Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра компьютерных систем в управлении и проектировании (КСУП)

РАЗРАБОТКА ПЛАГИНА «НАСТЕННЫЕ ЧАСЫ» ДЛЯ СИСТЕМЫ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ «SOLID WORKS»

Пояснительная записка

по дисциплине «Основы разработки САПР»

Выполнил:

Студент гр.580-1

\_\_\_\_\_\_\_Цуканов С.С.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2023 г.

Руководитель:

к.т.н., доцент

каф. КСУП

\_\_\_\_\_\_\_Калентьев А.А.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2023 г.

**СОДЕРЖАНИЕ**

[2 ПОСТАНОВКА ЗАДАЧИ И АНАЛИЗ ЗАДАЧИ 4](#_Toc154613839)

[3 ОПИСАНИЕ ПРЕДМЕТА ПРОЕКТИРОВАНИЯ 5](#_Toc154613840)

[4 ВЫБОР ИНСТРУМЕНТОВ И СРЕДСТВ РЕАЛИЗАЦИИ 7](#_Toc154613841)

[5 НАЗНАЧЕНИЕ ПЛАГИНА 8](#_Toc154613842)

[6 ОБЗОР АНАЛОГОВ 9](#_Toc154613843)

[7 ОПИСАНИЕ РЕАЛИЗАЦИИ 11](#_Toc154613844)

[8 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЯ 13](#_Toc154613845)

[9 ТЕСТИРОВАНИЕ ПЛАГИНА 16](#_Toc154613846)

[9.1 Функциональное тестирование 16](#_Toc154613847)

[9.2 Модульное тестирование 19](#_Toc154613848)

[9.3 Нагрузочное тестирования 20](#_Toc154613849)

[ЗАКЛЮЧЕНИЕ 23](#_Toc154613850)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 24](#_Toc154613851)

**1 ВВЕДЕНИЕ**

САПР (Системы автоматизированного проектирования) – позволяют быстро и без затрат спроектировать некоторую систему, что играет огромную роль в развитии моделирования и проектирования в целом. В настоящий момент данные системы позволяют создавать все более и более сложные модели, воспроизводить сложные инженерные расчеты, что экономит силы, время и деньги у их пользователей (инженеров, проектировщиков, архитекторов и т.д.).

Таким образом, целью данной работы является разработка плагина (расширения) для системы автоматизированного проектирования «Solid Works», который позволяет строить настенные часы по заданным параметрам. Необходимый стек технологий для разработки плагина является: среда разработки Visual Studio 2022, язык программирования C#, и фреймворк .NET.

# **2 ПОСТАНОВКА ЗАДАЧИ И АНАЛИЗ ЗАДАЧИ**

Требуется разработать плагин к САПР «Solid Works», который позволяет создавать модель настенных часов по входным параметрам.

Настенные часы предназначены для показа времени, являются частью интерьера домов, офисов, рабочих помещений и т.п. Тем самым, исходя из технического задания, требуется вручную задавать радиус циферблата, ширину бортика часов, высоту бортика, вырез по периметру бортика, длину стрелок часов и минут. Так же предусмотреть возможность отображать только риски часов или часов и минут вместе.

Для реализации данной задачи необходимо было сначала смоделировать настенные часы вручную, чтобы посмотреть какие параметры, операции и эскизы необходимы для построения. Самым главным и трудоемким было создать архитектуру плагина и изучить API к «Solid Works». API – «application programming interface» - программный интерфейс, описывающий набор способов и правил взаимодействия одной программы с другой.

# **3 ОПИСАНИЕ ПРЕДМЕТА ПРОЕКТИРОВАНИЯ**

Настенные часы – являются частью интерьера домов, офисов и людских помещений, главный функционал которых – показ времени. Модель часов с различными ее параметрами показана на рисунке 3.1.

