Задание №1 по курсу

«Проектирование Программных Систем»

***ДоставимВсе***

Выполнили:

Забелин А., 492б

Кошкин П., 492б

Зинов Н., 492в

Ракчеев Е., 492в

**1. Постановка задачи**

## **1.1. Состав первого задания**

Первое задание состоит в первоначальном изучении предметной области, формализации существующих классов и вариантов использования. По результатам построения CRC карточек и диаграмм взаимодействия для каждого варианта использования предлагаются улучшения диаграммы классов.

## **1.2. Описание задачи**

Служба выполняет заказы по доставке заказов клиентам. Доставка заказа осуществляется по маршруту, состоящему из последовательно соединенных плеч (leg). Каждое плечо маршрута может быть одним из следующих типов: морским транспортом, автотраспортом, ж/д транспортом, авиатранспортом. Отслеживание местонахождения заказа по его номеру: для каждого заказа известно, на каком плече он находится. Начальное плечо - передача заказа в доставку, конечное - получение. Служба ведет историю заказов для каждого клиента, в том числе историю прохождения заказа по маршруту.

Заказ может включать несколько товаров. Каждый товар обладает заявленной стоимостью и весом. Для каждого типа плеча известны ограничения по весу, время и стоимость пересылки, последняя зависит от веса и заявленной стоимости всех товаров в заказе.

Служба хранит граф, объединяющий возможные плечи доставки, ограничения и доступные типы на каждом плече. Оператор составляет маршрут для заказа.

Автоматическое построение маршрутов, оптимизация доставки по скорости, стоимости

**2. Описание предметной области**

## **2.1. Идентификация классов**

Для идентификации классов и построения первого варианта модели предметной области воспользуемся методом Аббота. Потенциальные классы приведены в таблице 1.

Условные обозначения критериев проверки классов:

– С : класс сохраняет информацию;

– И : предполагается наличие интерфейса для изменения хранимой информации;

– А : для хранения информации используется несколько атрибутов;

– О : класс реализует несколько действий;

– У : атрибуты и операции класса применимы ко всем экземплярам;

– Т : наличие класса в модели является существенным требованием.

Таблица 1. Потенциальные классы и проверка критериев

|  |  |  |  |
| --- | --- | --- | --- |
| **Потенциальный класс** | **Критерии** | **Хранимая информация** | **Выделенные действия** |
| Служба |  | - | - |
| Заказ | САУТ | местоположение, история прохождения, маршрут, товары, номер | - |
| Доставка |  | - | - |
| Клиент | СИУ | История заказов | - |
| Маршрут | СУ | плечи | - |
| Плечо | САУТ | тип, ограничение по весу, время и стоимость | - |
| Морской транспорт,  автотранспорт,  ж/д транспорт,  авиатранспорт | СУ | тип плеча (строка) | - |
| Местонахождение | СУ | плечо | - |
| Номер | СУ | номер заказа (число) | - |
| Передача,  получение | СУ | тип плеча (строка) | - |
| История заказов | СИУ | Заказы | - |
| История прохождения | СИАУ | плечи | - |
| Товар | САУТ | Стоимость, вес, описательные характеристики | - |
| Стоимость | СУ | стоимость | - |
| Вес | СУ | вес | - |
| Время | СУ | время | - |
| Граф | СИАУТ | плечи, вершины | - |
| Оператор |  | - | Составить марштрут |

TODO

Дать пояснения по составлению таблицы:

– какие существительные объединены, как синонимы;

– какие атрибуты предполагается использовать для хранения информации;

– как были получены действия.

Описать действия по анализу таблицы и принятию решений по классам.

Существительные, обозначающие одни и те же сущности объединены как синонимы (например “маршрут” и “история прохождения“). Служба - не класс, а вся система. Граф - на самом деле мультиграф, ребрами которого являются плечи различных типов. Тип плеча определяется его строковым атрибутом, который может быть равен одному из следующих вариантов: морской транспорт, автотранспорт, жд транспорт, авиатранспорт, передача, получение. Заказ хранит историю прохождения товаром по графу в виде пар (реальное время доставки, плечо). Система обрабатывает события завершения доставки товаров заказа по определенному плечу. Доставка абстрактное понятие не присутствующее в системе в виде класса или функции. Маршрут – массив плечей. Время, стоимость и т.п. описывается примитивными типами.

