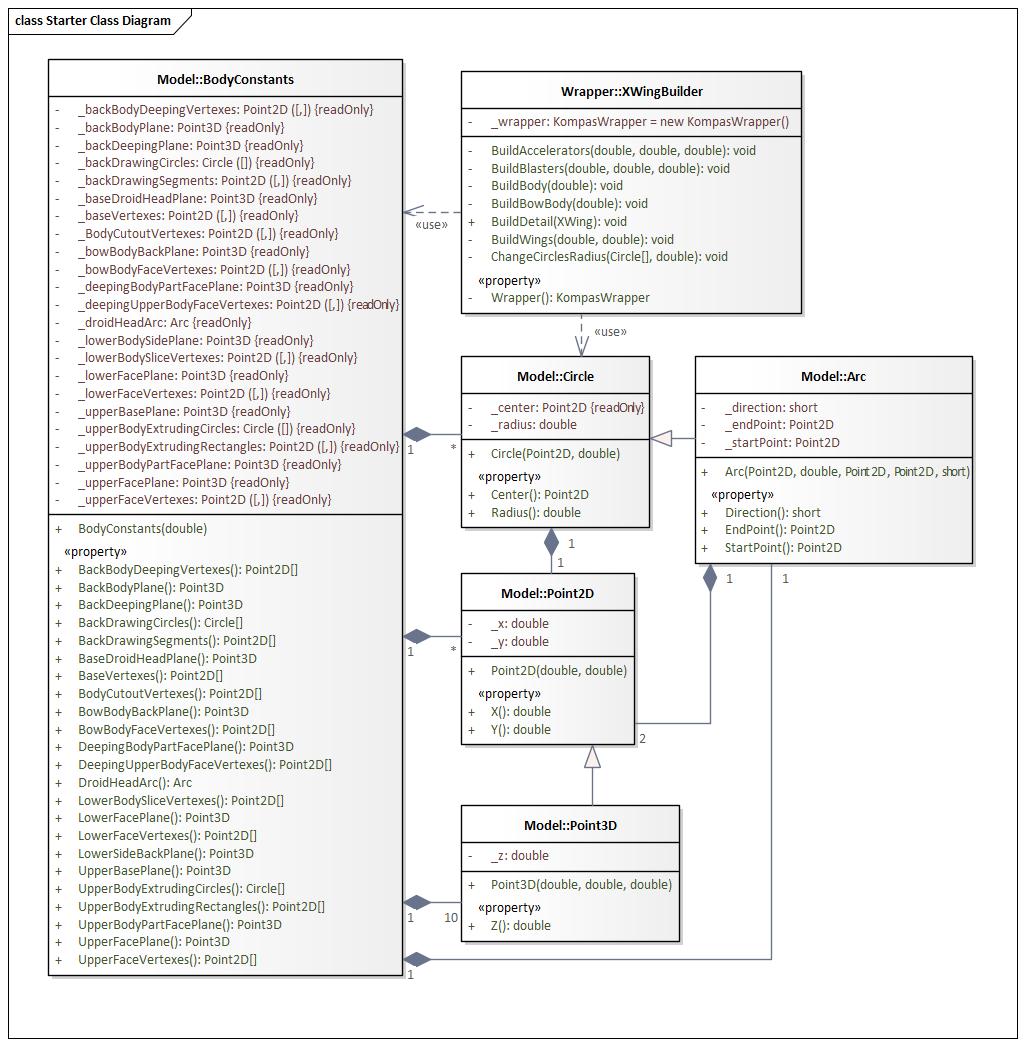
****

Рисунок 3.5 – Связь построителя с константами построения корпуса

Класс Parameter был предназначен для описания любого параметра истребителя и его валидацию, класс XWing был предназначен для объединения всех параметров воедино, также класс отвечает частично за валидацию. Класс KompasWrapper был предназначен для взаимодействия с САПР «Компас-3D», в нём содержатся методы для связи с САПР и рисования в ней, класс XWingBuilder был предназначен для построения звёздного истребителя, класс MainForm был предназначен для описания пользовательского интерфейса. Перечисление XWingParameters было необходимо для валидации параметров и корректной связи их с MainForm.

Классы Point2D, Point3D, Circle и Arc представляли собой соответственные фигуры точек двухмерного и трёхмерного пространства, круга и дуги. Классы BowBodyConstants, BodyConstants, WingsConstants, AcceleratorsConstants и BlastersConstants являлись классами заранее вычисленных констант, необходимых для построения детали, поля этих классов через свойства передавались в соответствующие методы класса XWingBuilder. Связь между классами фигур и классами констант – композиция, кратность можно увидеть на диаграммах классов на рисунках 3.3-3.5. Класс XWingBuilder использует все эти классы.

В процессе реализации библиотеки и её тестирования были изменены поля классов – теперь все приватные поля не имеют свойств, поля со стандартными геттерами и сеттерами имеют авто свойства. Также у авто свойств констант были изменены некоторые типы данных, чтобы исправить дублирование кода, из-за чего поменялась и кратность связей. Классам Arc, Circle, Parameter, Point2D и Point3D был добавлен интерфейс IEquatable для возможности сравнивать объекты этих классов.

В перечисление был добавлен новый элемент как дополнительная функциональность. В главной форме изменились методы для проверки ошибок введённых значений в режиме реального времени. Поэтому некоторые методы по перекраске полей ушли, вместо них появились другие методы, направленные на поиск ошибок.

Из-за большого числа классов диаграмму вновь необходимо делить на несколько частей. Метод представления архитектуры программы аналогичен тому, что представлен выше на рисунках 3.1-3.5. Итоговые диаграммы представлены на рисунках 3.6-3.9.

