МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Курганский государственный университет»

Кафедра информационных технологий
и методики преподавания информатики

ПРАКТИКУМ ПО ОБЪЕКТНО-ОРИЕНТИРОВАННОМУ ПРОЕКТИРОВАНИЮ

Методические рекомендации для студентов
направления 44.03.01 «Педагогическое образование»
(профиль «Информатика»)

Курган 2015

Кафедра: «Информационных технологий и методики преподавания информатики»

Дисциплина: «Объектно-ориентированное проектирование» (направление 44.03.01).

Составители: ст. преподаватель Ю.В. Адаменко.

Утверждены на заседании кафедры «27» августа 2014 г.

Рекомендованы методическим советом университета «19» декабря 2014 г.

Проектирование информационных систем с использованием среды объектно-ориентированного проектирования Rational Rose

Интерфейс Rational Rose

В **CASE**-средстве **Rational Rose** реализованы общепринятые стандарты на рабочий интерфейс программы.

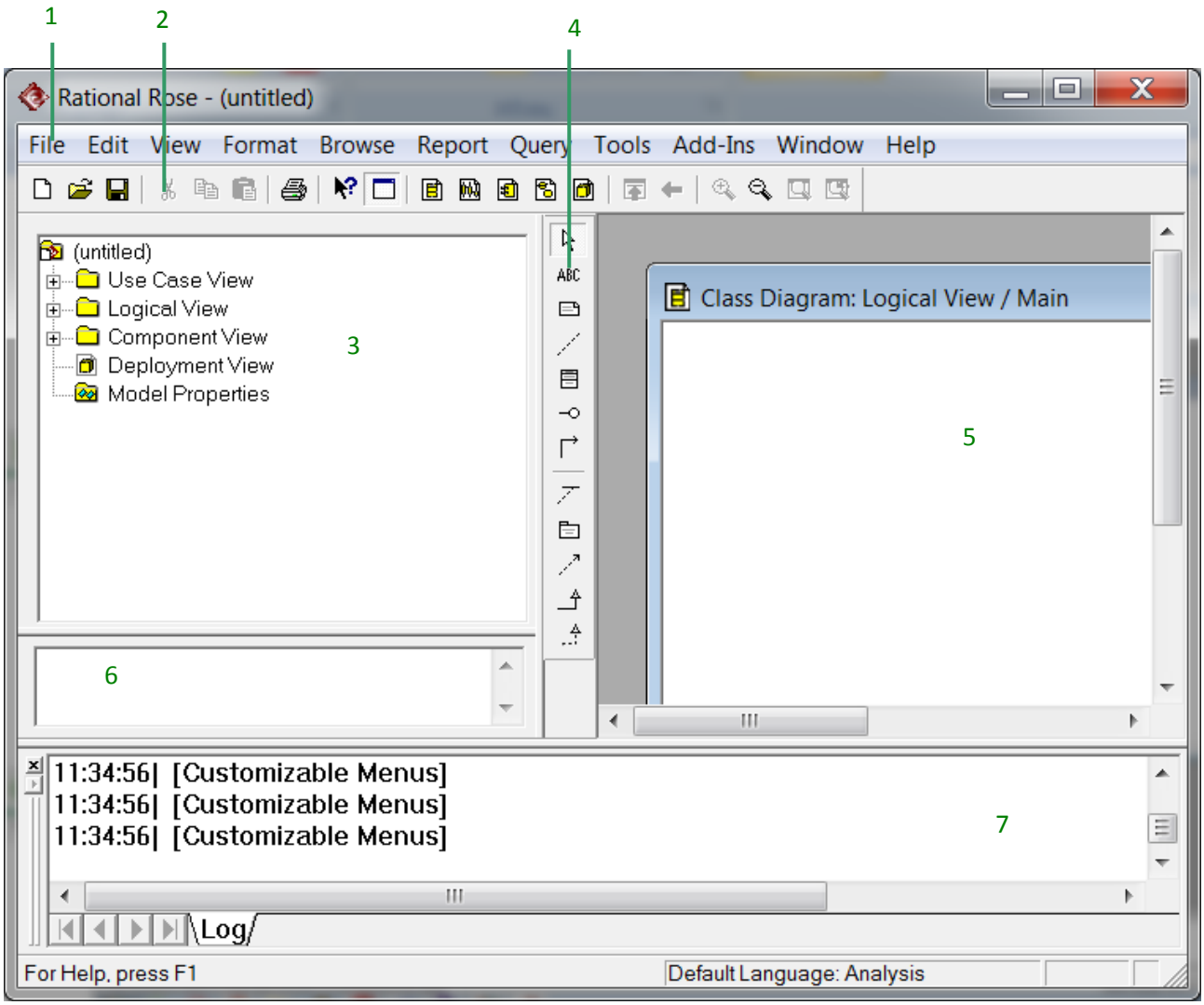

Рисунок 1 – Пользовательский интерфейс **Rational Rose**

Пользовательский интерфейс **Rational Rose** состоит из различных элементов, основными из которых являются:

1) главное меню программы;
2) стандартная панель инструментов;
3) окно браузера. Браузер организует представления модели в виде иерархической структуры, которая упрощает навигацию и позволяет отыскать любой элемент модели в проекте. При этом любой элемент, который разработчик добавляет в модель, сразу отображается в окне браузера. Браузер позволяет также организовывать элементы модели в пакеты и перемещать элементы между различными представлениями модели. При желании окно браузера можно расположить в другом месте рабочего интерфейса либо скрыть вовсе, используя для этого пункт меню **View** (Вид);

4) специальная панель инструментов. Располагается между окном браузера и окном диаграммы в средней части рабочего интерфейса. По умолчанию предлагается панель инструментов для построения диаграммы классов модели;

5) окно диаграммы. Является основной рабочей областью ее интерфейса, в которой визуализируются различные представления модели проекта. По умолчанию окно диаграммы располагается в правой части рабочего интерфейса, однако его расположение и размеры также можно изменить. Название диаграммы, которая располагается в данном окне, указывается в строке заголовка программы (самая верхняя строка программы) или, если окно не развернуто во весь экран, в строке заголовка окна диаграммы. Одновременно в окне диаграммы могут присутствовать несколько диаграмм, однако активной может быть только одна из них. Переключение между диаграммами можно осуществить выбором нужного представления на стандартной панели инструментов либо через пункт меню **Window** (Окно);

6) окно документации. Окно документации по умолчанию может не присутствовать на экране. В этом случае оно может быть активизировано через пункт меню **View → Documentation** (Вид → Документация), после чего появится ниже браузера. В окне документации активизируется та информация, которая относится к отдельному выделенному элементу диаграммы. При этом выделить элемент можно либо в окне браузера, либо в окне диаграммы;

7) окно журнала. Окно журнала (**Log**) предназначено для автоматической записи различной служебной информации, образующейся в ходе работы с программой. В журнале фиксируется время и характер выполняемых разработчиком действий, таких как обновление модели, настройка меню и панелей инструментов, а также сообщений об ошибках, возникающих при генерации программного кода. Активизировать окно журнала можно через меню **Window → Log** (Окно → Журнал).

Через меню **File → Open** (Файл → Открыть) можно ***открыть готовый проект*** для последующей модификации. В этом случае программа загрузит его со всеми имеющимися диаграммами, спецификациями и документацией.

Лабораторная работа № 1. Создание диаграммы вариантов использования

Задание 1. Фирма «Кухни Deluxe»

«Кухни Deluxe» - это маленькая компания, специализирующаяся на производстве стандартных и нестандартных кухонных шкафов. Компания началась с небольшой группы собравшихся вместе предпринимателей. Когда дело началось три года назад, поступало слишком мало заказов, и они вполне могли управляться с ними на бумаге. С ростом их репутации число заказов возрастало. Пришлось нанять новых рабочих, и за три года фирма выросла до магазина с более чем 50 сотрудниками.

Хотя это еще все-таки небольшая компания, но она выросла настолько, что уже не могла больше полагаться на обработку заказов вручную. Владельцы фирмы Иван Иванович и Мария Ивановы решили поговорить с Ириной Петровной, чтобы решить эту проблему. Ирина Петровна - это одна из небольшого числа специалистов по компьютерам, входящих в отделение этой фирмы, занимающееся информационными технологиями.

Иван Иванович пошел звонить Ирине Петровне.

«Совершенно очевидно, что нам требуется система по обработке заказов. Мы столкнулись с серьезным риском потерять клиентов».

«Согласна».

«Можешь ты разработать программу на Java, которая отслеживала бы заказы?»

«Пока что не волнуйтесь по поводу реализации. Давайте решим, чего вы хотите от системы?»

«Надо, чтобы она отслеживала заказы».

«Не мог бы ты быть более конкретным? Давай рассмотрим нынешний процесс».

«Хорошо, получив звонок, мы заполняем форму заказа. Мы передаем ее Петру Петровичу в магазин, он заполняет все необходимые документы и готовит отправку товара клиенту. Копию формы мы отдаем Дарье Павловне в бухгалтерию. Она вводит ее в бухгалтерскую систему и выписывает счет».

«И вы хотите, чтобы новая система поддерживала весь этот процесс?»

«Точно».

Из этого разговора Ирина Петровна смогла понять, что система должна обеспечивать возможность добавления новых заказов, изменения старых, выполнения заказов, проверки и возобновления инвентарных описей. При получении заказа система должна также послать сообщение бухгалтерской системе, которая выписывает счет. Если требуемого товара нет на складе, заказ должен быть отклонен. Затем Ирина Петровна преобразовала требования в ***диаграмму вариантов использования***, с помощью которой начала моделировать систему.

Создание диаграммы вариантов использования

Создайте диаграмму вариантов использования для системы обработки заказов. Требуемые для этого действия подробно перечислены далее. Готовая диаграмма вариантов использования должна выглядеть как на рисунке 2.

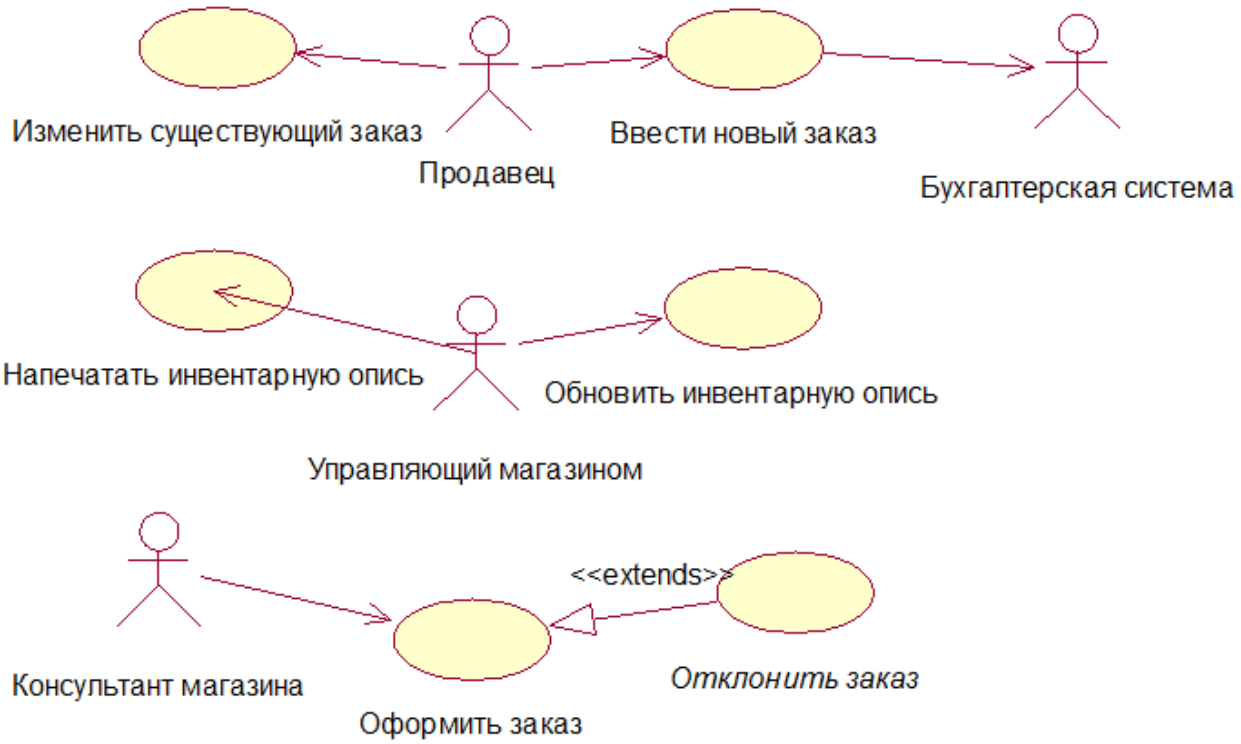

Рисунок 2 – Диаграмма вариантов использования для системы обработки заказов

Запуск приложения (**Пуск / Все программы / IBM Rational / IBM Rational Rose Enterprise Edition**).

I Создать диаграммы вариантов использования и действующих лиц:

1) дважды щелкните на **Главной диаграмме Вариантов Использования** (**Main**) в браузере, чтобы открыть ее (рисунок 3)

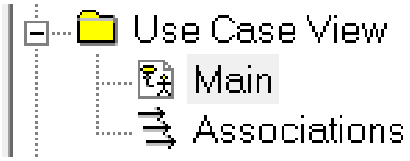

Рисунок 3 – Главная диаграмма Вариантов Использования

2) с помощью кнопки **Use Case**  (Вариант Использования) панели инструментов поместите на диаграмму новый вариант использования;
3) назовите этот новый вариант использования **Ввести новый заказ**;
4) повторите этапы 2 и 3, чтобы поместить на диаграмму остальные варианты использования: **Изменить существующий заказ**, **Напечатать инвентарную опись**, **Обновить инвентарную опись**, **Оформить заказ**, **Отклонить заказ**;
5) с помощью кнопки **Actor**  (Действующее лицо) панели инструментов поместите на диаграмму новое действующее лицо;
6) назовите его **Продавец**;
7) повторите шаги 5 и 6, поместив на диаграмму остальных действующих лиц: **Управляющий магазином**, **Консультант магазина**, **Бухгалтерская система**.

II Указать абстрактные варианты использования:

1) щелкните правой кнопкой мыши на варианте использования Отклонить заказ на диаграмме;
2) в открывшемся меню выберите пункт **Open Specification** (Открыть спецификацию);
3) пометьте контрольный переключатель **Abstract** (Абстрактный), чтобы сделать этот вариант использования абстрактным.

III Добавить ассоциации:

1) с помощью кнопки **Unidirectional Association**  (Однонаправленная ассоциация) панели инструментов нарисуйте ассоциацию между действующим лицом Продавец и вариантом использования Ввести новый заказ;
2) повторите этот этап, чтобы поместить на диаграмму остальные ассоциации.

IV Добавить связь расширения:

1) с помощью кнопки **Generalization**  панели инструментов нарисуйте связь между вариантом использования **Отклонить заказ** и вариантом использования **Оформить заказ**. Стрелка должна протянуться от первого варианта использования ко второму. Связь расширения означает, что вариант использования **Отклонить заказ** при необходимости дополняет функциональные возможности варианта использования **Оформить заказ**;
2) щелкните правой кнопкой мыши на новой связи между вариантами использования **Отклонить заказ** и **Оформить заказ**;
3) в открывшемся меню выберите пункт **Open Specification** (Открыть спецификацию);

4) в раскрывающемся списке стереотипов введите слово **extends** (расширение), затем нажмите **ОК**;
5) слово **<<extends**>> появится на линии данной связи.

V Добавить описания к вариантам использования:

1) выделите в браузере вариант использования **Ввести новый заказ**;
2) в окне документации введите следующее описание к этому варианту использования: Этот вариант использования дает продавцу возможность ввести новый заказ в систему;
3) с помощью окна документации введите описания ко всем остальным вариантам использования.

VI Добавить описания к действующему лицу:

1) выделите в браузере действующее лицо **Продавец;**
2) в окне документации введите для этого действующего лица следующее описание: «Продавец - это служащий, доставляющий и старающийся продать продукцию»;
3) с помощью окна документации введите описания к оставшимся действующим лицам.

VII Прикрепление файла к варианту использования:

1) для описания главного потока событий варианта использования **Ввести новый заказ** создайте файл **OrderFlow.doc**, содержащий следующий текст:
   а) продавец выбирает пункт «Создать новый заказ» из имеющегося меню;
   б) система выводит форму «Подробности заказа»;
   в) продавец вводит номер заказа, заказчика и то, что заказано;
   г) продавец сохраняет заказ;
   д) система создает новый заказ и сохраняет его в базе данных;
2) щелкните правой кнопкой мыши на варианте использования **Ввести новый заказ**;
3) в открывшемся меню выберите пункт **Open Specification** (Открыть спецификацию;
4) перейдите на вкладку файлов **Files**;
5) щелкните правой кнопкой мыши на белом поле и из открывшегося меню выберите пункт **Insert File** (Ввести файл);
6) укажите файл **OpenFlow.doc** и нажмите на кнопку **Open** (Открыть), чтобы прикрепить файл к варианту использования;
7) сохраните модель с именем **Кухни Deluxe**.

Задание 2. Система регистрации курсов университета. Постановка задачи

В начале каждого семестра студенты могут запросить каталог курсов, в который включен список учебных предметов, предлагаемых в данном семестре. Информация о курсах должна содержать фамилию преподавателя, название факультета и краткое описание, помогающее студентам сделать выбор.

Новая система позволит студенту выбрать четыре курса из предложенных в настоящем семестре. Кроме того, каждому студенту нужно дополнительно указать еще два варианта на случай, если курс будет переполнен или отменен. На курс не должно быть записано более 10 и менее 3 студентов. Курс, на который запишутся менее 3 студентов, будет отменен. По завершении регистрации система регистрации направляет информацию в систему оплаты для выставления счетов студентам.

Преподаватели должны иметь возможность онлайнового доступа к системе для указания курсов, которые они будут читать, и для просмотра списка записавшихся студентов.

В каждом семестре выделяется определенное время, в течение которого студенты могут менять свое расписание и получать доступ к системе для добавления или удаления выбранных курсов.

I Выделяем актеров:

Студент хочет зарегистрироваться на курсы. Преподаватель хочет выбрать курсы, которые он будет читать. Регистратор должен создать учебный план и составить каталог на семестр. Регистратор должен хранить информацию о курсах, преподавателях и студентах. Система оплаты должна получать необходимую информацию из системы регистрации. Из перечисленных вариантов выделяем актеров (студент, преподаватель, регистратор, система оплаты).

Этапы выполнения:

1) создайте новую модель;
2) в браузере щелкните правой кнопкой мыши по разделу **Use Case View** (Представление прецедентов). Выберите команду **New → Actor** (Создать → Действующее лицо). В список окна браузера будет добавлен новый актер с именем **New Class**;
3) выбрав новый пункт списка, введите нужное имя актера;
4) введите описание актеров:
    а) студент – человек, который регистрируется для посещения занятий в университете;
    б) преподаватель – человек, который читает лекции в университете;
    в) регистратор – человек, управляющий системой регистрации курсов;
    г) система оплаты – внешняя система, выполняющая функции расчетов за курсы.

II Выделяем прецеденты для системы:

В системе должны обеспечиваться следующие потребности:

а) актер **Студент** использует систему для регистрации на курсы;
б) по завершении выбора курсов в **Систему оплаты** должна поступить необходимая информация;
в) актер **Преподаватель** использует систему для выбора курсов, которые он будет читать в наступающем семестре, и должен получать от системы расписание занятий;
г) **регистратор** отвечает за составление каталога курсов на семестр, за управление информацией об учебных курсах, а также о студентах и преподавателях, работающих с системой.

Можно выделить следующие прецеденты (**Регистрация на курсы**, **Выбор курсов для преподавания**, **Запрос расписания курсов**, **Хранение информации о курсах**, **Хранение информации о преподавателях**, **Хранение информации о студентах**, **Создание каталогов курсов**). Выполните:

1) в браузере щелкните правой кнопкой мыши по разделу **Use Case View** (Представление прецедентов). Выберите команду **New → Use Case** (Создать → Прецедент). В список окна браузера будет добавлен новый прецедент с именем **NewUseCase**;
2) выбрав новый пункт списка, введите нужное название.

Внесите краткое описание прецедентов. Например, для прецедента **Регистрация на курсах** оно будет следующее: «Запускается студентом. Позволяет создавать, удалять, изменять и просматривать расписание студентов в указанном семестре».

Самостоятельно! Внесите краткое описание для остальных прецедентов.

III Создадим документ, описывающий поток событий:

Поток событий для прецедента – последовательность событий, необходимых для обеспечения требуемого поведения. Он должен определять:

1) когда и как прецедент начитается и заканчивается;
2) как он взаимодействует с актерами;
3) какие данные ему нужны;
4) нормальную последовательность событий для прецедента;
5) описание потоков в альтернативных и исключительных ситуациях.

Для создания документа, описывающего поток событий, должен использоваться стандартные шаблоны:

X Поток событий для прецедента <ИМЯ>.

X.1 Предусловия.

X.2 Главный поток.

X.3 Подпотоки (если применимы).

X.4 Альтернативные потоки.

Здесь X – число от 1 до количества прецедентов.