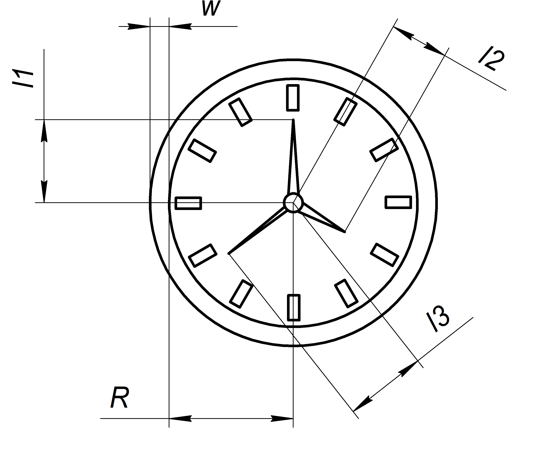


Рисунок 3.1 – модель часов с изменяемыми параметрами

Изменяемые параметры для плагина (также часть обозначений показаны на рис. 2.1):

* R – радиус циферблата (100мм – 200мм);
* l1 – длина минутной стрелки, R/2 + 4 < l1< R-2;
* l2 – длина часовой стрелки, R/5 < l2 < l1/2;
* l3 – длина секундной стрелки, l1/2 < l3 < l1-4;
* w – ширина бортика (30мм – 60мм);
* d – выступ бортика над циферблатом (глубина) (20мм – 40мм);
* Показывание только часов или часов и минут вместе;
* Вырез по периметру бортика, который задается малым радиусом r, в виде окружности, максимальный размер малого радиуса может быть r = R, тогда в окружность циферблата может войти 2 окружности выреза, количество вырезов k = π/arcsin(r/R+w), при r = R+w, получится k = 3,14 / 1,57 ≈ 2. Но надо и учитывать, что тогда вырезы перекроют циферблат, следовательно у нас малый радиус выреза должен ограничиваться шириной бортика w. 0 <= r < w;
* Так же можно задать и вручную количество этих самых вырезов k, которое ограничивается зависимостью k < = π/arcsin(r/R+w);

**4 ВЫБОР ИНСТРУМЕНТОВ И СРЕДСТВ РЕАЛИЗАЦИИ**

Программа выполнена на языке программирования C#, в среде интегрированной разработки Microsoft Visual Studio 2022 с использованием фреймворка .NET Framework 4.7.2. Библиотека для взаимодействия с САПР - «SolidWorks».

Инструментом тестирования и создания модульных тестов был выбран тестовый фреймворк NUnit.

# **5 НАЗНАЧЕНИЕ ПЛАГИНА**

Назначение данного плагина - это быстрое моделирование настенных часов разных параметров, позволяющее посмотреть как будут выглядеть настенные часы при разных размерах и видах, задаваемых пользователем.

# **6 ОБЗОР АНАЛОГОВ**

**«Редуктор – 3D V2.3»** для Компас 3D

Данный плагин косвенно похож на разрабатываемый плагин. Суть его заключается в том, что это отдельное приложение, использующее API САПР Компас3D, для построения редуктора по заданным параметрам.

Задаваемые параметры:

* вращающий момент на ведомом валу редуктора;
* угловая скорость ведомого вала;
* передаточное число редуктора;
* режим работы редуктора.

Скриншот пользовательского интерфейса представлен на рисунке 6.1.

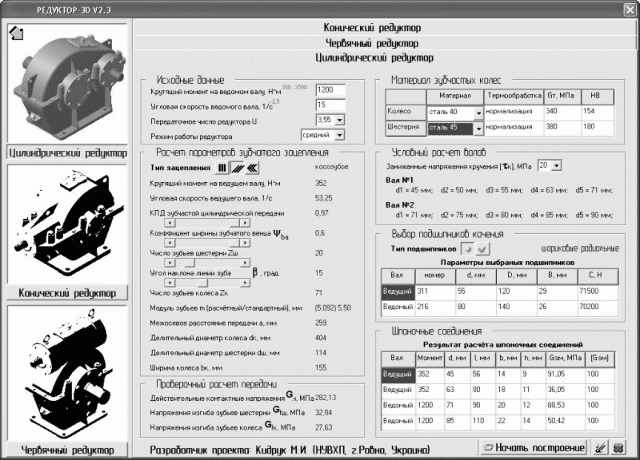


Рисунок 6.1 – Пользовательский интерфейс программы «Редуктор-3D»

По нажатию кнопки «Начать построение», приложение открывает Компас 3D и начинает построение редуктора. Пример построения показан на рисунке 6.2.

