ЗНАКОМСТВО С ГРАФИЧЕСКОЙ КОМПЬЮТЕРНОЙ ПРОГРАММОЙ RATIONAL ROSE И ЕЕ ФУНКЦИОНАЛЬНЫМИ ВОЗМОЖНОСТЯМИ

Цель работы – изучить состав, назначение, функциональные возможности и общие правила работы в CASE-средстве Rational Rose на примере построения диаграмм прецедентов и классов.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

UML диаграммы в Rational Rose

Rational Rose – мощное CASE-средство для проектирования программных систем любой сложности. Одним из достоинств этого программного продукта будет возможность использования диаграмм на языке UML. Можно сказать, что Rational Rose является графическим редактором UML диаграмм.

CASE-средство (Computer Aided Software Engineering) – это инструмент, который позволяет автоматизировать процесс разработки информационной системы и программного обеспечения.

UML (Unified Modeling Language) – это графический язык моделирования общего назначения, предназначенный для спецификации, визуализации, проектирования и документирования всех артефактов, создаваемых при разработке программных систем. В рамках языка UML все представления о модели сложной системы фиксируются в виде специальных графических конструкций, получивших название **диаграмм**.

В распоряжение проектировщика системы Rational Rose предоставляет следующие типы диаграмм, последовательное создание которых позволяет получить полное представление о всей проектируемой системе и об отдельных ее компонентах:

- Use case diagram (диаграммы прецедентов);
- Deployment diagram (диаграммы топологии);
- Statechart diagram (диаграммы состояний);
- Activity diagram (диаграммы активности);
- Interaction diagram (диаграммы взаимодействия);
- Sequence diagram (диаграммы последовательностей действий);
- Collaboration diagram (диаграммы сотрудничества);
- Class diagram (диаграммы классов);
- Component diagram (диаграммы компонент).

Интерфейс Rational Rose

После запуска программы Rational Rose автоматически создается новый проект и в рабочем окне диаграммы появляется по умолчанию окно диаграммы классов.

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

- Главное меню программы;
- Окно диаграммы;
- Стандартная панель инструментов;
- Окно документации;
- Окно браузера;
- Окно журнала;
- Специальная панель инструментов.