Создадим полный документ с описанием потока событий для прецедента **Выбор курсов для преподавания.**

1 Поток событий для прецедента **Выбор курсов для преподавания.**

1.1 Предусловия

Подпоток **создание учебных курсов** прецедента **Хранение информации о курсах** должен быть выполнен перед его началом.

1.2 Главный поток

Начало выполнения: подключение преподавателя к системе регистрации и ввод пароля. Система проверяет правильность пароля (E1) и просит преподавателя выбрать текущий или будущий семестр (E2). Преподаватель вводит нужный семестр. Система предлагает выбрать требуемую операцию: **добавить**, **удалить**, **просмотреть**, **напечатать** или **выйти**.

Если выбрана операция **добавить** (S1): выполняется поток **добавить учебный курс**.

Если выбрана операция **удалить** (S2): выполняется поток **удалить учебный курс**.

Если выбрана операция **просмотреть** (S3): выполняется поток **просмотреть расписание**.

Если выбрана операция **напечатать** (S4): выполняется поток **напечатать расписание**.

Если выбрана операция **выйти**: прецедент завершается.

1.3 Подпотоки

S1: **добавить учебный курс.** Система отображает окно, содержащее поле для ввода названия и номера предмета. Преподаватель вводит название и номер предмета (E3). Система отображает список учебных курсов для указанного предмета (E4). Преподаватель выбирает учебный курс. Система закрепляет за преподавателем выбранный учебный курс (E5). Затем прецедент начинается сначала.

S2: **удалить учебный курс.** Система отображает окно, содержащее поле для ввода названия и номера учебного курса. Преподаватель выбирает название и номер учебного курса (E6). Система удаляет взаимосвязь курса с преподавателем (E7). Затем прецедент начинается сначала.

S3: **просмотреть расписание.** Система получает (E8) и отображает информацию для всех учебных курсов, за которыми закреплен данный преподаватель: название предмета, номер предмета, номер учебного курса, день недели, время и место проведения занятий. Когда преподаватель отмечает, что просматривает список, прецедент начинается сначала.

S4: **напечатать расписание.** Система распечатывает расписание преподавателя (E9). Прецедент начинается сначала.

1.4 Альтернативные потоки

E1: введен неверный идентификационный номер преподавателя. Пользователь должен повторить ввод идентификационного номера или завершить прецедент.

E2: введен неверный семестр. Пользователь должен повторить ввод семестра или завершить прецедент.

E3: введено неверное название или номер предмета. Пользователь должен повторить ввод названия и номера предмета или завершить прецедент.

E4: список учебных курсов не может быть отображен. Пользователю сообщается, что данная команда в настоящий момент недоступна. Прецедент начинается сначала.

E5: преподаватель не может быть прикреплен к выбранному учебному курсу. Информация сохраняется, система осуществит прикрепление позже. Выполнение прецедента продолжается.

E6: введено неверное название или номер учебного курса. Пользователь должен повторить ввод названия и номера учебного курса или завершить прецедент.

E7: система не может удалить связь курса с преподавателем. Информация сохраняется, система удалит связь позже. Выполнение прецедента продолжается.

E8: система не может получить информацию о расписании. Прецедент начинается сначала.

E9: расписание не может быть распечатано. Пользователю сообщается, что данная операция в настоящий момент недоступна. Прецедент начинается сначала.

Сохраните в текстовом документе описание потока событий для прецедента **Выбор курсов для преподавания** и свяжите его с этим прецедентом.

IV Создадим диаграмму прецедентов:

1) дважды щелкните по пункту **Main** (**Главная диаграмма**) в разделе **Use Case View** в списке браузера;
2) в списке браузера выберите актера и перетащите его на диаграмму с помощью мыши. Переместите на диаграмму всех актеров;
3) в списке браузера выберите прецедент и перетащите его на диаграмму с помощью мыши. Переместите на диаграмму все прецеденты;
4) создайте коммуникативные ассоциации (связь между актером и прецедентом), подобные рисунку 4

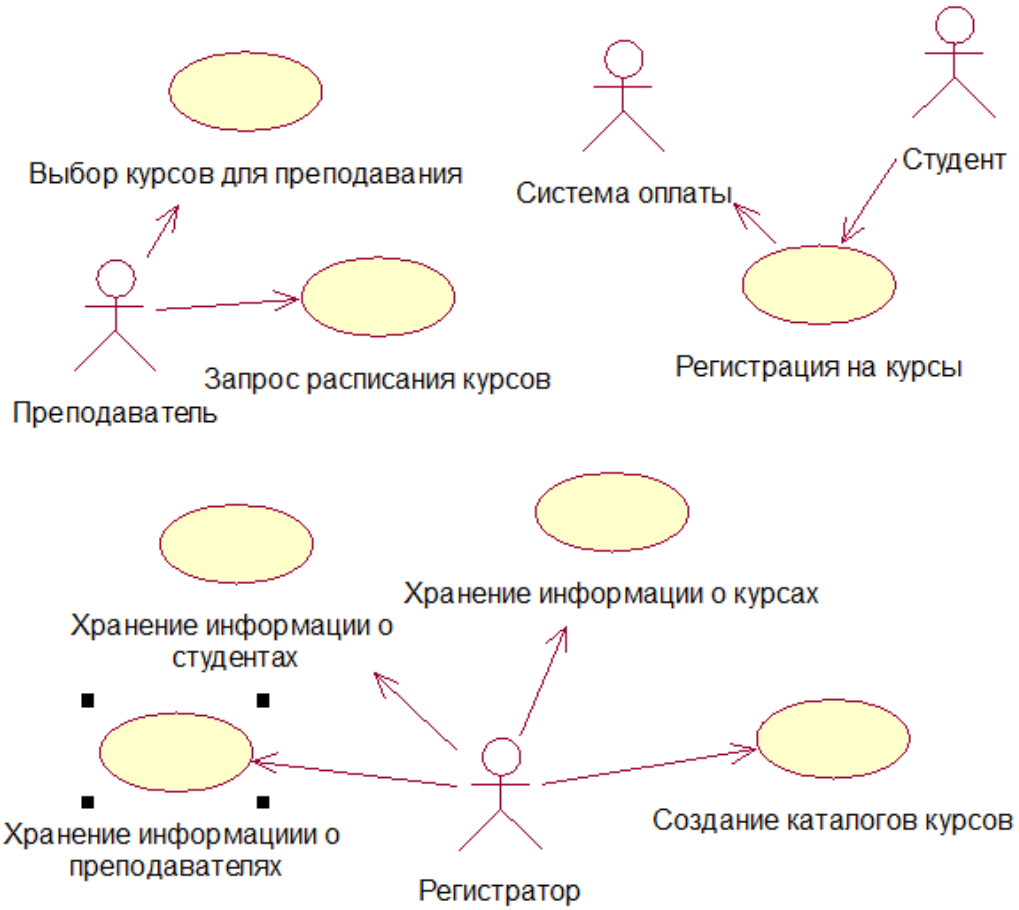

Рисунок 4 - Диаграмма вариантов использования для системы регистрации курсов университета

5) создадим дополнительную диаграмму прецедентов для преподавателя:
   а) щелкнуть правой кнопкой мыши по разделу **Use Case View** в списке браузера;
   б) в меню выбрать команду **New → Use Case Diagram** (Создать → Диаграмму прецедентов);
   в) ввести название диаграммы;
   г) открыть диаграмму и поместить необходимых актеров, прецеденты и связи (рисунок 5)

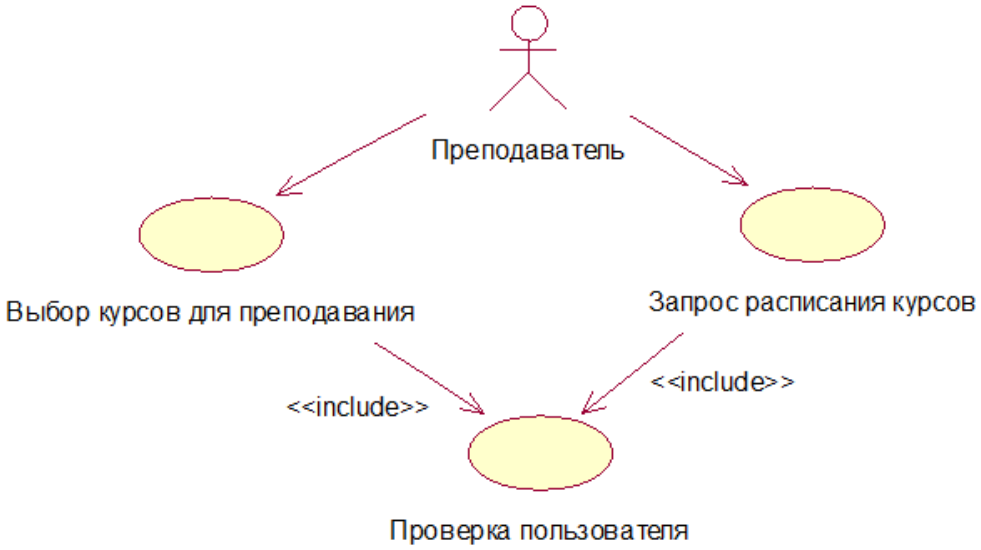

Рисунок 5 - Дополнительная диаграмма прецедентов

д) для создания отношения **include** (включает) нужно в диалоговом окне **Specification** в разделе **Stereotype** нужно выбрать значение **include**.

V Создадим диаграмму прецедентов в логическом представлении модели:

1) щелкните правой кнопкой мыши по папке **Logical View** в окне браузера;
2) в появившемся меню выберите команду **New → Use Case Diagram** (Создать → Диаграмма прецедентов). В разделе логического представления модели будет добавлена новая диаграмма прецедентов с названием **New Diagram**;
3) введите для новой диаграммы название **Realizations**.

VI Рассмотрим последовательность создания реализаций прецедентов:

1) дважды щелкните по диаграмме прецедентов **Realizations**, будет открыта диаграмма;
2) щелкните по кнопке **Use Case** на панели инструментов;
3) щелкните по диаграмме по прецедентов. В диаграмму и список браузера будет добавлен новый прецедент;
4) дважды щелкните по изображению прецеденту. На экране появится диалоговое окно параметров прецедента;
5) введите название прецедента (такое же, как у модели). Имя прецедента нужно указать в диалоговом окне параметров прецедента в поле **Name**. В списке **Stereotype** (Стереотип) выберите стереотип **use-case realization**. Закройте окно, нажатием на **ОК**. Диаграмма прецедентов **Realizations** показана на рисунке 6

Рисунок 6 - Диаграмма реализаций прецедентов

6) связь между прецедентами в логическом и use case-представлении отражается путем добавления прецедентов из представлений use case на диаграмму **Realizations** и соединения с их реализациями посредством однонаправленной ассоциативной связи с соответствующим стереотипом. На рисунке 7 показана связь реализаций с представлением прецедентов на диаграмме **Realizations.**

Создайте связь реализаций с остальными прецедентами. Сохраните модель с именем **Курсы**.

Самостоятельно! Создайте реализацию прецедентов для модели «**Кухни Deluxe**».

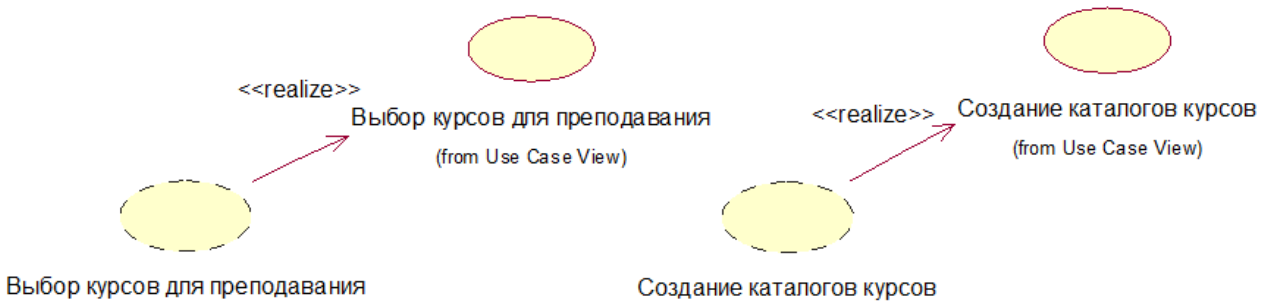

Рисунок 7 - Связь реализаций с представлением прецедентов

Лабораторная работа № 2. Создание диаграммы взаимодействия

Задание 1. Фирма «Кухни Deluxe»

Поговорив с Иваном Ивановичем, Ирина Петровна поняла, что должна делать система обработки заказов, разрабатываемая ей для фирмы «Кухни Deluxe». Она нарисовала диаграмму прецедентов. Изучив эту диаграмму, все пришли к согласию по поводу области применения системы.

Теперь наступило время анализа ее составных частей. Высший приоритет среди пользователей имеет вариант использования «Ввести новый заказ», он же связан с наибольшим риском. В связи с этим Ирина Петровна решила заняться им в первую очередь.

Она поговорила с Игнатом, заведующим отделом продаж. Они вдвоем обсудили поток событий, который будет реализовываться в варианте использования.

Получив нужную ей информацию, Ирина Петровна засела за описание сценариев. В результате ее описание выглядело следующим образом:

а) продавец вводит новый заказ;
б) продавец пытается ввести заказ, но товара нет на складе;
в) продавец пытается ввести заказ, но при его сохранении в базе данных произошла ошибка.

Затем она приступила к созданию диаграмм последовательности действий и диаграмм кооперации для сценария «Ввести новый заказ».

Создание диаграмм Взаимодействия

Диаграмма Взаимодействия только одна из диаграмм, необходимых для моделирования варианта использования **Ввести новый заказ**. Она соответствует успешному варианту хода событий. Для описания того, что случится, если возникнет ошибка, или если пользователь выберет другие действия из предложенных, придется разработать другие диаграммы. Каждый альтернативный поток варианта использования может быть промоделирован с помощью своих собственных диаграмм Взаимодействия.

I Настройка:

1) в меню модели выберите пункт **Tools → Options** (Инструменты → Параметры);
2) перейдите на вкладку диаграмм;
3) контрольные переключатели **Sequence Numbering**, **Collaboration Numbering** и **Focus of Control** должны быть помечены;
4) нажмите **ОК**, чтобы выйти из окна параметров.

II Создание диаграммы Последовательности:

1) щелкните правой кнопкой мыши на **Логическом представлении** браузера (**Logical View**);

2) в открывшемся меню выберите пункт **New → Sequence Diagram** (Создать → Диаграмма последовательности действий);
3) назовите новую диаграмму **Ввод заказа**;
4) дважды щелкните на ней, чтобы открыть ее.

III Добавление на диаграмму действующего лица и объектов:

1) перетащите действующее лицо **Продавец** с браузера на диаграмму;
2) на панели инструментов нажмите кнопку **Object**  (Объект);
3) щелкните мышью в верхней части диаграммы, чтобы поместить туда новый объект;
4) назовите объект **Форма выбора варианта заказа**;
5) повторите этапы 3 и 4, чтобы поместить на диаграмму следующие объекты: **Форма Детали заказа**, **Заказ №1234**.

IV Добавление сообщений на диаграмму

1) на панели инструментов нажмите кнопку **Object Message**  (Сообщение объекта);
2) проведите мышью от линии жизни актера **Продавец** к линии жизни объекта **Форма выбора варианта заказа**;
3) выделив сообщение, введите его имя **Создать новый заказ**;
4) повторите этапы 2 и 3, чтобы поместить на диаграмму дополнительные сообщения:
   а) **Открыть форму** (между **Выбором варианта заказа** и **Деталями заказа**);
   б) **Ввести номер заказа, заказчика и число заказываемых предметов** (между **Продавцом** и **Деталями заказа**);
   в) **Сохранить заказ** (между **Продавцом** и **Деталями заказа**);
   г) **Создать пустой заказ** (между **Деталями заказа** и **Заказом №1234**);
   д) **Ввести номер заказа, заказчика и число заказываемых предметов** (между **Деталями заказа** и **Заказом №1234**);
   е) **Сохранить заказ** (между **Деталями заказа** и **Заказом №1234**).

   Готовая диаграмма Последовательности представлена на рисунке 8.

Теперь надо позаботиться об управляющих объектах и взаимодействии с базой данных. Как видно из диаграммы, объект **Детали заказа** имеет множество ответственностей, с которыми лучше всего мог бы справиться управляющий объект. Кроме того, новый заказ должен сохранять себя в базе данных сам. Вероятно, эту обязанность лучше было бы переложить на другой объект.

V Добавление на диаграмму дополнительных объектов:

1) на панели инструментов нажмите кнопку **Object**;
2) щелкните мышью между объектами **Детали заказа** и **Заказ №1234**, чтобы поместить туда новый объект;
3) введите имя объекта - **Менеджер заказов**;
4) на панели инструментов нажмите кнопку **Object**;
5) новый объект расположите справа от **Заказа №1234**;
6) введите его имя **Менеджер транзакций**.

VI Назначение ответственностей объектам:

1) выделите сообщение 5 **Создать пустой заказ**;
2) нажмите комбинацию клавиш **CTRL + D**, чтобы удалить это сообщение;
3) повторите этапы 1 и 2, чтобы удалить два последних сообщения: **Вести номер заказа, заказчика и число заказываемых предметов, Сохранить заказ;**

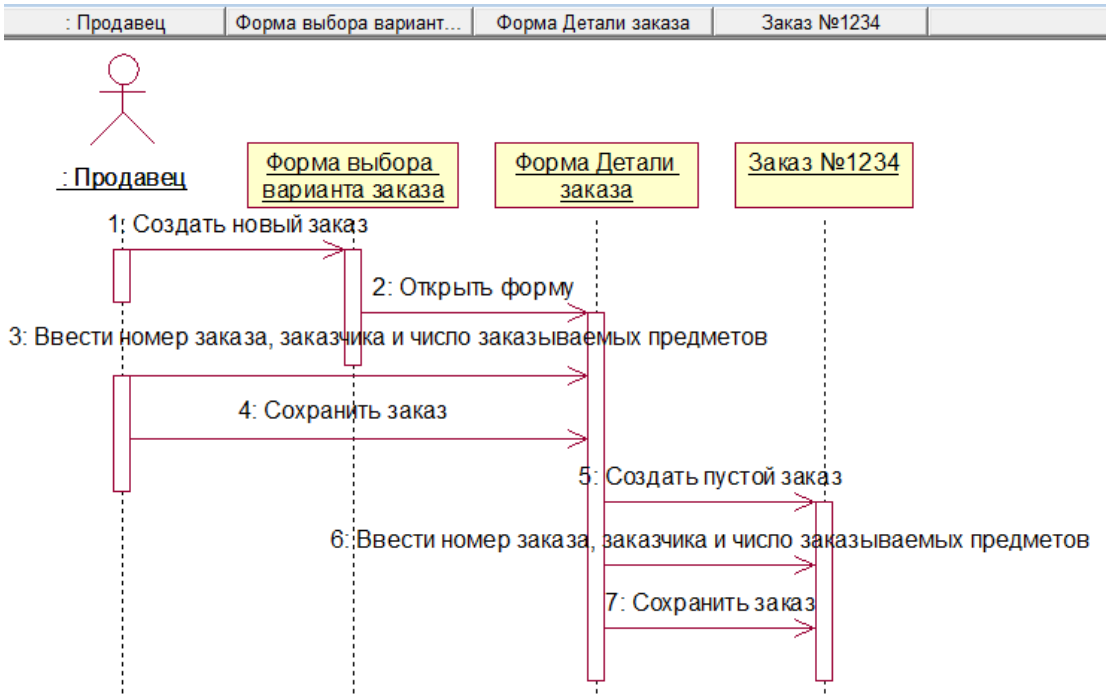

Рисунок 8 - Диаграмма Последовательности ввода нового заказа после завершения первого этапа работы

4) на панели инструментов нажмите кнопку **Object Message**;
5) поместите на диаграмму новое сообщение, расположив его под сообщением 4 между **Деталями заказа** и **Менеджером заказов**;
6) назовите его **Сохранить заказ**;
7) повторите этапы 4 - 6, добавив сообщения с шестого по девятое и назвав их:
   а) **Создать новый заказ** - между **Менеджером заказов** и **Заказом №1234**;
   б) **Ввести номер заказа, заказчика и число заказываемых предметов** - между **Менеджером заказов** и **Заказом №1234**;
   в) **Сохранить заказ** - между **Менеджером заказов** и **Менеджером транзакций**;
   г) **Информация о заказе** - между **Менеджером транзакций** и **Заказом №1234**;
8) на панели инструментов нажмите кнопку **Message to Self** (Сообщение себе);
9) щелкните на линии жизни объекта **Менеджер транзакций** ниже сообщения 9, добавив туда рефлексивное сообщение. Назовите его **Сохранить информацию о заказе в базе данных**.

Теперь диаграмма Последовательности должна выглядеть как на рисунке 9.

