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)

«Построение болта в системе КОМПАС-3D v.21»

Пояснительная записка по дисциплине

«Основы разработки САПР» (ОрСАПР)

Выполнил:

Студент гр.589-2

\_\_\_\_\_\_\_Кумарбеков Н.М.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2022 г.

Руководитель:

к.т.н., доцент каф. КСУП

\_\_\_\_\_\_\_Калентьев А.А.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2022 г.

**РЕФЕРАТ**

Лабораторная работа 29 с., 17 рис., 9 источников.

Ключевые слова: КОМПАС-3D, VISUAL STUDIO, C#, КОМПАС API, ПЛАГИН, САПР.

Целью данной работы является разработка плагина для автоматизации проектирования модели болта.

Результатом работы является плагин, осуществляющий построение болта по заданным пользователям параметрам.

Пояснительная записка выполнена в текстовом редакторе Microsoft Word 2016.

# 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 [1].

Таким образом, целью данной работы является разработка плагина, автоматизирующего построение модели «Болта» для системы автоматизированного проектирования КОМПАС-3D с помощью интегрированной среды разработки Visual Studio 2022 Сommunity. [2]

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.

# Описание API

Аббревиатура ***API*** расшифровывается как «Application Programming Interface» (интерфейс программирования приложений, программный интерфейс приложения). [3]

Сегодня встречаются задачи, решение которых не реализованы в CAD-системах. Чаще всего это очень узкоспециализированные задачи, которые встречаются на каком-то конкретном предприятии или подотрасли. Для решения подобных задач и их автоматизации используется API.

***API КОМПАС-3D*** — это ориентированные на прикладного программиста инструментальные средства разработки приложений (библиотек конструктивов, прикладных САПР) на базе системы КОМПАС. API КОМПАС-3D включает в свой состав API 5 и API 7.

Главным интерфейсом API системы КОМПАС является ***KompasObject.***[4] Методы этого интерфейса, реализуют наиболее общие функции работы с документами системы, системными настройками, файлами, а также дают возможность получить указатели на другие интерфейсы (интерфейсы динамического массива, работы с математическими функциями, библиотек моделей или фрагментов и различных структур параметров определенного типа).

Ниже в таблице 1.1 представлены основные свойства и методы интерфейса KompasObject.

Таблица 2.1 – Методы и свойства интерфейса KompasObject

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание |
| Document3D() |  | ksDocument | Метод для получения указателя на интерфейс трехмерного графического документа (детали или сборки) |

Продолжение таблицы 2.1

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание |
| GetParamStruct(short structType) | structType – тип интерфейса параметров | StructType2D | Метод для получения указателя на интерфейс графического документа (чертежа или фрагмента) |
| Visible |  | bool | Свойство видимости приложения |
| Quit() |  |  | Метод для закрытия активного окна приложения |

В таблице 2.2 представлены методы интерфейса ksEntity, необходимые для разработки плагина.

Таблица 2.2 – Методы интерфейса ksEntity

|  |  |  |
| --- | --- | --- |
| Название | Тип возвращаемых данных | Описание |
| Create() | bool | Создать объект в модели |
| GetDefinition() | IUnkown | Получить указатель на интерфейс параметров объектов и элементов |
| Update() | bool | Изменить свойства объекта (используя ранее установленные свойства) |
| NewEntity (short objType) | указатель на интерфейс ksEntity или IEntity. | Реальный объект создается в модели после вызова метода [Create](mk:@MSITStore:D:\KOMPAS\SDK\SDK.chm::/ksEntity_Create.htm). |

В таблице 2.3 представлены свойства и методы интерфейса ksDocument2D, необходимые для разработки плагина.

Таблица 2.3 – Методы интерфейса ksDocument2D

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание |
| ksRectangle(  ksRectangleParam param, int style) | param – параметры прямоугольника.  style – стиль линии. | int | Получить указатель на прямоугольник на двумерной плоскости либо 0 в случае ошибки |
| ksCircle(  double xc, double yc, double rad, int style) | xc, yc – координаты центра окружности.  rad – радиус окружности.  style – стиль линии. | int | Получить указатель на окружность на двумерной плоскости либо 0 в случае ошибки |

В таблице 2.4 представлены свойства и методы интерфейса ksDocument3D, необходимые для разработки плагина.

Таблица 2.4 – Методы интерфейса ksDocument3D

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание | |
| Create (bool invisible, bool \_typeDoc) | invisible – признак режима редактирования документа (true – невидимый режим, false –  видимый режим), typeDoc – тип документа (true – деталь, false – сборка). | bool | Создать документ-модель (деталь или сборку) | |
|  | |  |
| GetPart(int type) | type – тип компонента из перечисления Типы компонентов. | ksPart | | Получить указатель на интерфейс компонента в соответствии с заданным типом |

В таблице 2.5 представлены методы интерфейса ksPart, необходимые для разработки плагина.