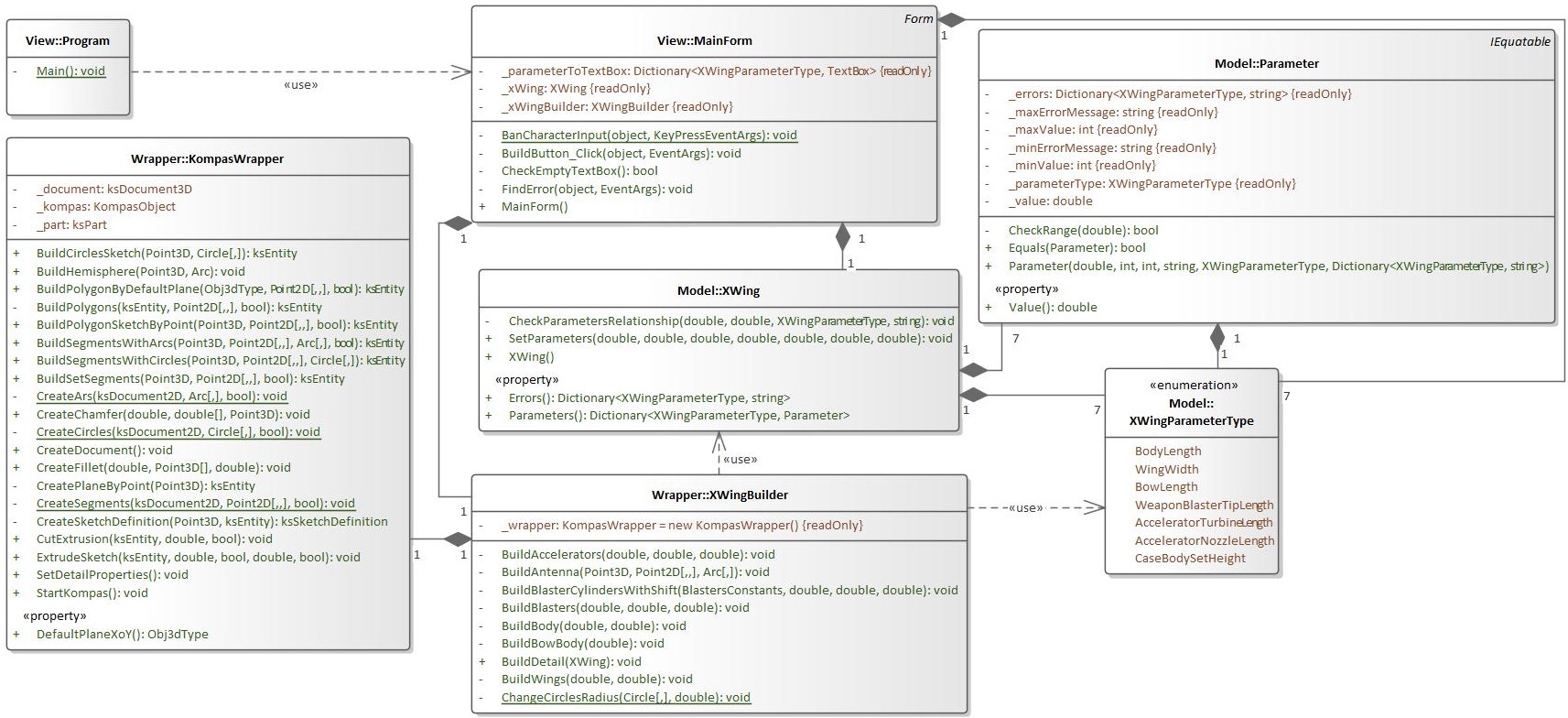


Рисунок 3.6 – Связь основной формы с классами параметров, построителя, обёртки

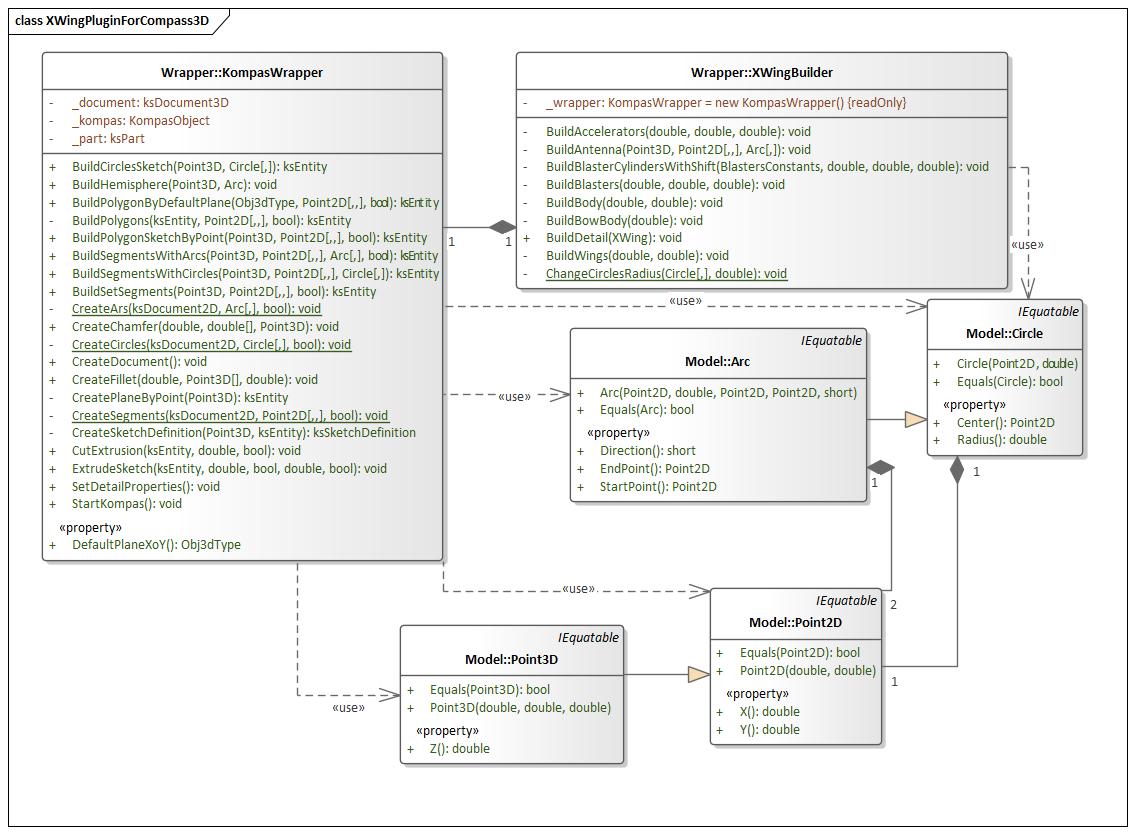