VII Соотнесение объектов с классами:

1) щелкните правой кнопкой мыши на объекте **Выбор варианта заказа**;
2) в открывшемся меню выберите пункт **Open Specification** (Открыть спецификацию);
3) в раскрывающемся списке классов выберите пункт **New** (Создать). Появится окно спецификации классов;
4) в поле имени введите имя **Выбор заказа**;
5) щелкните на кнопке **ОК**. Вы вернетесь к окну спецификации объекта;
6) в списке классов выберите теперь класс **Выбор заказа**;

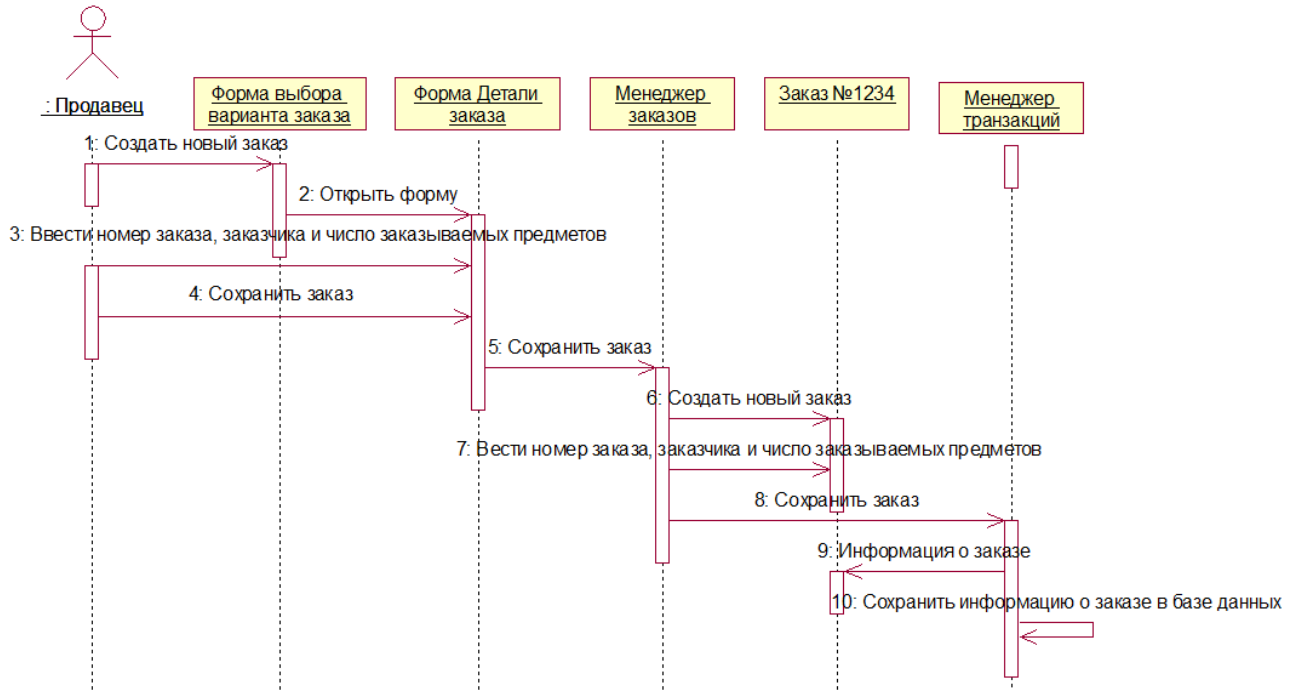

Рисунок 9 - Диаграмма Последовательности с новыми объектами

7) щелкните на кнопке **ОК**, чтобы вернуться к диаграмме. Теперь объект называется **Форма выбора варианта заказа : Выбор заказа**;
8) для соотнесения остальных объектов с классами повторите этапы с 1 по 7:
   а) класс **Детали заказа** соотнесите с объектом **Форма Детали заказа**;
   б) класс **Управляющий заказами** - с объектом **Менеджер заказов**;
   в) класс **Заказ** - с объектом **Заказ №1234**;
   г) класс **Управляющий транзакциями** - с объектом **Менеджер транзакций**.

После завершения этих действий ваша диаграмма должна выглядеть как на рисунке 10.

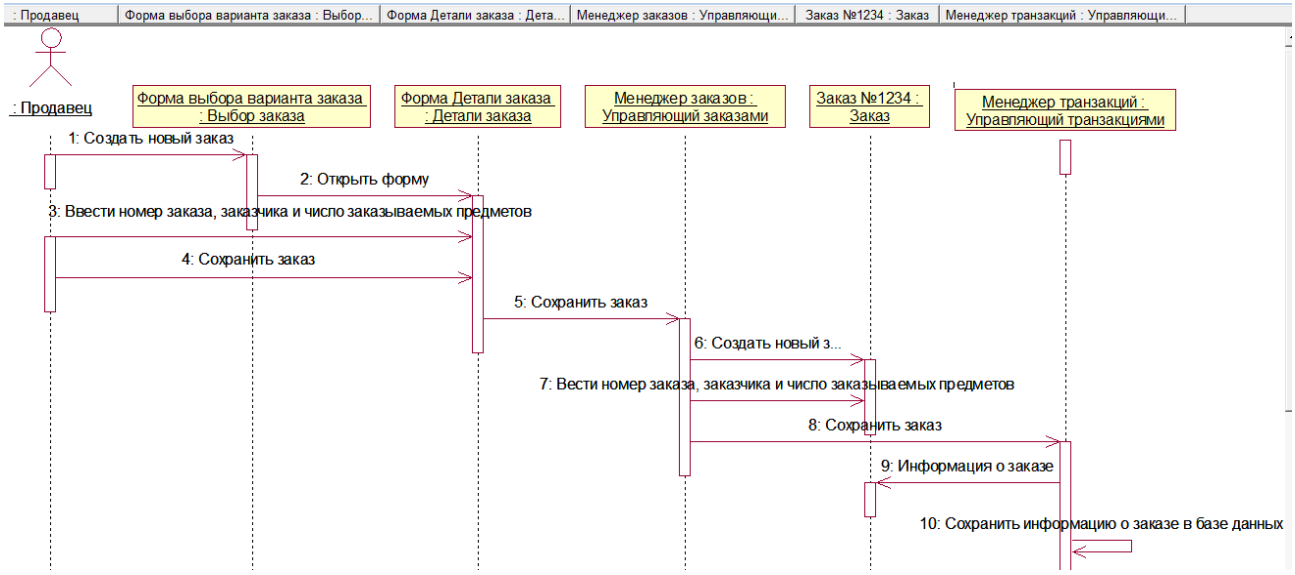

Рисунок 10 - Диаграмма Последовательности с именами классов

VIII Соотнесение сообщений с операциями

1) щелкните правой кнопкой на сообщении 1: **Создать новый заказ**;
2) в открывшемся меню выберите пункт **<new operation>** (создать операцию). Появится окно спецификации операции;
3) в поле имени введите имя операции – **Create**;
4) нажмите на кнопку **ОК**, чтобы закрыть окно спецификации операции и вернуться на диаграмму;
5) еще раз щелкните правой кнопкой мыши на сообщении 1;
6) в открывшемся меню выберите новую операцию **Create**();
7) повторите сообщения с 1 по 6, пока не соотнесете с операциями все остальные сообщения:
   а) сообщение 2: **Открыть форму** соотнесите с операцией **Open**();
   б) сообщение 3: **Ввести номер заказа, заказчика и число заказываемых предметов** - с операцией **SubmitInfo**();
   в) сообщение 4:**Сохранить заказ** - с операцией **Save**();
   г) сообщение 5:**Сохранить заказ** - с операцией **SaveOrder**();
   д) сообщение 6:**Создать пустой заказ** - с операцией **Create**();
   е) сообщение 7: **Ввести номер заказа, заказчика и число заказываемых предметов** - с операцией **SetInfo**();
   ж) сообщение 8:**Сохранить заказ** - с операцией **SaveOrder**();
   з) сообщение 9:**Информация о заказе** - с операцией **GetInfo**();
   и) сообщение 10:**Сохранить информацию о заказе в базе данных** - с операцией **Commit ()**.
   Ваша диаграмма должна выглядеть как на рисунке 11.

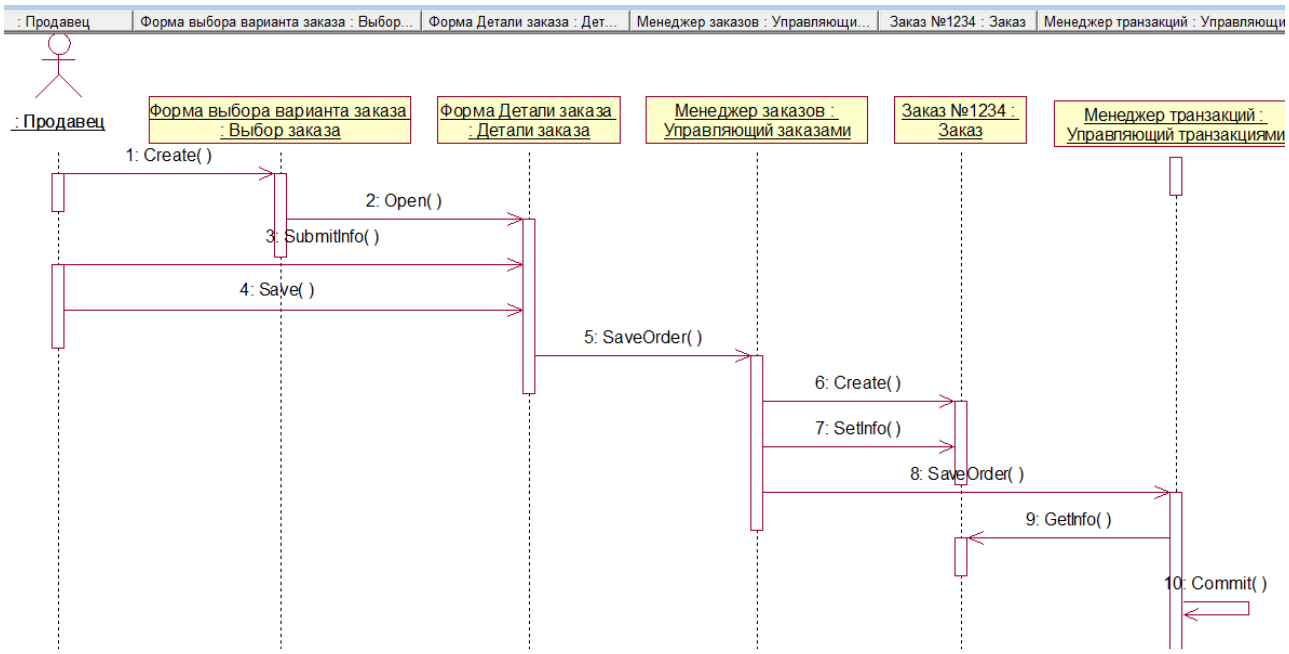

Рисунок 11 - Диаграмма Последовательности с показанными на ней операциями

IX Создание диаграммы взаимодействия

Для создания диаграммы взаимодействия достаточно просто нажать клавишу **F5**. Ваша диаграмма должна выглядеть как на рисунке 12.

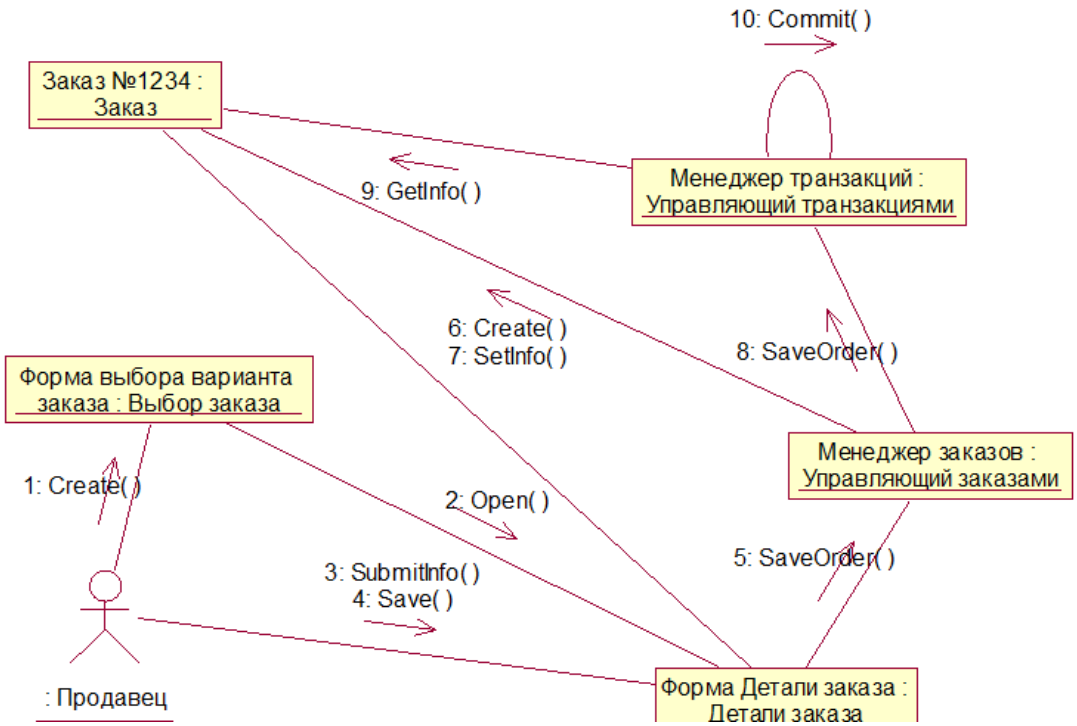

Рисунок 12 - Кооперативная диаграмма с показанными на ней операциями

Задание 2. Система регистрации курсов университета

1 Создайте диаграмму последовательности действий для сценария **Создание учебного предмета** (рисунок 13).

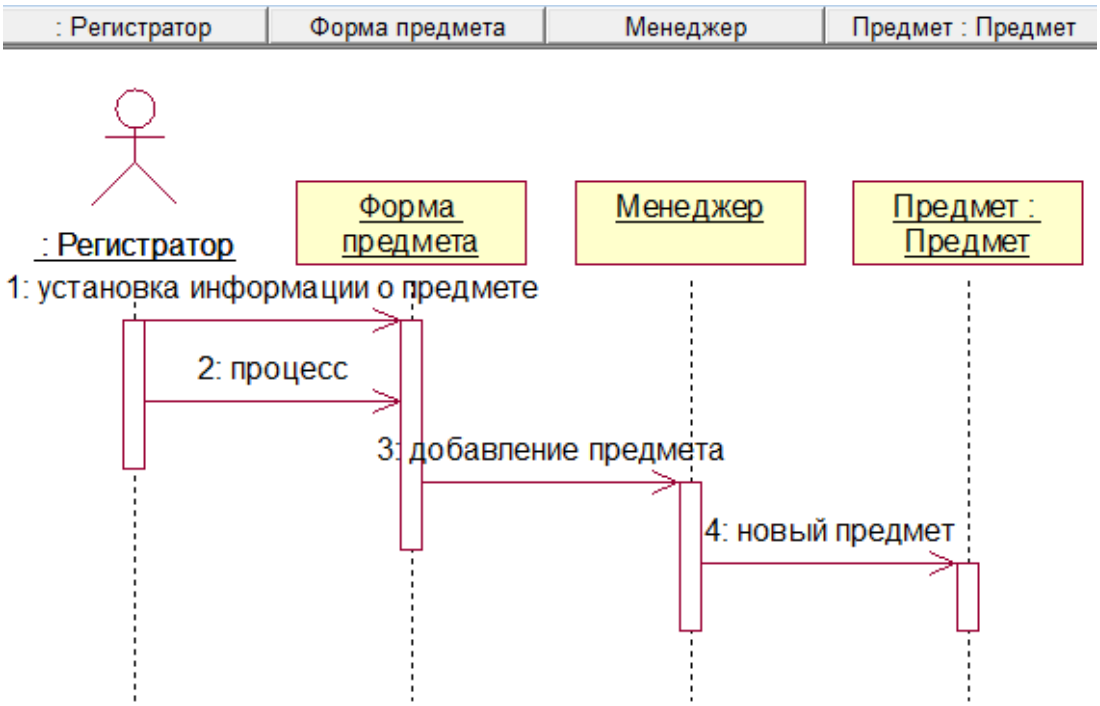

Рисунок 13 - Диаграмма последовательности действий для сценария **Создание учебного предмета**

2 Создайте диаграмму взаимодействий для сценария **Создание учебного предмета** (рисунок 14).

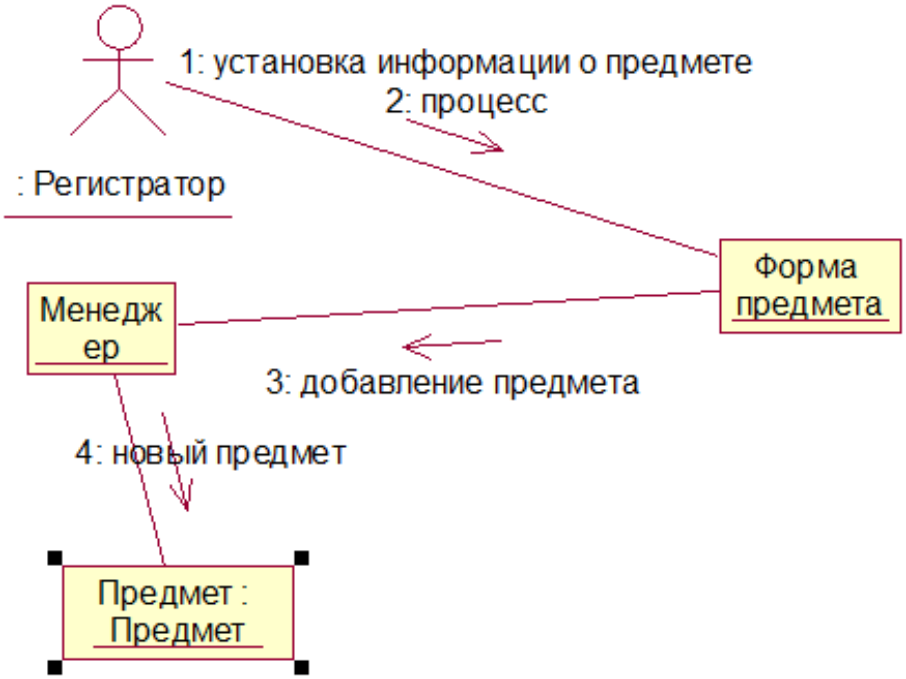

Рисунок 14 - Диаграмма взаимодействий для сценария **Создание учебного предмета**

Лабораторная работа № 3. Создание диаграмм классов

Задание 1. Фирма «Кухни Deluxe»

Изучив диаграммы Взаимодействия, Иван Иванович понял, что система соответствует деловым потребностям компании. После этого Ирина Петровна пришла к руководителю группы разработчиков Светлане:

«Вот диаграммы Взаимодействия, описывающие процесс ввода нового заказа».

«Прекрасно. Приступаем к разработке».

Светлана взглянула на классы модели Rose. Она решила объединить их в пакеты по стереотипу. Она создала пакеты **Сущности**, **Границы** и **Управление**, поместив в них соответствующие классы. Затем для каждого пакета были созданы **диаграммы Классов**; кроме того, на **Главной диаграмме** были показаны пакеты и на диаграмме **Ввода нового заказа** - все классы этого варианта использования.

Создание диаграммы Классов

Объедините обнаруженные нами классы в пакеты. Создайте диаграмму Классов для отображения пакетов, диаграммы Классов для представления классов в каждом пакете и диаграмму Классов для представления всех классов варианта использования «**Ввести новый заказ**».

I Настройка:

1) в меню модели выберите пункт **Tools → Options** (Инструменты → Параметры);
2) перейдите на вкладку диаграмм;
3) убедитесь, что помечен контрольный переключатель **Show Stereotypes** (Показать стереотипы);
4) убедитесь, что помечены контрольные переключатели **Show All Attributes** (Показать все атрибуты) и **Show All Operations** (Показать все операции);

5) убедитесь, что не помечены переключатели **Suppress Attributes** (Подавить вывод атрибутов) и **Suppress Operations** (Подавить вывод операций).

II Создание пакетов:

1) щелкните правой кнопкой мыши на **Логическом представлении** браузера;
2) в открывшемся меню выберите пункт **New Package** (Создать → Пакет). *Пакет* в логическом представлении модели – это набор классов и других связанных пакетов;
3) назовите новый пакет **Сущности**;
4) повторите этапы с первого по третий, создав пакеты **Границы** и **Управление**.

III Создание Главной диаграммы Классов:

1) дважды щелкните на **Главной диаграмме Классов** прямо под **Логическим представлением** браузера, чтобы открыть ее;
2) перетащите пакет **Сущности** из браузера на диаграмму;
3) перетащите пакеты **Границы** и **Управление** из браузера на диаграмму;

Главная диаграмма Классов должна выглядеть как на рисунке 15.

