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)

**РАЗРАБОТКА ПЛАГИНА «СТЕЛЛАЖ» ДЛЯ «КОМПАС-3D v20»**

Пояснительная записка по лабораторному проекту

по дисциплине «ОСНОВЫ РАЗРАБОТКИ САПР»

«Построение стеллажа в системе КОМПАС-3D v20»

Выполнил:

студент гр.588-3

\_\_\_\_\_\_\_\_\_\_\_\_\_\_Пичугин Е.А.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Руководитель:

к.т.н., доцент каф. КСУП

\_\_\_\_\_\_\_\_\_\_\_\_\_Калентьев А. А.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Томск 2021

Оглавление

[1 Описание Компас-3D 3](#_Toc91112714)

[2 Описание API 4](#_Toc91112715)

[3 Обзор аналогов 7](#_Toc91112716)

[4 Описание предмета проектирования 9](#_Toc91112717)

[5 Диаграмма классов 11](#_Toc91112718)

[6 Макет пользовательского интерфейса 13](#_Toc91112719)

[7 Функциональное тестирование 15](#_Toc91112720)

[8 Модульное тестирование 20](#_Toc91112721)

[9 Нагрузочное тестирование 22](#_Toc91112722)

[10 Заключение 25](#_Toc91112723)

[Список литературы 26](#_Toc91112724)

[Приложение А 27](#_Toc91112725)

**1 Описание Компас-3D**

«Компас» — семейство систем автоматизированного проектирования, универсальная система автоматизированного проектирования, позволяющая в оперативном режиме выпускать чертежи изделий, схемы, спецификации, таблицы, инструкции, расчетно-пояснительные записки, технические условия, текстовые и прочие документы. Изначально система ориентирована на оформления документации в соответствии с ЕСКД, ЕСТД, СПДС и международными стандартами, но этим возможности системы не ограничиваются.

Система «Компас-3D» предназначена для создания трёхмерных ассоциативных моделей отдельных деталей (в том числе, деталей, формируемых из листового материала путём его гибки) и сборочных единиц, содержащих как оригинальные, так и стандартизованные конструктивные элементы. Параметрическая технология позволяет быстро получать модели типовых изделий на основе проектированного ранее прототипа. Многочисленные сервисные функции облегчают решение вспомогательных задач проектирования и обслуживания производства.

Система «Компас-3D» включает следующие компоненты: система трёхмерного твердотельного моделирования, универсальная система автоматизированного проектирования «Компас-График» и модуль формирования спецификаций. Ключевой особенностью «Компас-3D» является использование собственного математического ядра и параметрических технологий. [1]

# 2 Описание API

API (англ. Application Programming Interface) — описание способов, которыми одна компьютерная программа может взаимодействовать с другой программой.

В КОМПАС-3D существуют API двух версий: API 5 и API 7. Обе версии реализуют различные функции системы и дополняют друг друга. 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-3D является KompasObject. Получить указатель на этот интерфейс 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 [2].

Ниже в таблицах 1.1-1.5 представлены основные свойства и методы интерфейсов.

Таблица 1.1. – Методы интерфейса KompasObject.

|  |  |  |
| --- | --- | --- |
| Метод | Возвращаемое значение | Описание |
| Document3D() | Указатель на интерфейс документа трёхмерной модели ksDocument3D | Даёт возможность получить указатель на интерфейс трёхмерного документа (детали или сборки) |
| Visible |  | Свойство видимости приложения |
| Quit |  | Метод для завершения программы Kompas-3D |
| ActivateControllerAPI |  | Метод для активации контроллера API |
| ksDocument2D |  | Интерфейс событий графического документа, события интерфейса позволяют контролировать состояние документа. |

Таблица 1.2 — Методы интерфейса IPart.

|  |  |  |  |
| --- | --- | --- | --- |
| Метод | Входные параметры | Возвращаемое значение | Описание |
| GetDefaultEntity  (short objType) | |  |  | | --- | --- | | objType | - тип объекта. | | |  | | --- | | Указатель на интерфейс [ksEntity](mk:@MSITStore:D:\INSTAL\KOMPAS-3D%20V17.1\KOMPAS\SDK\SDK.chm::/ksEntity.htm) или [IEntity](mk:@MSITStore:D:\INSTAL\KOMPAS-3D%20V17.1\KOMPAS\SDK\SDK.chm::/ksEntity.htm). | | Получить указатель на интерфейс объекта, создаваемого системой по умолчанию |
| GetPart(int type) | |  |  | | --- | --- | | type | - тип компонента. | | указатель на интерфейс компонента [ksPart](mk:@MSITStore:D:\INSTAL\KOMPAS-3D%20V17.1\KOMPAS\SDK\SDK.chm::/ksPart.htm) или [IPart](mk:@MSITStore:D:\INSTAL\KOMPAS-3D%20V17.1\KOMPAS\SDK\SDK.chm::/ksPart.htm). | Получить указатель на интерфейс компонента в соответствии с заданным типом |
| NewEntity(short objType) | |  |  | | --- | --- | | ob | jType- [тип объекта](mk:@MSITStore:D:\INSTAL\KOMPAS-3D%20V17.1\KOMPAS\SDK\SDK.chm::/Obj3dType_NewEntil_Part.htm). | | указатель на интерфейс [ksEntity](mk:@MSITStore:D:\INSTAL\KOMPAS-3D%20V17.1\KOMPAS\SDK\SDK.chm::/ksEntity.htm) или [IEntity](mk:@MSITStore:D:\INSTAL\KOMPAS-3D%20V17.1\KOMPAS\SDK\SDK.chm::/ksEntity.htm). | Создать новый интерфейс объекта и получить указатель на него |