Рисунок 3.7 – Связь обёртки с построителем и классами геометрических фигур

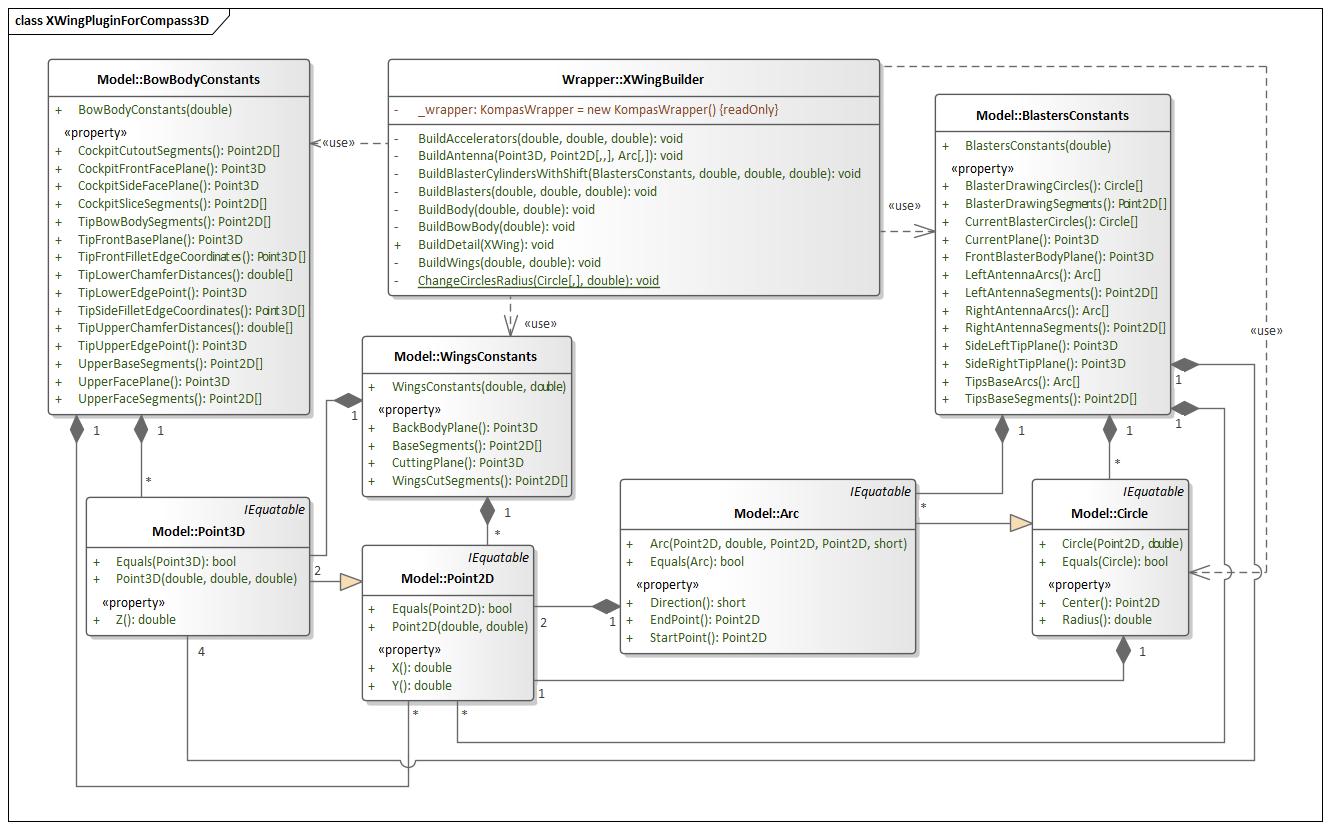


Рисунок 3.8 – Связь построителя с константами построения носовой части корпуса, крыльев и бластеров