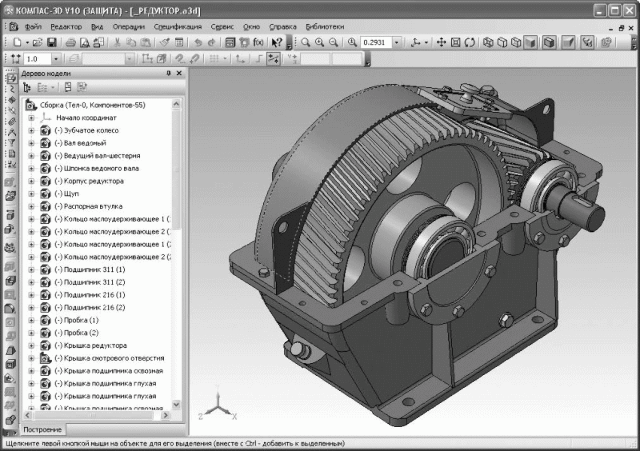


Рисунок 6.2 – Построенный редуктор из плагина «Редуктор-3D»

**7 ОПИСАНИЕ РЕАЛИЗАЦИИ**

Для описания графической модели проекта был использован стандарт UML – диаграмм.

UML –  язык графического описания для объектного моделирования в области разработки программного обеспечения, для моделирования бизнес-процессов, системного проектирования и отображения организационных структур.

Конечная диаграмма классов с использованием стандарта UML представлена на рисунке 7.1.