Таблица 1.3 — Методы интерфейса ksDocument3D.

|  |  |  |  |
| --- | --- | --- | --- |
| Метод | Входные параметры | Возвращаемое значение | Описание |
| Create (bool invisible, bool \_typeDoc) | invisible – признак режима редактирования документа  (TRUE – невидимый режим,  FALSE – видимый режим),  typeDoc – тип документа  (TRUE – деталь,  FALSE – сборка). | TRUE – в случае успешного завершения. | Дает возможность создать пустой документ (деталь или сборку) |
| GetPart(int type) | type – тип компонента из перечисления Типы компонентов. |  | Получить указатель на интерфейс компонента в соответствии с заданным типом |

Таблица 1.4 – Методы интерфейса ksDocument2D

|  |  |  |  |
| --- | --- | --- | --- |
| Метод | Входные параметры | Возвращаемое значение | Описание |
| ksCircle (double xc, double yc, double rad, long style) | xc, yc - координаты центра окружности, rad - радиус окружности, style - стиль линии. | указатель на окружность - в случае удачного завершения, 0 - в случае неудачи. | Метод для создания окружности. |

Продолжение таблицы 1.4

|  |  |  |  |
| --- | --- | --- | --- |
| Метод | Входные параметры | Возвращаемое значение | Описание |
| ksCircle (double xc, double yc, double rad, long style) | xc, yc - координаты центра окружности, rad - радиус окружности, style - стиль линии. | указатель на окружность - в случае удачного завершения, 0 - в случае неудачи. | Метод для создания окружности. |
| long ksLineSeg (double x1, double y1, double x2, double y2, long style) | x1, y1 - координаты первой точки отрезка, x2, y2 - координаты второй точки отрезка, style - стиль линии. | указатель на отрезок - в случае удачного завершения, 0 - в случае неудачи. | Метод для создания отрезка. |
| ksArcByPoint (double xc, double yc, double rad, double f1, double f2, short direction, long style); | xc, yc - координаты центра окружности, rad - радиус окружности, x1, y1 - координаты начальной точки дуги, x2, y2 - координаты конечной точки дуги, direction - направление отрисовки дуги: 1 - против часовой стрелки, -1 - по часовой стрелке, style - стиль линии. | указатель на отрезок - в случае удачного завершения, 0 - в случае неудачи. | Метод создает дугу по центру и конечным точкам |

Таблица 1.5 —Методы интерфейса [ksEntity](mk:@MSITStore:C:\Program%20Files\ASCON\KOMPAS-3D%20v18%20Study\SDK\SDK.chm::/ksEntity_props.htm).

|  |  |  |  |
| --- | --- | --- | --- |
| Метод | Входные параметры | Возвращаемое значение | Описание |
| ksBossExtrusionDefinition(BOOL forward, short type, double depth, double draftValue, BOOL draftOutward); | Forward- - направление выдавливания: TRUE - прямое направление, FALSE - обратное направление, type - тип выдавливания, depth - глубина выдавливания, draftValue - угол уклона, draftOutward - направление уклона: FALSE - уклон наружу, TRUE - уклон внутрь. | TRUE – в случае успешного завершения. | Метод выдавливает эскиз в одном направлении |
| ksCutExtrusionDefinition (BOOL forward, short type, double depth, double draftValue, BOOL draftOutward); | Forward- - направление выдавливания: TRUE - прямое направление, FALSE - обратное направление, type - тип выдавливания, depth - глубина выдавливания, draftValue - угол уклона, draftOutward - направление уклона: FALSE - уклон наружу, TRUE - уклон внутрь. | TRUE – в случае успешного завершения. | Метод вырезания выдавливанием эскиз в одном направлении |

# 3 Обзор аналогов

Проект ОБЪЕМНИК – ­­­­программа для проектирования кухонной и корпусной мебели для профессионалов и новичков в этой сфере [3].

Отличительной особенностью программы является использование параметрических изделий, таким образом составление проекта представляет собой набор готовых частей мебели и редактирование их под размер. Для продвинутых пользователей предусмотрен режим ручных построений. Программа рассчитана на то, что клиент в реальном времени получает готовый проект будущей мебели.

В программе ОБЪЕМНИК используются передовые технологий в обработке изображения, такие как OpenCL и OpenGL. Именно поэтому изображение высокого качества с тенями, глянцами и эффектами формируется на лету, не заставляя ждать конечного результата. Огромный опыт практического использования программы привел к созданию так называемого "цветового колеса" в программе ОБЪЕМНИК, которое позволяет вместе с клиентами подобрать цвет и материал будущей мебели, "втягивая" клиента в процесс проектирования и тем самым делая процесс создания мебели более творческим и непринуждённым.

Ниже на рисунке 3.1 изображена программа Объемник.



Рисунок 3.1 — Программа Объемник

Программа настраивается и работает под конкретное производство за счет использования библиотеки параметрических изделий. Изначально эта библиотека укомплектована массой изделий и также ее можно редактировать под свое производство.

Ниже на рисунке 3.2 изображено меню настроек мебели.



Рисунок 3.2 – Настройка элементов мебели

В процессе проектирования можно использовать параметрическую базу изделий либо создавать свои модели и для упрощения работы также эти модели можно сохранять в собственные каталоги, которые после бесплатного обновления программы останутся неизменными. Свои модели можно создавать на основе уже имеющихся моделей мебели, либо создавать их с нуля. В базовой комплектации в программе присутствуют изделия из массива дерева, из пластиков и ЛДСП, а также предусмотрено использования моделей из облака постоянно обновляющихся моделей мебели.