## **2.2. Предварительная логическая модель заказа товаров**

ПРОВЕРИТЬ, ЧТО СДЕЛАНО ВСЕ СИНЕЕ

Построить диаграмму классов, указать, каким образом были получены атрибуты, ассоциации и операций из хранимой информации и действий.

При разработке модели названия классам следует давать на английском языке, при этом должно быть понятно соответствие между классами и потенциальными классами.

Также нужно дать пояснения по выделенным ассоциациям и кратностям полюсов.

При оформлении рисунков следует придерживаться одного выбранного масштаба, например, 75%.

Создан вспомогательный класс вершины графа, который используют плечо Leg и заказ Order. Граф состоит из вершин и ребер, поэтому есть соответствующие **композиции (или как их там эти закрашенные ромбики)**. В Order необходимо хранить маршрут доставки, поэтому Leg и Order соединены агрегацией route, причем плечи в маршруте упорядочены. Еще в Order нужно хранить список доставляемых товаров, поэтому Products соединены **композицией** с Order.

Диаграмма классов модели приведена на рис. 1.

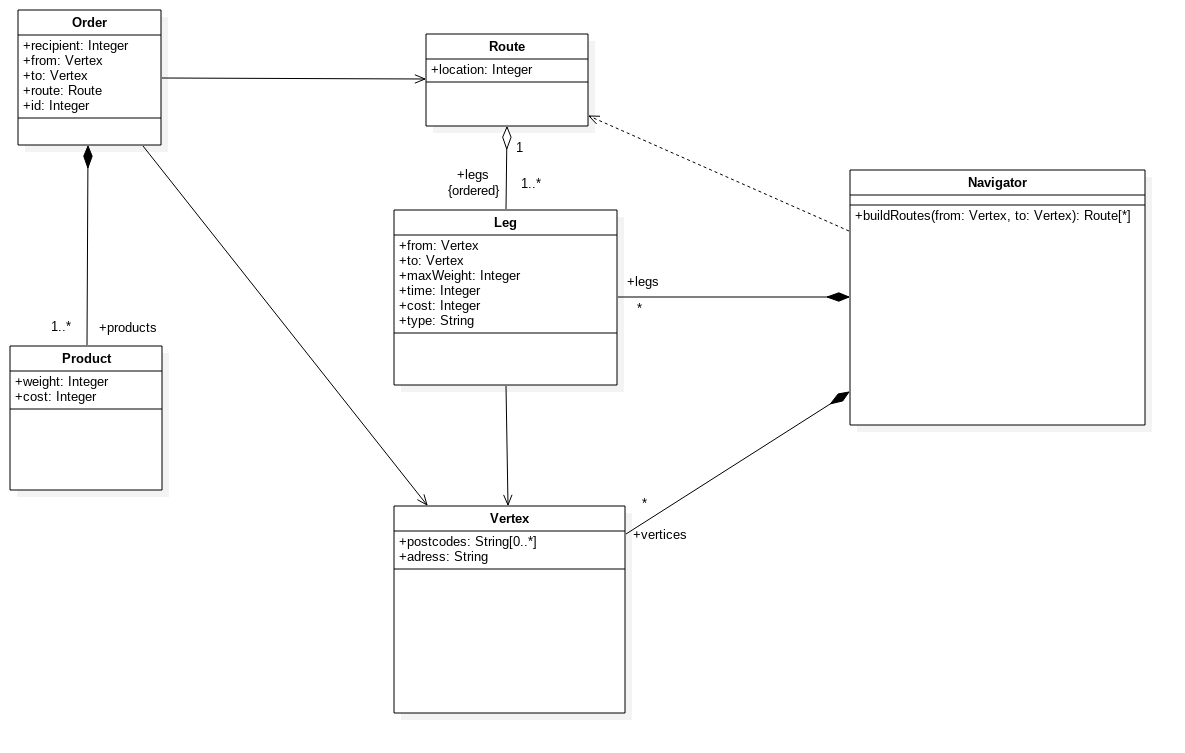


Рис. 1. Предварительная диаграмма классов бронирования билетов

## **2.3. Обновленная логическая модель заказа товаров**

TODO

На рис. 2 приведена диаграмма классов после рассмотрения варианта использования Registration.