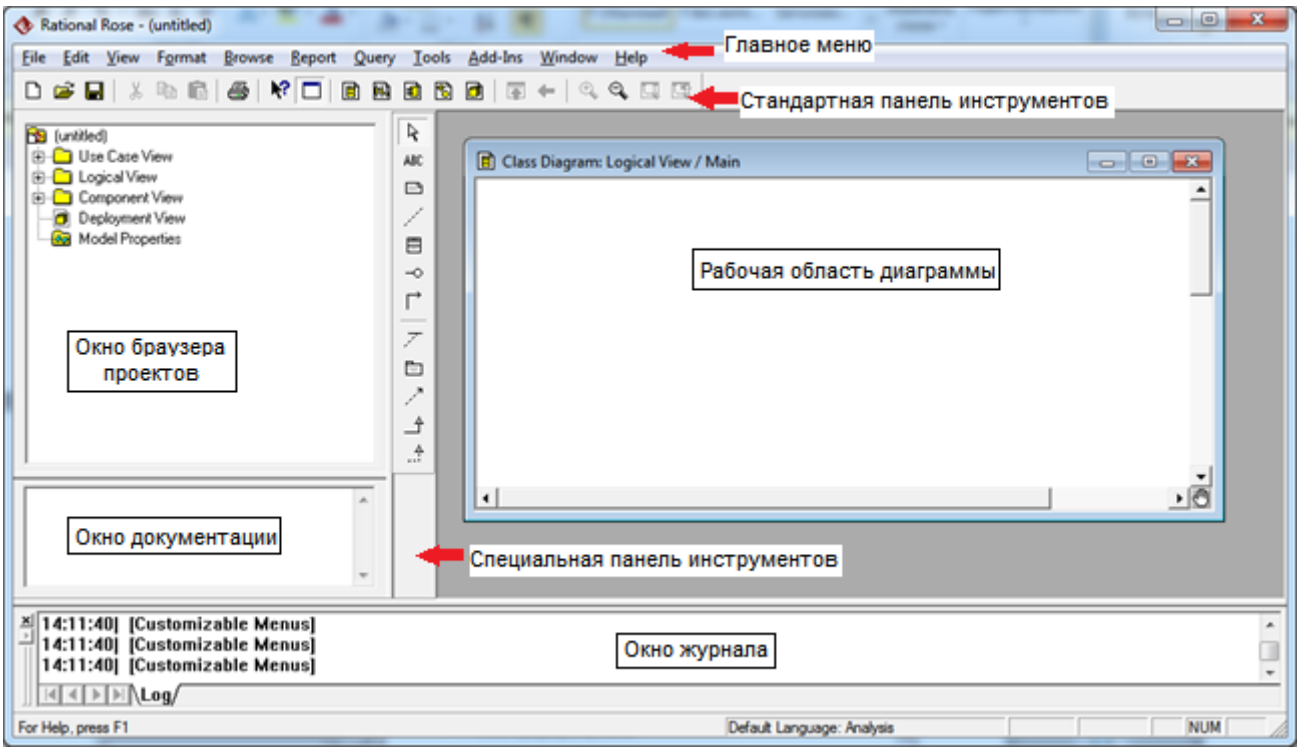

Рисунок 4.1 – Общий вид рабочего интерфейса CASE-средства Rational Rose

Отдельные пункты **главного мен**ю, назначение которых понятно из их названий, объединяют сходные операции, относящиеся ко всему проекту в целом. Некоторые из пунктов меню содержат хорошо знакомые функции (открытие проекта, вывод печать диаграмм, копирование в буфер и вставка из буфера различных элементов диаграмм). Другие настолько специфичны, что могут потребовать дополнительных усилий на изучение (опции генерации программного кода, проверка согласованности моделей, подключение дополнительных модулей).

Стандартная панель инструментов обеспечивает быстрый доступ к тем командам меню, которые выполняются разработчиками наиболее часто. Пользователь может настроить внешний вид этой панели по своему усмотрению.

Окно браузера предназначено для 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. Соответственно, выбрав элемент в окне браузера, мы можем его визуализировать в окне диаграммы или изменить его спецификацию. Браузер позволяет также организовывать элементы модели в пакеты и перемещать элементы между различными представлениями модели.

Специальная панель инструментов по умолчанию предназначена для построения диаграммы классов модели. Состав панели можно настраивать под конкретный вид диаграммы.

Окно диаграммы является основной рабочей областью ее интерфейса, в которой визуализируются различные представления модели проекта. При разработке нового проекта окно диаграммы представляет собой чистую область, не содержащую никаких элементов модели. Одновременно в окне диаграммы могут присутствовать несколько диаграмм, однако активной может быть только одна из них.

Окно документации предназначено для документирования элементов представления модели. В него можно записывать самую различную информацию, в том числе и на русском языке. Эта информация в последующем преобразуется в комментарии и никак не влияет на логику выполнения программного

кода. В окне документации отображается только та информация, которая относится к отдельному выделенному элементу диаграммы.

Окно журнала 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 сообщения об ошибках, возникающих при генерации программного кода.

Use case diagram (диаграмма прецедентов)

Этот вид диаграмм позволяет создать список операций, которые выполняет система. Часто этот вид диаграмм называют диаграммой функций, потому что на основе набора таких диаграмм создается список требований к системе и определяется множество выполняемых системой функций.

Каждая такая диаграмма или, как ее обычно называют, каждый Use case – это описание сценария поведения, которому следуют действующие лица (Actors).

Данный тип диаграмм используется при описании бизнес процессов автоматизируемой предметной области, определении требований к будущей программной системе. Отражает объекты как системы, так и предметной области и задачи, ими выполняемые.

При построении диаграммы прецедентов будем использовать пиктограммы типа "прецеденты" и "актеры".

Прецедент – это описание множества последовательных событий, выполняемых компьютерной системой, которые приводят к наблюдаемому актером результату. Графически прецедент изображается в виде ограниченного непрерывной линией эллипса, обычно содержащего только имя прецедента.

Актер – это кто-то (или что-то) внешний по отношению к компьютерной системе, кто взаимодействует с ней. Графически актер изображается в виде пиктограммы, представляющей человека, поскольку актер это человек или группа людей, использующих данные, предоставляемые компьютерной системой.