Таблица 2.5 – Свойства и методы интерфейса ksPart.

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание |
| EntityCollection(short objType) | objType – тип объектов, содержащихся в массиве. | ksEnintyCollection | Формирует массив объектов и возвращает указатель на его интерфейс |
| GetDefaultEntity(short objType) | objType – тип объекта | ksEntity | Получить указатель на интерфейс объекта, создаваемого системой по умолчанию |
| GetPart(int type) | Type – тип компонента | ksPart | Получить указатель на интерфейс компонента в соответствии с заданным типом |
| NewEntity(short objType) | objType – тип объекта | ksEntity | Создать новый интерфейс объекта и получить указатель на него |

В таблице 2.6 представлены типы объектов документа-модели, необходимые для разработки плагина.

Таблица 2.6 – Некоторые типы объектов документа-модели

|  |  |  |
| --- | --- | --- |
| Идентификатор объекта | Название объекта | Интерфейс параметров |
| o3d\_unknown | Неизвестный (включает все объекты) |  |
| o3d\_planeXOZ | Плоскость XOZ | ksPlaneParam |
| o3d\_planeYOZ | Плоскость YOZ | ksPlaneParam |
| o3d\_planeXOY | Плоскость XOY | ksPlaneParam |
| o3d\_sketch | Эскиз | ksSketchDefinition |
| o3d\_face | Грань | ksFaceDefinition |
| o3d\_bossExtrusion | Приклеивание выдавливанием | ksBossExtrusionDefinition |
| o3d\_cutExtrusion | Вырезать выдавливанием | ksCutExtrusionDefinition |
| o3d\_fillet | Операция "скругление" | ksFilletDefinition |
| o3d\_edge | Ребро | ksEdgeDefinition |
| o3d\_planeOffset | Смещённая плоскость | ksPlaneOffsetDefinition |

# Обзор аналогов

AutoCAD — двух- и трёхмерная система автоматизированного проектирования и черчения, разработанная компанией Autodesk. Первая версия системы была выпущена в 1982 году. AutoCAD и специализированные приложения на его основе нашли широкое применение в машиностроении, строительстве, архитектуре и других отраслях промышленности.