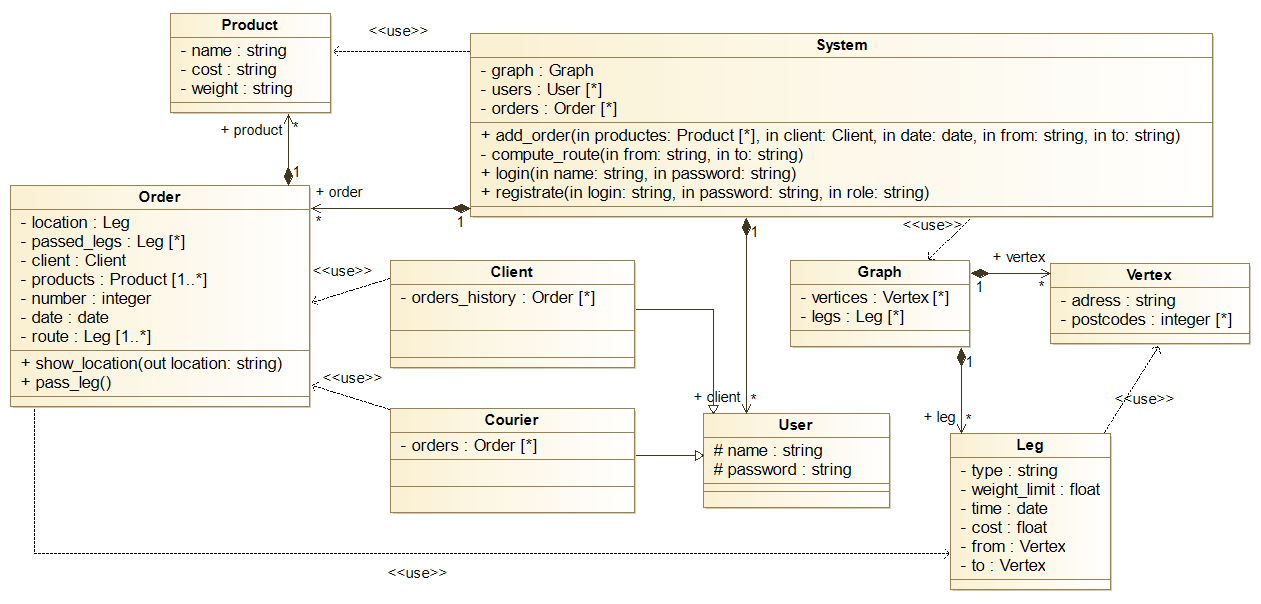


Рис. 2. Обновленная диаграмма классов с регистрацией

# **3. Описание требований к системе**

## **3.1. Модель вариантов использования**

Все пользователи системы (актор User) могут смотреть список заказов ShowOrders и местоположение определенного заказа ShowLocation - <<extend>> выбран потому, что этот вариант использования вызывается опционально из списка заказов.

В описании системы упоминается клиент. Он может сделать заказ в сценарии PlaceOrder. Этот вариант использования включает в себя сценарий выбора маршрута из предложенных SelectRoute, который также может выполняться для существующего заказа например, если добавлен новый маршрут. ShowOrders extend SelectRoute, так как в случае изменения маршурута уже существующего заказа Клиент должен сначала выбрать нужный заказ из списка. В случае, если ни один из вариантов не устраивает клиента, срабатывает точка расширения на вариант использования CancelOrder для отмены заказа.

Еще один актор - Оператор. Он может добавить возможный маршрут для выбранного заказа - AddRoute. И следит за тем, что товар проходит по плечам. При прохождении заказом очередного плеча он выполняет сценарий использования CompleteLeg.