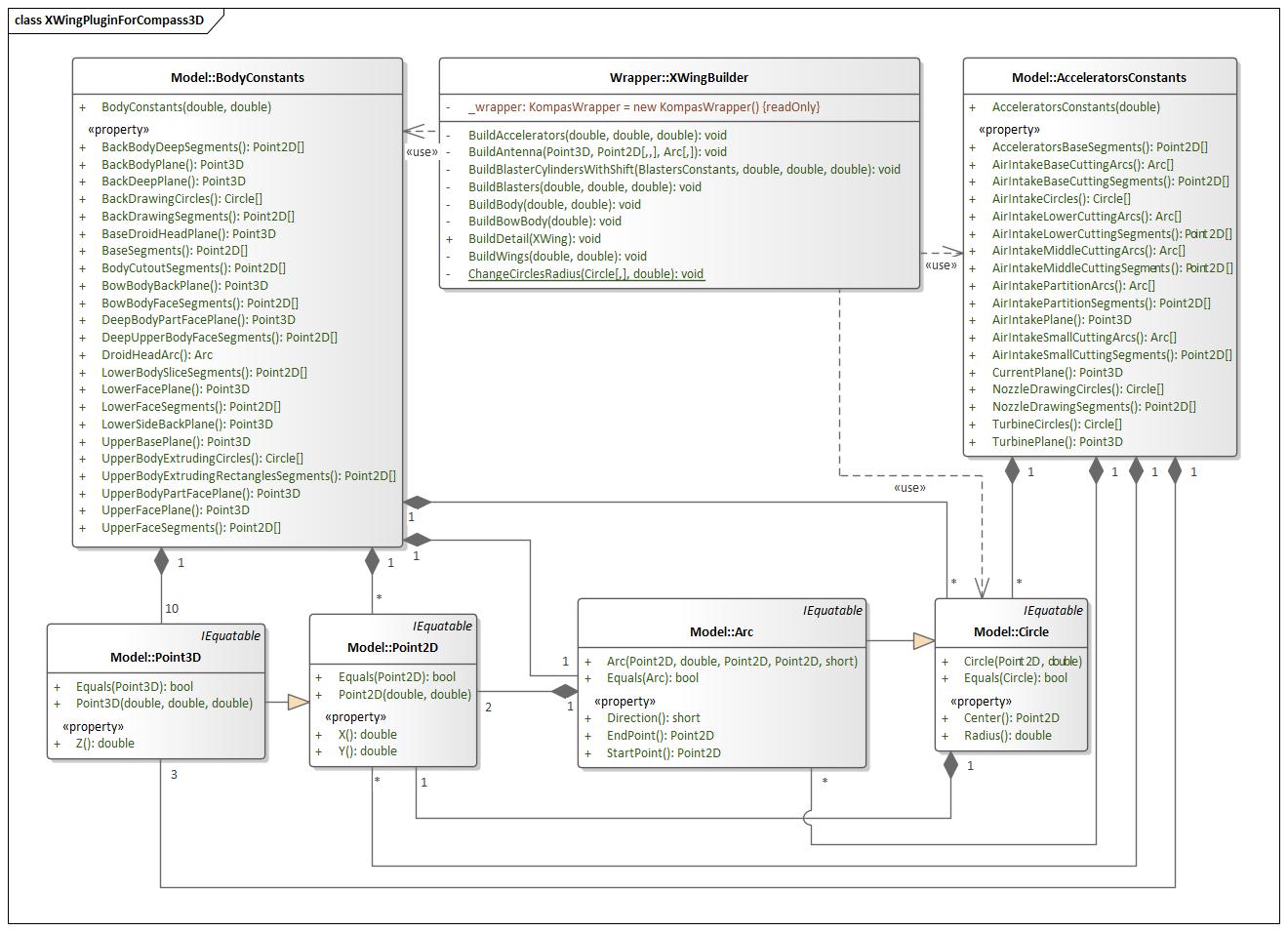


Рисунок 3.9 – Связи построителя с константами построения корпуса и ускорителей

# 4 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЯ

Пользовательский интерфейс (UI) – интерфейс, обеспечивающий передачу информации между пользователем-человеком и программно-аппаратными компонентами компьютерной системы [8].

Плагин представляет собой пользовательскую форму с полями для ввода соответствующих параметров, форму можно увидеть на рисунке 4.1. Посредством кнопки «Построить» осуществляется запуск САПР «Компас-3D», на рабочей области которой построится трехмерная модель детали по заданным параметрам. 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 в ходящих в допустимый диапазон, поле для ввода окрашивается в розовый цвет и выводится окно в режиме реального времени. При нажатии на кнопку построить появляется окно, информирующее пользователя о некорректности введенного значения. Пример неправильного ввода представлен на рисунке 4.2, пример вывода списка ошибок представлен на рисунке 4.3.

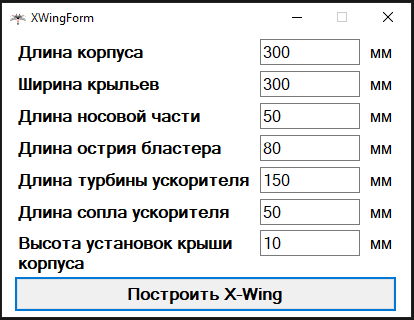


Рисунок 4.1 – Макет пользовательского интерфейса

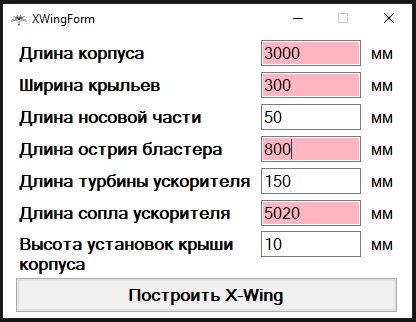


Рисунок 4.2 – Реакция приложения на ввод некорректных значений

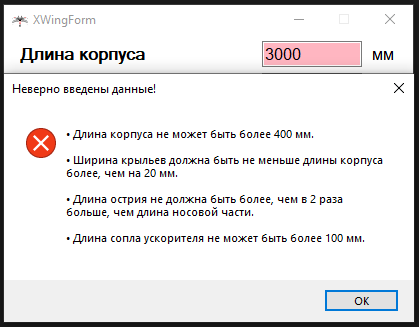


Рисунок 4.3 – Список ошибок на ввод некорректных значений

Если пользователь ввёл правильные значения, при нажатии кнопки «Построить» происходит построение детали. Трёхмерная модель звёздного истребителя T-65 «X-Wing», построенная в системе автоматизированного проектирования Компас-3D с помощью разработанной библиотеки представлена на рисунках 4.5 и 4.6.