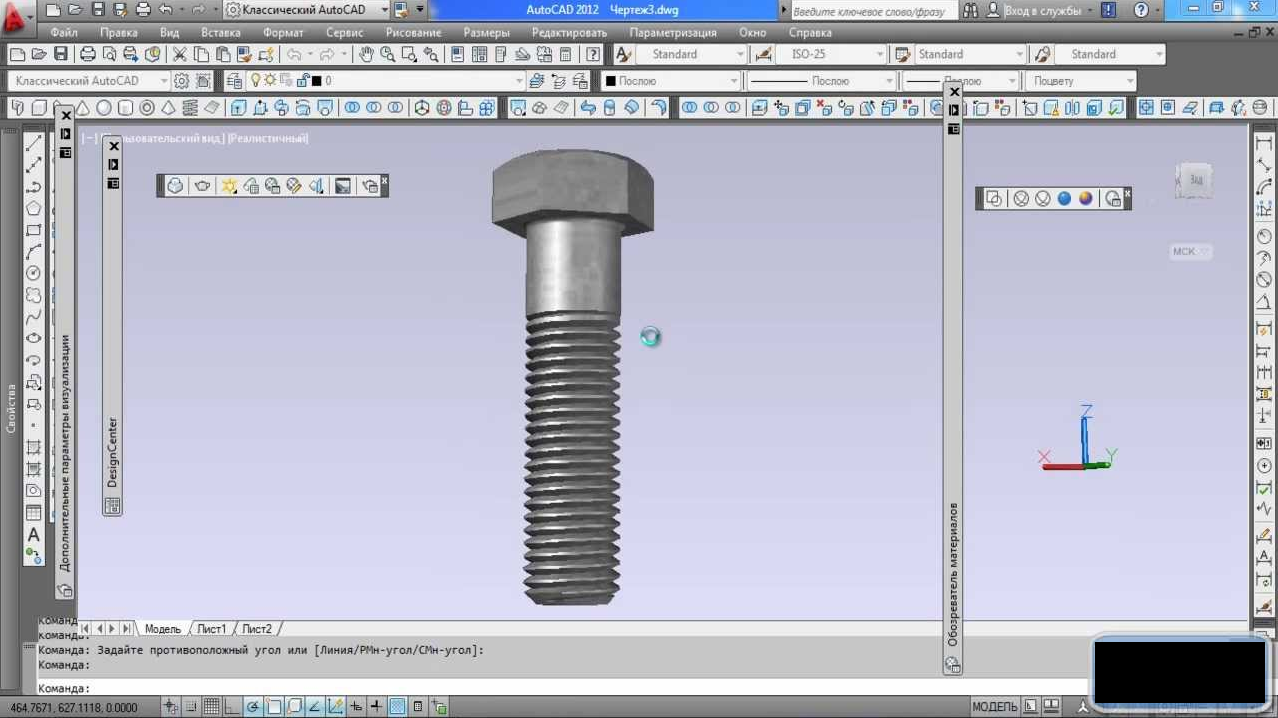


Рисунок 3.1 – Построение болта в AutoCAD.

В данном плагине позволено указывать следующие параметры: Размер под ключ(мм), высота головки(мм), фаска головки(мм), диаметр резьбы(мм), высота стержня(мм), длина резьбы(мм), шаг резьбы(мм). Так же для быстроты выбора стандартных болтов и их основных параметров, можно выбрать нужные для вас болты из предложенного списка.

# Описание предмета проектирования

Предметом проектирования является сборка из модели болта.

Болт – крепёжное изделие в виде стержня с наружной резьбой, как правило, с шестигранной головкой под гаечный ключ, образующее соединение при помощи гайки или иного резьбового отверстия.

Под желание заказчика можно изменять параметры болта, приведенные ниже:

1. Диаметр шляпки D (27≤D≤45) мм;
2. Длина гладкой части I (5≤I≤35) мм;
3. Длина резьбы b (5≤b≤80) мм;
4. Высота шляпки H (6≤H≤20) мм;
5. Глубина выреза m (4≤m≤8) мм;
6. Ширина выреза n (2.5≤m≤9) мм.

На рисунке 4.1 представлен общий вид стакана с крышкой.

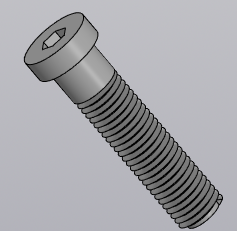


Рисунок 4.1 – Общий вид болта

Чертеж модели с обозначениями показан на рисунке 4.2

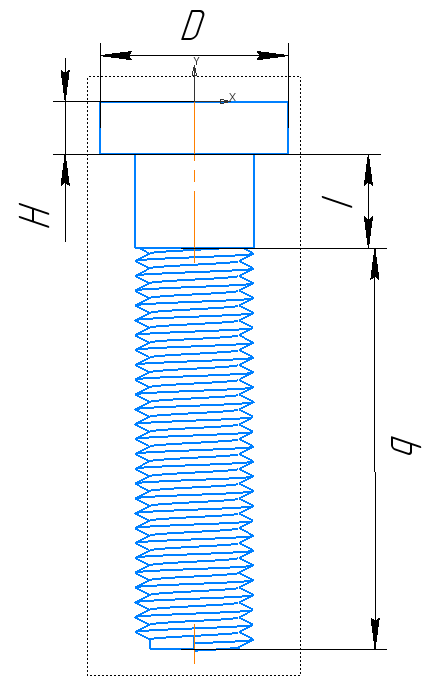


Рисунок 4.2 – Чертеж модели с обозначениями

## Выбор инструментов и средств реализации

На основе требований к техническому заданию программа выполнена на языке программирования C# в среде Microsoft Visual Studio 2022 с использованием .NET Framework 4.7.2 [2], библиотека «Kompas6API5» [6] для основных операций в САПР КОМПАС-3D.

Инструментом тестирования и создания модульных тестов был выбран тестовый фреймворк NUnit [7] версии 3.13.2.

Технология разработки графического интерфейса: Windows Forms [8].

## Назначение плагина

Назначение разрабатываемого плагина обусловлено быстрым моделированием болтов разного вида. Благодаря данному расширению, проектировщики могут наглядно рассмотреть спроектированную модель, при необходимости перестроить под необходимые им параметры.

# Описание реализации

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

UML язык графического описания для объектного моделирования в области 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. UML не является языком программирования, но на основании UML возможна генерация кода и наоборот. [9]

При использовании UML были простроена диаграмма классов.

## 5.1 Диаграмма классов

Диаграмма классов – структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы, их коопераций, атрибутов (полей), методов, интерфейсов и взаимосвязей между ними.

На рисунке 5.1 представлена изначальная диаграмма классов.