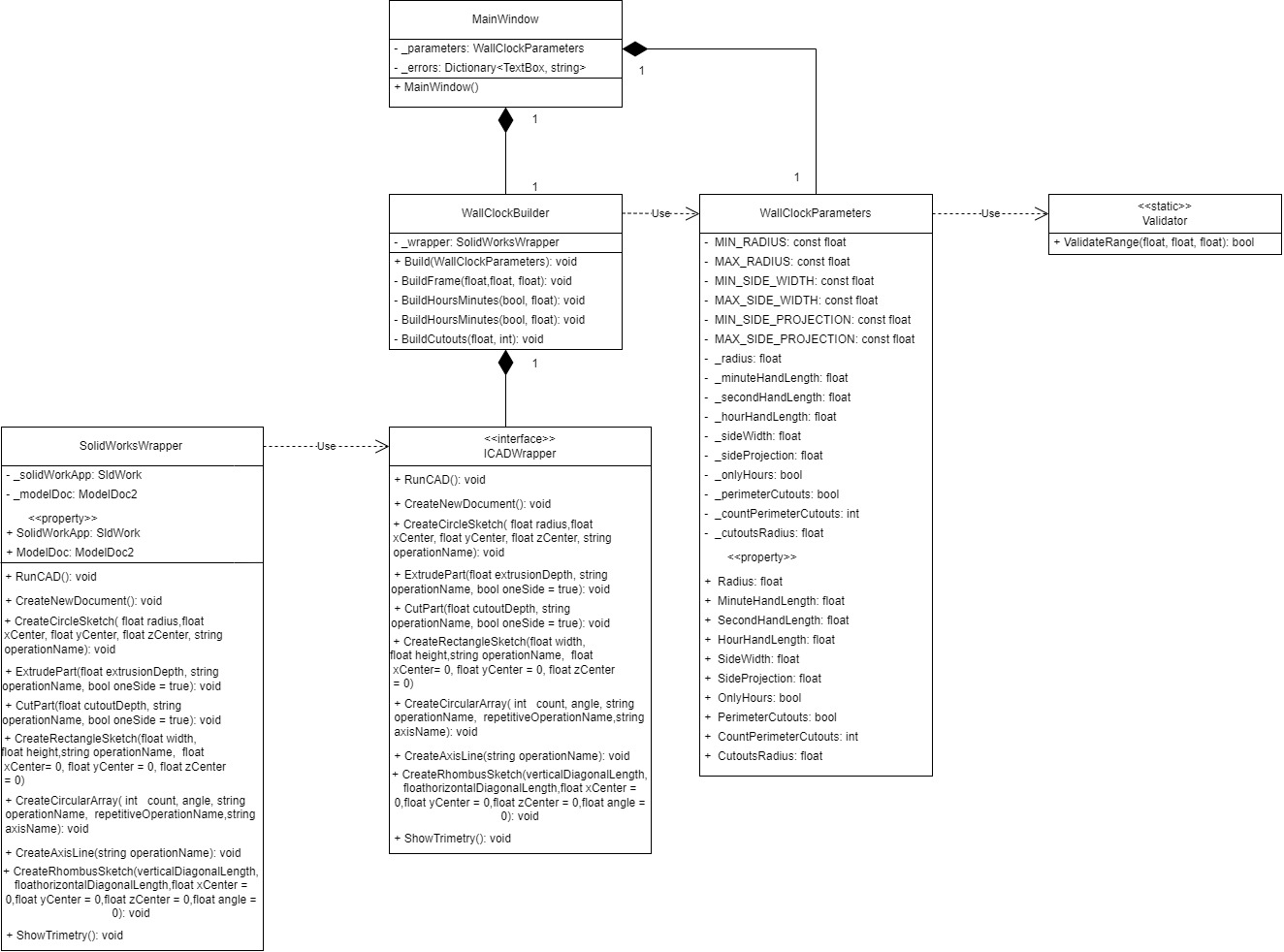


Рисунок 7.1 – итоговая диаграмма классов проекта

В ходе разработки проекта, была добавлена еще одна сущность к диаграмме классов – **ICADWrapper**. **ICADWrapper** – это абстрактная сущность, которая предоставляет интерфейс для САПР обертки. SolidWorksWrapper реализует данную сущность.

Такой подход позволяет нам на зависеть от выбранной САПР, и гибко для каждой из САПР предоставлять свою реализацию.

MainWindow – сущность, отвечающая за графические элементы и контролы графического интерфейса. Данная сущность композирует сущности WallCloclBuilder и WallClockParameters. Так же данная сущность отвечает за конкретную реализацию интерфейса ICADWrapper.

WallClockBuilder – сущность, описывающая процесс построения настенных часов в САПР. Сущность композирует интерфейс ICADWrapper, и использует для построения сущность WallClockParameters.

WallClockParameters – сущность, являющаяся бизнес-логикой приложения. Она содержит в себе все параметры настенных часов, необходимых для построения. Это самая независимая сущность в архитектуре. Для валидации параметров используется Validator.

Validator – статический класс, предоставляющий методы для валидации значений.

ICADWrapper – интерфейс, предоставляющий различные методы для построения различных элементов в САПР.

SolidWorksWrapper – класс, реализующий интерфейс ICADWrapper для САПР SolidWorks.

# **8 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЯ**

Пользовательский интерфейс состоит из одного окна (рисунок 8.1). В левой части интерфейса располагаются элементы для ввода параметров необходимых для построения детали. В правой части располагается изображение с построенными часами и кнопки для построения, перехода на github разработчика, переход на сайт документации API SolidWorks.

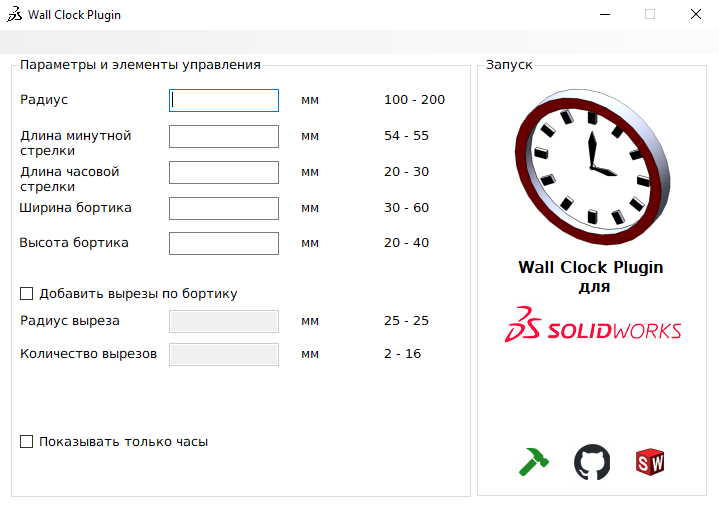


Рисунок 8.1 – пользовательский интерфейс плагина

При нажатии кнопки построения без введенных параметров, деталь построится по умолчанию с минимальными значениями (рисунок 8.2).