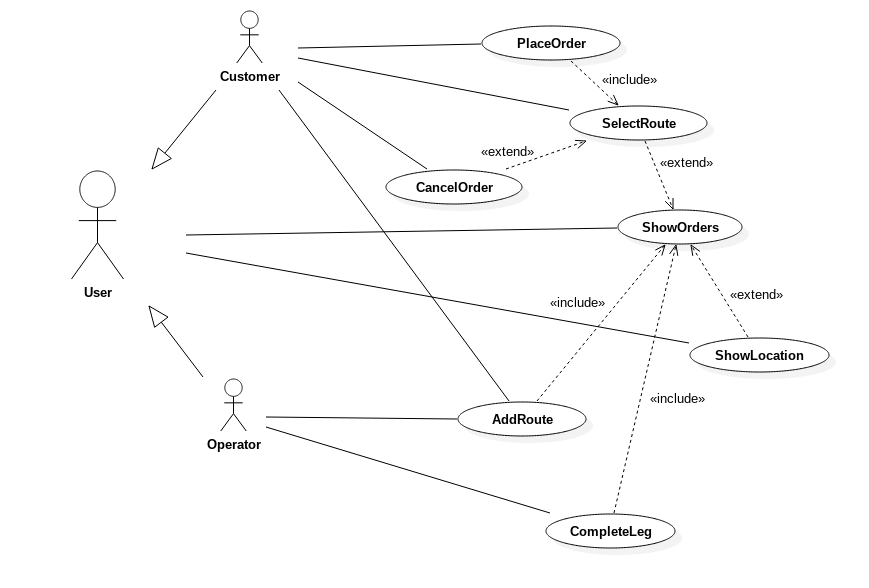


Рис. 3. Диаграмма вариантов использования.

## **3.2. Вариант использования ShowLocation**

**Акторы:** User

**Цель:** Пользователь хочет узнать местонахождение заказа в данный момент

**Предусловия:** Есть заказ в доставке

**Постусловия: -**

**Основной сценарий:**

1. Система показывает список заказов в доставке
2. Пользователь выбирает заказ, местоположение которого он хочет узнать (SelectRoute)
3. Система показывает карту с отмеченным положением заказа

**Альтернативные сценарии: -**

**Точки расширения: -**

## **3.3. Вариант использования PlaceOrder**

**Акторы:** Client

**Цель:** Клиент хочет заказать доставку

**Предусловия:** -

**Постусловия:** добавлен заказ Клиента

**Основной сценарий:**

1. Клиент вводит параметры отправляемых товаров; исходный и целевой адреса
2. Система рассчитывает и сохраняет оптимальный маршрут, показывает клиенту варианты стоимости и времени доставки
3. Клиент выбирает вариант (SelectRoute)
4. Система сохраняет заказ в базу

**Альтернативные сценарии: -**

**Точки расширения: -**

## **3.4. Вариант использования CompleteLeg**

**Акторы:** Operator

**Цель:** Оператор отмечает, что заказ прошел текущее плечо

**Предусловия:** Заказ прошел текущее плечо

**Постусловия:** Изменено местоположение заказа

**Основной сценарий:**

1. Система показывает список заказов в обработке (ShowOrders)
2. Оператор выбирает какой заказ прошел плечо
3. Система обновляет местоположение заказа

**Альтернативные сценарии: -**

**Точки расширения: -**

.

## **3.5. Вариант использования ShowOrders**

**Акторы:** User

**Цель:** Пользователю показывают список заказов.

**Предусловия:** В системе есть заказы

**Постусловия:** -

**Основной сценарий:**

1. Система отображает список заказов

**Альтернативные сценарии: -**

**Точки расширения:** Если Клиент хочет посмотреть местоположение какого-нибудь заказа, выполняется вариант использования ShowLocation. Если Клиент хочет изменить маршрут какого-нибудь заказа, выполняется вариант использования SelectRoute

## **3.6. Вариант использования SelectRoute**

**Акторы:** Client

**Цель:** Выбрать один из предложенных вариантов маршрута

**Предусловия:** Оператор добавил хотя бы один маршрут

**Постусловия:** -

**Основной сценарий:**

1. Система показывает список возможных маршрутов
2. Client выбирает из них тот, который ему больше нравится

**Альтернативные сценарии: -**

**Точки расширения:** Если пользователь не видит хороших вариантов и хочет отменить заказ, то выполняется вариант использования CancelOrder.