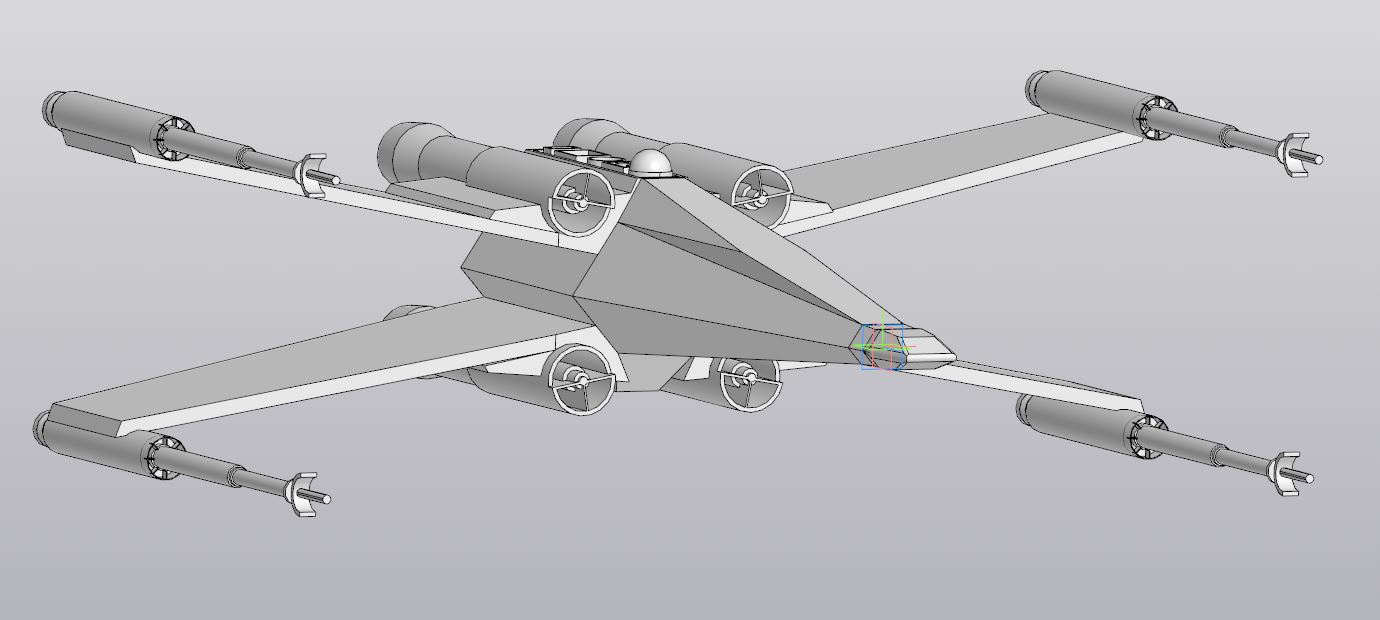


Рисунок 4.5 – Звёздный истребитель T-65 «X-Wing» в виде спереди

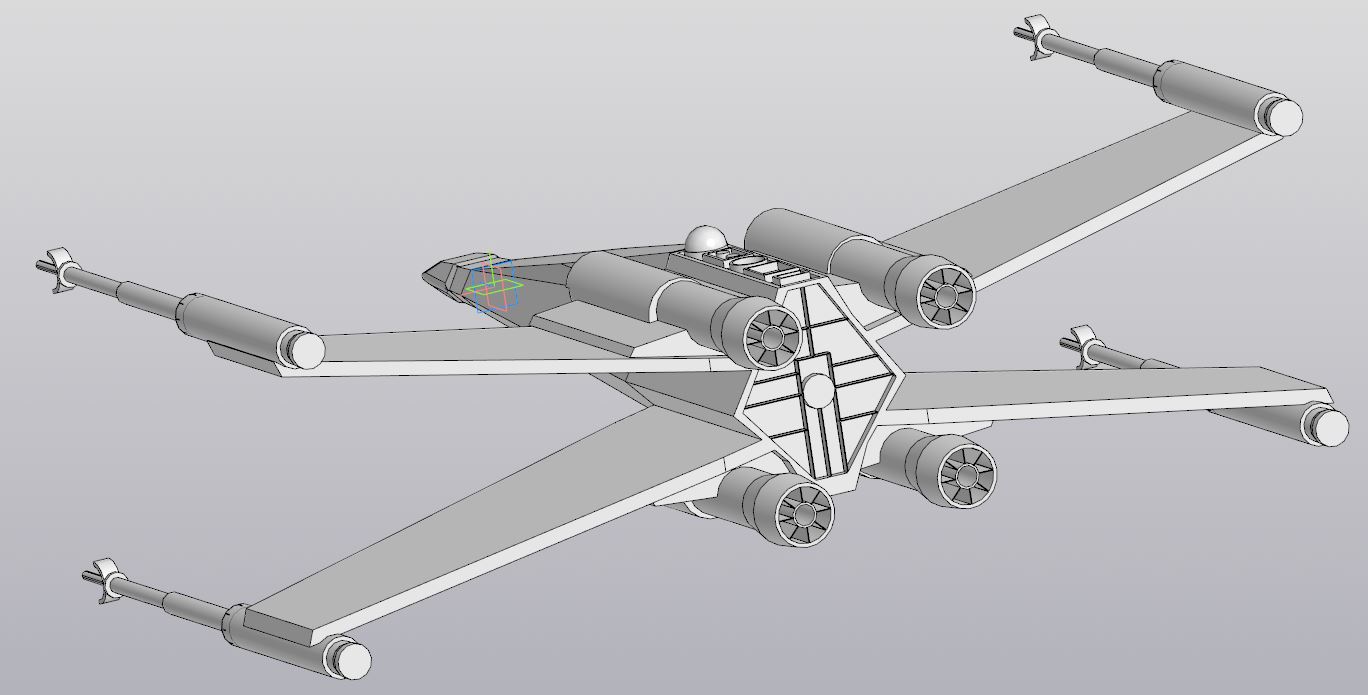


Рисунок 4.6 – Звёздный истребитель T-65 «X-Wing» в виде сзади

# 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 [9].

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

Тестирование при минимальных параметрах:

* длина корпуса равна 300 мм;
* ширина крыльев равна 300 мм;
* длина носовой части равна 50 мм;
* длина острия оружейного бластера равна 80 мм;
* длина турбины ускорителя равна 150 мм;
* длина сопла ускорителя равна 50 мм;
* высота установок крыши корпуса равна 10 мм.

