Министерство образования и науки 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)

**РАЗРАБОТКА ПЛАГИНА «БОЛТ С ГАЙКОЙ» ДЛЯ САПР «КОМПАС-3D v15»**

Пояснительная записка к курсовому проекту

по дисциплине «Основы разработки САПР» на тему

«Построение болта с гайкой в системе КОМПАС 3D v15»

Студент гр.583-1

\_\_\_\_\_\_ И.К. Харченко

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_2016 г.

К.т.н., м.н.с. ЛИКС, каф. КСУП

\_\_\_\_\_\_ А.А. Калентьев

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_2016 г.

Томск 2016

**Реферат**

Курсовой проект 44 страницы, 23 иллюстраций, 12 таблиц, 12 использованных источников, 1 приложение.

ПЛАГИН, БОЛТ С ГАЙКОЙ, САПР, КОМПАС-3D, ПРОЕКТИРОВАНИЕ, ПРОЕКТНАЯ ДОКУМЕНТАЦИЯ, API.

Объектом разработки является плагин «Болт с гайкой» для САПР «КОМПАС-3D».

Цель работы – создание плагина для построения трехмерной сборки деталей (болт с гайкой) по введенным параметрам в «КОМПАС-3D».

В процессе выполнения курсового проекта разработан плагин для построения болта с гайкой в рабочей плоскости САПР «КОМПАС-3D», а также проектная документация к плагину.

Плагин предназначен для построения трехмерной модели болта с гайкой. Взаимодействие с пользователем производится с помощью диалогового окна.

Содержание

[Введение 4](#_Toc477703882)

[1 Постановка и анализ задачи 5](#_Toc477703883)

[1.1 Описание САПР 5](#_Toc477703884)

[1.2 Описание предмета проектирования 12](#_Toc477703885)

[1.3 Описание технических и функциональных аспектов проекта 13](#_Toc477703886)

[2 Описание программы для пользователя 16](#_Toc477703887)

[3 Тестирование программы 17](#_Toc477703888)

[3.1 Функциональные тесты 17](#_Toc477703889)

[3.2 Конфигурационные тесты 24](#_Toc477703890)

[3.3 Модульные тесты 25](#_Toc477703891)

[3.4 Нагрузочные тесты 29](#_Toc477703892)

[4 Заключение 32](#_Toc477703893)

[Список литературы 33](#_Toc477703894)

[Приложение А 34](#_Toc477703895)

# Введение

В настоящее время проектирование в своем понимании представляет собой автоматизированный процесс и в некотором роде программно-аппаратный. Проектировщику, который занимается разработкой сложного механизма, или устройства, требующего больших расчетов, математических вычислений при построении модели и высокой точности, подходят системы автоматизации проектных решений — САПР [1].

САПР позволяют уменьшить финансовые затраты на разработку макета (модели) проекта (объекта), а также сократить время, которое тратит проектировщик на создание модели объекта и составление проектной документации.

В каждой крупной САПР есть свой средства для разработки, которые предоставляются с целью дать возможность разработчикам расширить функционал данной системы под свои конкретные нужды. Данным средством является API — программируемый интерфейс приложения [2]. Это набор готовых средств: классов, процедур, функций, структур и т.д. API позволяет определить функциональность, которую предоставляет приложение, при этом абстрагируясь от того, как она реализована.

Расширение функционала в основном подразумевает разработку плагина или библиотеки на основе предоставленного API. В данном курсовом проекте стоит задача разработки плагина для построения 3D модели болта с гайкой в автоматизированном режиме. Плагин — независимо компилируемый программный модуль, динамически подключаемый к основной программе, предназначенный для расширения или использования ее возможностей [3].

В качестве системы, которая предоставляет API и для которой стоит задача разработать плагин, была взята САПР «КОМПАС-3D» версии 15.

# Постановка и анализ задачи

## Описание САПР

* + 1. **Описание программы**

КОМПАС-3D — система трехмерного проектирования, ставшая стандартом для тысяч предприятий, благодаря сочетанию простоты освоения и легкости работы с мощными функциональными возможностями твердотельного и поверхностного моделирования [4]. Ключевой особенностью продукта является использование собственного математического ядра С3D и параметрических технологий, разработанных специалистами АСКОН. 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.

* + 1. Описание API

В КОМПАС на данный момент существуют API двух версий: API 5 и API 7. [5]. Обе версии реализуют различные функции системы и взаимно дополняют друг друга. Отсюда очевидно, что обе версии программных интерфейсов в равной мере поддерживаются и развиваются с учетом самих изменений в системе. В основном, для создания полноценных подключаемых модулей достаточно методов и свойств интерфейсов API 5.

Главным интерфейсом API системы КОМПАС является KompasObject. Получить указатель на этот интерфейс (если быть точным, на интерфейс приложения API 5) можно при работе под управлением внешнего приложения (контроллера) – после вызова стандартной системной функции. 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 свойства и методы интерфейса KompasObject, которые были использованы при разработке плагина.

Таблица 1.1 – Методы и свойства интерфейса KompasObject, используемые при разработке плагина

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Document3D() | ksDocument | Метод для получения указателя на интерфейс трехмерного графического документа (детали или сборки) |
| GetParamStruct(short structType) | [StructType2D](mk:@MSITStore:C:\Program%20Files\ASCON\KOMPAS-3D%20V16\SDK\SDK.chm::/StructType2D.htm) | Метод для получения указателя на интерфейс графического документа (чертежа или фрагмента) |
| Visible | bool | Свойство видимости приложения |
| Quit() |  | Метод для закрытия активного окна приложения КОМПАС |

В таблице 1.2 представлены методы интерфейса ksEntity, которые были использованы при разработке плагина.

Таблица 1.2 – Методы интерфейса ksEntity, используемые при разработке плагина

|  |  |  |
| --- | --- | --- |
| Название | Тип | 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) |

В таблице 1.3 представлены свойства и методы интерфейса ksDocument2D, которые были использованы при разработке плагина.

Таблица 1.3 – Методы интерфейса ksDocument2D, используемые при разработке плагина

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| ksLineSeg(double x1, double y1, double x2, double y2, int style) | int | Получить указатель на отрезок на двумерной плоскости либо 0 в случае ошибки |
| ksRegularPolygon(  ksRegularPolygonParam param, int style) | int | Получить указатель на многоугольник на двумерной плоскости либо 0 в случае ошибки |
| ksRectangle(  ksRectangleParam param, int style) | int | Получить указатель на прямоугольник на двумерной плоскости либо 0 в случае ошибки |
| ksCircle(  double xc, double yc, double rad, int style) | int | Получить указатель на окружность на двумерной плоскости либо 0 в случае ошибки |

В таблице 1.4 представлены свойства и методы интерфейса ksDocument3D, которые были использованы при разработке плагина.