Рисунок 4.2 – Пример диаграммы прецедентов

Class diagram (диаграммы классов)

Этот вид диаграмм позволяет создавать логическое представление системы, на основе которого создается исходный код описанных классов. Диаграмма классов служит для представления статической структуры модели системы в терминологии классов объектно-ориентированного программирования и содержит детальную информацию об архитектуре программной системы.

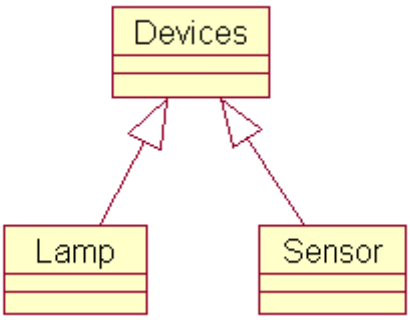

Рисунок 4.3 – Пример диаграммы классов

Отношения на диаграммах

Между компонентами диаграмм могут существовать различные отношения, которые описывают их взаимодействие. В языке UML имеется несколько стандартных видов отношений, например, ассоциация и обобщение.

Ассоциация – это структурное двунаправленное отношение, описывающее совокупность взаимоотношений между объектами. Ассоциация может иметь имя и кратность. Кратность (multiplicity) ассоциации указывается рядом с обозначением компонента диаграммы, который является участником данной ассоциации. Кратность характеризует общее количество конкретных экземпляров данного компонента, которые могут выступать в качестве элементов данной ассоциации. Наиболее распространенными являются следующие формы записи кратности отношения ассоциации:

1. Целое неотрицательное число (включая цифру 0). Предназначено для указания кратности, которая является строго фиксированной для элемента соответствующей ассоциации.

2. Два целых неотрицательных числа, разделенные двумя точками и записанные в виде: "первое число .. второе число", например, 1..5. Очевидно, что первое число должно быть строго меньше второго числа в арифметическом смысле, при этом первое число может быть равно 0.

Если кратность отношения ассоциации не указана, то по умолчанию принимается ее значение, равное 1.

Обобщение – это однонаправленное отношение, называемое "потомок/прародитель", в котором объект "потомок" может быть подставлен вместо объекта прародителя (родителя или предка). Потомок наследует структуру и поведение своего родителя. Графически обозначается сплошной линией со стрелкой в форме незакрашенного треугольника. Стрелка всегда указывает на родителя.

РАБОЧИЕ ЗАДАНИЯ

Задание 1. Построить в среде Rational Rose диаграмму прецедентов проекта регистрации учебных курсов. Порядок выполнения:

1.1. Изучить сценарий проекта, определить действующие лица системы и сценарии их поведения.

Сценарий проекта регистрации учебных курсов

Сначала каждый преподаватель университета заполняет специальную форму, в которой указывает, какие учебные курсы он намерен вести в следующем семестре. Данные из формы помещаются в университетский компьютер работником регистратуры.

После этого из полученных данных формируется каталог курсов, который раздается студентам. Студенты выбирают из каталога те курсы, на которых они собираются учиться, и подают заявки на обучение в

регистратуру. Все эти данные также попадают в компьютер, где происходит их обработка и формирование списков курсов и студентов. В задачи создаваемой системы входит, в частности, такое комплектование учебных курсов, чтобы каждый курс посещало бы от трех до десяти студентов. Если на какой-то курс не набирается трех студентов, он отменяется.

После формирования курсов преподаватели получают списки студентов, которых им предстоит обучать, а каждый студент получает подтверждение о зачислении на курс и счет на оплату.

1.2. Определить действующие лица системы.

Согласно сценарию действующих лиц в создаваемой системе четыре: преподаватель, студент, регистратор и биллинговая программа – система оплаты. Первые три выбраны действующими лицами, поскольку они активно взаимодействуют с создаваемой системой. Биллинговая же программа чаще всего является отдельным программным продуктом, а в нашем случае она получает информацию для своей работы от создаваемой курсовой системы, поэтому может считаться самостоятельным действующим лицом.

1.3. Определить сценарии поведения действующих лиц. Каждый сценарий описывает некоторое требование к функциям системы:

- Выбор курсов для преподавателя;
- Запрос расписания курсов;
- Регистрация на курсы;
- Создание каталогов ресурсов;
- Хранение информации о курсах;
- Хранение информации о преподавателях;
- Хранение информации о студентах.