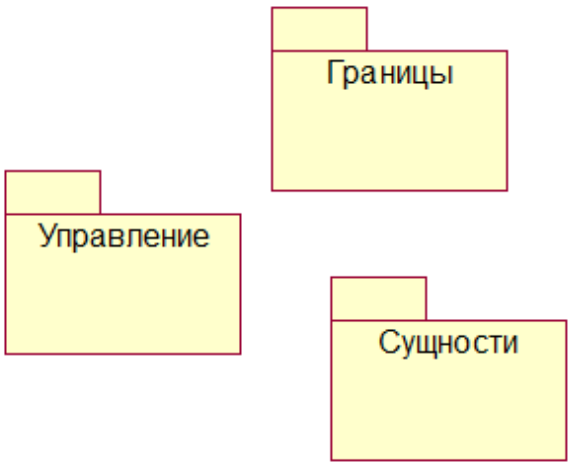

Рисунок 15 - Главная диаграмма Классов системы обработки заказов

IV Создание диаграммы Классов для сценария «Ввести новый заказ» со всеми классами:

1) щелкните правой кнопкой мыши на **Логическом представлении** браузера;
2) в открывшемся меню выберите пункт **New → Class Diagram** (Создать → Диаграмму Классов);
3) назовите новую диаграмму Классов **Введение нового заказа**;
4) щелкните в браузере на этой диаграмме дважды, чтобы открыть ее;
5) перетащите из браузера все классы (Выбор заказа, Детали заказа, Заказ, Управляющий заказами, Управляющий транзакциями).

Диаграмма Классов должна выглядеть как на рисунке 16.

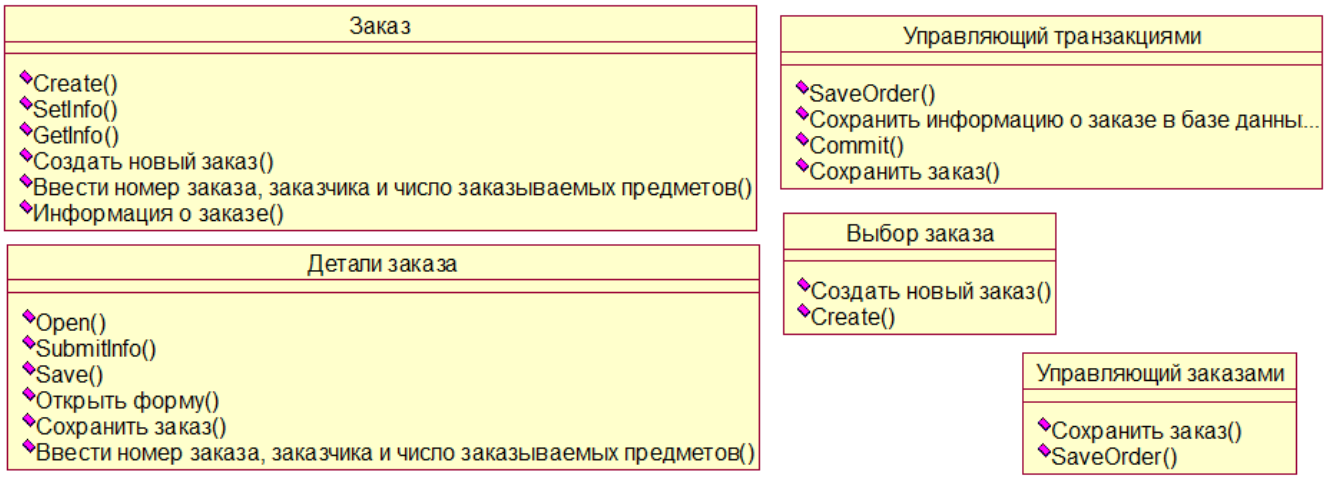

Рисунок 16 - Диаграмма Классов **Введение нового заказа**

V Добавление стереотипов к классам:

1) щелкните правой кнопкой мыши на классе **Выбор заказа** диаграммы;
2) в открывшемся меню выберите пункт **Open Specification** (Открыть спецификацию);
3) в поле стереотипа введите слово **boundary** (граничные классы обеспечивают взаимодействие между окружающей средой и внутренними элементами системы);
4) нажмите на кнопку **ОК**;
5) щелкните правой кнопкой мыши на классе **Детали заказа** диаграммы;
6) в открывшемся меню выберите пункт **Open Specification** (Открыть спецификацию);
7) в раскрывающемся списке в поле стереотипов теперь будет стереотип **boundary**. Укажите его;
8) нажмите на кнопку **ОК;**
9) повторите этапы 1 - 4, связав классы **Управляющий заказами** и **Управляющий транзакциями** со стереотипом **control** (управляющие классы служат для моделирования последовательного поведения одного или нескольких прецедентов и координации событий, реализующих заложенное в них поведение), а класс **Заказ** - со стереотипом **entity** (классы сущности используются для моделирования данных и поведения с длинным жизненным циклом, представляет сущности реального мира или внутренние элементы системы).

Теперь диаграмма Классов должна выглядеть как на рисунке 17.

VI Объединение классов в пакеты:

1) перетащите в браузере класс **Выбор заказа** на пакет **Границы**;
2) перетащите класс **Детали заказа** на пакет **Границы**;
3) перетащите классы **Управляющий заказами** и **Управляющий транзакциями** на пакет **Управление**;
4) перетащите класс **Заказ** на пакет **Сущности**.

VII Добавление диаграмм Классов к каждому пакету:

1) щелкните правой кнопкой на пакете **Границы** браузера;
2) в открывшемся меню выберите пункт **New → Class Diagram** (Создать → Диаграмму Классов);

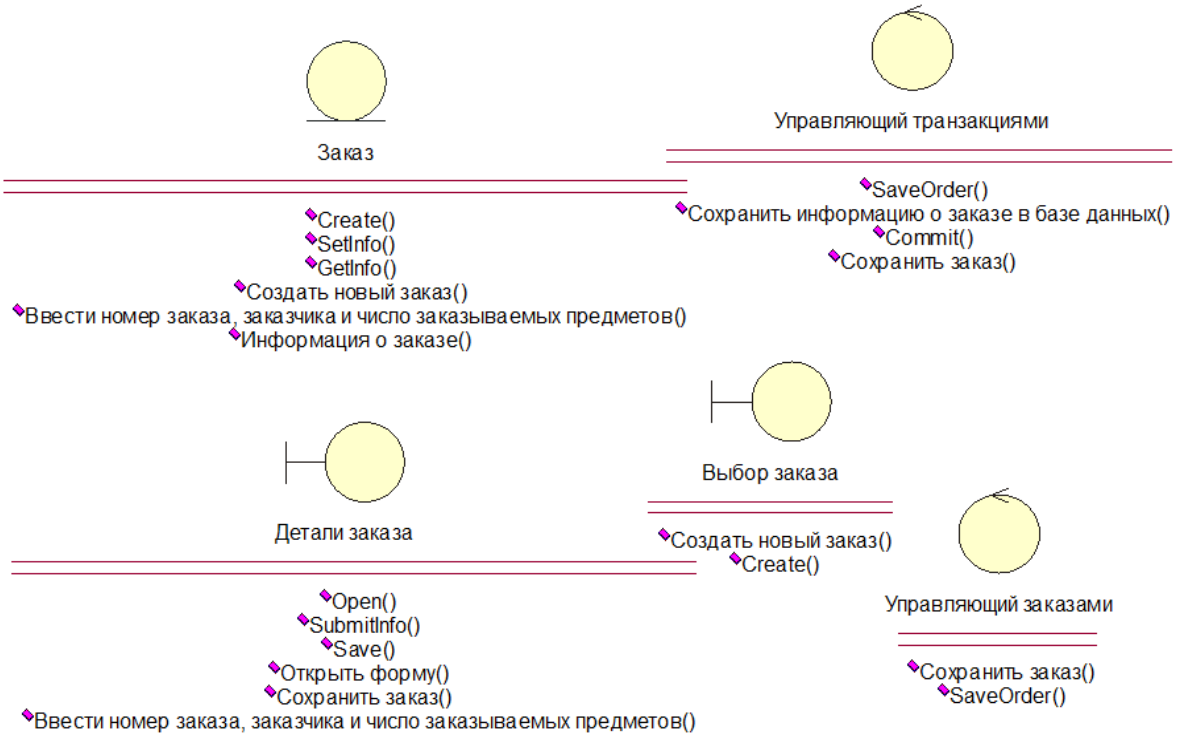

Рисунок 17 - Стереотипы классов для варианта использования **Ввести новый заказ**

3) введите имя новой диаграммы - **Main** (Главная);
4) дважды щелкните мышью на этой диаграмме, чтобы открыть ее;
5) перетащите на нее из браузера классы **Выбор заказа** и **Детали заказа**;
6) закройте диаграмму;
7) щелкните правой кнопкой на пакете **Сущности** браузера;
8) в открывшемся меню выберите пункт **New → Class Diagram** (Создать → Диаграмму Классов);
9) введите имя новой диаграммы - **Main** (Главная);
10) дважды щелкните мышью на этой диаграмме, чтобы открыть ее;
11) перетащите на нее из браузера класс **Заказ**;
12) закройте диаграмму;
13) щелкните правой кнопкой на пакете Управление браузера;
14) в открывшемся меню выберите пункт **New → Class Diagram** (Создать → Диаграмму Классов);
15) введите имя новой диаграммы - **Main** (Главная);
16) дважды щелкните мышью на этой диаграмме, чтобы открыть ее;
17) перетащите на нее из браузера классы **Управляющий заказами** и **Управляющий транзакциями**;
18) закройте диаграмму.

VIII Создание диаграмм классов (учет новых требований):

После того, как Светлана разработала диаграмму Классов для варианта использования **Ввести новый заказ**, она начала заполнять ее подробностями. В качестве языка программирования был выбран C++, что позволило добавить к классам параметры операций, типы данных и типы возвращаемых значений.

Для определения атрибутов Светлана вновь обратилась к потоку событий. В результате к классу **Заказ** диаграммы Классов были добавлены атрибуты **Номер**

заказа и **Имя клиента**. Она просмотрела также список заказываемых товаров. Так как в одном заказе можно указать большое количество товаров, и у каждого из них имеются свои собственные данные и поведение, Светлана решила моделировать их как самостоятельные классы, а не как атрибуты класса **Заказ**.

Чтобы привести модель в соответствие с новыми идеями, пришлось обновить диаграмму Последовательностей, как показано на рисунке 18.

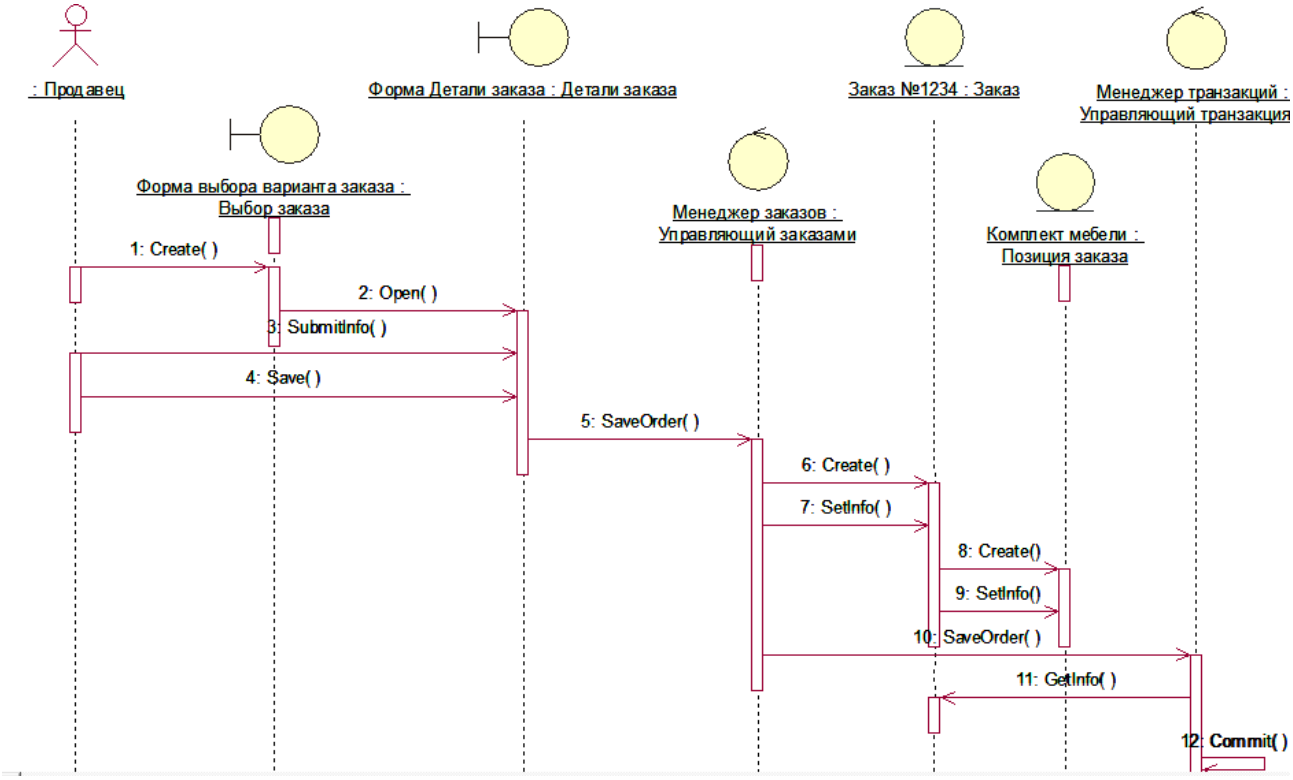

Рисунок 18 - Обновленная диаграмма Последовательностей

В этот момент Иван Иванович решил изменить требования: «Нам надо отслеживать дату заказа и дату его выполнения. Кроме того, так как у нас появились новые поставщики, слегка изменилась процедура инвентаризации».

Сначала Светлана документировала новые требования относительно дат и рассмотрела изменения в процедуре инвентаризации «на высоком уровне». Поскольку в данный момент она работала над вариантом использования **Ввести новый заказ**, ее больше всего интересовало, как эти процедурные изменения повлияют на данный вариант использования. Работа с вариантом использования **Провести инвентаризацию** была запланирована на следующий месяц, тогда она и позаботится о деталях соответствующих процедур. Оказалось, что хотя они чрезвычайно сильно повлияют на вариант использования **Провести инвентаризацию**, но совсем не отразятся на варианте использования **Ввести новый заказ**.

Новые требования, связанные с датами, привели к необходимости ввести пару новых атрибутов в класс **Заказ**. После этого модель опять стала соответствовать последним предъявленным к системе требованиям.

Добавление атрибутов и операций

Добавим атрибуты и операции к классам диаграммы Классов **Ввести новый заказ**. Для атрибутов и операций используем специфические для языка

особенности. Установим параметры так, чтобы показывать все атрибуты, все операции и их сигнатуры. Видимость покажем с помощью нотации UML.

I Настройка:

1) в меню модели выберите пункт **Tools → Options**;
2) перейдите на вкладку **Diagram**;
3) убедитесь, что переключатель **Show Visibility** помечен;
4) убедитесь, что переключатель **Show Stereotypes** помечен;
5) убедитесь, что переключатель **Show Operation Signatures** помечен;
6) убедитесь, что переключатели **Show All Attributes** и **Show All Operations** помечены;
7) убедитесь, что переключатели **Suppress Attributes** и **Suppress Operations** не помечены;
8) перейдите на вкладку **Notation**;
9) убедитесь, что переключатель **Visibility as Icons** не помечен.

II Добавление нового класса:

1) найдите в браузере диаграмму Классов варианта использования **Ввести новый заказ**;
2) щелкните на ней дважды, чтобы ее открыть;
3) нажмите кнопку **Class**  панели инструментов;
4) щелкните мышью внутри диаграммы, чтобы поместить там новый класс;
5) назовите его **OrderItem** (ПозицияЗаказа);
6) назначьте этому классу стереотип **Entity**;
7) в браузере перетащите класс в пакет **Сущности**.

III Добавление атрибутов:

1) щелкните правой кнопкой мыши на классе **Order** (Заказ);
2) в открывшемся меню выберите пункт **New Attribute** (Создать атрибут);
3) введите новый атрибут **OrderNumber : Integer** (Номер Заказа);
4) нажмите клавишу **Enter**;
5) введите следующий атрибут **CustomerName : String** (Наименование Заказчика);
6) повторите этапы 4 и 5, добавив атрибуты **OrderDate : Date** (ДатаЗаказа) и **OrderFillDate : Date** (ДатаЗаполненияЗаказа);
7) щелкните правой кнопкой мыши на классе **OrderItem** (Позиция заказа);
8) в открывшемся меню выберите пункт **New Attribute** (Создать атрибут);
9) введите новый атрибут **ItemID : Integer** (ИдентификаторПредмета);
10) нажмите клавишу **Enter**;
11) введите следующий атрибут **ItemDescription : String** (ОписаниеПредмета).

IV Добавление операций к классу OrderItem:

1) щелкните правой кнопкой мыши на классе **OrderItem**;
2) в открывшемся меню выберите пункт **New Operation** (Создать операцию);
3) введите новую операцию **Create**;
4) нажмите клавишу **Enter**;
5) введите следующую операцию **SetInfo;**
6) нажмите клавишу **Enter**;
7) введите следующую операцию **GetInfo**.

V Подробное описание операций с помощью диаграммы Классов:

1) щелкните мышью на классе **Order**, выделив его таким способом;

2) щелкните на этом классе еще один раз, чтобы переместить курсор внутрь;
3) отредактируйте операцию **Create**(), чтобы она выглядела следующим образом: **Create() : Boolean;**
4) отредактируйте операцию **SetInfo**(), чтобы она выглядела следующим образом: **SetInfo(OrderNum : Integer, Customer : String, OrderDate : Date, FillDate : Date) : Boolean;**
5) отредактируйте операцию **GetInfo**(), чтобы она выглядела следующим образом: **GetInfo() : String.**

VI Подробное описание операций с помощью браузера:

1) найдите в браузере класс **OrderItem**;
2) чтобы раскрыть этот класс, щелкните на значке «+» рядом с ним. В браузере появятся его атрибуты и операции;
3) дважды щелкните на операции **GetInfo()**, чтобы открыть окно ее спецификации;
4) в раскрывающемся списке **Return type** (возвращаемый класс) укажите **String**;
5) щелкните на кнопке **ОК**, закрыв окно спецификации операции;
6) дважды щелкните в браузере на операции **SetInfo** класса **OrderItem**, чтобы открыть окно ее спецификации;
7) в раскрывающемся списке **Return type** укажите **Boolean**;
8) перейдите на вкладку **Detail** (Подробно);
9) щелкните правой кнопкой мыши на белом поле в области аргументов, чтобы добавить туда новый параметр;
10) в открывшемся меню выберите пункт **Insert**. Rose добавит туда аргумент под названием **argname**;
11) щелкните один раз на этом слове, чтобы выделить его, и измените имя аргумента на **ID**;
12) щелкните на колонке **Type**, открыв раскрывающийся список типов. В нем выберите тип **Integer**;
13) щелкните на колонке **Default**, чтобы добавить значение аргумента по умолчанию. Введите туда число **0**;
14) нажмите на кнопку **ОК**, закрыв окно спецификации операции;
15) дважды щелкните на операции **Create**() класса **OrderItem**, чтобы открыть окно ее спецификации;
16) в раскрывающемся списке **Return type** укажите **Boolean**;
17) нажмите на кнопку **ОК**, закрыв окно спецификации операции.

VII Подробное описание операций с помощью любого из описанных методов:

1) используя браузер или диаграмму Классов, введите следующую сигнатуру операций класса **OrderDetail** (Детали заказа):
   Open() : Boolean
   SubmitInfo() : Boolean
   Save() : Boolean
2) используя браузер или диаграмму Классов, введите следующую сигнатуру операций класса **OrderOptions** (Выбор заказа):
   Create() : Boolean
3) используя браузер или диаграмму Классов, введите следующую сигнатуру операций класса **OrderMgr** (Управляющий заказами):
   SaveOrder(OrderID : Integer) : Boolean

4) используя браузер или диаграмму Классов, введите следующую сигнатуру операций класса **TransactionMgr** (Управляющий транзакциями):

SaveOrder(OrderID : Integer) : Boolean

Commit() : Integer

VIII Создание диаграмм классов (добавление связей между классами):

После добавления к классам атрибутов и операций Светлана была уже почти готова к генерации кода. Сначала, однако, она должна была изучить связи между классами.

Чтобы найти связи, Светлана изучила диаграммы Последовательности. Все взаимодействующие там классы нуждались в определении соответствующих связей на диаграммах Классов. После обнаружения связей Светлана добавила их в модель.

Добавление связей

Добавим связи к классам, принимающим участие в варианте использования «Ввести новый заказ».

I Настройка:

1) найдите в браузере диаграмму Классов **Ввод нового заказа;**
2) дважды щелкните на ней, чтобы открыть ее;
3) проверьте, имеется ли на панели инструментов диаграммы кнопка **Unidirectional Association**. Если ее нет, продолжайте настройку, выполнив этапы 4 и 5. Если есть, приступайте к выполнению самого упражнения;
4) щелкните правой кнопкой мыши на панели инструментов диаграммы и в открывшемся меню выберите пункт **Customize**;
5) добавьте на панель кнопку, называющуюся **Create A Unidirectional Association**.