Таблица 1.4 – Методы интерфейса ksDocument3D, используемые при разработке плагина

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Create (bool invisible, bool \_typeDoc) | bool | Создать документ-модель (деталь или сборку) |
| GetPart(int type) | ksPart | Получить указатель на интерфейс компонента в соответствии с заданным типом |

В таблице 1.5 представлены методы интерфейса ksPart, которые были использованы при разработке плагина.

Таблица 1.5 – Свойства и методы интерфейса ksPart, используемые при разработке плагина

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

В таблице 1.6 представлены типы объектов документа-модели, которые были использованы при разработке плагина.

Таблица 1.6 – 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](mk:@MSITStore:C:\Program%20Files\ASCON\KOMPAS-3D%20V16\SDK\SDK.chm::/ksSketchDefinition.htm) |
| o3d\_face | Грань | [ksFaceDefinition](mk:@MSITStore:C:\Program%20Files\ASCON\KOMPAS-3D%20V16\SDK\SDK.chm::/ksEdgeDefinition.htm) |
| o3d\_baseExtrusion | Базовая операция выдавливания | [ksBaseExtrusionDefinition](mk:@MSITStore:C:\Program%20Files\ASCON\KOMPAS-3D%20V16\SDK\SDK.chm::/ksBaseExtrusionDefinition.htm) |
| o3d\_cutExtrusion | Вырезать выдавливанием | [ksCutExtrusionDefinition](mk:@MSITStore:C:\Program%20Files\ASCON\KOMPAS-3D%20V16\SDK\SDK.chm::/ksCutExtrusionDefinition.htm) |
| o3d\_baseLoft | Создание элемента по сечениям | [ksBaseLoftDefinition](mk:@MSITStore:C:\Program%20Files\ASCON\KOMPAS-3D%20V16\SDK\SDK.chm::/ksBaseExtrusionDefinition.htm) |
| o3d\_baseEvolution | Создание кинематического элемента | [ksBaseEvolutionDefinition](mk:@MSITStore:C:\Program%20Files\ASCON\KOMPAS-3D%20V16\SDK\SDK.chm::/ksBaseExtrusionDefinition.htm) |
| o3d\_cutEvolution | Вырезать кинематический элемент | [ksCutEvolutionDefinition](mk:@MSITStore:C:\Program%20Files\ASCON\KOMPAS-3D%20V16\SDK\SDK.chm::/ksCutExtrusionDefinition.htm) |

* + 1. Обзор аналогов
       1. **Плагин PDF**

Данный плагин позволяет производить экспорт моделей и сборок из КОМПАС-3D в формат PDF формат [6]. 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. Это полезно для подготовки интерактивных сборочных инструкций, создания маркетинговых материалов, презентаций, а также для налаживания взаимодействия между проектировщиками и заказчиками. 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 плагина:

* сохранение деталей и сборок в формате 3D PDF для интерактивного просмотра при помощи бесплатной программы Adobe Reader;
* создание анимаций, 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.
  + - 1. **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 [7].

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, маркетинговых материалов, при подготовке интерактивных сборочных конструкций. Он значительно упрощает взаимодействие между заказчиками и проектировщиками.

Компания Visual Technology Services Ltd. из Великобритании разработала плагин PDF3D, предоставляющий доступ к описанным выше возможностям.

Основной функционал плагина:

* сохранение сборок и деталей для интерактивного просмотра в формате 3D PDF с помощью программы Adobe Reader;
* создание анимации, имитирующей естественный порядок разборки и сборки;
* создание анимации, имитирующей гибки листового тела;
* возможность вставки в PDF документ, содержащий основной текст, фоновых картинок, таблиц, логотипов, эмблем, спецификаций и т.д.

## Описание предмета проектирования

Предметом проектирования является сборка из модели болта и модели гайки.

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.[8]

Гайка – крепёжное изделие с резьбовым отверстием, образующее соединение с помощью винта, болта или шпильки. Обычно гайки изготавливаются шестигранной формы под гаечный ключ, но могут быть и квадратными, круглыми с насечкой, с выступами под пальцы («барашки») или другой формы.[9]

На рисунке 1.1 представлен вид на 3D модель болта.

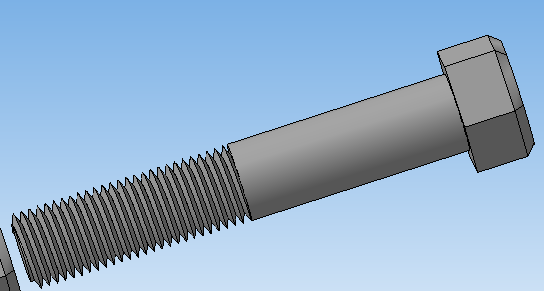


Рисунок 1.1 – Вид на 3D модель болта

На рисунке 1.2 представлен вид на 3D модель гайки.

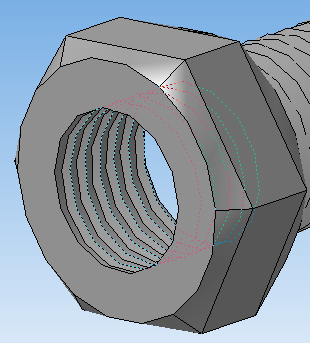


Рисунок 1.2 – Вид на 3D модель гайки

## Описание технических и функциональных аспектов проекта

Для графического описания абстрактной модели проекта, а также пользовательского взаимодействия (сценария действий) использован стандарт UML[10].

UML – это 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-моделей возможна генерация кода.

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

* + 1. Диаграмма вариантов использования (Use Cases)

На рисунке 1.3 представлена диаграмма вариантов использования.

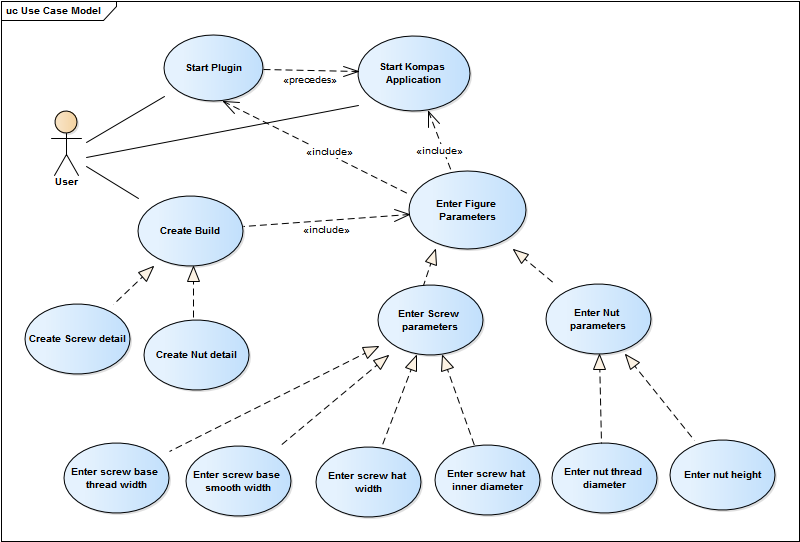


Рисунок 1.3 – Диаграмма вариантов использования

* + 1. Диаграмма классов

На рисунке 1.4 показана диаграмма классов.