# 4 Описание предмета проектирования

Стеллаж  — складское специализированное мебельное оборудование для хранения предметов/грузов, состоящее из металлических/деревянных стоек, балок и поперечных балок либо [многоярусных](https://ru.wikipedia.org/wiki/%D0%AF%D1%80%D1%83%D1%81) настилов (полок), закрепленных на балках, либо состоящее из закреплённых на стойках консолей (консольные стеллажи).

В мебели стеллаж относится к корпусной мебели для хранения предметов различного назначения на полках, включая самую верхнюю полку/настил.

Стеллажи минимизируют занимаемую площадь и делают удобным доступ к хранимым на них предметам.[4]

Плагин, предназначен для создания стеллажей заранее определенного дизайна (рисунок 4.1). Плагин должен уметь изменять такие параметры как:

1. Глубина стеллажа L (от 300 до 600 мм);
2. Ширина стеллажа S (от 300 до 800 мм);
3. Высота стеллажа H (от 1000 до 3000 мм);
4. Высота от пола до нижней полки h2 (от 80 до 100 мм);
5. Толщина материала s1 (от 10 до 20 мм);
6. Количество полок n;
7. Высота полки h1 (от 200 до ((H – h2 – s1 \* n) / n) мм).



Рисунок 4.1 — Схема стеллажа с параметрами

# 5 Диаграмма классов

Диаграмма классов описывает типы объектов системы и различного рода статические отношения, которые существуют между ними. На диаграммах классов отображаются также свойства классов, операции классов и ограничения, которые накладываются на связи между объектами.[5]

Диаграмма классов приведена на рисунке 5.1.

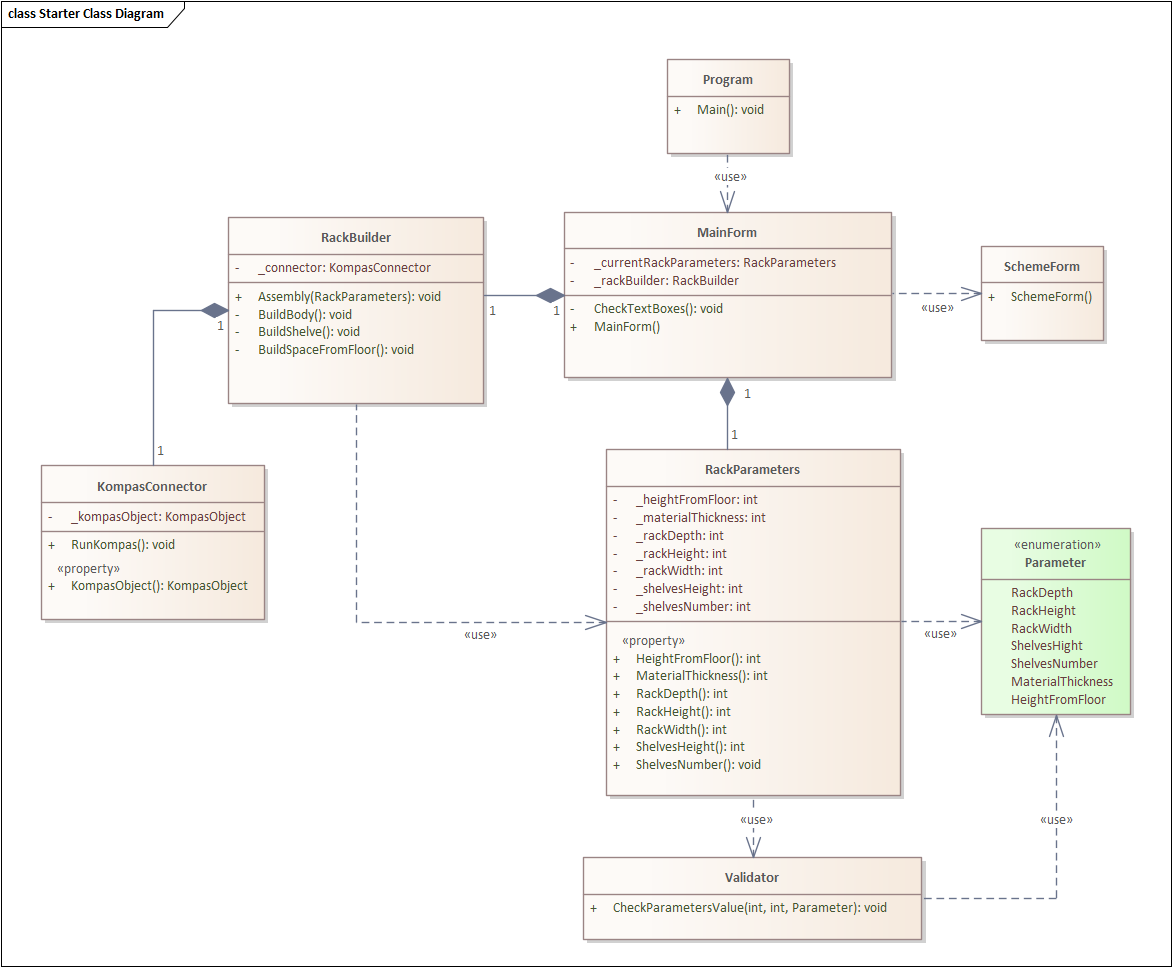


Рисунок 5.1 – Диаграмма классов

Для реализации был выбран следующий набор классов:

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

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

3) Validator − класс, проверяющий правильность введенных параметров;

4) KompasConnector – класс для работы с API КОМПАС 3D.