1.4. Запустить программу Rational Rose. Ярлык программы:  . При запуске в окне **Create New Model** нажать кнопку **Cancel**.

1.5. В программе Rational Rose создать пустой проект. Для этого:

1.5.1. Переключиться на папку **Use Case View** – для работы с диаграммами прецедентов (Use case) и открыть контекстное меню нажатием правой кнопки мыши. Если теперь выбрать пункт **New** → **Actor** (рис. 4.4), то вы получите действующее лицо, которому следует дать имя **Преподаватель**.

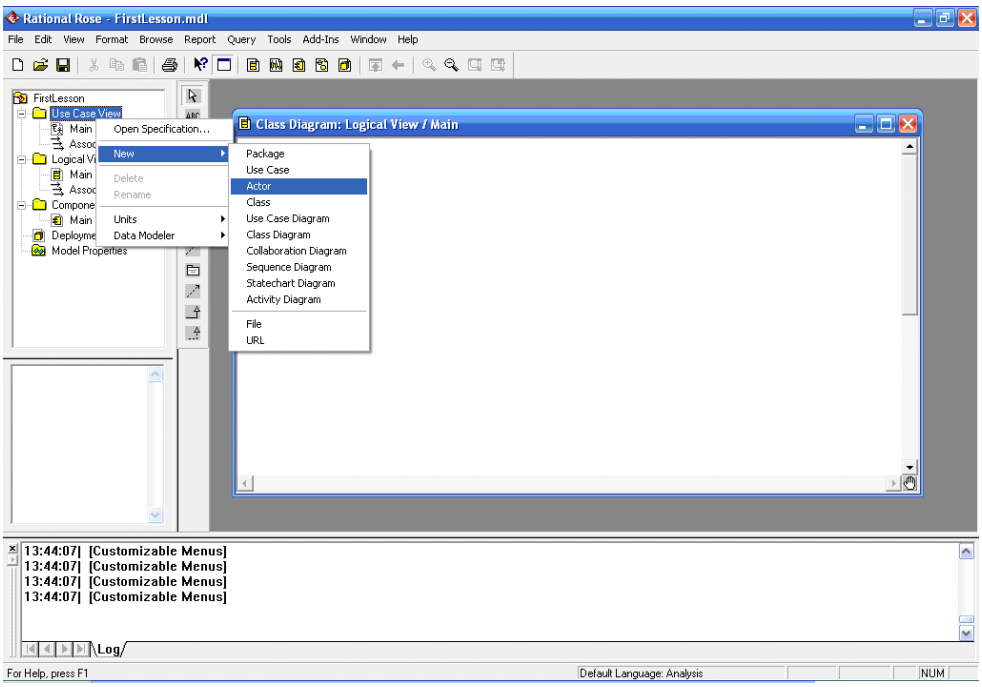

Рисунок 4.4 – Создание действующего лица

Далее аналогичным образом необходимо создать всех действующих лиц системы (рис. 4.5).

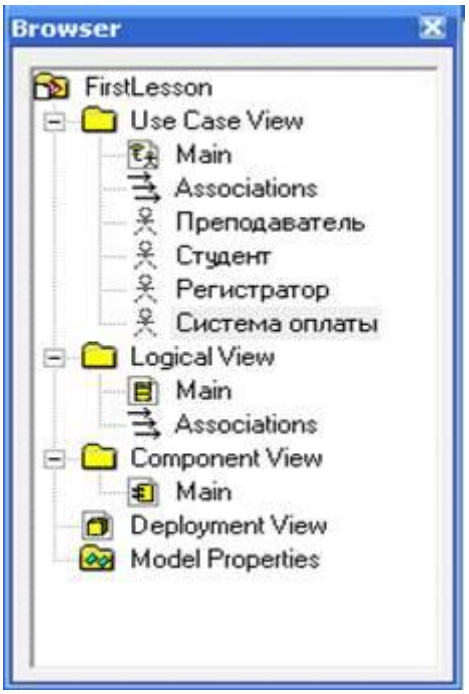

Рисунок 4.5 – Вид окна браузера проекта после создания всех действующих лиц

1.5.2. С помощью контекстного меню папки **Use Case View** и команды **New** → **Use Case** создать сценарий поведения действующих лиц, перечисленные в п. 3.