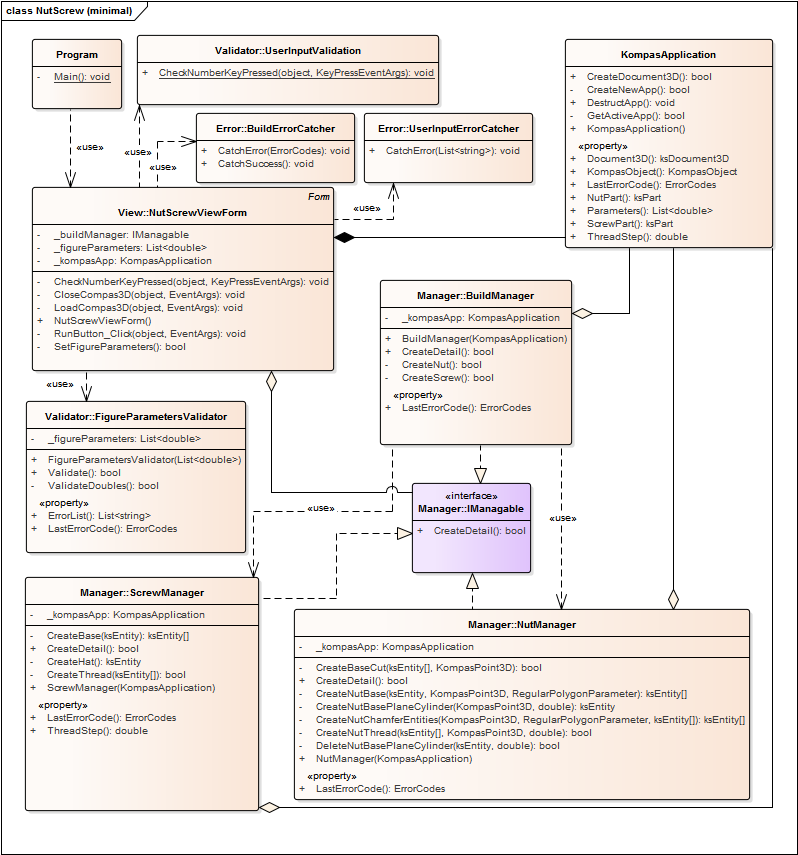


Рисунок 1.4 – Диаграмма классов

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.

* + 1. Макет пользовательского интерфейса

Плагин представляет собой пользовательскую форму с ячейками для ввода параметров. Запуск построения осуществляется кнопкой «Build nut with screw». На рисунке 1.5 представлен интерфейс плагина.

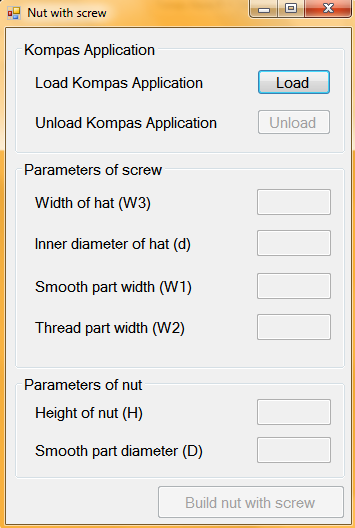


Рисунок 1.5 – Пользовательский интерфейс при запуске программы

# Описание программы для пользователя

Чтобы построить сборку болта с гайкой, используя данный плагин, необходимо запустить плагин. 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. Далее необходимо ввести данные: параметры болта с гайкой. При вводе некорректных значений программа выдаст сообщение об ошибке, в котором будут указаны все некорректные значения с пояснениями и возможными диапазонами корректных значений. После корректного ввода всех значений, нужно нажать на кнопку «Build nut with screw», чтобы создать сборку болта с гайкой на рабочей плоскости программы КОМПАС-3D.

# Тестирование программы

## Функциональные тесты

Функциональное тестирование — это тестирование в целях проверки реализуемости функциональных требований, то есть способности программного обеспечения (ПО) в определённых условиях решать задачи, нужные пользователям. Функциональные требования определяют, что именно делает ПО, какие задачи оно решает. [13]

При запуске плагина открывается окно, показанное на рисунке 3.1. Ячейки для ввода данных недоступны, а строка состояния уведомляет о необходимости запуска программы «КОМПАС-3D» нажатием на кнопку «Load». После нажатия кнопки пользовательский интерфейс изменится на вид, показанный на рисунке 3.2. Если при запуске плагина приложение «КОМПАС-3D» было уже открыто, то при запуске плагина сразу отобразится окно, представленное на рисунке 3.2. Появляется возможность вводить данные – параметры болта и параметры гайки.

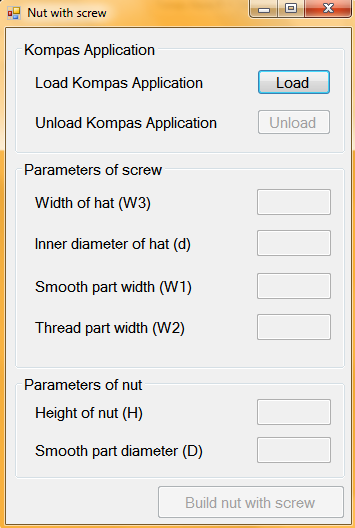


Рисунок 3.1 – Окно плагина без запущенной программы «КОМПАС-3D»

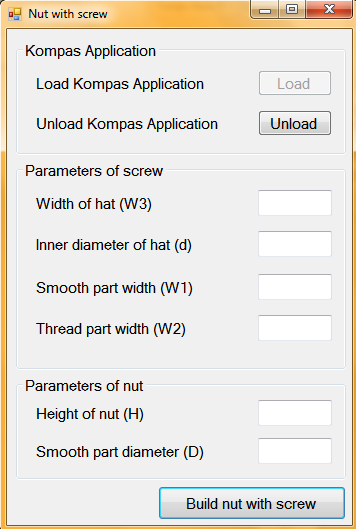


Рисунок 3.2 – Окно плагина с запущенной программой «КОМПАС-3D»

Далее, при попытке ввода неправильных символов в ячейки ввода параметров, а также при пустой ячейке ввода параметров программа выдает ошибку проверки правильности ввода данных (рисунки 3.3, 3.4).