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

6) Parameter – перечисление, хранящее виды всех параметров;

7) Scheme – класс, выводящий схему проектируемой 3D-модели.

Диаграмма классов, после добавления дополнительной функциональности изображена на рисунке 5.2.

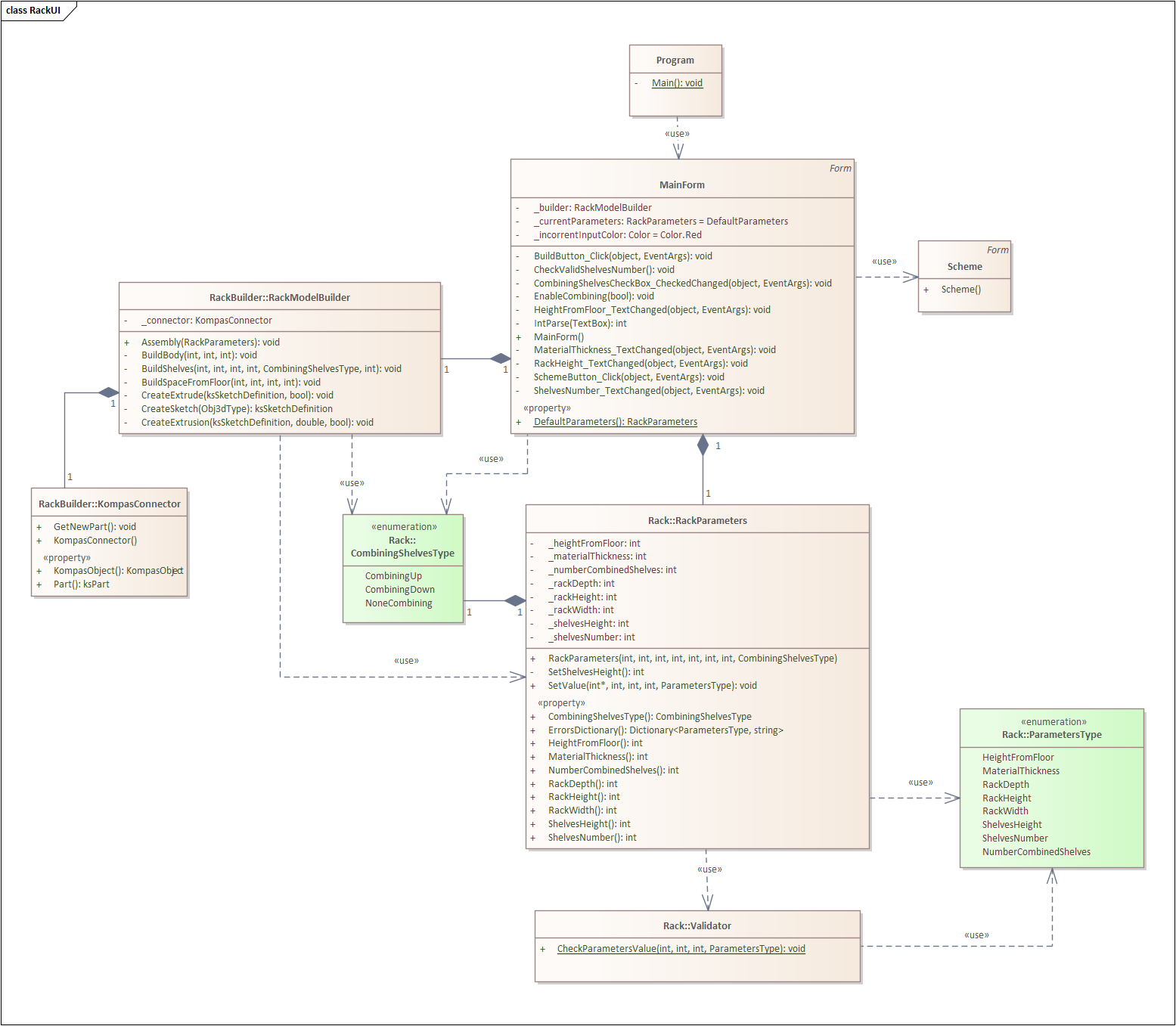


Рисунок 5.2 – Диаграмма классов c дополнительной функциональностью

Опишем некоторые изменение возникшие после добавления дополнительной функциональности и переработки кода в ходе разработки плагина.

Было добавлено: перечисление CombiningShelvesType для реализации дополнительного функционала по объединению полок стеллажа; различные методы позволяющие отслеживать и выделять неправильно введенные значения; отображать возможное для построения количество полок; словарь для хранения ошибок ввода параметров; метод для автоматического подсчета зависимых параметров; методы для выдавливания различных объектов.

# 6 Макет пользовательского интерфейса

Макет пользовательского интерфейса представляет собой форму для ввода параметров. При запуске программы в полях для ввода параметров находятся среднеарифметические значения. Пользователь может менять данные параметры (рисунок 6.1).

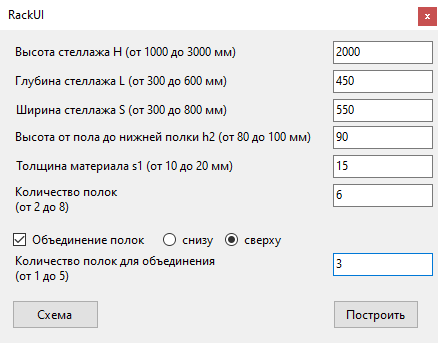


Рисунок 6.1 — Макет пользовательского интерфейса

При вводе некорректных данных всплывает окно с описанием ошибки (рисунок 6.2).