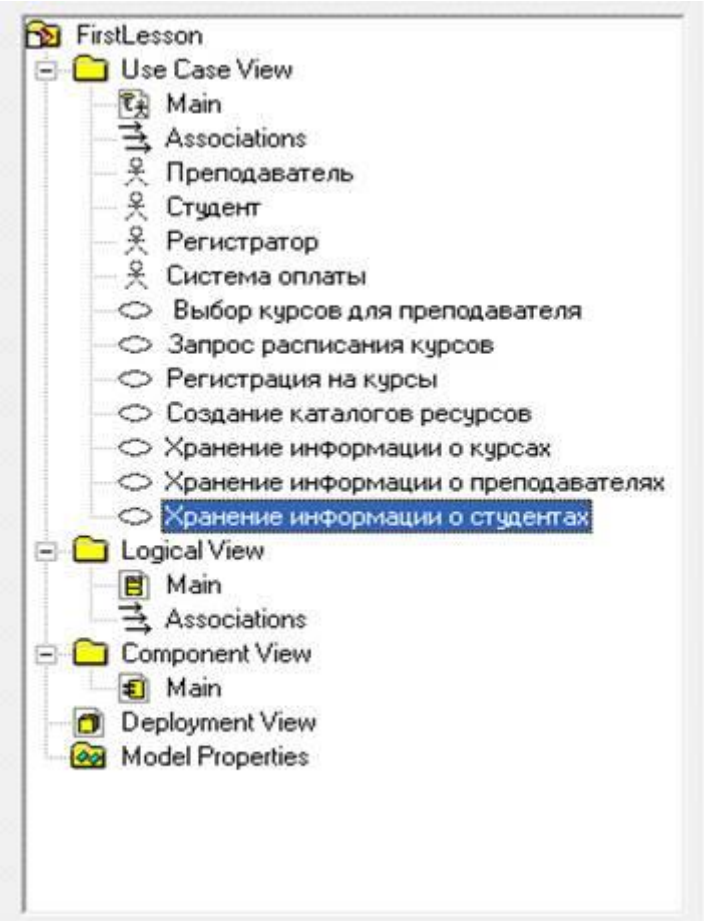

Рисунок 4.6 – Вид окна браузера проекта после создания всех сценариев поведения

1.6. Построить диаграмму сценариев поведения (прецедентов). Для этого:

1.6.1. Открыть окно построения диаграммы прецедентов двойным щелчком на пиктограмме **Main** из папки **Use Case View**.

1.6.2. Из окна браузера перетащить в окно построения диаграммы прецедентов все созданные действующие лица и сценарии поведения (рис. 4.7).