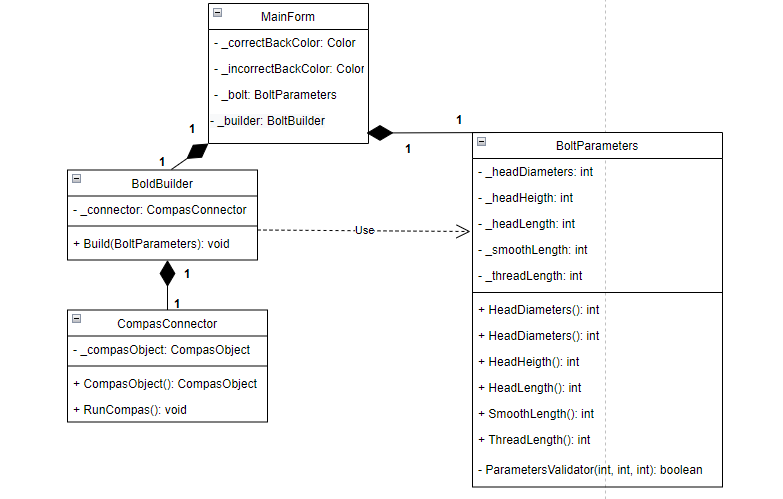


Рисунок 5.1.1 – Изначальная диаграмма классов

1)MainForm – класс диалогового окна, который обеспечивает взаимодействие между пользователем и программой;

2)BoltParametrs − класс, хранящий в себе все параметры 3D-модели;

3)CompasConnector – класс для работы с API КОМПАС 3D.

4)BoltBuilder – класс, осуществляющий вызов методов API, необходимых для постройки 3D-модели.

Класс MainForm использует классы BoltBuilder для отображения построенной детали в главном окне, так же он композирует класс BoltParameters. Класс BoltBuilder использует класс BoltParameters для получения параметров болта, так же он композирует класс CompasConnector, для работы с САПР.

В итоговом проекте созданы следующие классы и методы, которые отображены на итоговой диаграмме классов (рисунок 5.2).

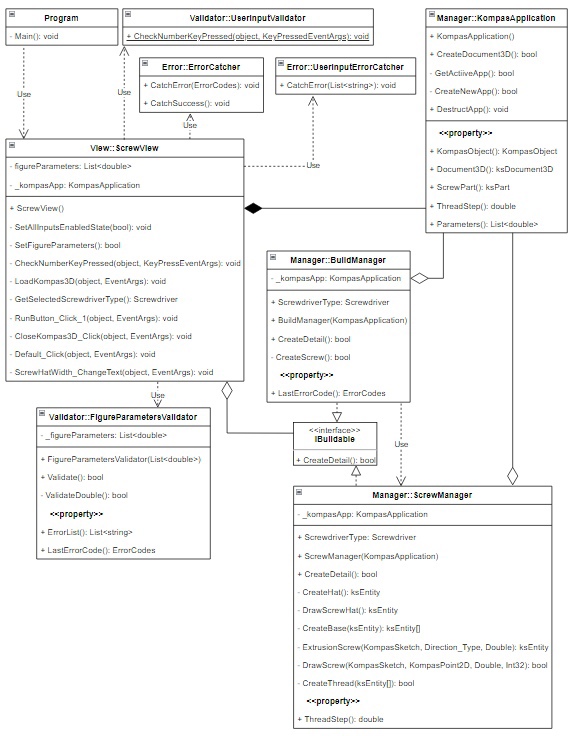


Рисунок 5.1.2 – Итоговая диаграмма классов

В процессе разработки была переосмыслена изначальная диаграмма классов, в результате чего:

В диаграмму классов не была включена структура ErrorCodes, содержащая в себе коды ошибок, так как она используется во всех классах. В приложении А приведены описания данных классов.

# Тестирование программы

На рисунках 6.1 – 6.4 показан болт с минимальными параметрами, болт с максимальными параметрами и болт с шестигранной головкой с максимальными и минимальными параметрами. При построении болта с шестигранной головкой, Глубина выреза и Ширина выреза отсутствуют.