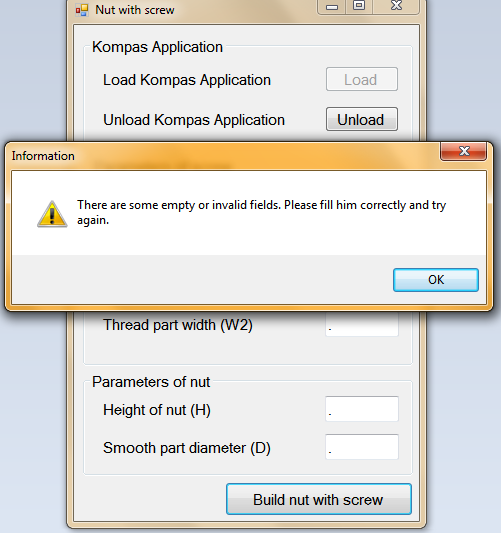


Рисунок 3.3 – Ошибка проверки при вводе неправильных символов в ячейки ввода параметров

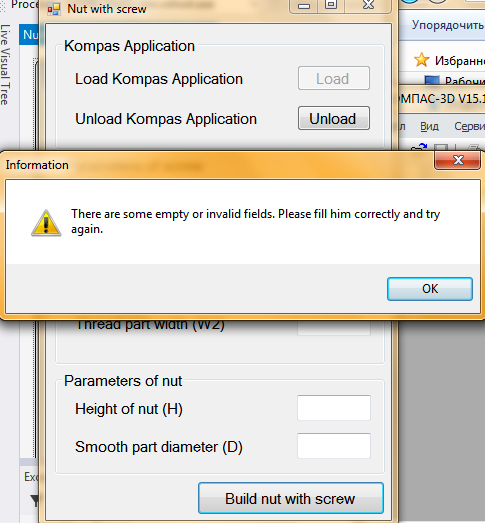


Рисунок 3.4 – Ошибка проверки при пустых ячейках ввода параметров

Тестируется ввод некорректных данных (рисунок 3.5).

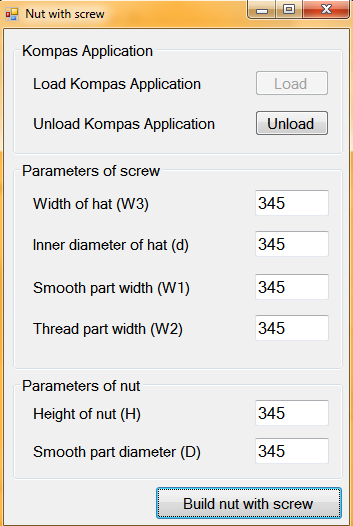


Рисунок 3.5 – Ввод недопустимых данных

При нажатии на кнопку «Build nut with screw» программа откроет окно со списком ошибок, которые были допущены при вводе параметров болта и гайки, а также с диапазонами корректных значений параметра (рисунок 3.6).

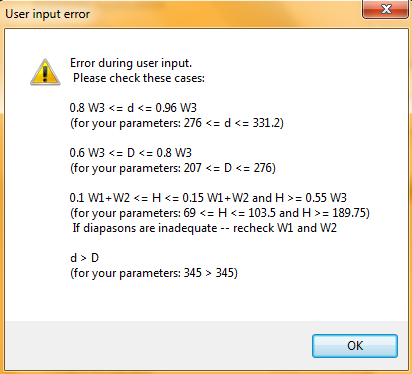
****

Рисунок 3.6 – Ошибка ввода недопустимых значений

Данная ошибка будет выводиться до тех пор, пока пользователь не введет допустимые значения параметров во все ячейки ввода данных (рисунок 3.7).

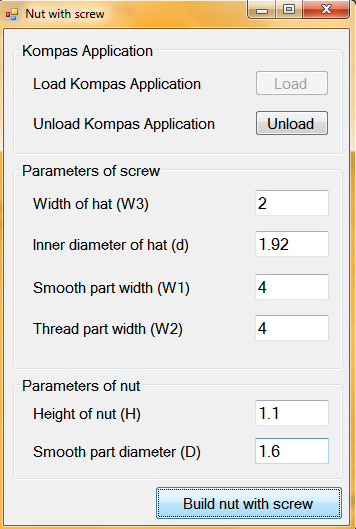


Рисунок 3.7 – Один из допустимых вариантов значения параметров

При вводе корректных данных и нажатии кнопки «Build nut with screw», на рабочей плоскости «КОМПАС-3D» построится модель сборки (рисунок 3.8).