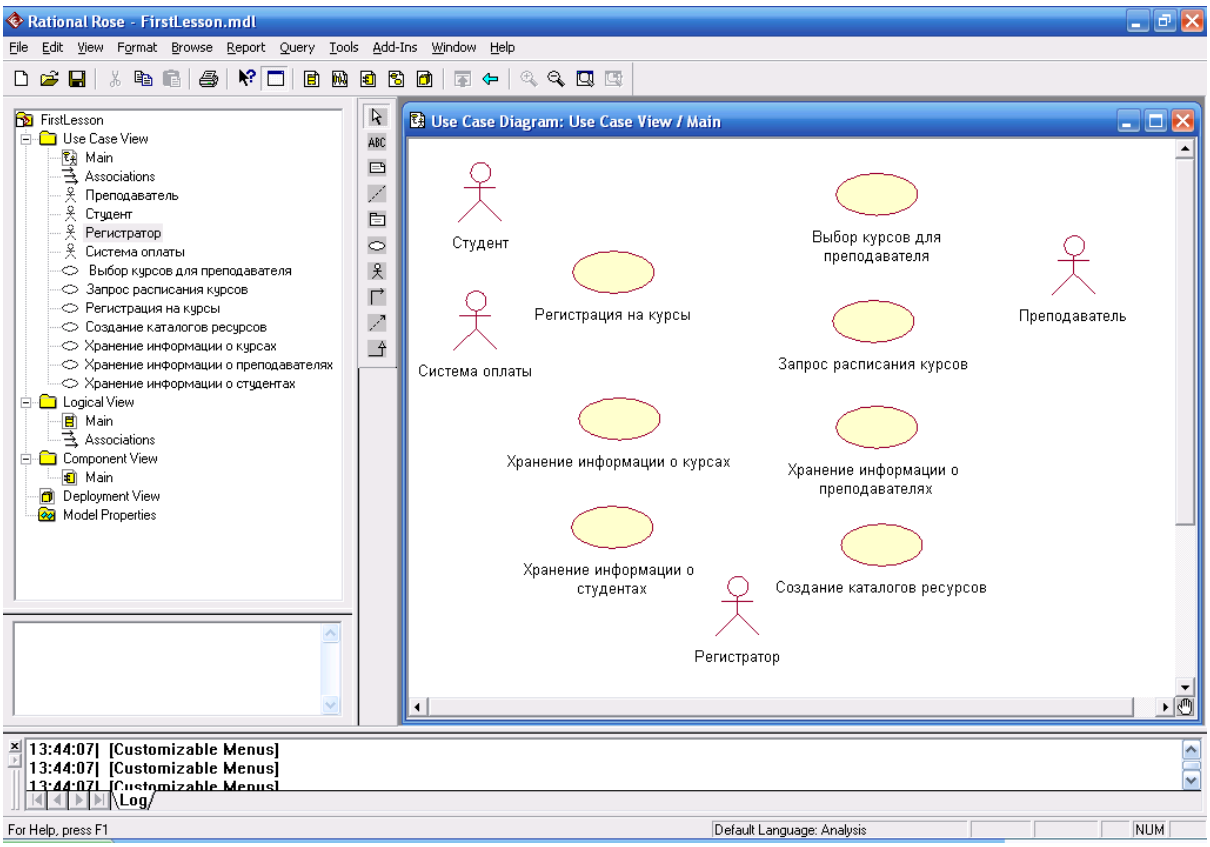

Рисунок 4.7 – Диаграмма сценариев поведения без связей

1.6.3. Установить взаимосвязи между действующими лицами и сценариями поведения (рис. 5.8). Для этого на специальной панели инструментов выбрать тип связи **Однонаправленная ассоциация (Unidirectional Association)**  , после чего протянуть линию между действующим лицом и сценарием поведения. В результате на диаграмме возникнет стрелка.

1.6.4. Создать границы между актерами и прецедентами, воспользовавшись пунктом меню **Tools → Create → Note Anchor** или пиктограммой специальной панели инструментов **Anchor Note to Item**. Выбрав инструмент необходимо щелкнуть один раз на середине одной из стрелок отношения, далее во всех углах интерфейса и завершить границу в месте начала ее рисования. Если прямоугольник получился не очень ровный то это можно исправить, выбрав в меню **Format → Line Style → Rectilinear**.