II Добавление ассоциаций:

1) нажмите кнопку панели инструментов **Unidirectional Association**;
2) нарисуйте ассоциацию от класса **OrderOptions** (ВыборЗаказа) к классу **OrderDetail** (ДеталиЗаказа);
3) повторите этапы 1 и 2, создав еще ассоциации:
   - от класса **OrderDetail** к классу МенеджерЗаказов (**OrderMgr**);
   - от класса **OrderMgr** к классу Заказ (**Order**);
   - от класса **OrderMgr** к классу МенеджерТранзакций (**TransactionMgr**);
   - от класса **TransactionMgr** к классу **Order;**
   - от класса **TransactionMgr** к классу ПозицияЗаказа (**OrderItem**);
   - от класса **Order** к классу **OrderItem;**
4) щелкните правой кнопкой мыши на однонаправленной ассоциации между классами **OrderOptions** и **OrderDetail**, со стороны класса **OrderOptions**;
5) в открывшемся меню выберите пункт **Multiplicity → Zero or One**;
6) щелкните правой кнопкой мыши на другом конце однонаправленной ассоциации;
7) в открывшемся меню выберите пункт **Multiplicity → Zero or One**;
8) повторите этапы 4 - 7, добавив на диаграмму значения множественности для остальных ассоциаций, как показано на рисунке 19.

   Реализовать ассоциации сценария, выполненного Вами самостоятельно.

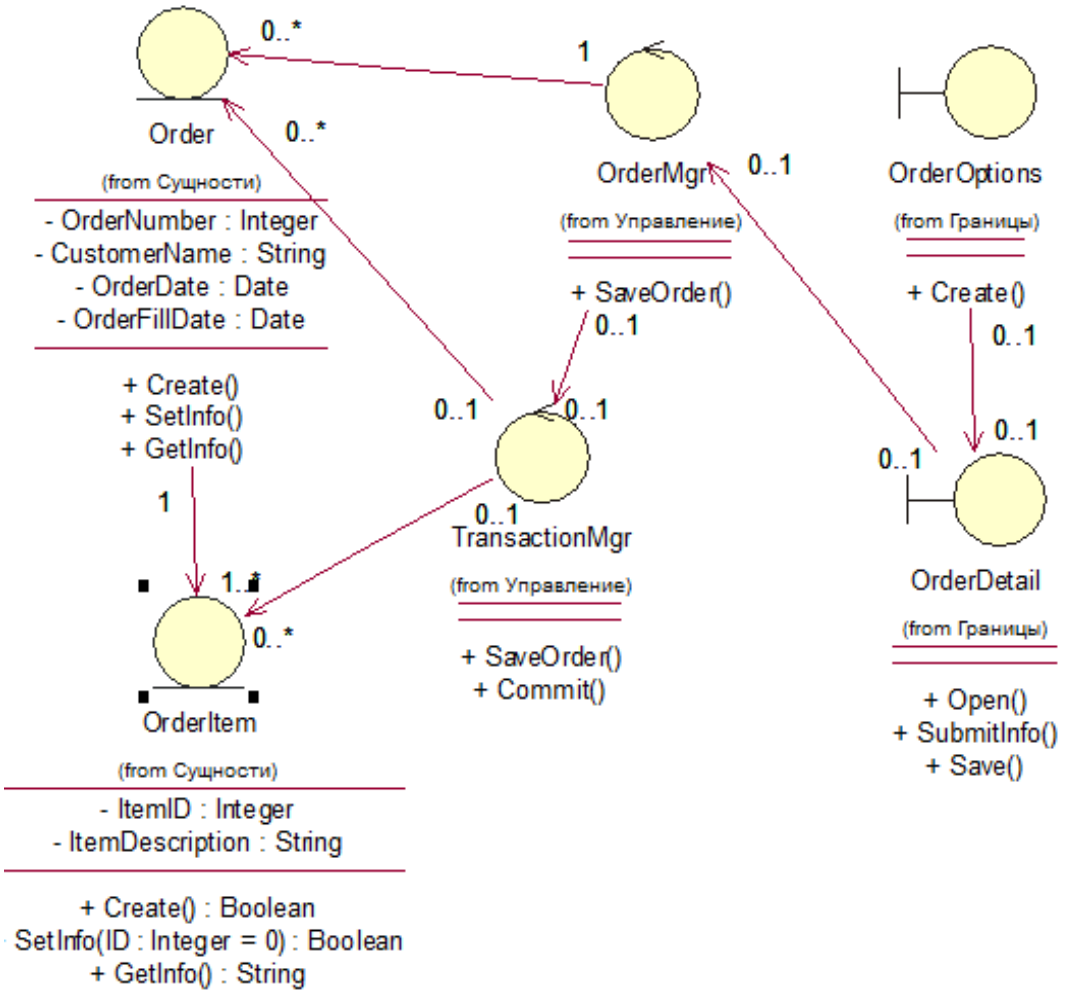

Рисунок 19 - Ассоциации сценария **Ввести новый заказ**

Задание 2. Регистрация курсов

Рассмотрим сценарий **Добавление учебного курса**, который является внутренним потоком для прецедента **Выбор предметов для преподавания**. Данный сценарий позволяет преподавателю выбрать учебный курс для конкретного семестра.

I Выбор граничных классов:

1) прецедент взаимодействует с актером **Преподаватель**;
2) потребности преподавателя: изменение, удаление, просмотр и печать курса. Для их обеспечения создайте класс **Параметры курса преподавателя** (ProfessorCourseOptions)**;**
3) дополнительно укажите класс, который служит для добавления новых курсов, доступных преподавателю – **Добавление учебного курса** (AddCourseOffering).

II Выбор классов сущностей:

1) сценарий состоит из **предметов**, **учебных курсов** и назначения **преподаватель**;
2) выделите классы-сущности: Предмет (**Course**), Учебный курс (**CourseOffering**), Преподаватель (**Professor**).

III Выбор управляющих классов:

1) добавьте управляющий класс – **Менеджер курсов преподавателя** (ProfessorCourseManager).

IV Создание пакетов:

1) разделите все классы на три логические группы (рисунок 20): Interfaces (Интерфейсы), UniversityArtifacts (Объекты университета), PeopleInfo (Сведения о людях).

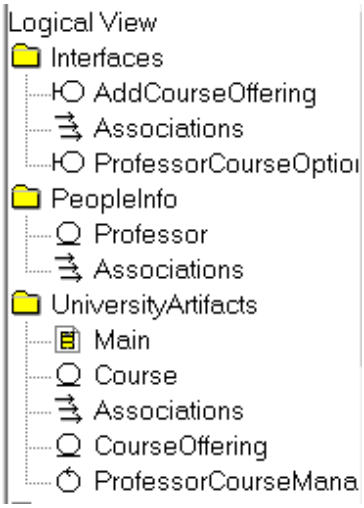

Рисунок 20 – Пакеты и классы

V Создание диаграмм классов:

1) создайте главную диаграмму классов системы регистрации курсов;
2) визуализируйте главные диаграммы пакетов системы регистрации курсов.

VI Создание отношений:

1) создайте диаграмму классов с указанием отношений между классами (рисунок 21). Между сущностями **Учебный курс** и **Предмет** установлено агрегационное отношение (установить связь и в контекстном меню выбрать использовать кнопку **Aggregate**). Агрегационное отношение – это специальная форма ассоциации между целым и его частью или частями (отношение типа «часть от» или «содержит»). Например, предмет может читаться несколько раз в течение семестра, предмет содержит несколько учебных курсов;
2) именование отношений. Ассоциации можно присвоить имя. Присваивать имя необязательно. Укажите название отношения **manages** для связи классов **Предмет** и **Менеджер курсов преподавателей**. Для этого щелкните два раза по связи мышью и введите название;
3) именование ролей. Окончание линии ассоциации в месте, где она соединяется с классом, называется ролью ассоциации. Название роли может быть использована вместо названия ассоциации. Для этого выбирается существительное, описывающее роль, в которой один класс выступает в связи с другим классом. Для ввода названия роли нужно:
    а) щелкнуть правой кнопкой мыши по линии ассоциации рядом с классом, к которому применяется роль;
    б) в контекстном меню выбрать команду **Role Name** (Название роли). Введите имя роли;

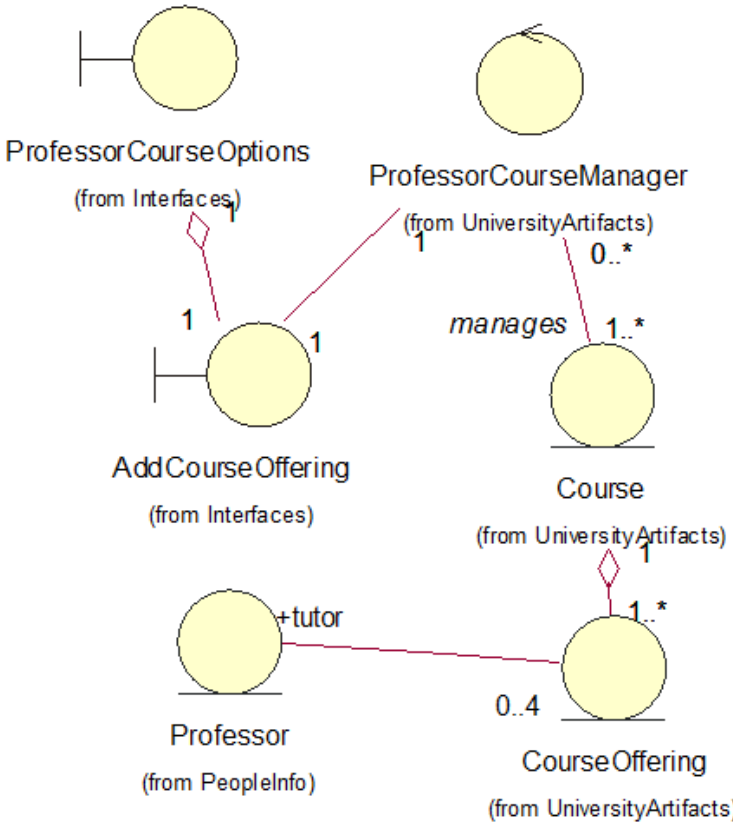

Рисунок 21 – Отношения в сценарии **Добавить учебный курс**

4) определение мощности отношения. *Мощность* отношения указывается для классов и определяет допустимое количество объектов, участвующих в отношении с каждой стороны. Для определения мощности (рисунок 22):
   а) дважды щелкните по линии связи на диаграмме – откроется диалоговое окно **Параметры** (Specification);
   б) перейдите на вкладку **Детально** (Detail) для нужной роли;
   в) укажите требуемое значение мощности в поле **Численное отношение** (Myltiplicity). Щелкните ОК;

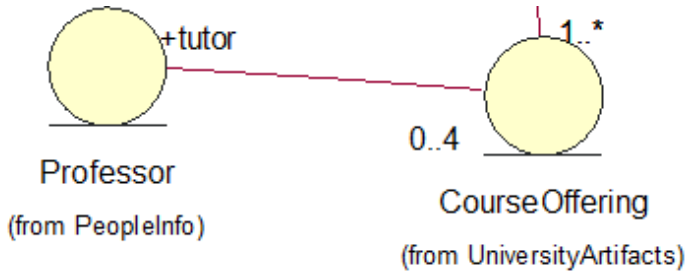

Рисунок 22 – Мощность отношения

5) отношение между пакетами. Это отношение зависимости изображается в виде пунктирной стрелки, направленной к зависимому пакету. Если пакет А (пакет-клиент) зависит от пакета В (пакет-поставщик), значит, один или несколько классов в пакете А инициируют связь с одним или более общедоступными классами в пакете В. В сценарии **Добавить учебный курс** класс **Добавление учебного курса** отправляет сообщение классу **Менеджер курсов преподавателя** (рисунок 23).

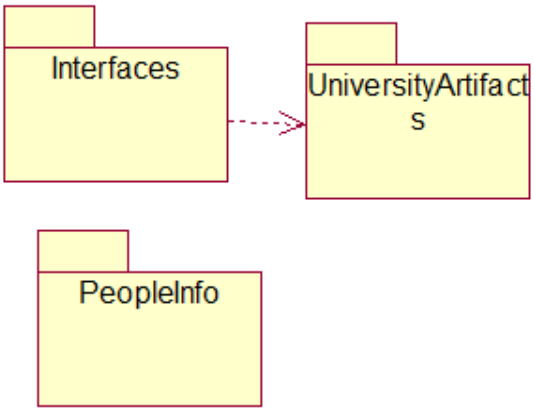

Рисунок 23 - Отношение между пакетами

6) для создания отношений между пакетами нужно:
   а) щелкнуть по кнопке **Отношение зависимости** (Dependency) на панели инструментов;
   б) щелкните по пакету-клиенту и перетащите линию связи к пакету-поставщику.

VII Добавление поведения и структуры:

Структура объекта описывается атрибутами класса. Каждый атрибут - это поле данных, содержащееся в объекте класса. Объект, созданный на основе класса, наделен значениями всех атрибутов класса. Например, класс **Предмет** (Course) имеет атрибуты: название (**name**), описание (**definition**), количество учебных часов (**credits hours**). Каждый объект **Предмет** будет содержать значения перечисленных атрибутов. Они могут повторяться, т.к. в университете существуют учебные предметы с одинаковым количеством академических часов.

1) создайте диаграмму действий с операциями (рисунок 24);
2) добавьте операции для классов и добавьте их описание. Например, добавлена операция **SetProfessor** (Добавление преподавателя) класса **Course** (Предмет) и ее описание (рисунок 25);
3) входными параметрами для операции **SetProfessor** в классе **Course** являются классы **Professor** и **CourseOffering**. Существуют отношения:
   а) между классами **Course** и **Professor**;
   б) между классами **Course** и **CourseOffering**. На основе этих отношений пересмотрите взаимосвязи между пакетами (добавить отношение зависимости между пакетами UniversityArtifacts и PeopleInfo);
4) создание атрибутов. Добавим атрибуты для классов пакета UniversityArtifacts (рисунок 26);
5) отобразим атрибуты и операции на диаграмме классов (рисунок 27).

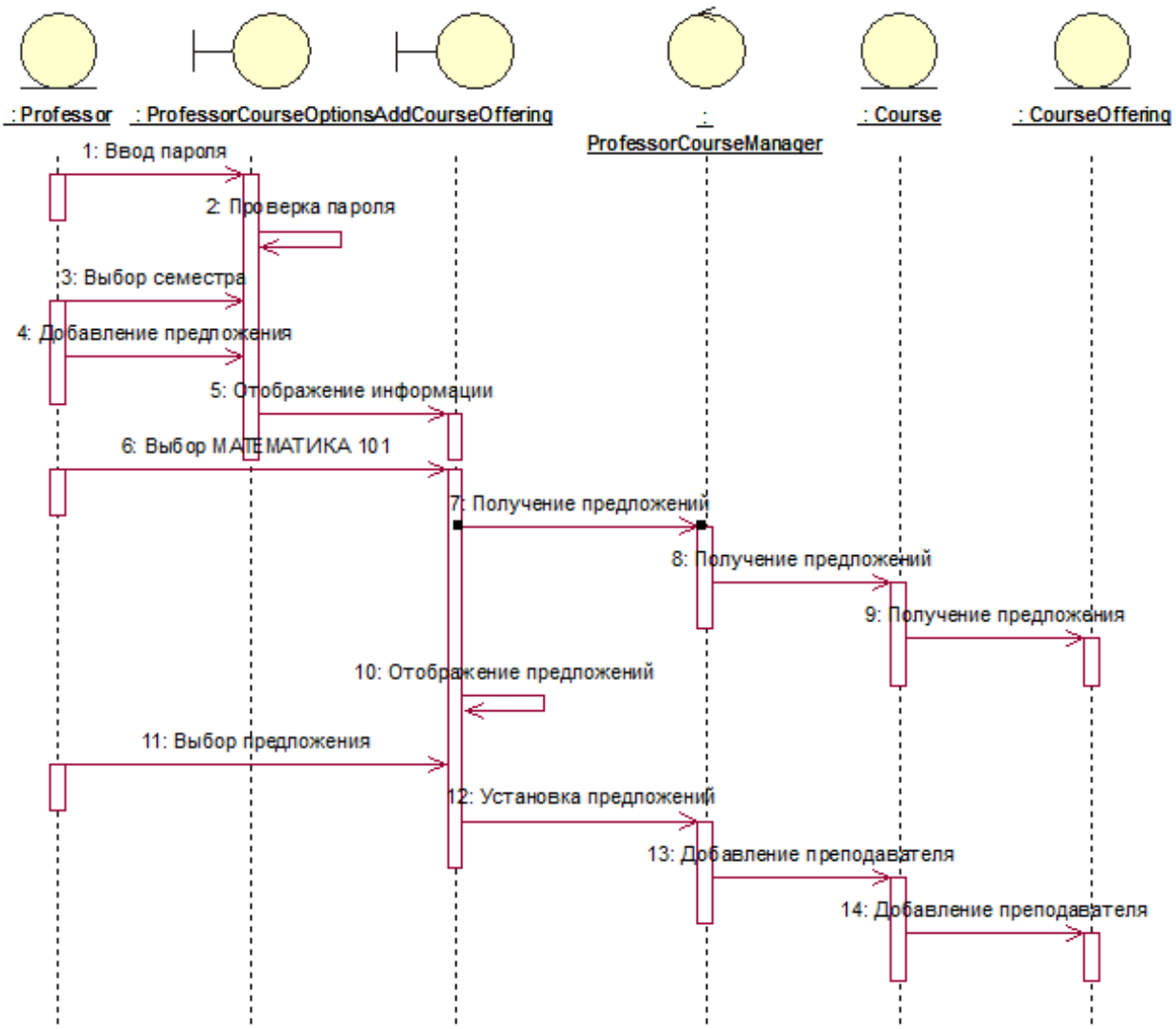

Рисунок 24 - Диаграмма последовательности действий с операциями

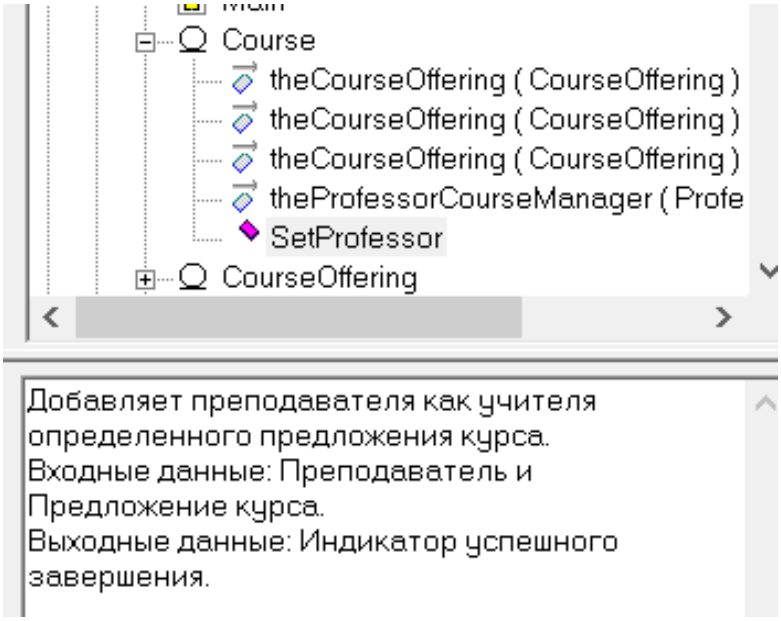

Рисунок 25 - Операция **SetProfessor** и ее описание

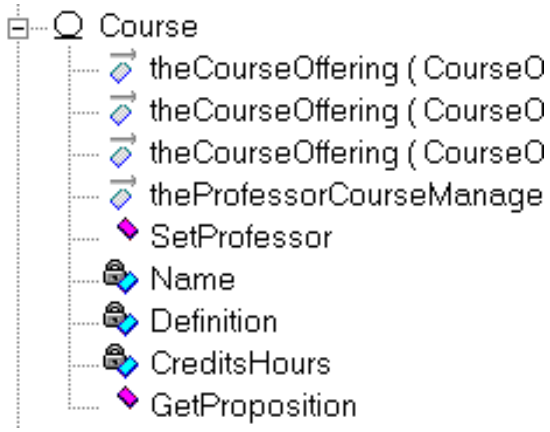

Рисунок 26 - Атрибуты класса **Course**

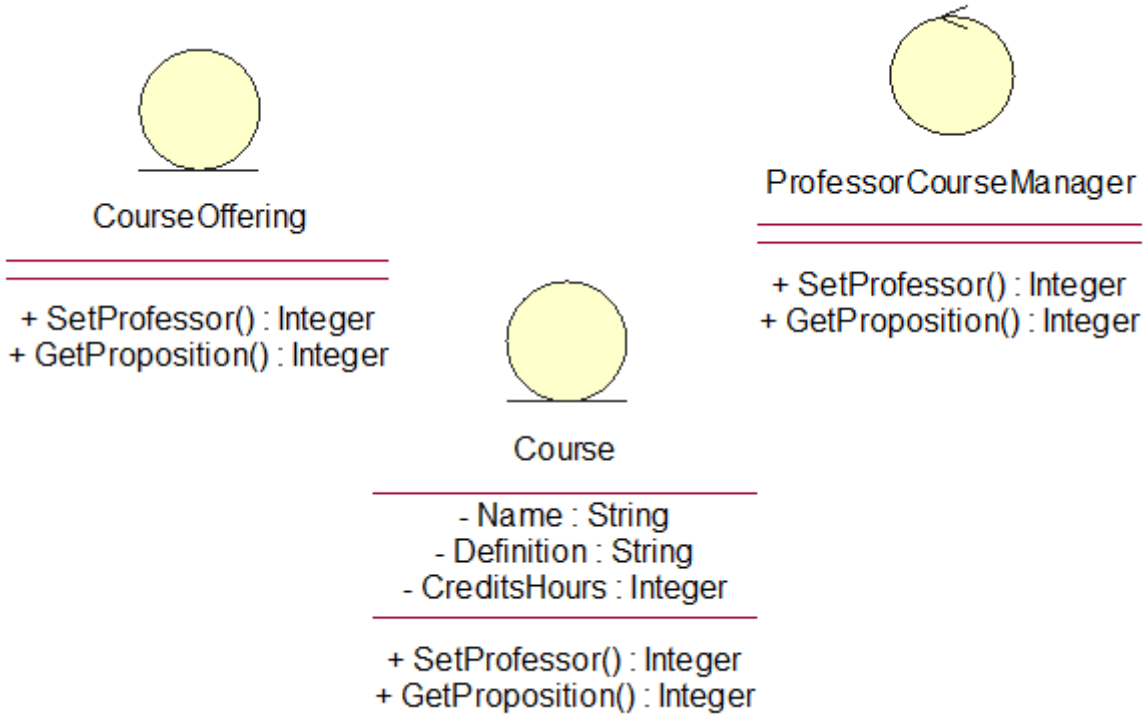

Рисунок 27 - Отображение атрибутов и операций на диаграмме классов

Лабораторная работа № 4. Создание диаграммы состояний

В этом упражнении будет создана диаграмма Состояний для класса Order.