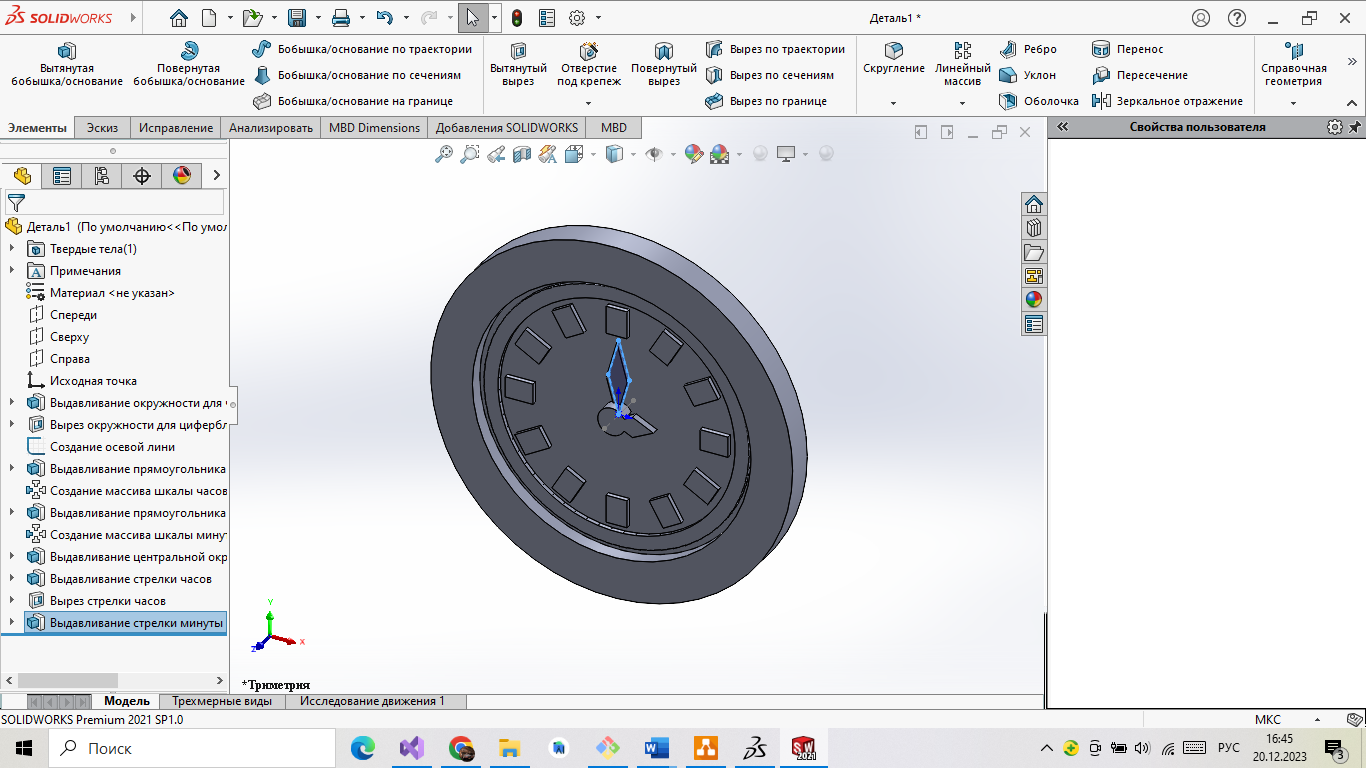


Рисунок 8.2 – деталь с параметрами по умолчанию

При вводе некорректных значениях (выходящих за границы допустимых, неудовлетворяющих зависимым параметрам), текстовое поле подсвечивается красным цветом и при попытке построить деталь, выводится сообщение о неправильности введенных значений (рисунок 8.3).

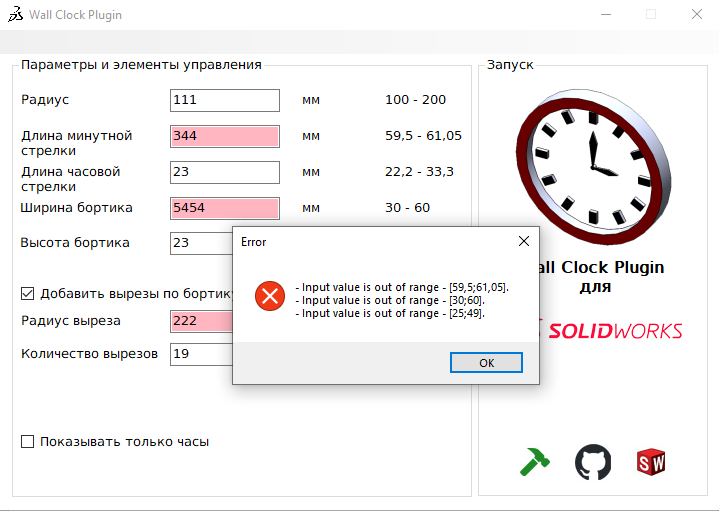


Рисунок 8.3 - попытка построения детали при некорректных значениях

# **9 ТЕСТИРОВАНИЕ ПЛАГИНА**

Тестирование данного плагина проводилось тремя различными видами тестирования:

* Функциональное тестирование – проверка функциональных требований приложения.
* Модульное тестирование – вид тестирования, позволяющий изолировано проверить наименьшую единицу программы.
* Нагрузочное тестирование – позволяет проверить работоспособность программы при экстремальных условиях использования.