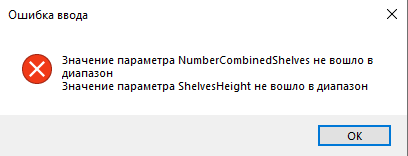


Рисунок 6.2 — Сообщение об ошибке

Поле, где было введено некорректное значение изменит цвет на красный (рисунок 6.3).

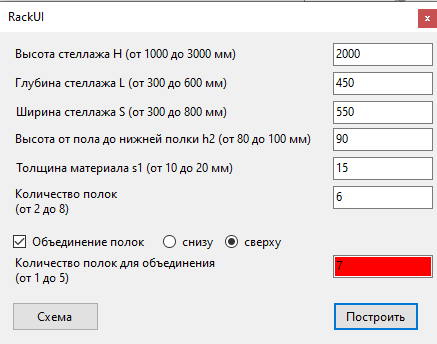


Рисунок 6.3 — Поле с некорректным параметром

При нажатии на кнопку «Схема», будет выведено изображение со схемой моделируемого объекта (рисунок 6.4).

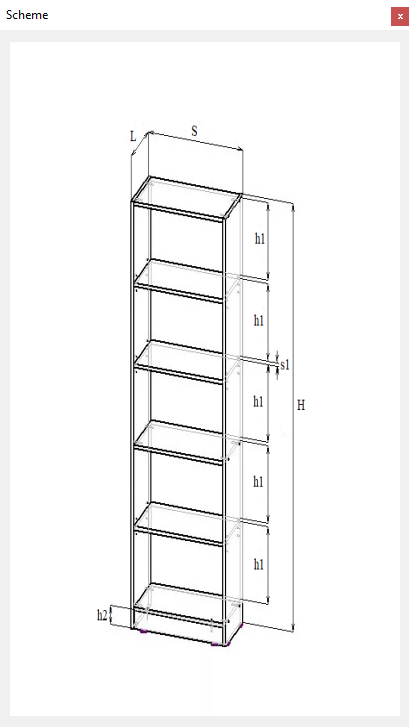


Рисунок 6.4 — Схема стеллажа в окне

# 7 Функциональное тестирование

Функциональное тестирование – это тестирование функциональности и поведения программы на соответствие требованиям функциональной спецификации. Функциональная спецификация определяет, что именно делает ПО, какие задачи оно решает. В данном случае будет проверяться правильность построения детали при различных входных параметрах

Построение модели с минимальными входными данными (рисунки 7.1):

1. Глубина стеллажа 300 мм;
2. Ширина стеллажа 300 мм;
3. Высота стеллажа 1000 мм;
4. Высота от пола до нижней полки 80 мм;
5. Толщина материала 10 мм;
6. Количество полок 2 штуки.

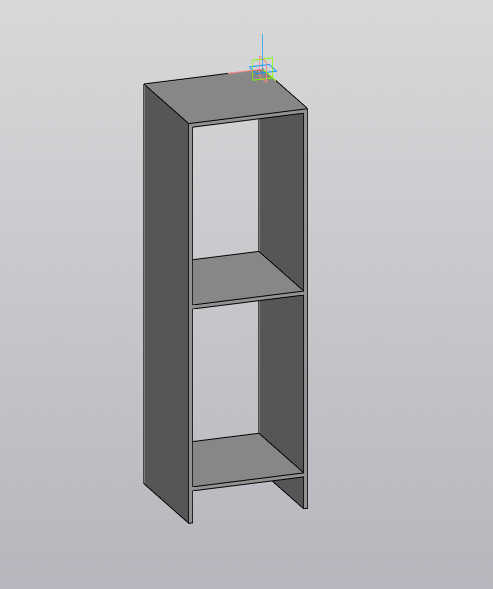


Рисунок 7.1 — Модель стеллажа, построенная с минимальными входными параметрами, без объединения полок

Построение модели с максимальными входными данными (рисунки 7.3-7.5):

1. Глубина стеллажа 600 мм;
2. Ширина стеллажа 800 мм;
3. Высота стеллажа 3000 мм;
4. Высота от пола до нижней полки 100 мм;
5. Толщина материала 20 мм;
6. Количество полок 13 штук;
7. Полок для объединения 8 штук.

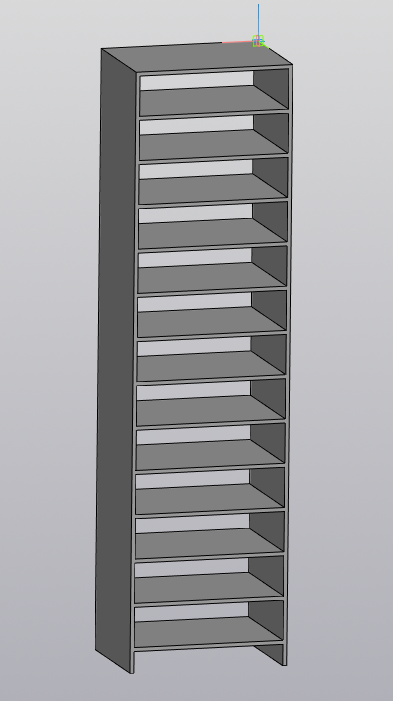


Рисунок 7.3 - Модель стеллажа, построенная с минимальными входными параметрами, без объединения полок