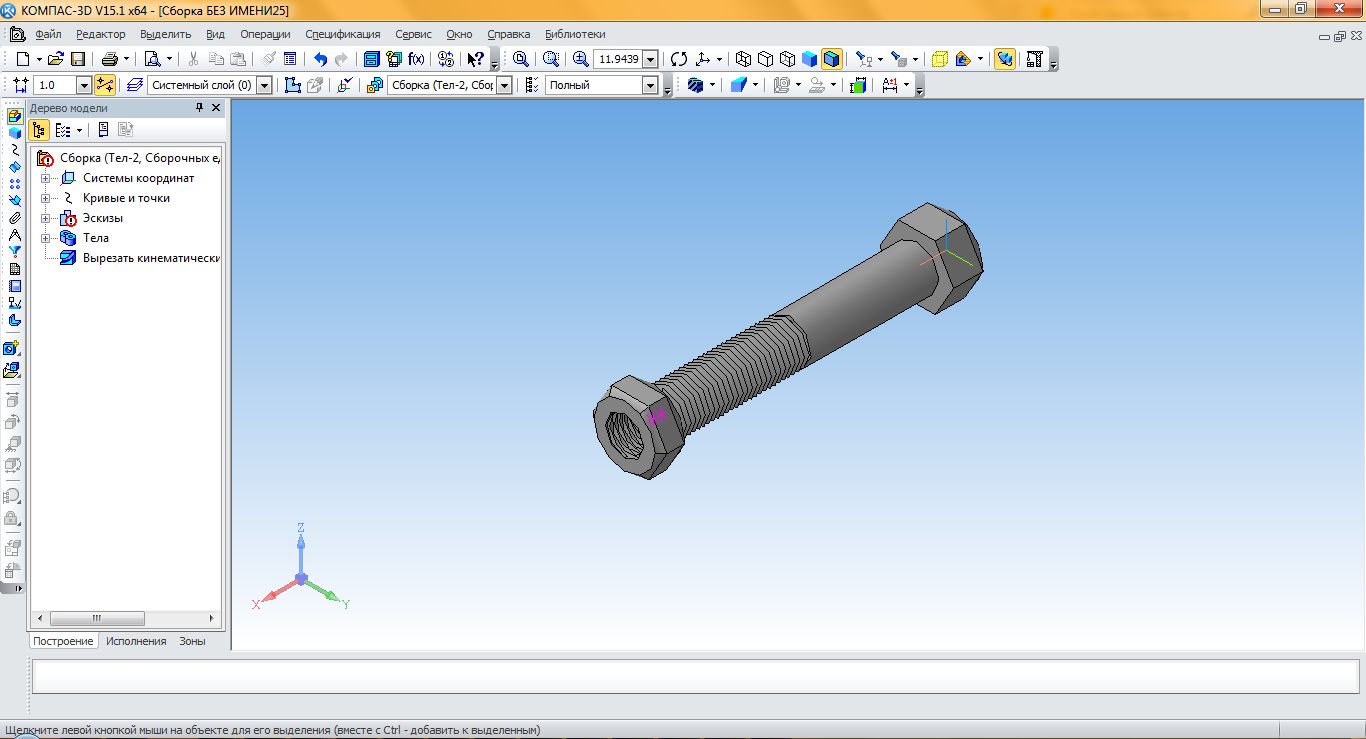


Рисунок 3.8 – Построенная сборка

Ввод минимальных допустимых значений представлен на рисунке 3.9.

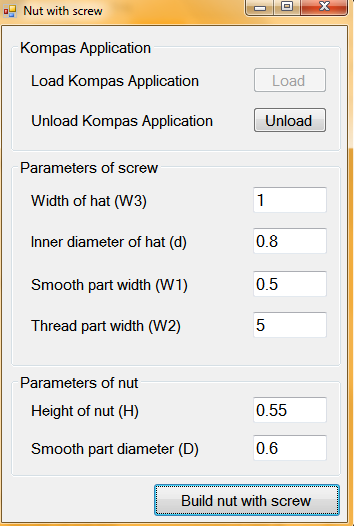


Рисунок 3.9 – Ввод минимально допустимых значений

Результат при вводе минимально допустимых значений представлен на рисунке 3.10.

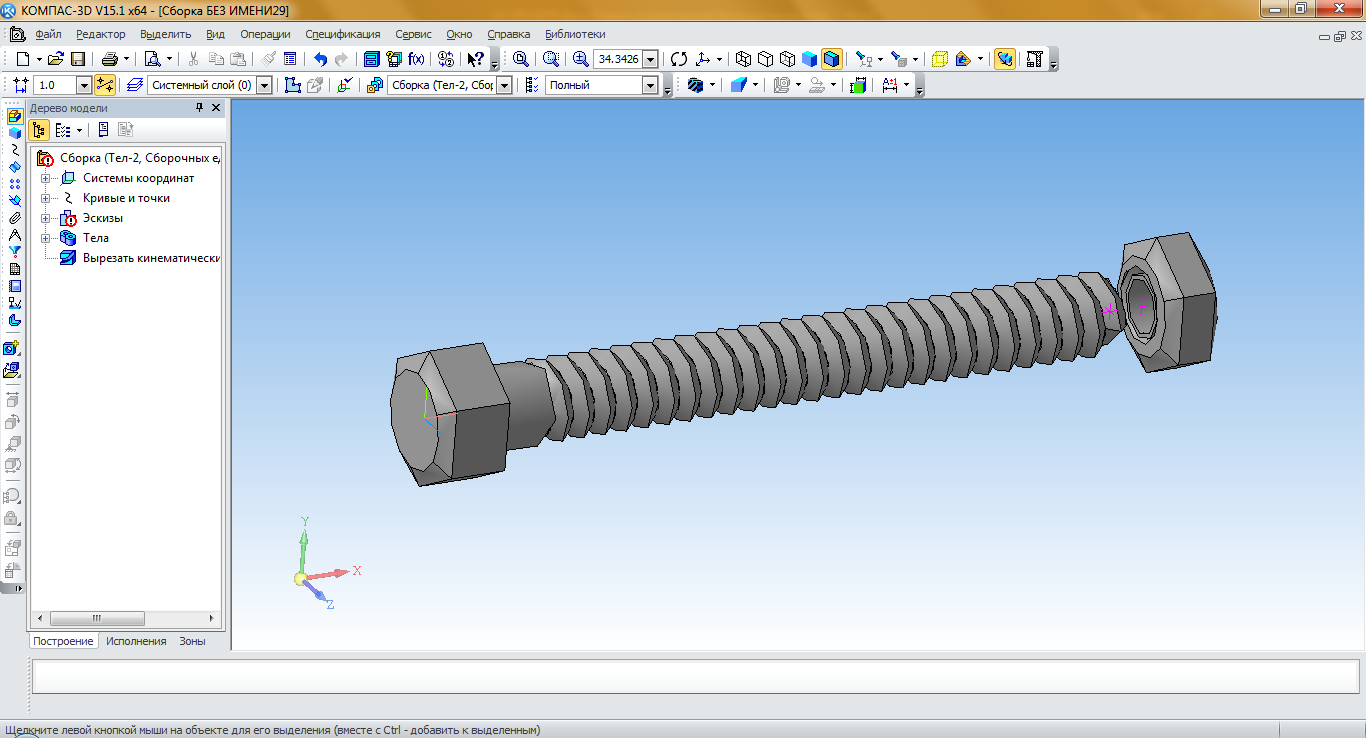


Рисунок 3.10 – Результат ввода минимально допустимых значений

Ввод максимально допустимых значений представлен на рисунке 3.11.

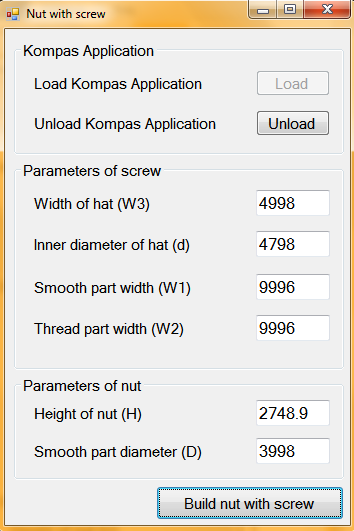


Рисунок 3.11 – Ввод максимально допустимых значений

Результат при вводе максимально допустимых значений показан на рисунке 3.12.