Минимальные параметры:

1. Диаметр шляпки D = 27
2. Длина гладкой части I = 5
3. Длина резьбы b = 5
4. Высота шляпки H = 6
5. Глубина выреза m = 4
6. Ширина выреза n = 2.5

Максимальные параметры:

1. Диаметр шляпки D = 45
2. Длина гладкой части I = 35
3. Длина резьбы b = 80
4. Высота шляпки H = 20
5. Глубина выреза m = 8
6. Ширина выреза n = 9

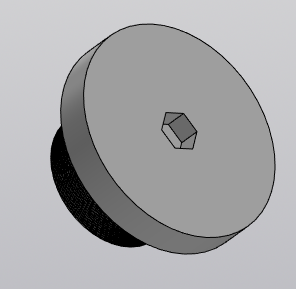


Рисунок 6.1 – Болт с минимальными параметрами

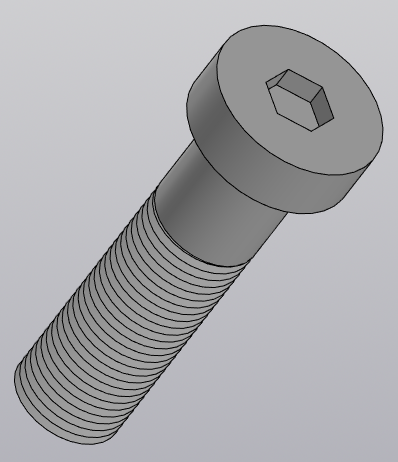


Рисунок 6.2 – Болт с максимальными параметрами

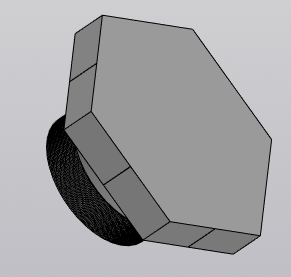


Рисунок 6.3 – Болт с шестигранной головкой с минимальными параметрами

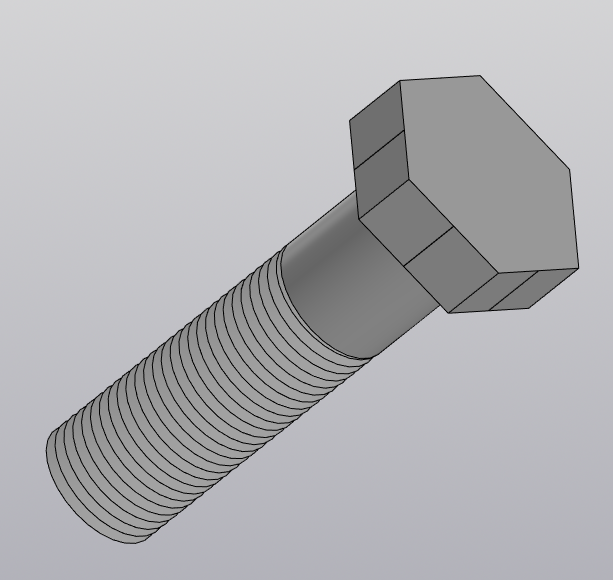


Рисунок 6.4 – Болт с шестигранной головкой с максимальными параметрами

## Unit Тестирование

Далее было произведено юнит-тестирование плагина.

Юнит-тестирование (англ. «unit-testing», или модульное тестирование) — тестирование отдельного элемента изолированно от остальной системы. Относительно парадигмы объектно-ориентированного программирования системой является вся программа, а отдельным элементом — класс или его метод. Юнит-тестирование предназначено для проверки правильности работы отдельно взятого элемента. Чтобы исключить из результатов тестирования влияние потенциальных ошибок других элементов, тестируемый элемент должен быть максимально изолирован, то есть не использовать объекты и методы других классов [10].

На основе таблицы приведенных в приложении А тестовых сценариев (таблица А.1), проводилось тестирование корректности входных параметров 3D-модели.

Ниже, на рисунке 6.1.1, представлен список тестов.

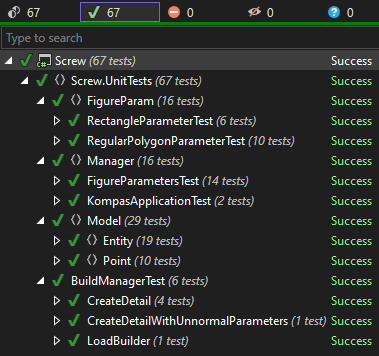


Рисунок 6.1.1 – Список тестов

Результат покрытия Параметров приведен на рисунке 6.1.2.