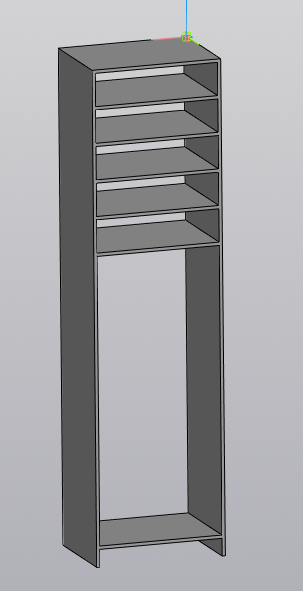


Рисунок 7.4 - Модель стеллажа, построенная с максимальными входными параметрами, c объединением полок снизу

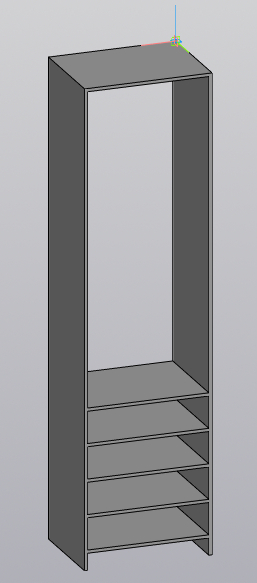


Рисунок 7.5 - Модель стеллажа, построенная с максимальными входными параметрами, c объединением полок сверху

Построение модели с входными данными по умолчанию (рисунки 7.6 -7.8):

1. Глубина стеллажа 450 мм;
2. Ширина стеллажа 550 мм;
3. Высота стеллажа 2000 мм;
4. Высота от пола до нижней полки 90 мм;
5. Толщина материала 15 мм;
6. Количество полок 6 штук;
7. Полок для объединения 3 штуки.

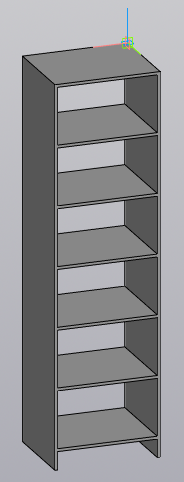
****

Рисунок 7.6 - Модель стеллажа, построенная с входными параметрами по умолчанию, без объединения

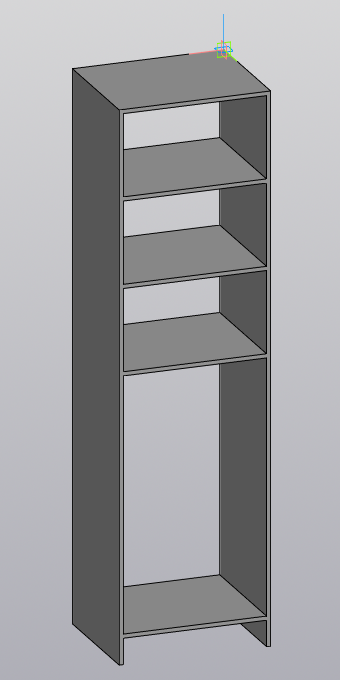


Рисунок 7.7 - Модель стеллажа, построенная с входными параметрами по умолчанию, с объединением снизу

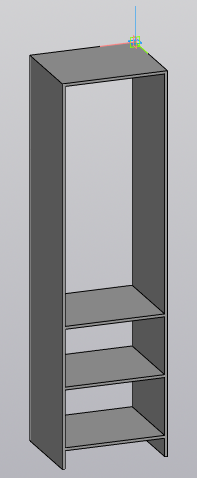
****

Рисунок 7.8 - Модель стеллажа, построенная с входными параметрами по умолчанию, с объединением сверху

# 8 Модульное тестирование

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 [6].

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

Тестирование проводилось с помощью фреймворка модульного тестирования NUnit 3.13.2 для языков платформы .NET.

Результаты прохождения всех модульных тестов приведены на рисунках 8.1 и 8.2.

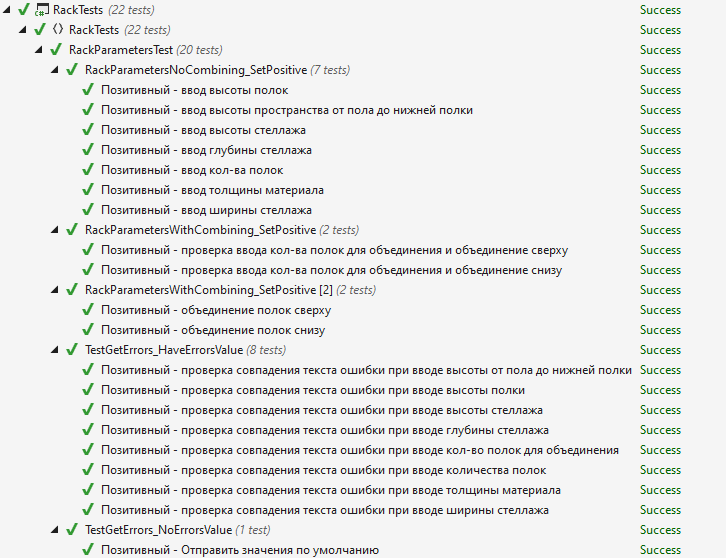
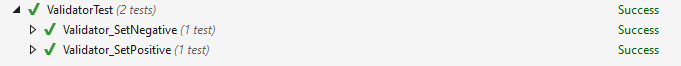


Рисунок 8.1 – Список пройденных юнит-тестов для класса RackParameters

Рисунок 8.2 – Список пройденных юнит-тестов для класса Validator

Результат покрытия моделей Validator и RackParameters тестами приведен на рисунке 8.3. Цикломатическая сложность для класса RackParameters равна 23, а для класса Validator равна 3.