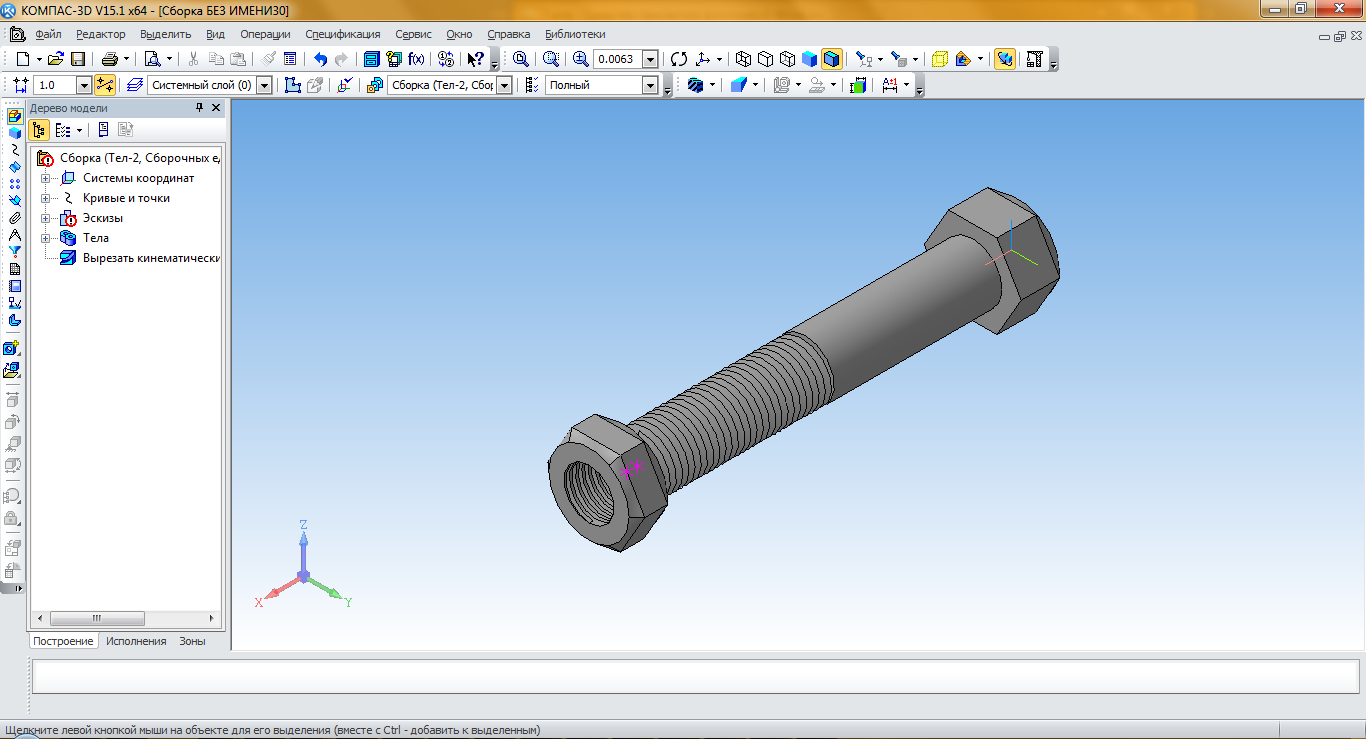


Рисунок 3.12 – Результат ввода максимально допустимых значений

## Конфигурационные тесты

При сборке проекта была выбрана конфигурация «Release» и конечная платформа «x86». После удачной сборки программа была запущена в операционной системе Windows 7 (x86), запущенная поверх основной системы с помощью программной среды виртуализации VirtualBox (версия среды на момент тестирования – 5.1.18) [11]. Результаты запуска и работы программы при заданной конфигурации приведены на рисунке 3.13.

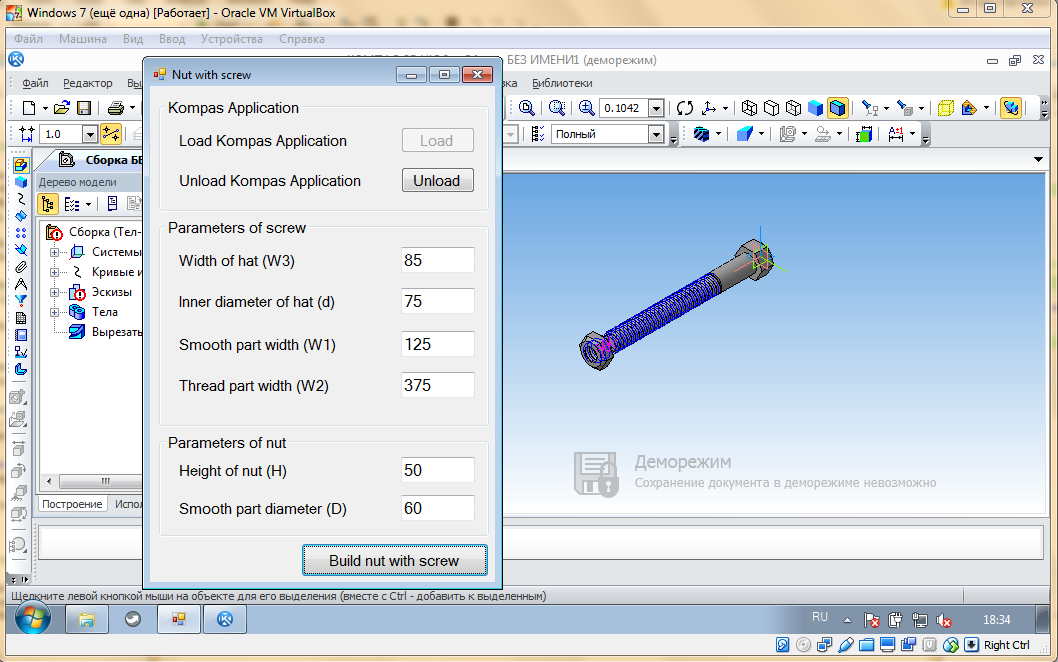


Рисунок 3.13 – Тестирование программы в операционной системе Windows 7 (x86) в среде виртуализации VirtualBox

## Модульные тесты

Список тестовых сценариев для модульного тестирования представлен в таблице 3.1.

Таблица 3.1 – Список тестовых сценариев

|  |  |  |
| --- | --- | --- |
| Название тестового класса | Название тестового метода | Назначение |
| FigureParametersTest | TestValidation(bool expectedResult, params double[] parameters) | Тестирование параметров деталей: болта и гайки. Присвоение корректных и некорректных значений. |
| KompasApplicationTest | TestConstructKompasApplication(  ErrorCodes res, params double[] parameters) | Тестирование создания объекта приложения КОМПАС. |
| TestDestructKompasApplication() | Тестирование удаления объекта приложения КОМПАС. |
| BuildManagerTest | CreateDetail(bool expected, params double[] parameters) | Тестирование создания сборки с деталями. |
| LoadBuilder(int amountOfLoads) | Тестирование создания сборки заданное количество раз. |