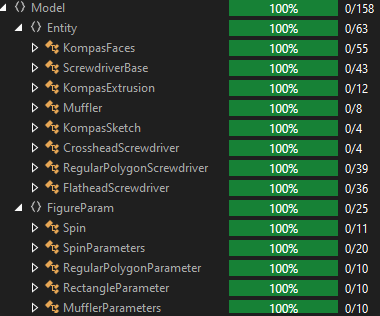


Рисунок 6.1.2 – Степень покрытия тестами бизнес-логики плагина

# Описание программы для пользователя

Пользовательский интерфейс представляет собой форму для ввода параметров. В запущенном окне нажать на кнопку «Load», слева от которой располагается надпись «Load Kompas Application», чтобы запустить приложение «КОМПАС-3D» или сделать его активным, в случае если оно уже запущено. Так же на форме представлен рисунок моделируемого объекта. Пользователь вводит значения самостоятельно, либо установить стандартные значения. Ввод проверяется на соответствие значения диапазону, отображенному рядом с заполняемым полем, а также на корректность ввода, то есть ввод других символов, кроме цифр. При вводе некорректных значений программа выдаст сообщение об ошибке, в котором будут указаны все некорректные значения с пояснениями и возможными диапазонами корректных значений. При переходе на построение болта с шестигранной головкой, поля с параметрами выреза исчезают. После корректного ввода всех значений, нужно нажать на кнопку «Build screw», чтобы создать сборку болта на рабочей плоскости программы КОМПАС-3D.

Макет пользовательского интерфейса программы, а также примеры отображения при вводе представлен на рисунках 7.1 – 7.4

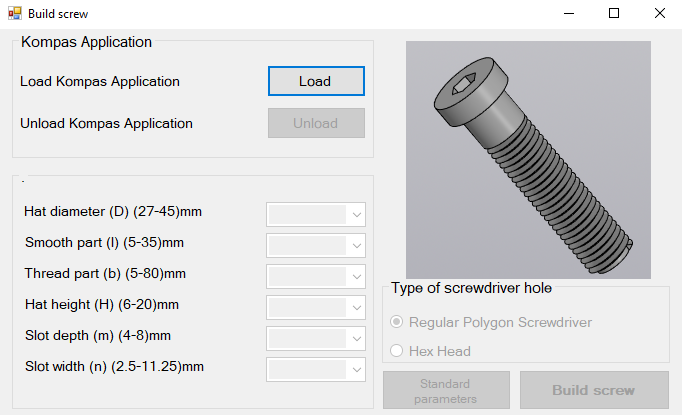


Рисунок 7.1 – Окно плагина без запущенной программы «КОМПАС-3D»

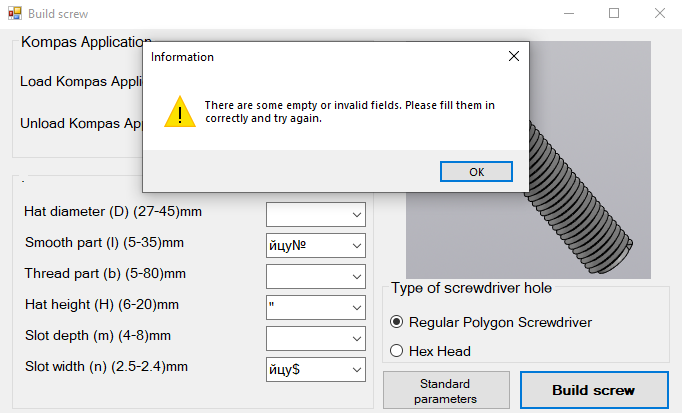


Рисунок 7.2 – Ошибка проверки при пустых ячейках ввода параметров или ввода других символов

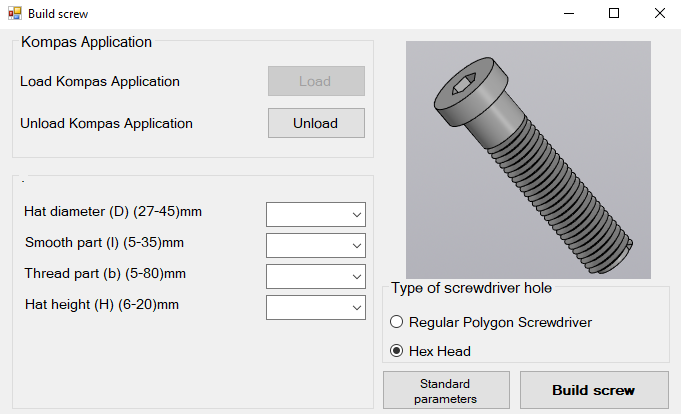


Рисунок 7.3 – Окно плагина при выборе болта с шестигранной головкой

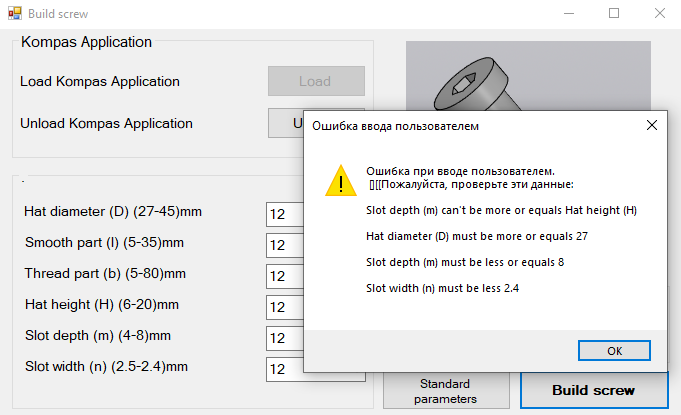


Рисунок 7.4 – Ошибка ввода недопустимых данных

# Нагрузочное тестирование

В целях проверки производительности работы плагина, было проведено нагрузочное тестирование [8]. Тестирование производилось на ПК со следующей конфигурацией:

* ЦП Intel Core i5-7100U;
* 8 ГБ ОЗУ;
* Графический процессор объемом памяти 2 ГБ.