## **3.7. Вариант использования AddRoute**

**Акторы:** Operator

**Цель:** Добавить вариант маршрута

**Предусловия:** Есть заказ в системе

**Постусловия:** Добавлен вариант маршрута

**Основной сценарий:**

1. Выполняется выбор маршрута (ShowOrders)
2. Оператор добавляет вариант маршрута
3. Система проверяет корректность
4. Система уведомляет Клиента о том, что добавлен маршрут

**Альтернативные сценарии: -**

**Точки расширения:**

## **3.8. Вариант использования CancelOrder**

**Акторы:** Сustomer

**Цель:** Отменить заказ

**Предусловия:** Есть заказ в состоянии доставки

**Постусловия:** Заказ удаляется

**Основной сценарий:**

1. Система запрашивает подтверждение
2. Клиент подтверждает отмену

**Альтернативные сценарии: -**

**Точки расширения:**

# **4. Моделирование взаимодействий и поведения**

В данном разделе нужно провести анализ взаимодействий, описанных в вариантах использования. Это означает, что для каждого варианта использования нужно проработать, какие классы будут участвовать в его реализации и какие обязанности они будут иметь.

Что делать: выделить взаимодействия, для каждого используя карточки CRC выделить и назначить обязанности, затем построить диаграммы последовательности. Не забыть реализовать обязанности в виде операций и атрибутов, показать их на диаграмме классов.

На данном этапе в модель нужно добавить кооперации, реализующие варианты использования. В кооперациях будут участвовать акторы.

Раздел заполняется после анализа взаимодействий в подразделах.

Варианты использования AddOrder, ShowLocation и PassLeg реализуются кооперацией Ordering