Продолжение таблицы 3.1

|  |  |  |
| --- | --- | --- |
| KompasExtrusionTest | TestBaseBySketch(ErrorCodes errorCode, Obj3dType extrusionType, Direction\_Type directionType) | Тестирование операции выдавливания по заданной плоскости |
| TestCutBySketch(ErrorCodes errorCode, Obj3dType extrusionType, Direction\_Type directionType) | Тестирование вырезания выдавливанием по заданной плоскости |
| CutBySketchUnsupportedDirection(  ErrorCodes errorCode, Obj3dType extrusionType, Direction\_Type directionType) | Тестирование присваивания некорректного направления выреза |
| KompasSketchTest | CreateOnBasePlaneAxis(  ErrorCodes errorCode, Obj3dType basePlaneAxis) | Тестирование создания эскиза на заданной базовой плоскости |
| CreateOnNull(ErrorCodes errorCode, ksEntity entity) | Тестирование присвоения некорректного значения базовой плоскости |
| RectangleParameterTest | TestRectangleParameterNormal(  ErrorCodes errorCode, double width, double height) | Тестирование создания класса RectangleParameter. Присвоение корректных и некорректных значений. |

Окончание таблицы 3.1

|  |  |  |
| --- | --- | --- |
| RegularPolygonParameterTest | TestRegularPolygonParameterNormal(  ErrorCodes errorCode, double width, double height) | Тестирование создания класса RegularPolygonParameter. Присвоение корректных и некорректных значений. |
| KompasPoint2D | TestKompasPointNormal(  ErrorCodes errorCode, double xs, double yc) | Тестирование создания класса KompasPoint2D. Присвоение корректных и некорректных значений. |
| KompasPoint3D | TestKompasPointNormal(  ErrorCodes errorCode, double xs, double yc, double zc) | Тестирование создания класса KompasPoint3D. Присвоение корректных и некорректных значений. |

Тестирование проводилось с помощью фреймворка модульного тестирования NUnit 3.6.1 для языков платформы .NET. [12]

Результаты успешного прохождения всех модульных тестов приведены на рисунке 3.14.

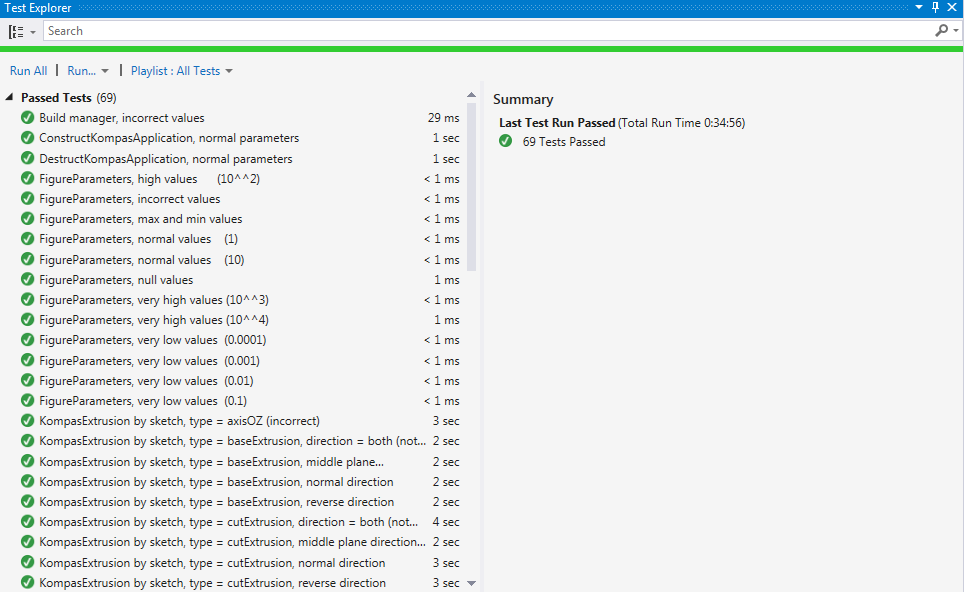


Рисунок 3.14 – Результаты модульных тестов

## Нагрузочные тесты

Построение сборки из деталей осуществляется множество раз последовательно с помощью одного экземпляра приложения «КОМПАС-3D».

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.

Результаты нагрузочного тестирования приведены на рисунках 3.15 и 3.16.

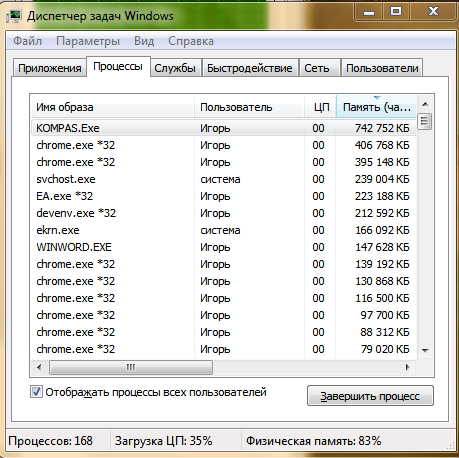


Рисунок 3.15 – Уровень загруженности оперативной памяти процессом приложения «КОМПАС-3D», Мб

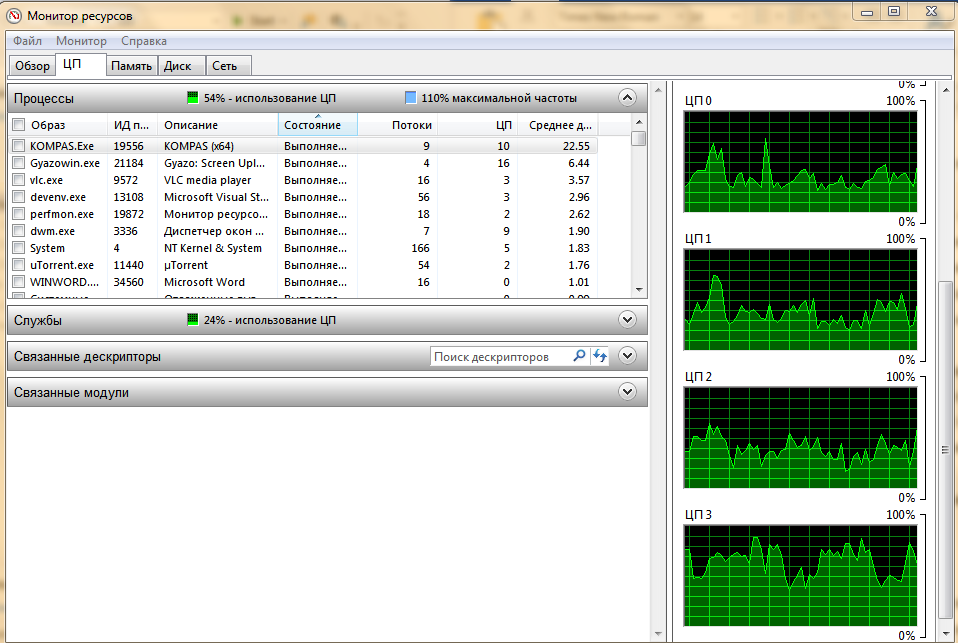


Рисунок 3.16 – Процент загрузки ядер ЦП процессом приложения «КОМПАС-3D»

На рисунках 3.17 и 3.18 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).