Результат построения при заданных параметрах представлен на рисунках 5.1-5.2.

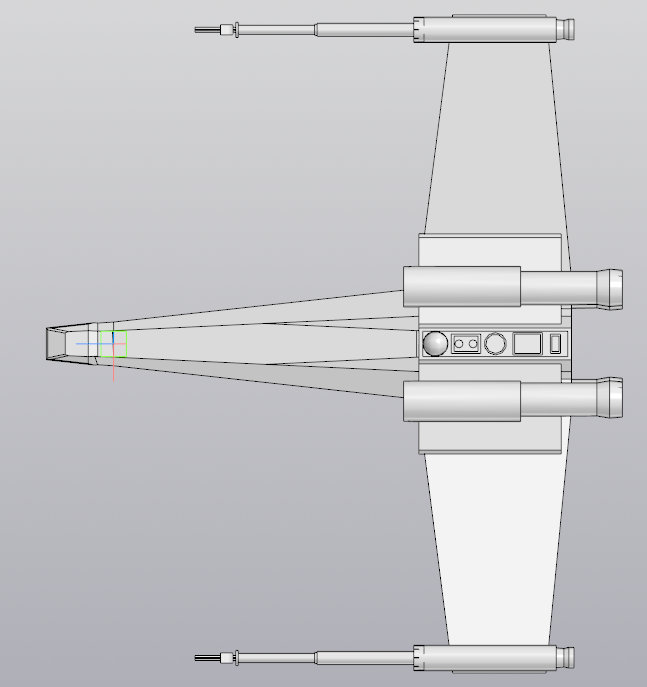


Рисунок 5.1 – Вид на модель в плоскости XZ

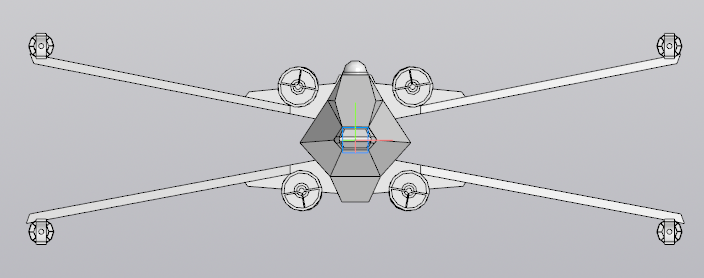


Рисунок 5.2 – Вид на модель в плоскости XY

Тестирование при средних параметрах:

* длина корпуса равна 350 мм;
* ширина крыльев равна 350 мм;
* длина носовой части равна 75 мм;
* длина острия оружейного бластера равна 105 мм;
* длина турбины ускорителя равна 200 мм;
* длина сопла ускорителя равна 75 мм;
* высота установок крыши корпуса равна 15 мм.

Результат построения при заданных параметрах представлен на рисунках 5.3-5.4.

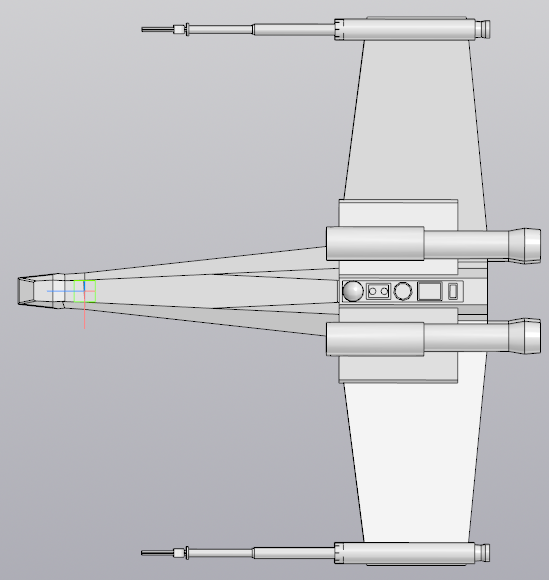


Рисунок 5.3 – Вид на модель в плоскости XZ

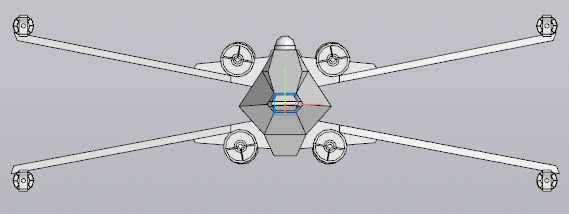


Рисунок 5.4 – Вид на модель в плоскости XY

Тестирование при максимальных параметрах:

* длина корпуса равна 400 мм;
* ширина крыльев равна 400 мм;
* длина носовой части равна 100 мм;
* длина острия оружейного бластера равна 130 мм;
* длина турбины ускорителя равна 250 мм;
* длина сопла ускорителя равна 100 мм;
* высота установок крыши корпуса равна 20 мм.

Результат построения при заданных параметрах представлен на рисунках 5.5-5.6.