Все виды тестирования позволяют проверить работоспособность программы, соответствие функциональным требованиям и корректность работы отдельных модулей программы.

## **9.1 Функциональное тестирование**

Проверим работу плагина при попытке построения с минимальными параметрами часов, максимальными параметрами и при попытке построения с некорректными значениями параметров.

Построим деталь с минимальными параметрами детали (рисунок 9.1).

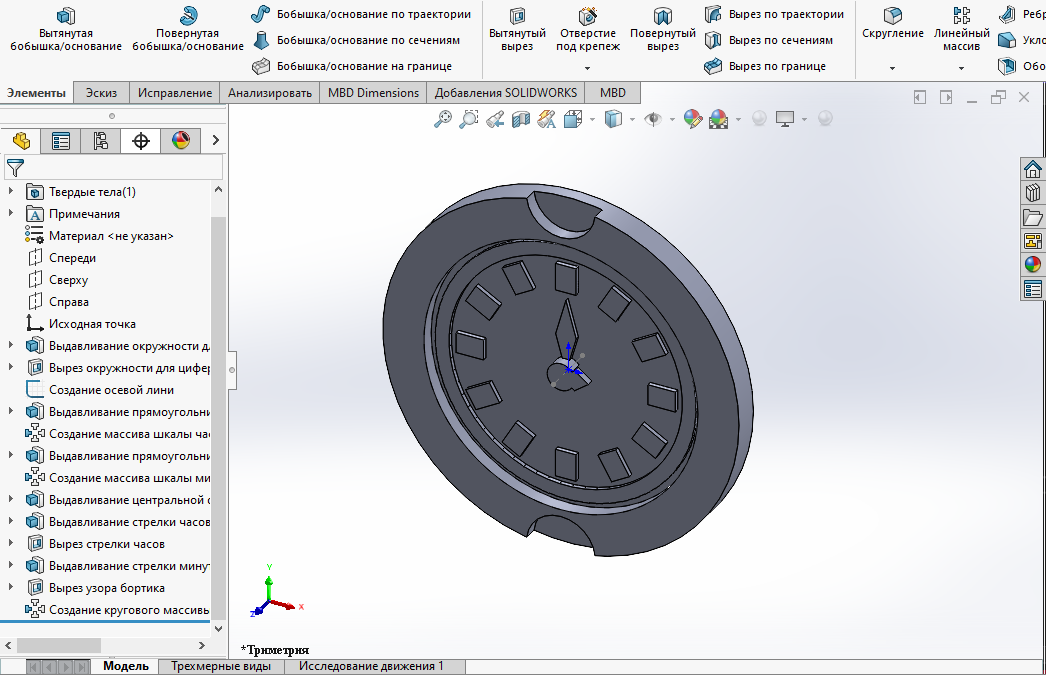


Рисунок 9.1 – построение детали с минимальными значениями параметров

Как видно из результата деталь правильно строится при минимальных значениях параметров.

Построим деталь, но теперь с максимальными значениями (рисунок 9.2).