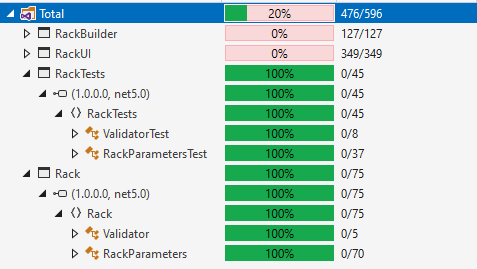


Рисунок 8.3 – Результат покрытия тестами

# 9 Нагрузочное тестирование

Нагрузочное тестирование – тип тестирования, который позволяет оценить поведение системы при возрастающей нагрузке, целью нагрузочного тестирования является также определение максимальной нагрузки, которую может выдержать система [7].

Нагрузочное тестирование проводилось на компьютере со следующими характеристиками:

* Процессор: AMD Ryzen 5 5600X 4650 MHz;
* Оперативная память: 16GB DDR4 3200 MHz;
* Графический процессор: AMD Radeon 480;
* Операционная система: Windows 10 Pro 64-bit.

При задании максимального количества последовательно строящихся моделей равным 200, САПР «КОМПАС-3D» остановила дальнейшие построения после 197 модели, окно программы перестало реагировать на ввод. В диспетчере задач у процесса запущенной САПР отображался статус «Не работает». Исходя из этого, для выполнения нагрузочного тестирования, количество построенных деталей было принято равным 197.

Зависимость потребления оперативной памяти САПР «КОМПАС-3D» при построенных 197 моделях представлена на графике (рисунке 9.1).

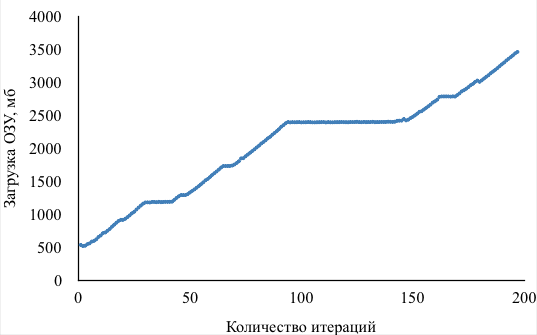
Рисунок 9.1 – Зависимость потребление ОЗУ от номера итерации

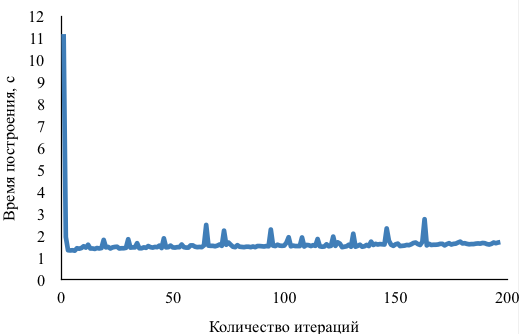
Потребление памяти резко возросло после построения 23 моделей. Пик потребления был достигнут на 29 построении и составил 810 МБ. При дальнейших построениях количество используемой ОЗУ уменьшается, при построении падает с 704 МБ до 554 MB, затем снова постепенно возрастает, не превышая 700 МБ.

График на рисунке 9.2 отображает зависимость застрачиваемого времени на каждое новое построение от количества уже построенных моделей. С начала время построения модели максимально, так как идет запуск САПР «КОМПАС-3D».

Рисунок 9.2 – Зависимость затрат времени на построение от номера итерации

Возможно, что такое потребление оперативной памяти связано с использованием файла подкачки, а точнее переход в виртуальную память. Виртуальная память — метод [управления памятью](https://ru.wikipedia.org/wiki/%D0%A3%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C%D1%8E) [компьютера](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80), позволяющий выполнять программы, требующие больше [оперативной памяти](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C), чем имеется в компьютере, путём автоматического перемещения частей программы между основной памятью и вторичным хранилищем (например, [твердотельным](https://ru.wikipedia.org/wiki/%D0%96%D1%91%D1%81%D1%82%D0%BA%D0%B8%D0%B9_%D0%B4%D0%B8%D1%81%D0%BA) накопителем)[8]. Для проверки данной гипотезы в Windows 10 был отключен файл подкачки и заново запущенно нагрузочное тестирование (рисунки 9.3 и 9.4).

Рисунок 9.3 – Зависимость потребление ОЗУ от номера итерации

Рисунок 9.4 – Зависимость затрат времени на построение от номера итерации

Из графика загрузки ОЗУ можно видеть, что он почти на протяжении всей длинны имеет линейны вид, а понижение потребления оперативной памяти, скорее всего, связано с переносом данных на внутренний накопитель данных. Время построения скорее всего имеет такие пики потребления из-за того-что, в процессе проведения тестирования были запущенны другие процессы, которые мешали работе Компаса-3D.

# 10 Заключение

При выполнения лабораторных работ были изучены основные этапы проектирования программного продукта и его реализации, предметная область объекта проектирования, SDK «КОМПАС-3D». Было составлено техническое задание, разработан проект системы, составлены UML диаграммы классов, разработан макет пользовательского интерфейса.

В результате работы был разработан и реализован плагин для САПР «КОМПАС-3D», выполняющий построение 3D-модели стеллажа по задаваемым параметрам.

Над реализованным плагином были проведены функциональное, модульное и нагрузочное тестирование на платформе Windows 10.