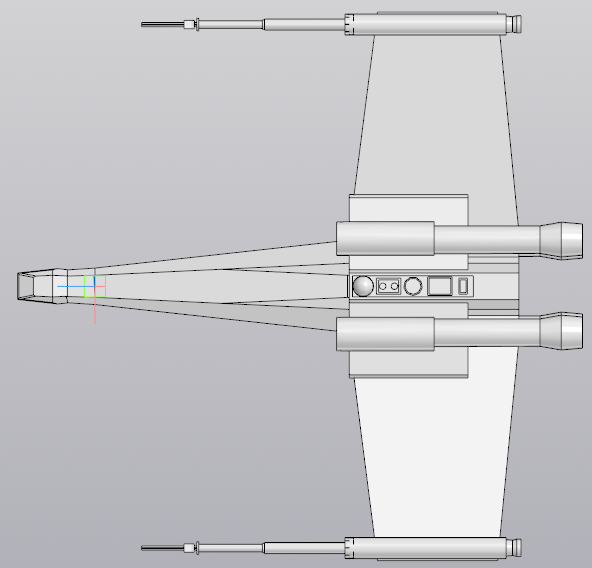


Рисунок 5.5 – Вид на модель в плоскости XZ

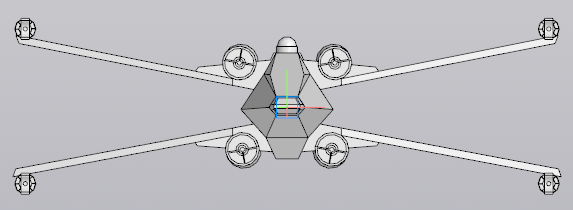


Рисунок 5.6 – Вид на модель в плоскости XY

# 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 [10].

Для тестирования используется библиотеки NUnit 3.13.3 и NUnit3TestAdapter 4.2.1. Тестами покрыта вся логика программы. Всего было написано 94 теста. На рисунках 5.7-5.12 представлено тестирование логики.