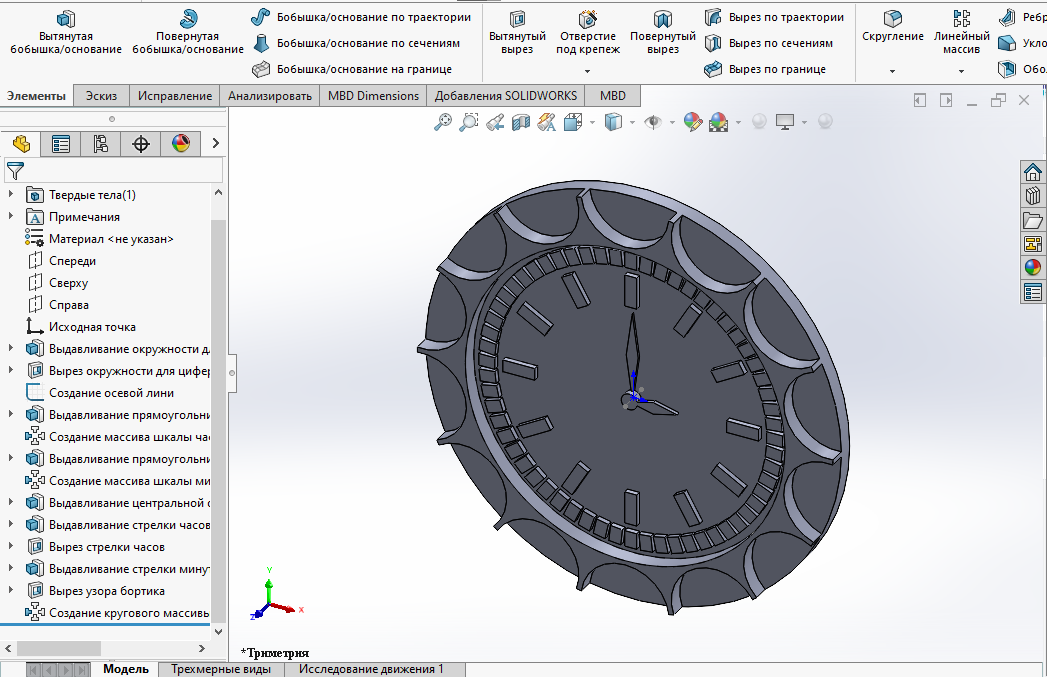


Рисунок 9.2 – построение детали с максимальными значениями параметров

Сделаем вывод, что деталь правильно строится как при минимальных, так и при максимальных значениях параметров.

Теперь произведем ситуацию, когда вводятся некорректные значения параметров, допустим не удовлетворяющие зависимым параметрам. Результат представлен на рисунке 9.3.

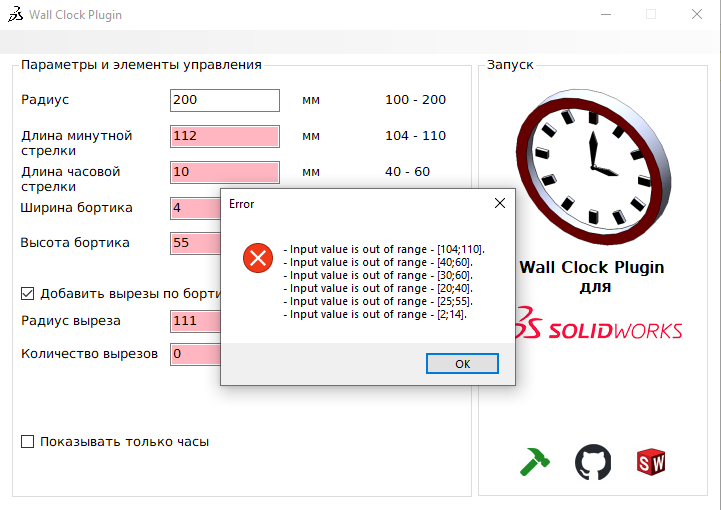


Рисунок 9.3 – попытка построения детали при некорректных значениях

При попытке построения, программа выдает ошибку, что не позволяет пользователю получить деталь с некорректными значениями.

**9.2 Модульное тестирование**

В целях проверки правильности работы отдельных методов и свойств, было проведено модульное тестирование. С помощью фреймворка NUnit были написаны модульные тесты на классы WallClockParameters и Validator. Прохождение тестов представлено на рисунке 9.4.

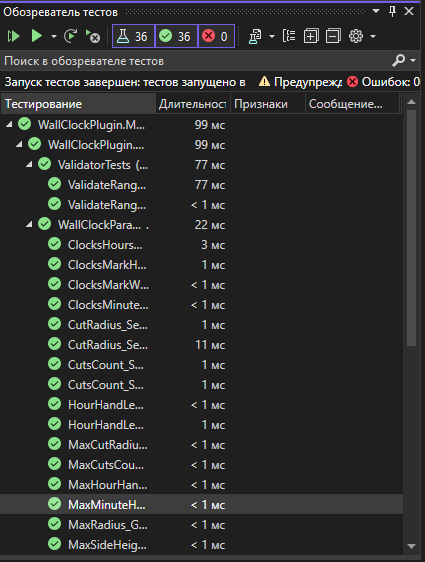


Рисунок 9.4 – прохождение unit-тестов

Процент покрытия тестами составил сто процентов.