Рис. 4. Реализация вариантов использования с помощью кооперации Ordering

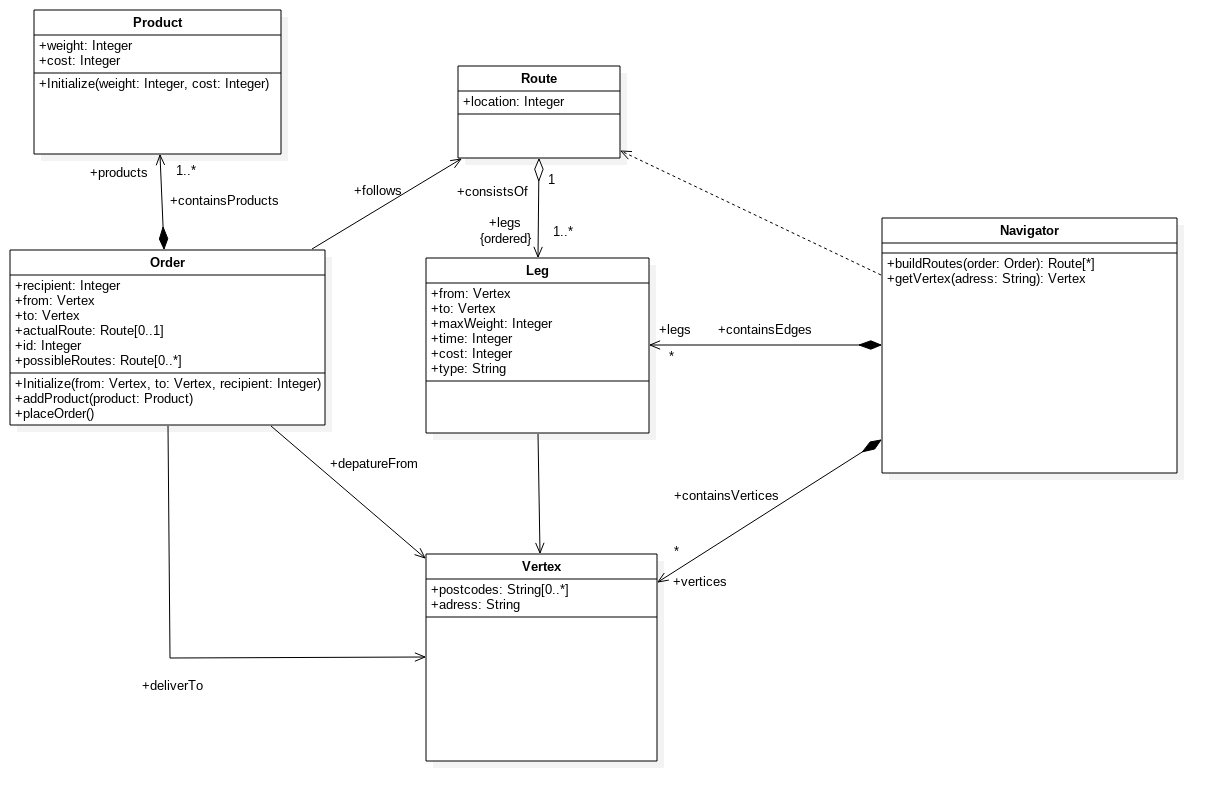
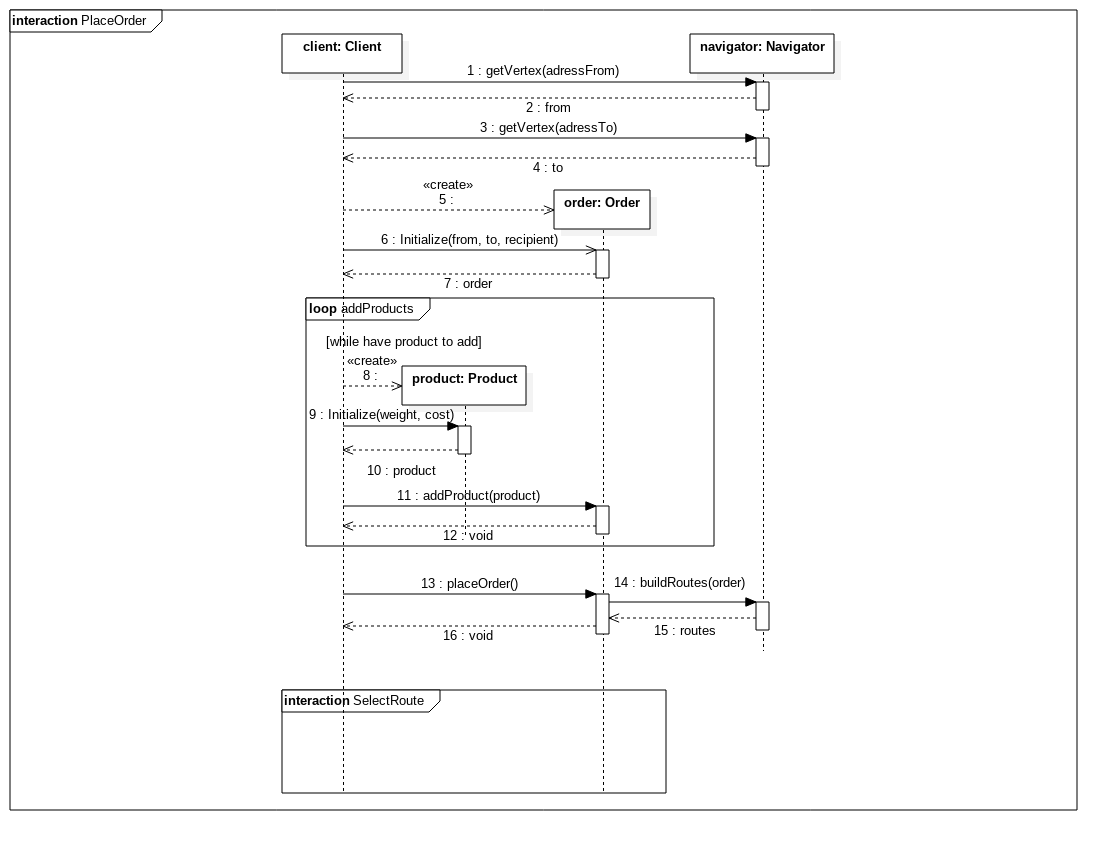
## **4.1. Реализация варианта использования PlaceOrder**

Результаты анализа основного сценария с помощью карточек CRC приведены в таблице 2.

Таблица 2. Обязанности и коллеги участников реализации варианта использования PlaceOrder.