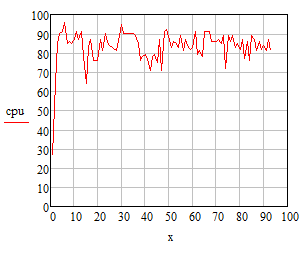


Рисунок 3.17 – График зависимости количества последовательно строящихся деталей (ось OX) от загрузки одного ядра ЦП процессом (ось OY), %

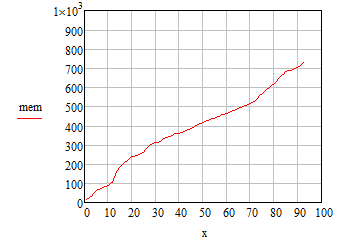


Рисунок 3.18 – График зависимости количества последовательно строящихся деталей (ось OX) от объёма памяти (ось OY), Мб

По графикам можно определить, что используемая память при увеличении количества последовательно строящихся сборок увеличивается линейно, при этом загруженность ядра ЦП, используемого приложением «КОМПАС-3D», остаётся непрерывной на уровне около 90% за всё время построения.

По результатам тестирования были получены следующие выводы:

– предел нагрузки одного ядра ЦП, используемого приложением «КОМПАС-3D», не превышал 25% в связи с ограничениями операционной системы Windows 7 на использование ядер процессора отдельно взятым приложением,

* при построении сборки использование оперативной памяти увеличивалось равномерно, около 7-10 мегабайт за одну операцию построения сборки.

# Заключение

В ходе курсового проекта были изучены основные этапы проектирования программного продукта, изучена предметная область предмета проектирования, также было изучено API системы автоматизированного проектирования «КОМПАС-3D».

В результате полученные знания были применены для реализации плагина для автоматизации построения модели объекта «Болт с гайкой» в рабочей плоскости программы «КОМПАС-3D».

Реализованный плагин протестирован на платформах Windows 7 «x86» и «x64». Также проводилось функциональное, нагрузочное и модульное тестирование приложения «КОМПАС-3D».

# Список литературы

Норенков И.П. «Основы автоматизированного проектирования». Издательство: МГТУ; Москва:, 2002 – 336 с.

API – Википедия. [Электронный ресурс]. — Режим доступа: <https://ru.wikipedia.org/wiki/API> (дата обращения 21.12.2016)

Плагин – Википедия. [Электронный ресурс]. — Режим доступа: <https://ru.wikipedia.org/wiki/Плагин> (дата обращения 21.12.2016)

КОМПАС-3D: О программе. Официальный сайт САПР КОМПАС [Электронный ресурс]. – Режим доступа: <http://kompas.ru/kompas-3d/about/> (дата обращения 21.12.2016)

Кидрук Максим. КОМПАС-3D V10 на 100% / М. Кидрук. – СПб.: Питер, 2009 – 560 с.

Плагин PDF [Электронный ресурс]. — Режим доступа: <http://gkmsoft.ru/allcatalog/pdf2dkompas_plugin/> (дата обращения 21.12.2016)

3D PDF [Электронный ресурс]. — Режим доступа: <http://sapr-journal.ru/novosti/eksport-iz-kompas-3d-v-formate-3d-pdf/> (дата обращения 21.12.2016)

Болт – Википедия. [Электронный ресурс]. – Режим доступа: <http://ru.wikipedia.org/wiki/Болт> (дата обращения 21.12.2016)

Гайка – Википедия [Электронный ресурс] – Режим доступа: <https://ru.wikipedia.org/wiki/Гайка> (дата обращения: 21.12.2016)

UML. [Электронный ресурс]. – Режим доступа: <http://www.uml.org/> (дата обращения 21.12.2016)

End user documentation – Oracle VirtualBox [Электронный ресурс] – Режим доступа: <https://www.virtualbox.org/wiki/End-user_documentation> (дата обращения 21.12.2016)

NUnit Documentation – Github [Электронный ресурс] – Режим доступа: <https://github.com/nunit/docs/wiki/NUnit-Documentation> (дата обращения 21.12.2016)

Функциональное тестирование – Википедия. [Электронный ресурс]. – Режим доступа: [http://ru.wikipedia.org/wiki/Функциональное тестирование](http://ru.wikipedia.org/wiki/Функциональное%20тестирование) (дата обращения 21.12.2016)

# 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 NutScrewViewForm

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс NutScrewViewForm – 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 и гайки |
| - \_kompasApp | KompasApplication | Переменная для хранения объекта сущности для работы с приложением «КОМПАС-3D» |
| Методы | | |
| + NutScrewViewForm() |  | Конструктор класса |
| - RunButton\_Click(object, EventArgs) | void | Событие при нажатии на кнопку «Build nut with 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 гайки 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 и гайки |

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» |
| + NutPart | ksPart | Переменная для хранения объекта тела гайки в приложении «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 и гайки |
| +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 и гайки | | |
| 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: последовательный вызов методов CreateNut и CreateScrew, в процессе проверяющий последние на код ошибки |
| - CreateNut() | bool | Метод, создающий экземпляр класса NutManager, в процессе строящий деталь гайки |
| - 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 |

Таблица А.5 – Описание класса NutManager

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Тип | Описание | |
| Описание класса | | | |
| Класс NutManager – сущность-менеджер для построения детали гайки | | | |
| Поля | | | |
| - \_kompasApp | KompasApplication | Переменная для хранения объекта сущности для работы с приложением «КОМПАС» |

Окончание таблицы А.5

|  |  |  |
| --- | --- | --- |
| Свойства | | |
| + LastErrorCode | ErrorCodes | Переменная для хранения кода последней ошибки экземпляра класса |
| Методы | | | |
| + NutManager(  KompasApplication) |  | Конструктор класса |
| + CreateDetail() | bool | Метод создания детали |
| - CreateNutBase  (ksEntity, KompasPoint3D, RegularPolygonParameter) | ksEntity[] | Метод создания базовой части на гайки на её базовой плоскости |
| * CreateNutBasePlaneCylinder(KompasPoint3D, double) | ksEntity | Метод создания цилиндра для базовой плоскости гайки |
| * CreateNutChamferEntities (KompasPoint3D, RegularPolygonParameter, ksEntity[]) | ksEntity[] | Метод создания скруглённых фасок гайки |
| * CreateNutThread (ksEntity[], KompasPoint3D, double) | bool | Метод создания резьбовой части гайки |
| * DeleteNutBasePlaneCylinder(ksEntity, double) | bool | Метод удаления цилиндра для базовой плоскости гайки |