Диаграмма состояний показывает положение одиночного объекта, события или сообщения, которые вызывают переход из одного состояния в другое, и действия, являющиеся результатом смены состояния.

Диаграмму состояний создают только для классов с динамическим поведением. Для определения динамических объектов в системе (объектов, принимающих и отсылающих большое количество сообщений) могут использоваться диаграммы взаимодействий.

Состояние - это некоторое положение в жизни объекта, при котором он удовлетворяет определенному условию, выполняет некоторое действие или ожидает события. Состояние объекта можно описать с помощью значений одного или нескольких атрибутов класса.

Диаграмма состояний включает все сообщения, которые объект получает или отправляет. Сценарий - это одиночный проход по диаграмме состояний.

Переходы между состояниями - это смена исходного состояния последующим (которое может быть тем же, что и исходное). Переход может сопровождаться некоторым действием.

Есть два способа выхода из состояния:

1) автоматический - автоматическая смена состояния происходит, когда действие исходного состояния будет завершено - с переходом не связано каких-либо событий;

2) неавтоматический - неавтоматический переход между состояниями вызывается определенным событием (от другого объекта или из внешней среды).

Считается, что оба типа переходов выполняются на нулевое время и не могут быть прерваны. Переход между состояниями изображается стрелкой, направленной от исходного состояния к последующему.

Есть два особых состояния, присутствующих на диаграмме состояний, - начальное и конечное. Каждая диаграмма должна иметь одно и только одно начальное состояние, т.к. объект должен находиться в целостном состоянии сразу после создания. Объект может иметь несколько конечных состояний.

С переходом между состояниями может быть связано условие (guard condition) и (или) определенное действие (action). Переход может также вызывать событие (event). Действие - это поведение, проявляющееся при возникновении перехода. Событие - сообщение, отправляемое другому объекту системы. Условием будет булево выражение значений атрибутов, которое допускает переход, только если оно верно. И действие, и проверка условия представляют собой поведение объекта и обычно реализуются в виде операций. Часто такие операции являются скрытыми (private), т.е. используются только самим объектом.

Действия, сопровождающие возможные переходы в определенное состояние, можно рассматривать как входные действия (entry) для этого состояния. И наоборот, действия, сопровождающие переходы из данного состояния, являются для него выходными (exit). Поведение, возникающее внутри состояния, называется деятельностью (activity). Деятельность начинается при входе в состояние и завершается или прерывается при переходе из него. Поведение может быть простым действием или событием, посылаемым другому объекту. Поведение внутри состояния обычно реализуется в виде операций.

Задание 1. Фирма «Кухни Deluxe»

Проектируя класс Order, Светлана поняла, что за поведением этого класса надо наблюдать. Многие требования к классу значительно изменялись при изменении состояния его экземпляра. Например, заказы, выполнение которых было приостановлено, вели себя не так, как выполненные заказы, а те, в свою очередь, не так, как отмененные заказы.

Чтобы убедиться, что проект удовлетворяет всем этим требованиям, Светлана со своей группой разработчиков создала диаграмму Состояний для класса Order (Заказ). С помощью этой диаграммы разработчики смогли окончательно понять, как надо писать код для этого класса.

Создание диаграммы Состояний:

I Создание диаграммы:

1) найдите в браузере класс **Order** (Заказ);

2) щелкните на классе правой кнопкой мыши и в открывшемся меню укажите пункт **New Statechart Diagram**. В список браузера будет добавлена диаграмма **New Diagram**;

3) введите название **Состояние заказа**;
4) чтобы открыть диаграмму, щелкните по значку «+» слева от имени подкласса в окне браузера, потом по значку «+» слева от пункта **State/Activity Model** (Модель состояний и действий), а затем дважды по диаграмме состояний.

II Добавление начального и конечного состояний:

1) на панели инструментов нажмите кнопку **Start State** (Начальное состояние);
2) поместите это состояние на диаграмму;
3) на панели инструментов нажмите кнопку **End State** (Конечное состояние);
4) поместите это состояние на диаграмму.

III Добавление суперсостояния и других состояний:

1) на панели инструментов нажмите кнопку **State** (Состояние);
2) поместите это состояние на диаграмму;
3) используя кнопку **State** панели инструментов поместите следующие состояния: **Cancelled** (Отменен), **Filled** (Выполнен), **Initialization** (Инициализация), **Pending** (Выполнение заказа приостановлено).

IV Подробное описание состояний:

1) дважды щелкните на состоянии **Initialization** (Инициализация);
2) щелкните правой кнопкой мыши на окне **Actions** (Действия);
3) в открывшемся меню выберите пункт **Insert** (Вставить);
4) дважды щелкните мышью на новом действии;
5) назовите его **Store Order Date** (Сохранить дату заказа);
6) убедитесь, что в окне **When** (Когда) указан пункт **On Entry** (На входе);
7) повторите этапы 3 - 7, добавив следующие действия:
   - **Collect Customer Info** (Собрать клиентскую информацию), в окне **When** указать пункт **Do;**
   - **Add Order Items** (Добавить к заказу новые графы), в окне **When** указать **Do;**
8) нажмите на кнопку **ОК** два раза, чтобы закрыть спецификацию;
9) дважды щелкните на состоянии **Cancelled** (Отменен);
10) повторите этапы 2 - 7, добавив действие **Store Cancellation Data** (Сохранить дату отмены), указать пункт **On Exit** (на выходе);
11) нажмите на кнопку **ОК** два раза, чтобы закрыть спецификацию;
12) дважды щелкните на состоянии **Filled** (Выполнен);
13) повторите этапы 2 - 7, добавив действие **Bill Customer** (Выписать счет), указать пункт **Do**;
14) нажмите на кнопку **ОК** два раза, чтобы закрыть спецификацию.

V Добавление переходов:

1) на панели инструментов нажмите кнопку **Transition** (Переход);
2) щелкните мышью на начальном состоянии;
3) проведите линию перехода к состоянию **Initialization** (Инициализация);
4) повторите этапы с первого по третий, создав следующие переходы:
   - от состояния **Initialization** (Инициализация) к состоянию **Pending** (Выполнение заказа приостановлено);
   - от состояния **Pending** (Выполнение заказа приостановлено) к состоянию **Filled** (Выполнен);
   - от суперсостояния к состоянию **Cancelled** (Отменен);
   - от состояния **Cancelled** (Отменен) к конечному состоянию;
   - от состояния **Filled** (Выполнен) к конечному состоянию;
5) на панели инструментов нажмите кнопку **Transition to Self** (Переход к себе);

6) щелкните на состоянии **Pending** (Выполнение заказа приостановлено).

VI Подробное описание переходов:

1) дважды щелкните на переходе от состояния **Initialization** (Инициализация) к состоянию **Pending** (Выполнение заказа приостановлено), открыв окно его спецификации;
2) в поле **Event** (Событие) введите фразу **Finalize order** (Выполнить заказ);
3) щелкните на кнопке **ОК**, закрыв окно спецификации;
4) повторите этапы с первого по третий, добавив событие **Cancel Order** (Отменить заказ) к переходу между суперсостоянием и состоянием **Cancelled** (Отменен);
5) дважды щелкните на переходе от состояния **Pending** (Выполнение заказа приостановлено) к состоянию **Filled** (Выполнен), открыв окно его спецификации;
6) в поле **Event** (Событие) введите фразу **Add Order Item** (Добавить к заказу новую позицию);
7) перейдите на вкладку **Detail** (Подробно);
8) в поле **Condition** (Условие) введите **No unfilled items remaining** (Не осталось незаполненных позиций);
9) щелкните на кнопке **ОК**, закрыв окно спецификации;
10) дважды щелкните мышью на рефлексивном переходе (**Transition to Self**) состояния **Pending** (Выполнение заказа приостановлено);
11) в поле **Event** (Событие) введите фразу **Add Order Item** (Добавить к заказу новую позицию);
12) перейдите на вкладку **Detail** (Подробно);
13) в поле **Condition** (Условие) введите **Unfilled items remaining** (Остаются незаполненные позиции);
14) щелкните на кнопке **ОК**, закрыв окно спецификации.

В результате получим диаграмму состояний для класса **Order** (Заказ), представленную на рисунке 28.

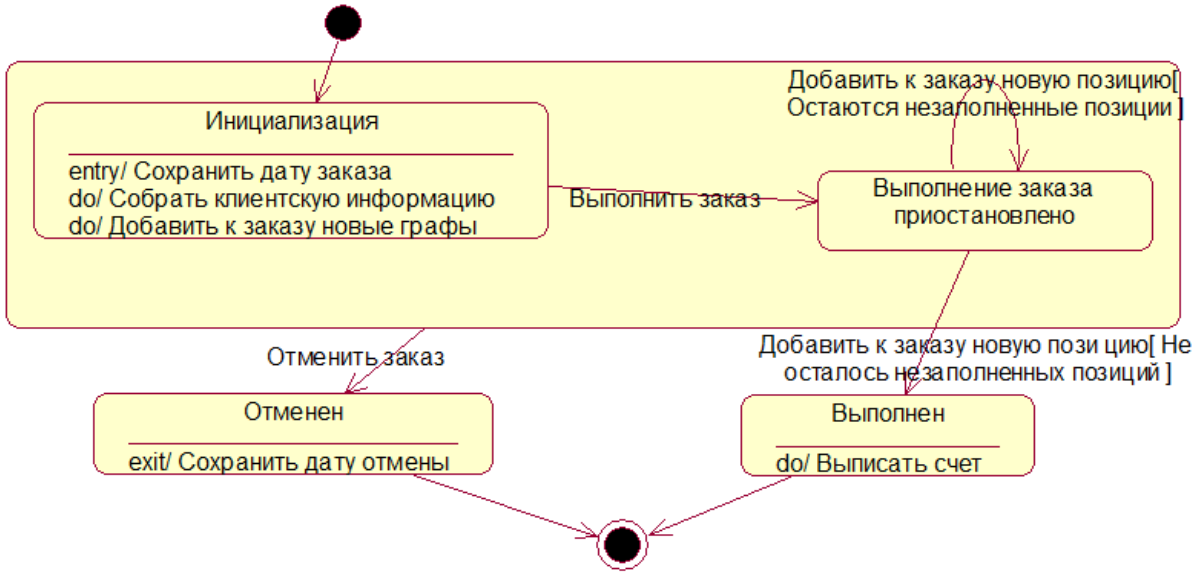

Рисунок 28 - Диаграмма состояний для класса Заказы

Задание 2. Регистрация курсов

Объект **Учебный курс** может быть открыт (доступен для записи студентов) или закрыт (максимальное число студентов уже записано на курс). Состояние зависит от числа студентов, прикрепленных к объекту **Учебный курс**. Кроме того,

оно может определяться наличием связи с другим объектом. Актер **Преподаватель** может быть читать лекции или находится в отпуске. Это зависит от наличия связи с объектом **Учебный курс**.

I Добавление состояний класса CourseOffering:

Объекты класса **CourseOffering** (Учебный курс) могут находиться в одном из следующих состояний: инициализация (создан до регистрации, но студенты не прикреплены), открыт (доступен для записи студентов), закрыт (на курс записано максимальное число студентов), отменен (больше не читается).

II Добавление переходов между состояниями:

- **Добавление студента** (между состояниями **Инициализация** и **Открыт**);
- **Добавление студента** (рефлексивный переход состояния **Открыт**);
- **Отмена** (между состояниями **Открыт** и **Отменен**);
- **Отмена** (между состояниями **Закрыт** и **Отменен**).

III Добавление особых состояний:

1) добавьте начальное состояние и проведите линию перехода от него к **Инициализация;**
2) добавьте конечное состояние и проведите линию перехода от состояний **Отменен** и **Закрыт** к конечному состоянию.

IV Добавление параметров переходов:

- **Добавление студента** (между состояниями **Инициализация** и **Открыт**):
  Действие - Установка счетчика = 0;
  Сообщение - Расписание курсов.Создать;
- **Добавление студента** (рефлексивный переход состояния **Открыт**):
  Действие - счетчик < 0;
- между состояниями **Открыт** и **Закрыт**:
  Граничное условие - счетчик = 10;
- между состояниями **Отменен** и **конечным состоянием**:
  Сообщение - Расписание курсов.Удалить.

V Добавление параметров состояний:

- Инициализация:
  Делать / Инициализация данных предложения курса (действие);
- Открыт:
  Вход / Регистрация студента (действие);
  Выход / Расписание курсов.Добавление студента (Студент) (событие);
- Закрыт:
  Делать / Завершение курса (действие);

В результате получим диаграмму состояний для класса **CourseOffering** (Учебный курс), представленную на рисунке 29.

Лабораторная работа № 5. Создание диаграммы компонентов

Представление реализации определяет реальную организацию программных модулей в среде разработки. Оно учитывает потребности в простоте разработки, управлении программными средствами, повтором использовании кода, а также языковых и инструментальных ограничениях.

Элементами моделирования в представлении компонентов (component view) являются пакеты, компоненты и связи между ними.

Рисунок 29 - Диаграмма состояний для класса Учебный курс

Пакет в данном представлении архитектуры - это физический раздел системы. Пакеты организованы в виде иерархии уровней или слоев, где каждый уровень имеет четко определенный интерфейс. Типичная последовательность уровней системы: интерфейс пользователя, пакеты, связанные с приложениями, бизнес-пакеты, ключевые механизмы, пакеты аппаратных средств и операционной системы.

В представлении компонентов модели компоненты исходного кода - это программные файлы, содержащиеся внутри пакетов. Тип файлов зависит от языка программирования (например, в C++ - файлы .h. и .cpp). Каждый компонент связан с каким-либо языком. Классы в логическом представлении отображаются на компоненты в представлении компонентов. Для C++ один класс отображается в один компонент. Иногда на один компонент может быть отображено больше одного класса. Это происходит в том случае, когда между классами существует очень тесная связь.

Представление процессов отражает структуру программной реализации системы. Представление процессов учитывает такие потребности, как производительность, надежность, масштабируемость, целостность, управление системой и синхронизация. Для представления программных и исполняемых компонентов системы создается диаграмма компонентов.

Задание 1. Фирма «Кухни Deluxe»

Завершив анализ и проектирование системы, один из разработчиков проекта Денис разработал диаграммы Компонентов. Выбрав в качестве языка программирования C++, для каждого класса Денис создал соответствующие этому языку компоненты.

I Создание пакетов компонентов:

1) щелкните правой кнопкой мыши на представлении компонентов в браузере (Component View);
2) в открывшемся меню выберите пункт **New → Package** (Создать → Пакет);

3) назовите этот пакет **Entities** (Сущности);
4) повторите этапы с первого по третий, создав пакеты **Boundaries** (Границы) и **Control** (Управление).

II Добавление пакетов на Главную диаграмму Компонентов:

1) откройте Главную диаграмму Компонентов (Main), дважды щелкнув на ней;
2) перетащите пакеты **Entities**, **Boundary** и **Control** из браузера на Главную диаграмму.

III Рисование зависимостей между пакетами:

1) на панели инструментов нажмите кнопку **Dependency** (Зависимость);
2) щелкните мышью на упаковке **Boundaries** Главной диаграммы Компонентов;
3) проведите линию зависимости до упаковки **Control**;
4) повторите этапы 1 - 3, проведя еще зависимость от пакета **Control** до пакета **Entities**.

IV Добавление компонентов к пакетам и рисование зависимостей:

1) дважды щелкните мышью на пакете **Entities** Главной диаграммы Компонентов, открыв Главную диаграмму Компонентов этого пакета (Main);
2) на панели инструментов нажмите кнопку **Package Specification** (Спецификация пакета);
3) поместите спецификацию пакета на диаграмму;
4) введите имя спецификации пакета **OrderItem**;
5) повторите этапы 2 - 4, добавив спецификацию пакета **Order**;
6) на панели инструментов нажмите кнопку **Package Body** (Тело пакета);
7) поместите его на диаграмму;
8) введите имя тела пакета **OrderItem**;
9) повторите этапы 6 - 8, добавив тело пакета **Order**;
10) на панели инструментов нажмите кнопку **Dependency** (Зависимость);
11) щелкните мышью на теле пакета **OrderItem**;
12) проведите линию зависимости от него к спецификации пакета **OrderItem**;
13) повторите этапы 10 - 12, добавив линию зависимости между телом пакета **Order** и спецификацией пакета **Order**;
14) повторите этапы 10 - 12, добавив линию зависимости от спецификации пакета **Order** к спецификации пакета **OrderItem**;
15) с помощью описанного метода создайте следующие компоненты и зависимости:

а) для пакета **Boundaries**:
- спецификацию пакета **OrderOptions;**
- тело пакета **OrderOptions;**
- спецификацию пакета **OrderDetail;**
- тело пакета **OrderDetail;**

б) зависимости в пакете **Boundaries**:
- от тела пакета **OrderOptions** до спецификации пакета **OrderOptions;**
- от тела пакета **OrderDetail** до спецификации пакета **OrderDetail;**
- от спецификации пакета **OrderOptions** до спецификации пакета **OrderDetail;**

в) для пакета **Control**:
- спецификацию пакета **OrderMgr;**
- тело пакета **OrderMgr;**

- спецификацию пакета **TransactionMgr;**
- тело пакета **TransactionMgr;**

г) зависимости в пакете **Control**:

- от тела пакета **OrderMgr** до спецификации пакета **OrderMgr;**
- от тела пакета **TransactionMgr** до спецификации пакета **TransactionMgr;**
- от спецификации пакета **OrderMgr** до спецификации пакета **TransactionMgr.**

V Создание диаграммы Компонентов системы:

1) щелкните правой кнопкой мыши на представлении Компонентов в браузере;
2) в открывшемся меню выберите пункт **New → Component Diagram;**
3) назовите новую диаграмму **System**;
4) дважды щелкните на этой диаграмме.

VI Размещение компонентов на диаграмме Компонентов системы:

1) если это еще не было сделано, разверните в браузере пакет компонентов **Entities**, чтобы открыть его;
2) щелкните мышью на спецификации пакета **Order** в пакете компонентов **Entities**;
3) перетащите эту спецификацию на диаграмму;
4) повторите этапы 2 и 3, поместив на диаграмму спецификацию пакета **OrderItem**;
5) с помощью этого метода поместите на диаграмму следующие компоненты:

а) из пакета компонентов **Boundaries**:

- спецификацию пакета **OrderOptions;**
- спецификацию пакета **OrderDetail;**

б) из пакета компонентов **Control**:

- спецификацию пакета **OrderMgr;**
- спецификацию пакета **TransactionMgr;**

6) на панели инструментов нажмите кнопку **Task Specification** (Спецификация задачи);
7) поместите спецификацию задачи на диаграмму и назовите ее **OrderClientExe**;
8) повторите этапы 6 и 7 для спецификации задачи **OrderServerExe**.

VII Добавление оставшихся зависимостей на диаграмму Компонентов системы:

Уже существующие зависимости будут автоматически показаны на диаграмме Компонентов системы после добавления туда соответствующих компонентов. Теперь надо добавить остальные зависимости.