# Список литературы

1. Компас (САПР) — Википедия. [Электронный ресурс] — Режим доступа: <https://ru.wikipedia.org/wiki/Компас_(САПР)> (дата обращения 25.03.2021)
2. Кидрук Максим. КОМПАС-3D V10 на 100% / М. Кидрук. – СПб.: Питер, 2009 – 560 с.
3. Мебельный софт, топ 10 лучших программ для работы с мебелью. [Электронный ресурс] — Режим доступа: https://gidmaster.info/soft.php?id=proekt\_ob-emnik (дата обращения 28.10.2021
4. Стеллаж – Википедия. [Электронный ресурс] — Режим доступа: https://ru.wikipedia.org/wiki/Стеллаж (дата обращения 28.10.2021)
5. М. Фаулер. UML. Основы, 3-е издание. — Пер. с англ. — СПб: символ-Плюс, 2004– 192 с.
6. Новые технологии в программировании: учебное пособие / А.А.Калентьев, Д.В.Гарайс, А.Е.Горяинов – Томск: Эль Контент, 2014.—176 с.
7. Нагрузочное тестирование vs Тестирование производительности. [Электронный ресурс]. – https://performance-lab.ru/blog/load-testing/testirovanie-proizvoditelnosti (дата обращения 22.12.2021)
8. Виртуальная память — Википедия. [Электронный ресурс] — Режим доступа: https://ru.wikipedia.org/wiki/Виртуальная память (дата обращения 17.04.2021)

# Приложение А

(справочное)

Таблица А.1 – Тестовые сценарии

|  |  |  |
| --- | --- | --- |
| Тестовый метод | Входные параметры | Описание тестового случая |
| Validator\_SetPositive(int correctValue, int minValue,int maxValue, ParametersType parametersType) | 1005, 1000, 3000, ParametersType.RackHeight | Позитивный - ввод значений в диапазоне |
| Validator\_SetNegative(int incorrectValue,int minValue,int maxValue, ParametersType parametersType) | 999, 1000, 3000, ParametersType.RackHeight | Негативный - ввод значений вне диапазона |
| RackParametersWithCombining\_SetPositive( CombiningShelvesType combiningType) | CombiningShelvesType.CombiningUp | Позитивный - объединение полок сверху |
| CombiningShelvesType.CombiningDown | Позитивный - объединение полок снизу |
| TestGetErrors\_NoErrorsValue() |  | Позитивный - Отправить значения по умолчанию |
| RackParametersWithCombining\_SetPositive(int correctValue, CombiningShelvesType combiningType) | 1, CombiningShelvesType.CombiningUp | Позитивный - проверка ввода кол-ва полок  для объединения и объединение сверху |
| 1, CombiningShelvesType.CombiningDown | Позитивный - проверка ввода кол-ва полок для объединения и объединение снизу |
| TestGetErrors\_HaveErrorsValue(ParametersType parametersType, string errorText) | ParametersType.NumberCombinedShelves, "Значение параметра NumberCombinedShelves не вошло в диапазон | Позитивный - проверка совпадения текста ошибки при вводе кол-во полок для объединения (также является негативным для ввода параметров) |

Продолжение таблицы А.1

|  |  |  |
| --- | --- | --- |
| Тестовый метод | Входные параметры | Описание тестового случая |
| TestGetErrors\_HaveErrorsValue(ParametersType parametersType, string errorText) | ParametersType.RackHeight, "Значение параметра RackHeight не вошло в диапазон" | Позитивный - проверка совпадения текста ошибки  при вводе высоты стеллажа (также является негативным для ввода параметров) |
| ParametersType.HeightFromFloor, "Значение параметра HeightFromFloor не вошло в диапазон" | Позитивный - проверка совпадения текста ошибки  при вводе высоты от пола до нижней полки (также является негативным для ввода параметров) |
| ParametersType.RackDept,"Значение параметра RackDepth не вошло в диапазон" | Позитивный - проверка совпадения текста ошибки  при вводе глубины стеллажа (также является негативным для ввода параметров) |
| ParametersType.RackWidt, "Значение параметра RackWidth не вошло в диапазон" | Позитивный - проверка совпадения текста ошибки  при вводе ширины стеллажа (также является негативным для ввода параметров) |

Продолжение таблицы А.1

|  |  |  |
| --- | --- | --- |
| Тестовый метод | Входные параметры | Описание тестового случая |
| TestGetErrors\_HaveErrorsValue(ParametersType parametersType, string errorText) | ParametersType.ShelvesNumber,"Значение параметра ShelvesNumber не вошло в диапазон" | Позитивный - проверка совпадения текста ошибки  при вводе количества полок (также является негативным для ввода параметров) |
| ParametersType.MaterialThickness,"Значение параметра MaterialThickness не вошло в диапазон" | Позитивный - проверка совпадения текста ошибки  при вводе толщины материала (также является негативным для ввода параметров) |
| ParametersType.ShelvesNumber,  "Значение параметра ShelvesNumber не вошло в диапазон" | Позитивный - проверка совпадения текста ошибки  при вводе количества полок (также является негативным для ввода параметров) |
| RackParametersNoCombining\_SetPositive( int correctValue,ParametersType parameter) | 10, ParametersType.MaterialThickness, | Позитивный - ввод толщины материала |
| 80, ParametersType.HeightFromFloor | Позитивный - ввод высоты пространства  от пола до нижней полки |

Продолжение таблицы А.1

|  |  |  |
| --- | --- | --- |
| Тестовый метод | Входные параметры | Описание тестового случая |
| RackParametersNoCombining\_SetPositive( int correctValue,ParametersType parameter) | 1000, ParametersType.RackHeight, | Позитивный - ввод высоты стеллажа |
| 300, ParametersType.RackDepth | Позитивный - ввод глубины стеллажа |
| 300, ParametersType.RackWidth | Позитивный - ввод ширины стеллажа |
| 2, ParametersType.ShelvesNumber | Позитивный - ввод кол-ва полок |
| 200, ParametersType.ShelvesHeight | Позитивный - ввод высоты полок |