TODO (Pasha, изменить CRC)

|  |  |  |
| --- | --- | --- |
| **Объект класса** | **Контракт** | **Коллеги** |
| Customer | Выполняет создание товара  Вызывает установку точки отправления и назначения  Выполняет выбор маршрута | Product  Order  Route |
| Navigator | Выполняет создание маршрутов  Хранит список плечей  Хранит список вершин | Route  Leg  Vertex |
| Order | Хранит список продуктов  Знает клиента  Хранит точку отправления и назначения  Хранит маршрут  Хранит свой номер | Product  Customer  Vertex  Route |
| Route | Знает список плечей в маршруте  Хранит местоположение | Leg |
| Leg | Хранит свои концы  Хранит ограничение по весу  Хранит время прохождения  Хранит стоимость  Хранит тип | Vertex |
| Vertex | Хранит обслуживаемые почтовые индексы  Хранит свой адрес |  |
| Product | Хранит стоимость  Хранит вес |  |



Затем дать пояснения, каким образом обязанности уже отражены в модели предметной области. Указать, какие не отражены. Их разложить на атрибуты, ассоциации и операции и добавить в модель.

*Обязанность класса Order по хранению списка товаров и маршрута в модели реализуется ассоциациями ContainsProducts и Follows. А хранение точки отправления и назначения реализуется ассоциациями DepartureFrom и DeliverTo.*

## **4.2. Реализация варианта использования ShowOrders**

TODO (Lesha)

|  |  |  |
| --- | --- | --- |
| **Объект класса** | **Контракт** | **Коллеги** |
| User | Использует заказ | Order |
| Navigator | Выполняет создание маршрутов  Хранит список плечей  Хранит список вершин | Route  Leg  Vertex |
| Order | Хранит список продуктов  Знает клиента  Хранит точку отправления и назначения  Хранит маршрут  Хранит свой номер | Product  Customer  Vertex  Route |
| Route | Знает список плечей в маршруте  Хранит местоположение | Leg |
| Leg | Хранит свои концы  Хранит ограничение по весу  Хранит время прохождения  Хранит стоимость  Хранит тип | Vertex |
| Vertex | Хранит обслуживаемые почтовые индексы  Хранит свой адрес |  |
| Product | Хранит стоимость  Хранит вес |  |

## **4.3. Реализация варианта использования AddRoute**

TODO (Kolya)

|  |  |  |
| --- | --- | --- |
| **Объект класса** | **Контракт** | **Коллеги** |
| Customer | Выполняет создание товара  Вызывает установку точки отправления и назначения  Выполняет выбор маршрута | Product  Order  Route |
| Navigator | Выполняет создание маршрутов  Хранит список плечей  Хранит список вершин | Route  Leg  Vertex |
| Order | Хранит список продуктов  Знает клиента  Хранит точку отправления и назначения  Хранит маршрут  Хранит свой номер | Product  Customer  Vertex  Route |
| Route | Знает список плечей в маршруте  Хранит местоположение | Leg |
| Leg | Хранит свои концы  Хранит ограничение по весу  Хранит время прохождения  Хранит стоимость  Хранит тип | Vertex |
| Vertex | Хранит обслуживаемые почтовые индексы  Хранит свой адрес |  |
| Product | Хранит стоимость  Хранит вес |  |

## **4.4. Описание собственного поведения объектов класса А**

В некоторых случаях объекты предметной области обладают собственным поведением. Например, классов лифт, переезд. Для описания их поведения следует составить схему состояний.

На данном этапе диаграмма служит для пояснения и прояснения поведения. Важно показать состояния, события и условия переходов.

Здесь нужно привести схему состояний, дать пояснения в тексте: обосновать, почему выделено каждое состояния, переходы между состояниями.

## **4.5. Описание алгоритма или процесса Б**

В других случаях для описания предметной области может быть важно показать алгоритм выполнения какого-либо действия или процесса. Для этого можно использовать диаграммы деятельности.

На диаграмме обозначить основные шаги процесса, последовательность действий, зависимости по управлению. Для назначения действий участникам (акторам, классам) рекомендуется использовать разделы (плавательные дорожки).

Здесь привести диаграмму деятельности, дать пояснения по действиям и разделам.

# **5. Приложение 1. Артефакты проектирования**

В приложении нужно привести фотографии / копии артефактов, использованных при разработке, которые подтверждают применение методов проектирования.

Хорошим примером являются карточки CRC, обсуждения на бумаге проектных решений.