При задании максимального количества последовательно строящихся сборок равным 1000, приложение «КОМПАС-3D» завершилось со сбоем на этапе построения 93 сборки по счёту из-за нехватки памяти; при этом на момент сбоя количество потребляемой приложением памяти было в районе 753 мегабайт. Поэтому максимальным количеством для нагрузочного тестирования было выбрано 90 последовательно строящихся сборок.

На рисунках 8.1 и 8.2 представлены графики зависимости количества непрерывно строящихся деталей (ось OX) от объема памяти в MB, используемой приложением «КОМПАС-3D» – mem и уровень загрузки одного ядра ЦП процессом в процентах – cpu (ось OY).

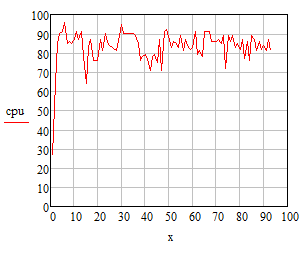


Рисунок 8.1 – График зависимости количества последовательно строящихся деталей (ось OX) от загрузки одного ядра ЦП процессом (ось OY), %

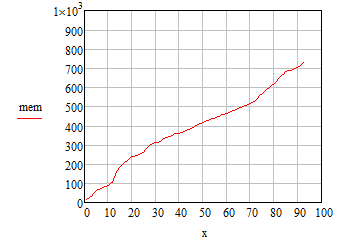


Рисунок 8.2 – График зависимости количества последовательно строящихся деталей (ось OX) от объёма памяти (ось OY), Мб

Исходя из результатов тестирования, можно сделать вывод, что программа выполняет заявленную функциональность в полном объёме. Использовать её для построения сразу нескольких моделей и удержания их в памяти не представляется возможным, так как среда «КОМПАС-3D» V.20 не выдерживает нагрузки подобного рода. Рекомендуется строить модель болта для разового применения.

# Заключение

В результате выполнения работы в рамках курса «Основы разработки систем автоматизированного проектирования» была разработана программа-плагин для системы «КОМПАС-3D», выполняющая построение трёхмерной модели болта.

Плагин выполнен в виде отдельного приложения, подключающегося к системе «КОМПАС-3D» и отдающего ему команды на построение модели. Заявленная функциональность реализована полностью, включая дополнительную функциональность. Тестирование программы проведено в три этапа: функциональное, модульное и нагрузочное, по итогам которого были проверены основные функции программы и защита от ввода некорректных данных, и измерено влияние количества одновременно открытых построенных моделей на расход оперативной памяти компьютера.

# Список использованных источников

1. Автоматизация вычислительных процедур в прикладных задачах инженерного проектирования [Электронный ресурс]. – URL: https://scienceforum.ru/2014/article/2014000201 (дата обращения: 10.12.2022).
2. Visual Studio [Электронный ресурс]. – URL: https://visualstudio.microsoft.com/ru/ (дата обращения: 10.12.2022).
3. Профессиональный софт для работы с 3D моделями. [Электронный ресурс] — www.tflex.ru/products/konstructor/cad3d/index.php / (дата обращения: 29.10.2022).
4. КОМПАС-3D для разработчиков. [Электронный ресурс]. – URL: https://kompas.ru/solutions/developers/ (дата обращения: 10.12.2022).
5. NUnit [Электронный ресурс]. – URL: https://nunit.org/ (дата обращения: 10.12.2022).
6. Введение в Windows Forms [Электронный ресурс]. – URL: https://metanit.com/sharp/windowsforms/1.1.php (дата обращения: 10.12.2022).
7. UML. [Электронный ресурс]. – Режим доступа: http://www.uml.org/ (дата обращения 10.12.2022).
8. Функциональное тестирование [Электронный ресурс]. – URL: https://daglab.ru/funkcionalnoe-testirovanie-programmnogo-obespechenija/ (дата обращения: 10.12.2022).
9. Виртуальная память — Википедия. [Электронный ресурс] — Режим доступа: https://ru.wikipedia.org/wiki/Виртуальная память (дата обращения 17.12.2022)

# Приложение А

(справочное)

Описания классов

В таблицах приложения для обозначения модификаторов доступа полей приняты следующие условные знаки:

- «#» − обозначение protected (защищенного) поля,

- «−» − обозначение private (закрытого) поля,

- «+» − обозначение public (открытого) поля.