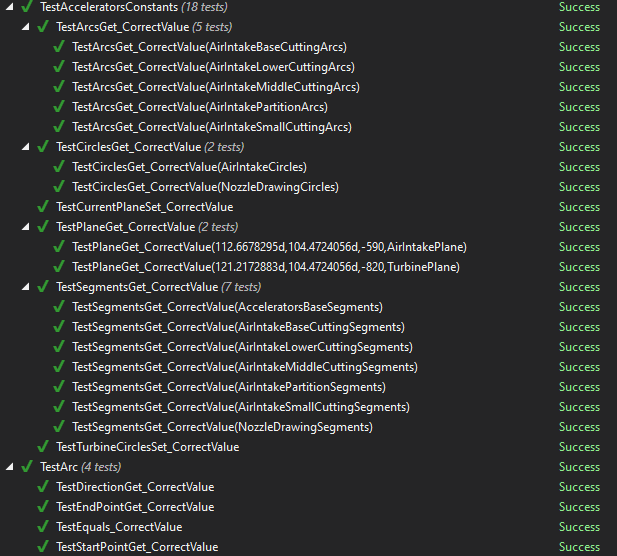


Рисунок 5.7 – Тестирование логики классов AcceleratorsConstants и Arc

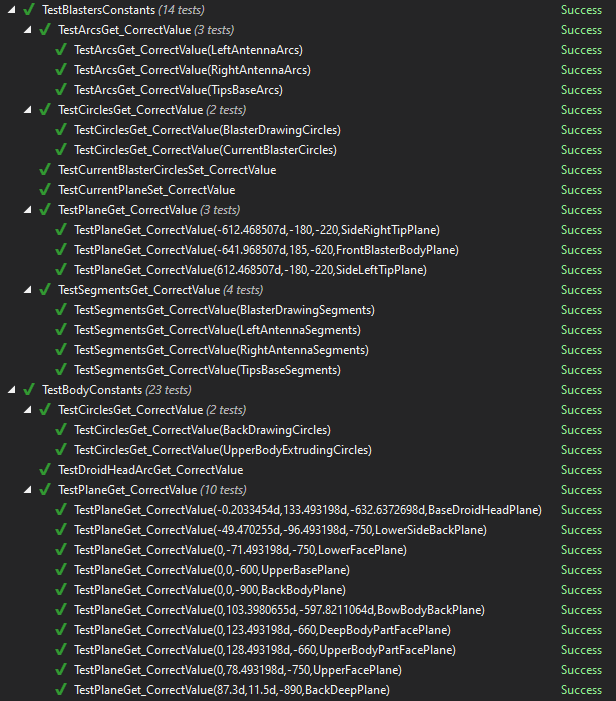


Рисунок 5.8 – Тестирование логики класса BlastersConstants

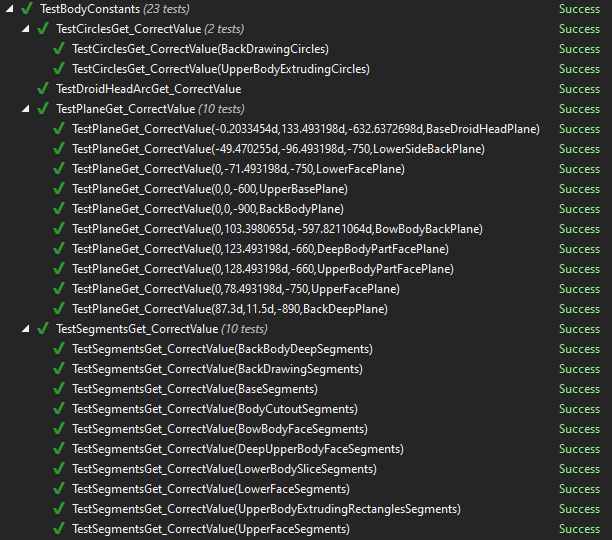


Рисунок 5.9 – Тестирование логики класса BodyConstants

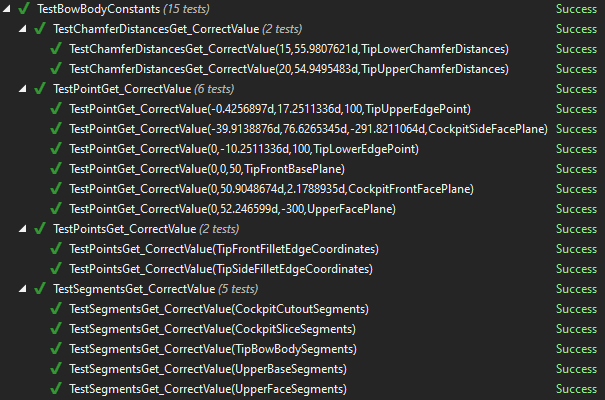


Рисунок 5.10 – Тестирование логики класса BowBodyConstants

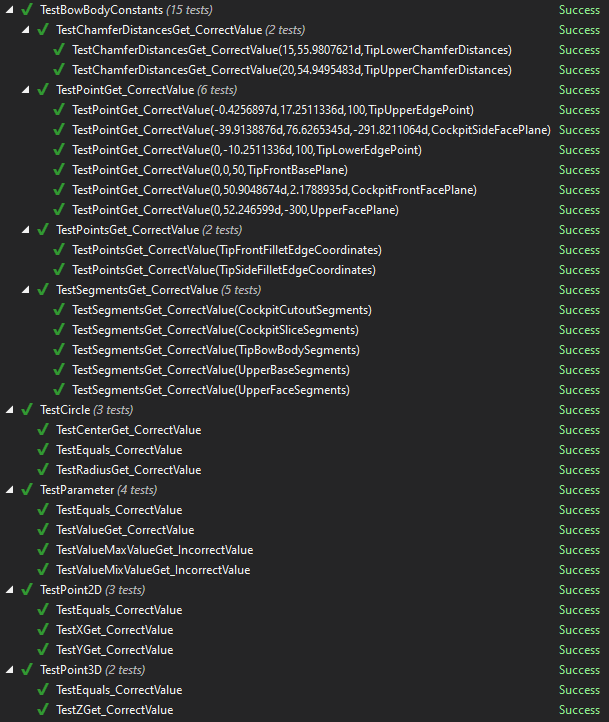


Рисунок 5.11 – Тестирование логики классов Circle, Parameter, Point2D и Point3D

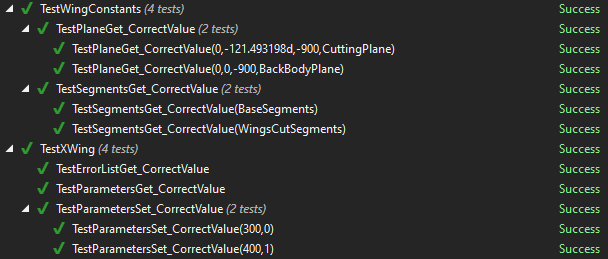


Рисунок 5.12 – Тестирование логики классов WingsConstants и XWing

# 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 персональном компьютере со следующей конфигурацией:

* процессор Intel(R) Core(TM) i7-7700K CPU @ 4.20GHz 4.20 GHz;
* оперативная память 16,0 ГБ;
* графический процессор объёмом памяти 12 ГБ.

Для тестирования будет зациклено построение модели со следующими параметрами:

* длина корпуса равна 300 мм;
* ширина крыльев равна 300 мм;
* длина носовой части равна 50 мм;
* длина острия оружейного бластера равна 80 мм;
* длина турбины ускорителя равна 150 мм;
* длина сопла ускорителя равна 50 мм;
* высота установок крыши корпуса равна 10 мм.

Диаграммы нагрузки на оперативную память и на процессор представлены на рисунках 5.13-5.14.

Здесь мог быть какой-то рисунок

Рисунок 5.13 – Нагрузка на оперативную память

Здесь мог быть какой-то рисунок

Рисунок 5.14 – Нагрузка на процессор

Ну и здесь должен быть какой-то анализ графиков…

# Заключение

В ходе 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 библиотеки с дополнительной функциональностью. Тестирование библиотеки включало в себя функциональное, модульное и нагрузочное тестирование. На основании готовой реализации библиотеки была сформирована UML-диаграмма классов, показывающая архитектуру приложения.

# Список использованных источников

1. Компас (САПР) [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Компас\_(САПР) (дата обращения: 30.09.2022)
2. Звёздный истребитель T-65 «X-wing» [Электронный ресурс] – Режим доступа: https:// starwars.fandom.com/ru/wiki/Звёздный\_истребитель\_T-65\_«X-wing» (дата обращения: 31.10.2022)
3. External workbenches [Электронный ресурс] – Режим доступа: https://wiki.freecadweb.org/External\_workbenches (дата обращения: 06.10.2022)
4. Airplane Design Workbench [Электронный ресурс] – Режим доступа: https://github.com/FredsFactory/FreeCAD\_AirPlaneDesign (дата обращения: 06.10.2022)
5. Rocket Workbench [Электронный ресурс] – Режим доступа: https://wiki.freecadweb.org/Rocket\_Workbench (дата обращения: 06.10.2022)
6. UML-диаграммы классов [Электронный ресурс] – Режим доступа: https://prog-cpp.ru/uml-classes (дата обращения: 06.10.2022)
7. Enterprise Architect (software) [Электронный ресурс] – Режим доступа: https://en.wikipedia.org/wiki/Enterprise\_Architect\_(software) (дата обращения: 06.10.2022)
8. Интерфейс пользователя [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Интерфейс\_пользователя (дата обращения: 06.10.2022)
9. Функциональное (ручное) тестирование [Электронный ресурс] – Режим доступа: https://logrocon.ru/functional\_testing (дата обращения: 30.10.2022)
10. Модульное тестирование [Электронный ресурс] – Режим доступа: https://logrocon.ru/news/unit\_testing (дата обращения: 30.10.2022)
11. Нагрузочное тестирование [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Нагрузочное\_тестирование (дата обращения: 30.10.2022)