1) на панели инструментов нажмите кнопку **Dependency** (Зависимость);
2) щелкните на спецификации пакета **OrderDetail**;
3) проведите линию зависимости к спецификации пакета **OrderMgr**;
4) повторите этапы 1 - 3, создав следующие зависимости:

- от спецификации пакета **OrderMgr** к спецификации пакета **Order;**
- от спецификации пакета **TransactionMgr** к спецификации пакета **OrderItem;**
- от спецификации пакета **TransactionMgr** к спецификации пакета **Order;**
- от спецификации задачи **OrderClientExe** к спецификации пакета **OrderOptions;**

– от спецификации задачи **OrderServerExe** к спецификации пакета **OrderMgr.**

VIII Соотнесение классов с компонентами:

1) в Логическом представлении браузера найдите класс **Order** пакета **Entities**;
2) перетащите этот класс на спецификацию пакета компонента **Order** в представлении Компонентов браузера. В результате класс **Order** будет соотнесен со спецификацией пакета компонента **Order**;
3) перетащите класс **Order** на тело пакета компонента **Order** в представлении Компонентов браузера. В результате класс **Order** будет соотнесен с телом пакета компонента **Order**;
4) повторите этапы 1 - 3, соотнеся с классами следующие компоненты:
   – класс **OrderItem** со спецификацией пакета **OrderItem;**
   – класс **OrderItem** с телом пакета **OrderItem;**
   – класс **OrderOptions** со спецификацией пакета **OrderOptions;**
   – класс **OrderOptions** с телом пакета **OrderOptions;**
   – класс **OrderDetail** со спецификацией пакета **OrderDetail;**
   – класс **OrderDetail** с телом пакета **OrderDetail;**
   – класс **OrderMgr** со спецификацией пакета **OrderMgr;**
   – класс **OrderMgr** с телом пакета **OrderMgr;**
   – класс **TransactionMgr** со спецификацией пакета **TransactionMgr;**
   – класс **TransactionMgr** с телом пакета **TransactionMgr.**

В результате получим диаграмму компонентов системы, представленную на рисунке 30.

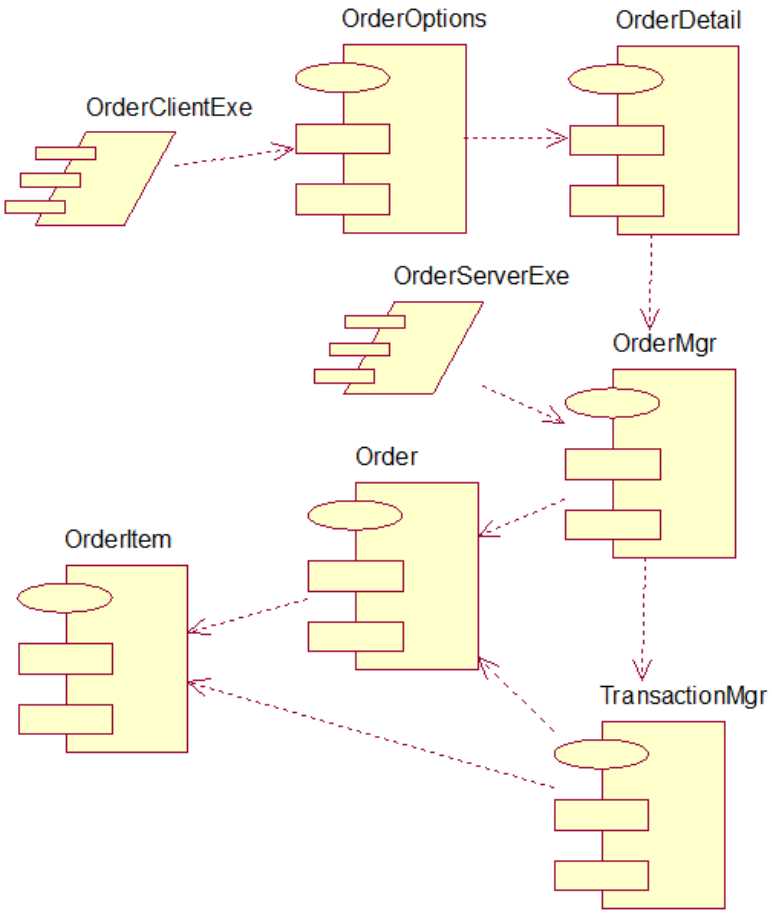

Рисунок 30 - Диаграмма компонентов системы

Задание 2. Регистрация курсов

Разработчики архитектуры выяснили, что для создания графического интерфейса пользователя потребуется определенный набор графических элементов управления, и в модель был добавлен пакет **Элементы управления ГИП** (GUI Controls). Стратегия хранения данных предусматривает использование отдельного класса доступа к базе данных (теневого класса) для каждого информационного класса в системе. На текущем этапе в модель добавляется пакет для доступа к базе данных, содержащий необходимые теневые классы. В модель добавлен общий пакет **Обработка ошибок** (Error Handling) для обработки исключительных ситуаций. В систему добавлен набор классов для реализации основных коммерческих операций **Базовые средства** (Foundation).

Так как **Обработка ошибок** и **Базовые средства** используются всеми остальными пакетами системы, они являются глобальными пакетами.

Результатом рассуждений должна стать диаграмма классов логического представления (рисунок 31).

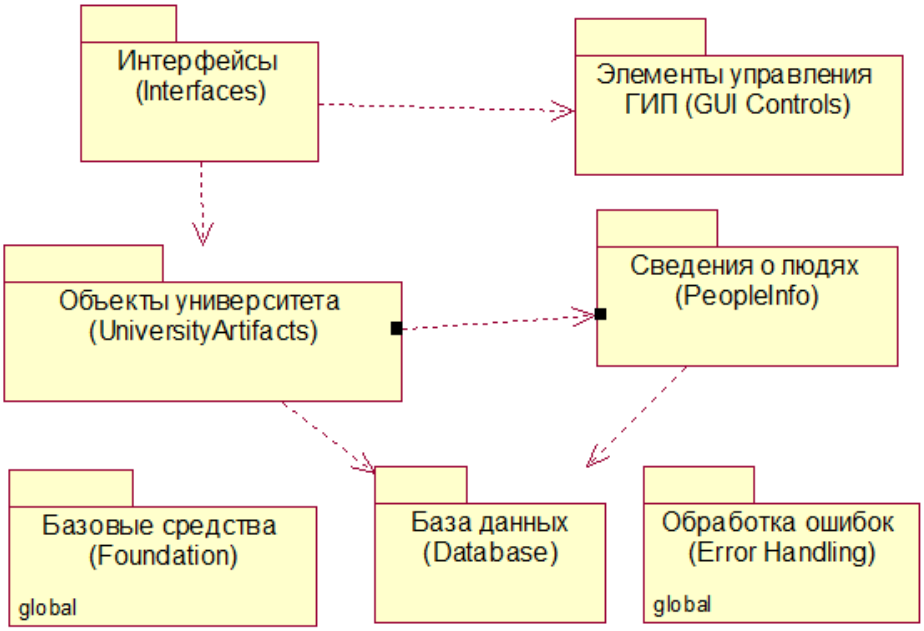

Рисунок 31 - Система регистрации учебных курсов

Реализуйте главную диаграмму компонентов для системы регистрации курсов. Для этого следует щелкнуть правой кнопкой мыши по разделу Представление компонентов (Component View), выбрать **Создать → Пакет** (New → Package) и ввести имя пакета. Главная диаграмма компонентов представлена на рисунке 32.

Рассмотрим проектирование программных компонентов. Система регистрации учебных курсов достаточно проста, поэтому было принято решение обеспечить отображение один в один между классами и компонентами - каждый класс имеет собственный файл заголовка и .cpp-файл.

Для создания компонентов нужно открыть диаграмму, выбрать на панели инструментов кнопку **Компонент** (Component) и ввести имя компонента. Программные компоненты для пакета Объекты университета приведены на рисунке 33.

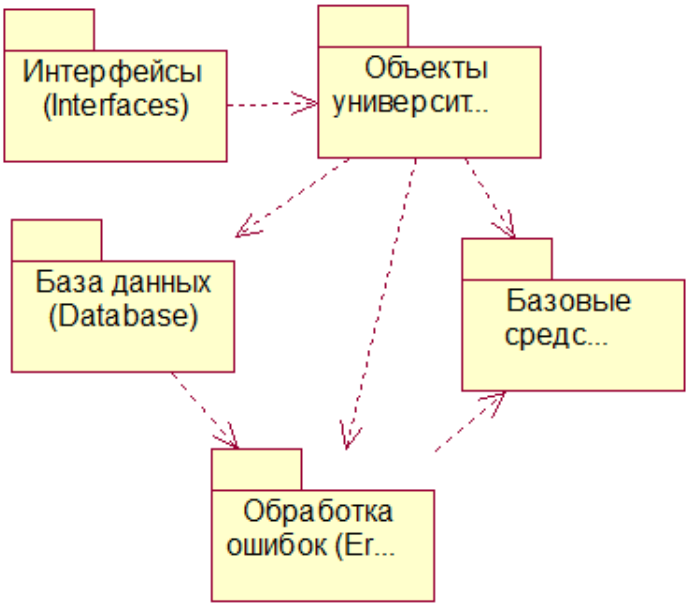

Рисунок 32 - Главная диаграмма компонентов

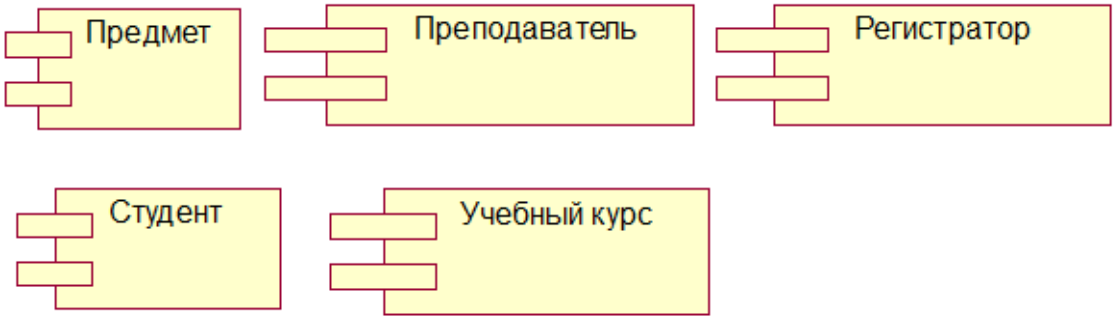

Рисунок 33 - Программные компоненты

Необходимо настроить отображение классов на компоненты. Для этого:

1) щелкните правой кнопкой мыши по компоненту в списке браузера и откройте окно параметров компонента;
2) перейдите на вкладку **Реализации** (Realize). Щелкните правой кнопкой мыши по нужному классу в списке классов;
3) в меню выбрать команду **Присвоить** (Assign). Щелкнуть по кнопке ОК.

Класс можно присвоить компоненту путем перетаскивания класса из окна браузера на изображение компонента в браузере или на диаграмме компонентов.

В системе регистрации курсов созданы две динамические библиотеки - для обработки информации о предметах и учебных курсах и для работы с базой данных. Такой подход был выбран исходя из возможных изменений в структуре курсов и в стратегии взаимодействия с базой данных.

При наличии изменений достаточно воспользоваться другой библиотекой. В системе есть три исполняемых модуля - один для регистратора, чтобы осуществлять ввод данных и управление информацией в системе; один для студента и один для преподавателя с целью получения доступа и использования системы. Между исполняемыми модулями нет никакого взаимодействия. Диаграмма компонентов для исполняемого модуля преподавателя (ProfessorOptions.exe) показана на рисунке 34.

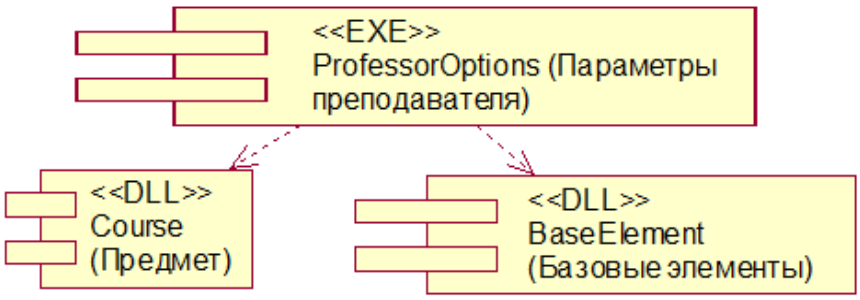

Рисунок 34 - Исполняемый модуль для **Преподаватель**

Лабораторная работа № 6. Создание диаграммы размещения

Представление средств внедрения отображает программные средства на узлы вычислительных систем. Оно показывает конфигурацию элементов обработки и работающих на них программных процессов. Представление средств внедрения учитывает такие потребности, как доступность системы, надежность, быстродействие и масштабируемость. Чтобы показать различные узлы вычислительных систем и связи между ними, создаются диаграммы внедрения.

Такая диаграмма демонстрирует распределение компонентов по предприятию. Элементы обработки представлены узлами вычислительных систем, которые соединены линиями, показывающими коммуникационные каналы между ними.

Программные процессы показываются в виде текста, привязанного к узлу или группе узлов.

Такая диаграмма позволяет разработчикам архитектуры понять топологию системы и отобразить компоненты на исполняемые процессы. Здесь учитываются следующие вопросы: процессорная архитектура, скорость, емкость, пропускная способность каналов для взаимодействия процессов, физическое расположение аппаратных ресурсов, технология распределенной обработки.

Задание 1. Фирма «Кухни Deluxe»

Команда разработчиков завершила весь предшествующий данному моменту анализ и проектирование системы. Варианты использования, взаимодействия между объектами и компоненты четко описаны. Тем не менее подразделению администрирования сети необходимо знать, на каких компьютерах будут размещаться различные компоненты системы. В связи с этим пришлось еще разработать диаграмму Размещения для системы обработки заказов.

Создание диаграммы Размещения (внедрения):

I Добавление узлов к диаграмме Размещения:

1) дважды щелкните мышью на представлении **Размещения** (Deployment View) в браузере, чтобы открыть диаграмму Размещения;
2) на панели инструментов нажмите кнопку **Processor** (Процессор);
3) щелкните на диаграмме, поместив туда процессор;
4) введите имя процессора **Сервер базы данных**;
5) повторите этапы 2 - 4, добавив следующие процессоры:
   - сервер приложении;
   - клиентская рабочая станция №1;
   - клиентская рабочая станция №2;
6) на панели инструментов нажмите кнопку **Device** (Устройство);
7) щелкните на диаграмме, поместив на нее устройство;

8) назовите его Принтер.

II Добавление связей:

1) на панели инструментов нажмите кнопку **Connection** (Связь);
2) щелкните на процессоре **Сервер базы данных**;
3) проведите линию связи к процессору **Сервер приложения**;
4) повторите этапы 1 - 3, добавив следующие связи:
   - от процессора **Сервер приложения** к процессору **Клиентская рабочая станция №1;**
   - от процессора **Сервер приложения** к процессору **Клиентская рабочая станция №2;**
   - от процессора **Сервер приложения** к устройству **Принтер.**

III Добавление процессов:

1) щелкните правой кнопкой мыши на процессоре **Сервер приложения** в браузере;
2) в открывшемся меню выберите пункт **New → Process** (Создать → Процесс);
3) введите имя процесса **OrderServerExe**;
4) повторите этапы 1 - 3, добавив еще процессы:
   - на процессоре **Клиентская рабочая станция №1** - процесс **OrderClientExe;**
   - на процессоре **Клиентская рабочая станция №2** - процесс **ATMClientEXE.**

IV Показ процессов на диаграмме:

1) щелкните правой кнопкой мыши на процессоре **Сервер приложения**.
2) в открывшемся меню выберите пункт **Show Processes** (Показать процессы).
3) повторите этапы 1 и 2, показав процессы на следующих процессорах:
   - клиентская рабочая станция №1;
   - клиентская рабочая станция №2.

Диаграмма размещения представлена на рисунке 35.

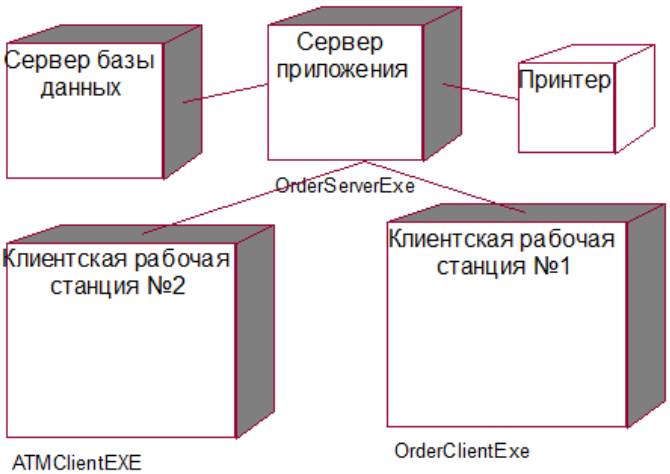

Рисунок 35 - Диаграмма Размещения для системы обработки заказов

Задание 2. Регистрация учебных курсов

После изучения определения для данной задачи компонентов, существующих аппаратных средств и оценки загруженности системы в период регистрации на курсы разработчики архитектуры решили выделить следующие вычислительные

системы: одну - для запуска исполняемого модуля преподавателя, одну - для работы с базой данных и три - для регистрации студентов (рисунок 36).

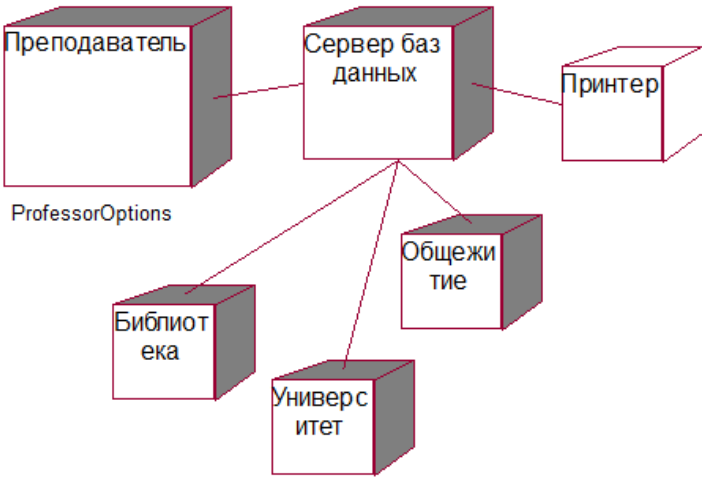

Рисунок 36 - Диаграмма внедрения

Лабораторная работа № 7. Генерация кода C++

Для класса, роли, атрибута, операции и проекта в целом существуют параметры генерации кода. К параметрам, применимым ко всему проекту, относятся имя файла, имя основного контейнера и место генерации кода. Параметры для класса определяют генерацию конструктора, деструктора, копирующего конструктора, операторов сравнения и методов установки/получения данных. Параметры для роли управляют созданием методов установки/получения данных, видимостью методов и определяют используемый класс-контейнер. Параметры операции задают тип операции (общая, виртуальная, абстрактная, статическая, дружественная) и позволяют ей стать константой. Наборы параметров могут редактироваться.

Для каждого класса создаются два файла: файл заголовка (.h) и файл спецификации (.cpp).

Рассмотрим последовательность шагов для генерации кода.

Шаг 1. Создание необходимого набора параметров:

Для создания наборов параметров нужно выполнить команду **Tools** → **Options** (Сервис → Параметры), перейти на вкладку **ANSI C++** и в списке **Type** (Тип) указать нужный тип набора параметров.

Шаг 2. Создание компонентов тела пакета на диаграмме компонентов:

Для компонентов без стереотипов создается h-файл, содержащий определение и декларацию класса. Для компонентов со стереотипом **заголовок пакета** (Package Specification) создается h-файл, включающий определение класса. Если существует компонент со стереотипом **тело пакета** (Package Body), то для него создается файл .cpp, содержащий декларацию класса.

Шаг 3. Назначение языка C++ компонентам:

Можно установить язык C++ для всех компонентов, выбрав его на вкладке **Notation** (Нотация) диалогового окна настройки параметров. Для конкретного компонента язык указывается в списке **Language** (Язык) в окне параметров.

Шаг 4. Связывание классов с компонентами:

После создания компонентов устанавливается связь классов с компонентами, представляющими файлы заголовков.

Шаг 5. Привязка наборов параметров к элементам моделирования:

Каждый элемент моделирования (класс, атрибут, роль) изучаются на предмет особенностей генерации кода. Если для элемента требуется набор параметров, отличный от используемого по умолчанию, то он привязывается к элементу моделирования.

Шаг 6. Выбор компонентов и генерация кода:

Код может быть сгенерирован для всего пакета, для компонента или группы компонентов. Название компонента используется в качестве имени файла, содержащего полученный код. Файл с кодом помещается в папку, соответствующую названию пакета в представлении компонентов.

Задание 1. Фирма «Кухни Deluxe»

В предыдущих упражнениях была создана модель для системы обработки заказов (Order Entry). Теперь сгенерируем программный код C++ для этой системы. При этом воспользуемся диаграммой Компонентов системы, представленной на рисунке 37. Для генерации программного кода необходимо выполнить описанные ниже шаги.

Этапы выполнения упражнения

I Ввод тел пакетов на диаграмму Компонентов системы:

1) откройте диаграмму Компонентов системы;
2) выберите в браузере **Entities**: тело пакета **Order**;
3) перетащите тело пакета **Order** на диаграмму Компонентов системы;
4) повторите пп. 2 и 3 для следующих компонентов:
   - **Entities**: тело пакета **OrderItem**;
   - **Boundaries**: тело пакета **OrderOptions**;
   - **Boundaries**: тело пакета **OrderDetail**;
   - **Control**: тело пакета **TransactionMgr;**
   - **Control**: тело пакета **OrderMgr**.