Таблица A.1 – Описание класса ScrewView

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс ScrewView – сущность для описания пользовательского интерфейса | | |
| Поля | | |
| - \_buildManager | BuildManager | Переменная для объекта менеджера сборки |
| - \_figureParameters | List<double> | Переменная для хранения значений, введённых пользователем на форме параметров болта |
| - \_kompasApp | KompasApplication | Переменная для хранения объекта сущности для работы с приложением «КОМПАС-3D» |
| Методы | | |
| + ScrewView() |  | Конструктор класса |
| - RunButton\_Click\_1(object, EventArgs) | void | Событие при нажатии на кнопку «Build screw» |
| * LoadCompas3D(object, EventArgs) | void | Событие при нажатии на кнопку «Load» (кнопка, запускающая приложение КОМПАС) |

Окончание таблицы А.1

|  |  |  |
| --- | --- | --- |
| * CloseCompas3D(object, EventArgs) | void | Событие при нажатии на кнопку «Unload» (кнопка, закрывающая запущенное приложение «КОМПАС») |
| * SetFigureParameters() | bool | Проверка на ввод параметров болта и установка переменной \_figureParameters параметрами, введёнными пользователем; в случае неудачной проверки выброс ошибки |
| * CheckNumberKeyPressed(object, KeyPressEventArgs) | void | Событие при нажатии на клавиши клавиатуры пользователя во время ввода параметров болта |

Таблица A.2 – Описание класса KompasApplication

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс KompasApplication – сущность для работы с приложением «КОМПАС-3D» | | |
| Свойства | | |
| + Document3D | ksDocument3D | Переменная для хранения трёхмерного документа приложения ««КОМПАС-3D» |
| + KompasObject | KompasObject | Переменная для хранения объекта приложения «КОМПАС-3D» |

Окончание таблицы А.2

|  |  |  |
| --- | --- | --- |
| + ScrewPart | ksPart | Переменная для хранения объекта тела болта в приложении «КОМПАС-3D» |
| + Parameters | List<double> | Переменная для хранения параметров болта |
| +ThreadStep | Double | Переменная для хранения шага резьбы |
| + LastErrorCode | ErrorCodes | Переменная для хранения последнего кода ошибки экземпляра объекта |
| Методы | | |
| + CreateDocument3D() | bool | Метод для создания трёхмерного документа в приложении КОМПАС |
| - CreateNewApp() | bool | Метод для создания нового приложения «КОМПАС-3D» |
| + DestructApp() | void | Метод для закрытия уже запущенного приложения «КОМПАС-3D» |
| - GetActiveApp() | bool | Метод для подключения к уже запущенному приложению «КОМПАС-3D» |
| + KompasApplication() | bool | Конструктор класса |

Таблица А.3 – Описание класса BuildManager

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс BuildManager – сущность-менеджер для последовательного построения сборки болта | | |
| Поля | | |
| - \_kompasApp | KompasApplication | Переменная для хранения объекта сущности для работы с приложением «КОМПАС-3D» |
| Свойства | | |
| + LastErrorCode | ErrorCodes | Переменная для хранения кода последней ошибки экземпляра класса |
| Методы | | |
| + BuildManager(  KompasApplication) |  | Конструктор класса |
| + CreateDetail() | bool | Метод создания детали менеджером: вызов метода CreateScrew, в процессе проверяющий последние на код ошибки |
| - CreateScrew() | bool | Метод, создающий экземпляр класса ScrewManager, в процессе строящий деталь болта |

Таблица А.4 – Описание класса ScrewManager

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс ScrewManager – сущность-менеджер для построения детали болта | | |
| Поля | | |
| - \_kompasApp | KompasApplication | Переменная для хранения объекта сущности для работы с приложением «КОМПАС-3D» |
| Свойства | | |
| + LastErrorCode | ErrorCodes | Переменная для хранения кода последней ошибки экземпляра класса |
| Методы | | |
| + ScrewManager(  KompasApplication) |  | Конструктор класса |
| + CreateDetail() | bool | Метод создания детали |
| - CreateBase(ksEntity) | ksEntity[] | Метод создания базовой части болта |
| - CreateThread(  ksEntity[]) | bool | Метод создания резьбовой части болта |
| - CreateHat() | ksEntity | Метод создания шляпки болта |