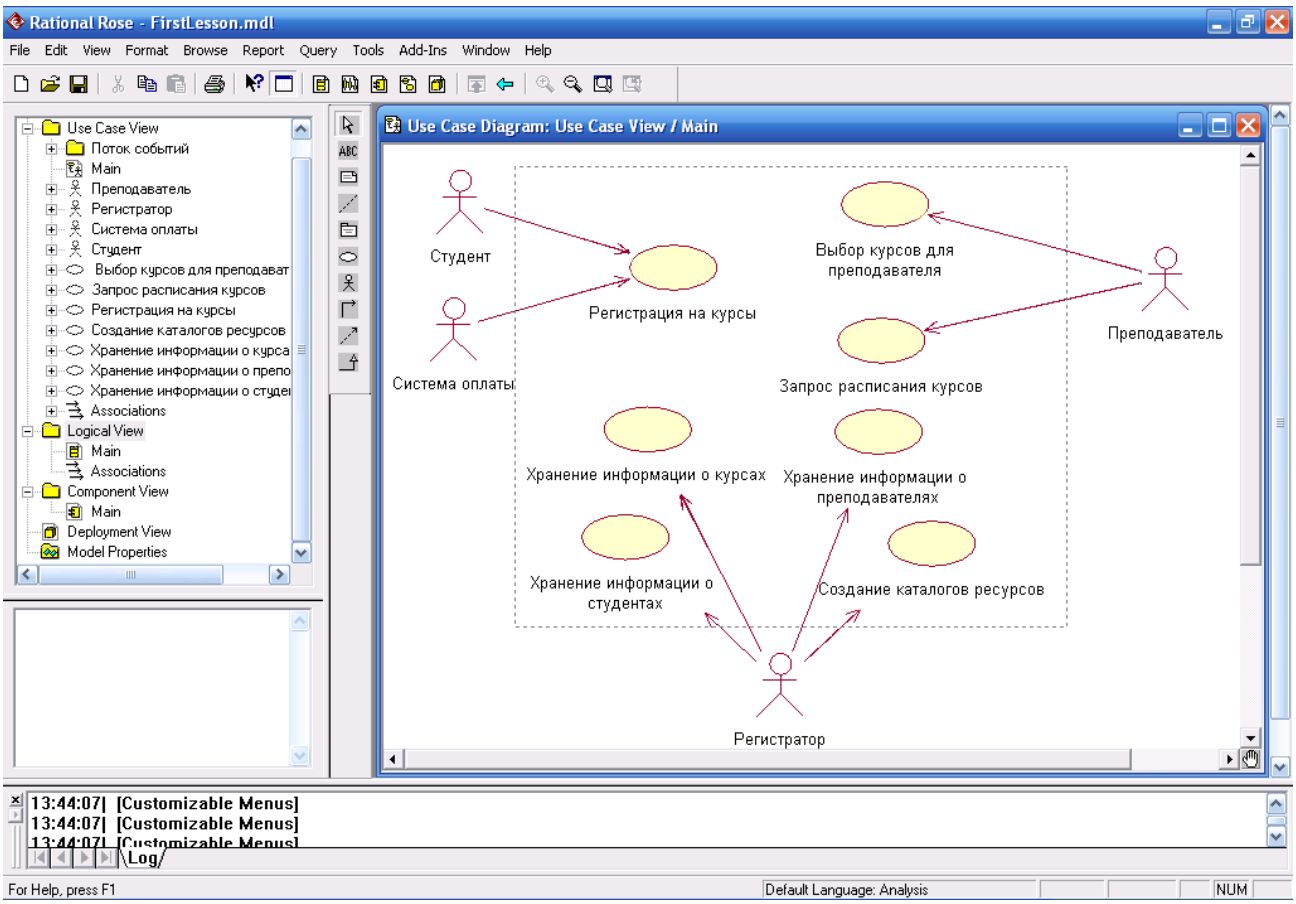

Рисунок 4.8 – Окончательный вариант диаграммы прецедентов системы регистрации учебных курсов

Задание 2. Построить диаграмму классов, представляющую фрагмент концептуальной схемы базы данных автоматической информационной системы (АИС) регистрации учебных курсов. Порядок выполнения:

2.1. Открыть окно построения диаграммы классов. Для этого можно воспользоваться одним из следующих вариантов:

1) раскрыть папку **Logical View** в браузере проекта и дважды щелкнуть на пиктограмме **Main**;

2) выполнить команды главного меню: **Browse** → **Class Diagram**.

2.2. В окне построения диаграммы классов создать четыре класса:

- Пользователь;
- Преподаватель;
- Студент;
- Учебный курс.

Для этого можно воспользоваться:

1) контекстным меню папки **Logical View** в браузере проекта, в котором выбрать команды **New** → **Class**, а затем перетащить созданный класс из окна браузера проекта в область окна диаграммы классов;

2) кнопкой **Class** на специальной панели инструментов, после нажатия которой щелкнуть левой кнопкой мыши на свободном месте окна диаграммы классов.