II Установка языка C++:

1) откройте спецификацию компонента **Order** в пакете компонентов **Entities**;
2) выберите в качестве языка C++;
3) повторите пп. 1 и 2 для следующих компонентов:
   - **Entities**: спецификация пакета **OrderItem**;
   - **Boundaries**: спецификация пакета **OrderOptions**;
   - **Boundaries**: спецификация пакета **OrderDetail**;
   - **Control**: спецификация пакета **TransactionMgr;**
   - **Control**: спецификация пакета **OrderMgr**;
   - спецификация задачи **OrderClientExe**;
   - спецификация задачи **OrderServerExe.**

III Генерация программного кода C++:

1) откройте диаграмму Компонентов системы;
2) выберите все объекты на диаграмме Компонентов системы;
3) выберите **Tools → C++ → Code Generation** в меню.

Задание 2. Регистрация учебных курсов

Рассмотрим вопрос о планировании выпуска версий (расписание для каждого шага развития системы). В плане выпуска версий должны излагаться специфичные для версии цели: реализуемые возможности; уменьшаемые с данной версией риски; устраняемые версией дефекты.

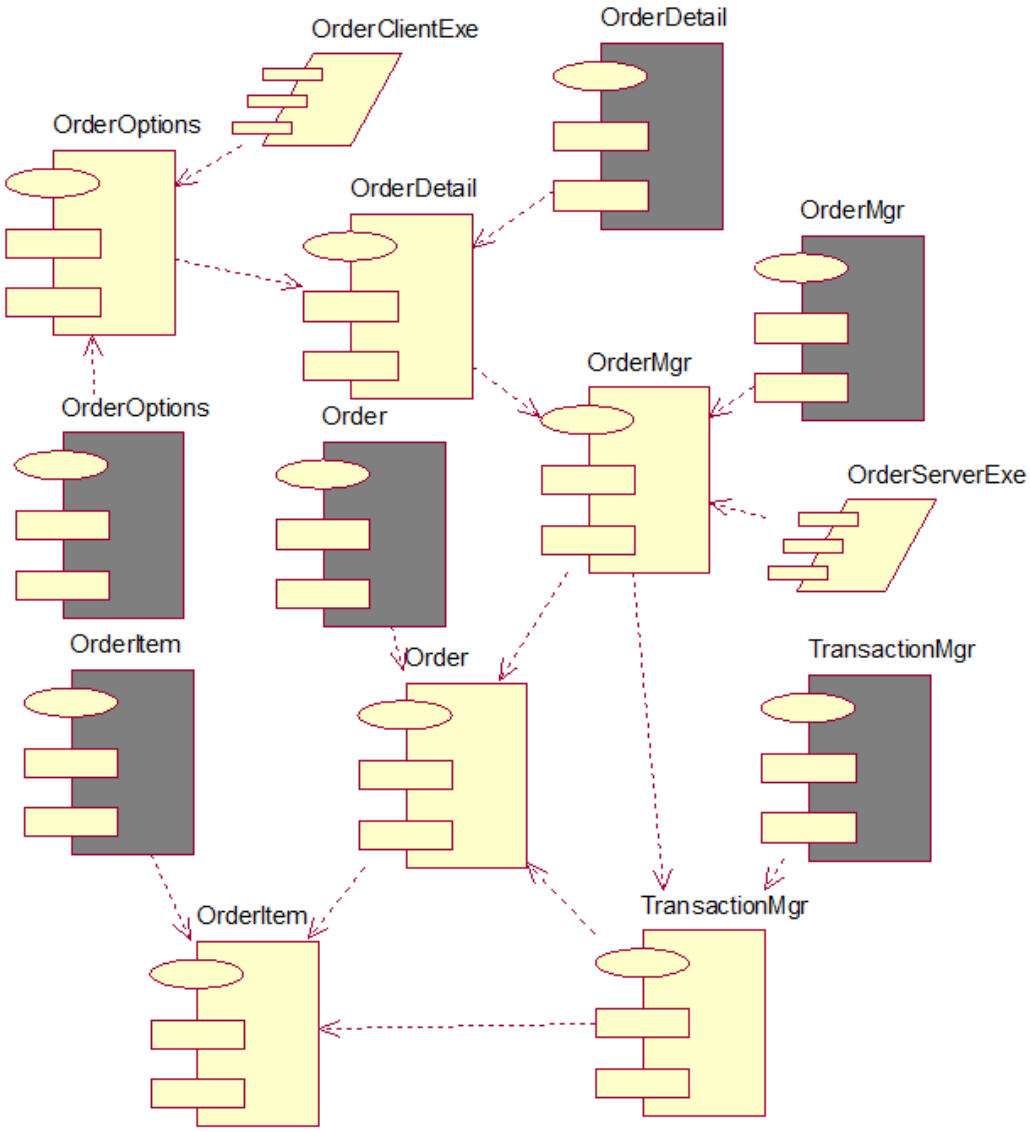

Рисунок 37 - Диаграмма компонентов системы Order Entry

Сценарии, созданные на этапе анализа, являются основными входными данными для этой стадии разработки. Сценарии изучаются и сортируются согласно степени риска, важности для заказчика и потребности в первоочередной разработке определенных базовых сценариев.

Для задачи регистрации учебных курсов план выпуска версий будет следующим: версия 1: Сохранение сведений о преподавателе, Выбор курсов для преподавателя, Создание учебного плана; версия 2: Сохранение сведений о студенте; Составление каталога; версия 3: Регистрация на курсы, Запрос списка учебных классов.

Версия 1 содержит риск, связанный с базой данных. Сведения о курсах должны храниться в базе данных, которая доступна для всех. Сценарии *Сохранение сведений о преподавателе* и *Выбор курсов для преподавателя* включены в данную версию, т.к. их нужно завершить до составления каталога. С помощью второй версии добавляется функциональность, необходимая для регистрации студента (сведения о студентах должны быть помещены в базу данных и каталог должен быть доступен студентам). Версия 3 завершает систему.

Рассмотрим стадию проектирования пользовательского интерфейса. На ранних стадиях жизненного цикла проекта для граничных классов системы были

созданы пустые классы. Доработаем их. Внесем данные о количестве и расположении окон и создадим обработчики событий, поступающих от пользователей. Для получения сведений о требованиях к пользовательскому интерфейсу применяются диаграммы последовательностей. Интерфейс пользователя позволяет получать все сообщения, поступающие от актера, и должен обеспечивать отправку всех сообщений, адресованных актеру.

После изучения сценариев, связанных с прецедентом **Выбор курсов для преподавания**, созданы окна для добавления, удаления и просмотра и определены следующие требования: 1) преподаватель должен ввести пароль для входа в систему (для ввода пароля создано отдельное окно; 2) если пароль введен, на экране появится окно **Параметры курса преподавателя** (ProfessorCourseOptions). Оно содержит поле для указания семестра и группу кнопок: **Добавить курс** (Add Course), **Удалить курс** (Delete Course), **Просмотр расписания** (Review Schedule), **Печать расписания** (Print Schedule).

Рассмотрим реализацию операции **Добавить курс** (Add Course), которая подразумевает добавление преподавателя в качестве учителя для данного курса. Этот сценарий требует отдельного окна для ввода преподавателем необходимой информации. Назовем окно **Добавление** (Addition), оно содержит следующие элементы: поля ввода (**Название предмета** (Course name) и **Номер предмета** (Course number)), список **Учебные курсы** (Course offerings), кнопки (**ОК, Отмена** (Cancel)**, Выход** (Quit)).

После того, как преподаватель ввел название и номер предмета, система получит и отобразит список курсов, из которого преподаватель может выбрать учебный курс. При щелчке на кнопке ОК объекту **Предмет** направляется сообщение.

Рассмотрим добавление классов уровня проектирования. Классы обычно добавляются в модель для выяснения способов реализации чего-либо в системе. Например, преподаватель должен ввести пароль, который обязательно проверяется. Для реализации этой задачи в систему добавляется класс, проверяющий пароль. По мере добавления классов в систему они отображаются на соответствующих диаграммах. Обновленная диаграмма классов для задачи регистрации учебных курсов приведена на рисунке 38.

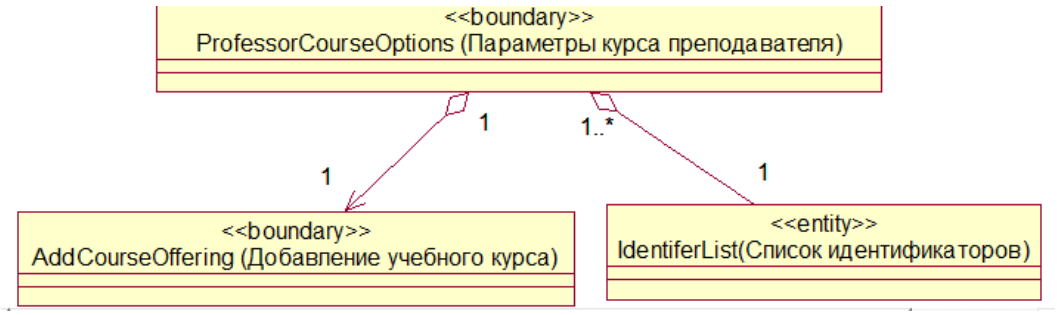

Рисунок 38 - Класс уровня проектирования

Для изменения направленности отношений можно воспользоваться командой **Navigation** (Направленность) контекстно-зависимого меню ассоциативной или агрегационной связи. На рисунке 39 представлены некоторые однонаправленные ассоциации.

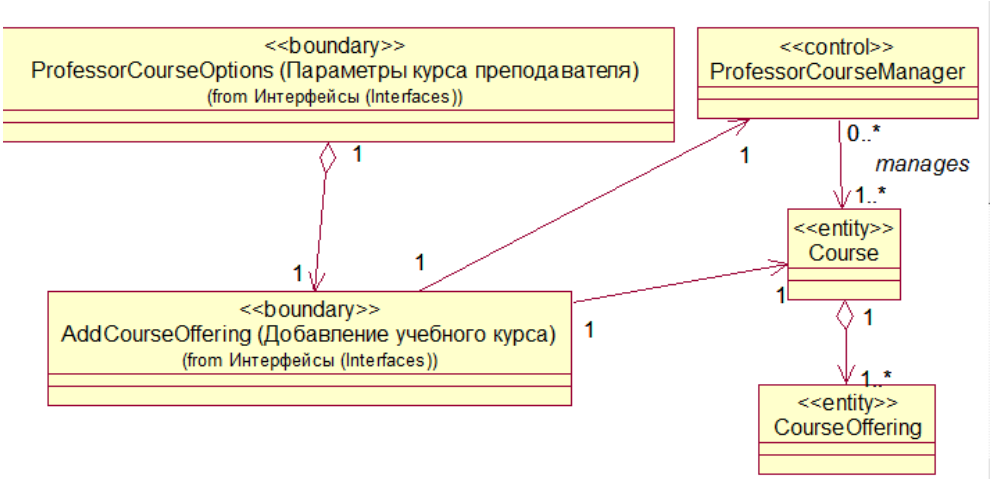

Рисунок 39 - Направленность отношений

В модели необходимо определить тип содержания в агрегационном отношении. Содержание может быть реализовано по значению (предполагает эксклюзивное владение для содержащего класса - закрашенные ромб) или по ссылке (не предполагает эксклюзивное владение для содержащего класса - не закрашенный ромб). Для указания типа агрегационного содержания нужно в диалоговом окне настройки параметров отношения на вкладке **Detail** (Детально) для роли, представляющей «целое» в агрегации, установить нужный тип содержания в группе переключателей **Containment** (Содержание).

Отношение с содержанием по значению (класс **Параметры курса преподавателя** (ProfessorCourseOptions) содержит класс **Добавление учебного курса** (AddCourseOffering)) и отношение с содержание по ссылке (отношение класса **Параметры курса преподавателя** (ProfessorCourseOptions) к классу **Список доступных идентификаторов** (IdentiferList) показаны на рисунке 40.

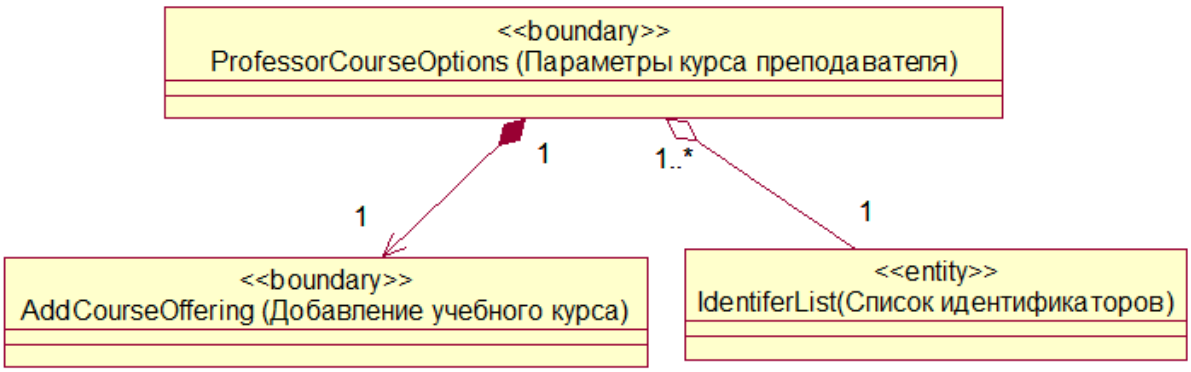

Рисунок 40 - Тип агрегационной связи

На данном этапе ассоциативные отношения могут быть преобразованы в отношения зависимости (объект, запросивший услугу (клиент) у другого объекта (поставщик услуги), не имеет представления о расположении объекта-поставщика). Ему необходимо сообщить, где находится объект-поставщик. Обычно он передается как параметр в один из методов класса-клиента или объявляется локально в области действия метода класса-клиента. Отношение зависимости изображаются пунктирной стрелкой, направленной от клиента к поставщику. Для создания отношений зависимости нужно щелкнуть по кнопке **Dependency Relationship** (Отношение зависимости) на панели инструментов и

соединить класс-клиент с классом-поставщиком. Отношение зависимости между классами **Учебный курс** (CourseOffering) и **Учебный курс БД** (DBCourseOffering) показано на рисунке 41 (диаграмма Занятия курса).

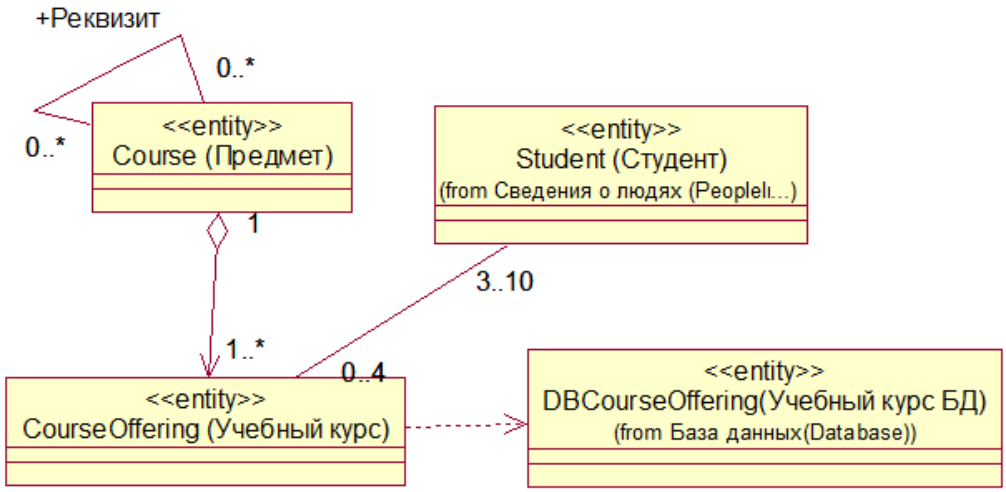

Рисунок 41 - Отношение зависимости

Создайте диаграмму компонентов для h- и cpp-файлов C++ как показано на рисунке 42 и сгенерируйте код.

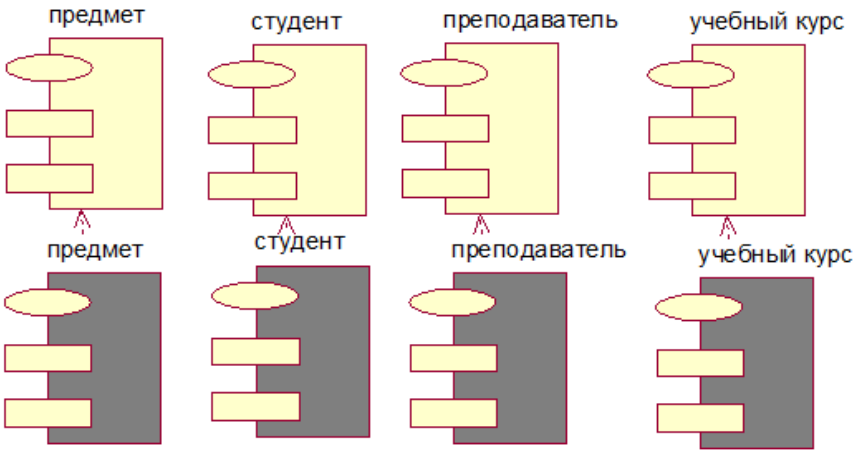

Рисунок 42 - Обновленная диаграмма компонентов

Задания для самостоятельной работы

Разработать модель и реализовать интерфейс в любой доступной среде программирования для одной из предметных областей:

1 Спроектировать покупку и продажу путевок в санаторий.
2 Спроектировать предварительную запись в поликлинику.
3 Спроектировать работу касс автовокзала.
4 Спроектировать работу салона красоты.
5 Спроектировать информационную систему «Библиотека».
6 Спроектировать систему выдачи методической литературы на кафедре.
7 Спроектировать информационную систему «Аптека».
8 Спроектировать информационную систему «Мебельный магазин».

9 Спроектировать информационную систему «Рекламное агентство».
10 Спроектировать систему записи детей в группу развития.
11 Спроектировать систему записи на подготовительные курсы в институт.
12 Спроектировать систему записи на курсы дизайна.
13 Спроектировать систему записи учеников на элективные курсы.
14 Спроектировать систему записи студентов на курсы дистанционного обучения.
15 Спроектировать систему записи слушателей на курсы английского языка.
16 Спроектировать систему записи на курсы в автошколу.
17 Спроектировать систему для предоставления услуг в автосервисе.
18 Спроектировать модель продажи билетов на железнодорожном вокзале.
19 Спроектировать модель программного обеспечения банкомата.
20 Спроектировать модель программного обеспечения кондиционера.
21 Спроектировать модель продажи авиабилетов.
22 Спроектировать модель продажи бытовой техники и электроники.
23 Спроектировать модель продажи компьютерной техники.
24 Спроектировать модель работы туроператора.
25 Спроектировать модель работы диспетчера такси.
26 Спроектировать модель работы магазина музыкальных инструментов.
27 Спроектировать работу компьютерного салона.
28 Спроектировать работу нотариальной конторы по оформлению документов.
29 Спроектировать работу копицентра.
30 Спроектировать работу фотосалона.
31 Спроектировать работу фитнес клуба.
32 Спроектировать работу школы танцев.
33 Спроектировать работу касс бассейна.
34 Спроектировать работу студии звукозаписи.
35 Спроектировать работу касс кинотеатра.

Список литературы

1 Буч Г., Джеймс Р., Джекобсон А. UML руководство пользователя. М. : ДМК Пресс, 2000. 458 с.
2 Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений. 3-е изд. / пер. с англ. М. : ООО «И.Д. Вильямс», 2008. 720с.
3 Гагарина Л.Г., Кокорева Е.В., Виснадул Б.Д. Технология разработки программного обеспечения : учебное пособие / под ред. Л.Г. Гагариной. М. : ИД «ФОРУМ» : ИНФРА-М, 2008. 400 с.
4 Кватрани Т. Rational Rose 2000 и UML. Визуальное моделирование / пер. с англ. М. : ДМК Пресс, 2001. 176 с.
5 Боггс М., Боггс У. UML и Rational Rose, 2008. 508с.
6 Иванова Г. С. Технология программирования. М. : Изд-во МГТУ им. Баумана, 2002.
7 Леоненков А. В. Самоучитель UML. СПб. : BHV, 2006. 578 с.
8 Леоненков А.В. Объектно-ориентированный анализ и проектирование с использованием UML и IBM Rational Rose, 2009. 320с.

Адаменко Юлия Владимировна

ПРАКТИКУМ ПО ОБЪЕКТНО-ОРИЕНТИРОВАННОМУ ПРОЕКТИРОВАНИЮ

Методические рекомендации для студентов
направления 44.03.01 «Педагогическое образование»
(профиль «Информатика»)

Редактор Н.Л. Борисова

……

Подписано в печать	Формат 60×84 1/16	Бумага 65 г/м2
Печать цифровая	Усл. печ. л. 3,25	Уч.-изд. л. 3,25
Заказ №	Тираж 25	Не для продажи

……...

РИЦ Курганского государственного университета.
640000, г. Курган, ул.Советская, 63/4.
Курганский государственный университет.