**9.3 Нагрузочное тестирования**

В целях проверки производительности плагина, было проведено нагрузочное тестирование. Тестирование проводилось на ПК со следующими параметрами:

* ЦП AMD Ryzen 3 3200U with Radeon Vega Mobile Gfx 2.60 GHz;
* 8 гб ОЗУ;
* Встроенный графический процессор AMD Radeon Vega 3;

Для тестирования был создан отдельный консольный проект с подключенным проектом модели. Запустим в бесконечном цикле построение детали со средними значениями параметров. Полученные результаты представим в виде графиков. График зависимости времени построения от номера итерации построения показан на рисунке 9.5.

Рисунок 9.5 – график зависимости времени построения от номера построения

Из графика видно, что кривая скачкообразно растет вверх. Это означает, что при большем количестве построений время построения сильно растет, что может быть связано с расходами ресурсов компьютера повышения нагрузки на оперативную память, процессор и на графический процессор.

На рисунке 9.6 изображен график зависимости расходуемой оперативной памяти от количества построений.

Рисунок 9.6 – график зависимости расходуемой ОЗУ от номера построения

Из графика видно, что снятые показания используемой ОЗУ не значительно увеличиваются в зависимости от роста итерации построения, что может говорить о хорошем результате. Однако, тестирование длилось 16:34, общее количество построений - 43 модели, тестирование прекратилось от того, что процесс SolidWorks завершился с сообщением о том, что количество ресурсов памяти израсходовано. Получившийся график и явный результат работы противоположны друг другу. Это может означать, что управлением занимаемой памяти занимался файл подкачки, который временно выделял память с диска, зарезервированную операционной системой, тем самым освобождая оперативную память. И когда выделенная память достигла своего предела, процесс просто завершился тем самым остановив тестирования. Вторая версия такого поведения это то, что у SolidWorks есть свое внутреннее управление памятью – SolidWorksResourceManager, который решает такие ситуации и отвечает за выделение памяти под построение 3D деталей.

# **ЗАКЛЮЧЕНИЕ**

В ходе выполнения лабораторных работы была изучена предметная область проектирования, изучены технологии и принципы проектирования плагинов для автоматизированных систем проектирования. Так же были изучены аналоги разрабатываемого плагины, различные виды САПР, программный интерфейс для SolidWorks. Было спроектированы с помощью UML диаграммы классов, написан собственный плагин и проведено для него модульное и нагрузочное тестирование. В процессе разработки плагина был добавлен дополнительный функционал, выбранный заказчиком (преподавателем), что являлось имитацией настоящего случая, когда в процессе разработки продукта у заказчика появляется желание внести правки в ход разработки. Дополнительный функционал был также протестирован и сдан заказчику.

# **СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ**

1. SolidWorks API Help [Электронный ресурс]. – Режим доступа <https://help.solidworks.com/2019/russian/SolidWorks/sldworks/c_solidworks_api.htm> Дата обращения (12.12.2023)
2. Habr Зачем нам UML? Или как сохранить себе нервы и время [Электронный курс]. – Режим доступа <https://habr.com/ru/articles/458680/> Дата обращения (17.12.2023)
3. Autodesk Inventor ru Wikipedia [Электронный ресурс]. – Режим доступа <https://ru.wikipedia.org/wiki/Autodesk_Inventor#:~:text=Autodesk%20Inventor%20%E2%80%94%20%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0%20%D1%82%D1%80%D1%91%D1%85%D0%BC%D0%B5%D1%80%D0%BD%D0%BE%D0%B3%D0%BE%20%D1%82%D0%B2%D0%B5%D1%80%D0%B4%D0%BE%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%BE%D0%B3%D0%BE,%D1%81%D0%BE%D0%B7%D0%B4%D0%B0%D0%BD%D0%B8%D1%8F%20%D1%86%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D1%8B%D1%85%20%D0%BF%D1%80%D0%BE%D1%82%D0%BE%D1%82%D0%B8%D0%BF%D0%BE%D0%B2%20%D0%BF%D1%80%D0%BE%D0%BC%D1%8B%D1%88%D0%BB%D0%B5%D0%BD%D0%BD%D1%8B%D1%85%20%D0%B8%D0%B7%D0%B4%D0%B5%D0%BB%D0%B8%D0%B9> Дата обращения (17.12.2023)
4. UML Wikipedia [Электронный курс]. Режим доступа - <https://ru.wikipedia.org/wiki/UML> Дата обращения (18.12.2023)