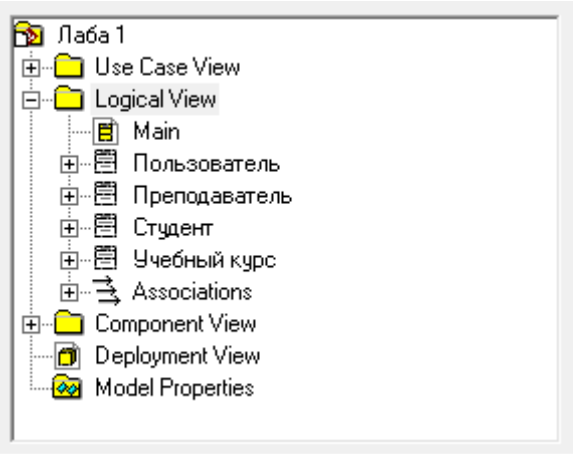

Рисунок 4.9 – Браузер проектов, отображающий все созданные классы

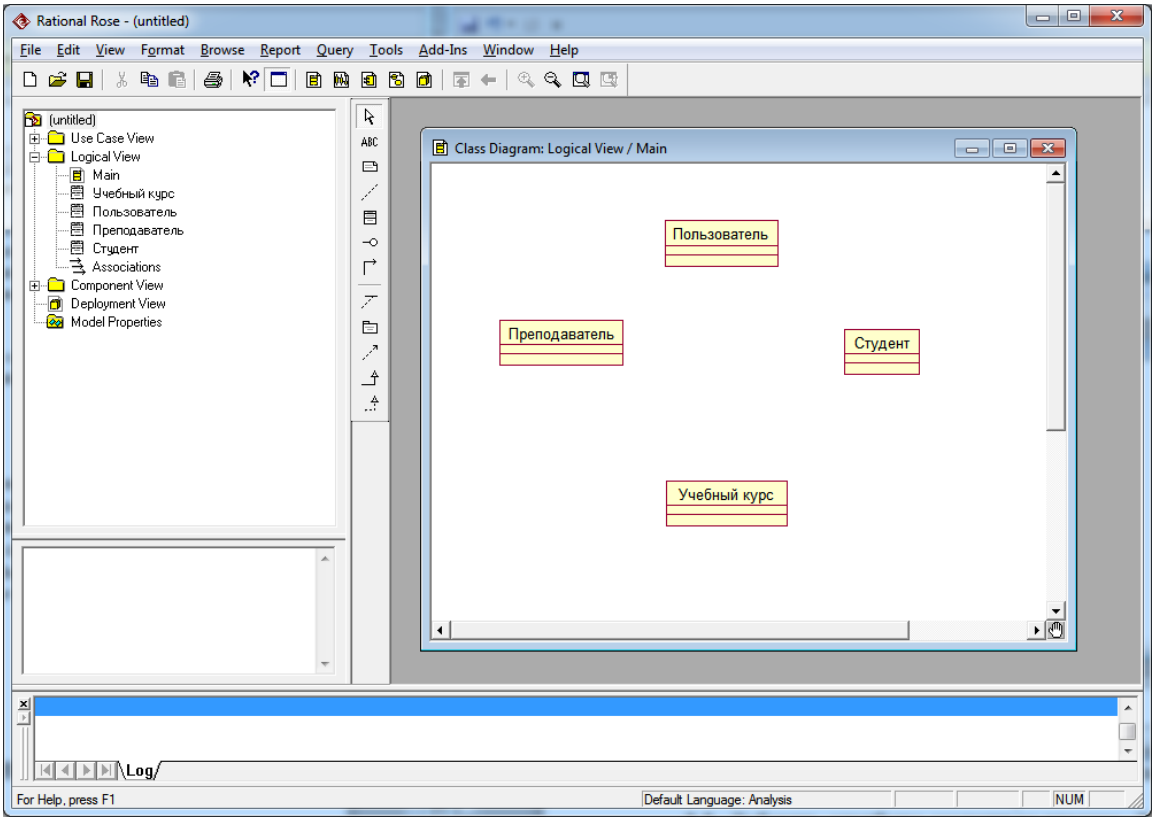

Рисунок 4.10 – Созданные классы в окне диаграммы классов

2.3. Добавить атрибуты созданным классам. Для класса **Пользователь** добавить атрибуты **Имя** и **Ид. номер**; для класса **Преподаватель** – атрибуты **Стаж работы**, **Имя** и **Ид. номер**; для класса **Студент** – атрибуты **Курс**, **Имя** и **Ид. номер**. Для класса **Учебный курс** атрибуты не добавлять.

Добавить атрибут можно одним из следующих способов:

1) В окне диаграммы классов щелкнуть правой кнопкой на графическом изображении нужного класса и выполнить команду контекстного меню **New Attribute**. В этом случае активизируется курсор ввода текста в области графического изображения класса на диаграмме

2) В окне браузера проекта щелкнуть правой кнопкой на нужном классе и выполнить команду контекстного меню **New Attribute**. В этом случае активизируется курсор ввода текста в области иерархического представления класса в браузере проекта под именем соответствующего класса.

3) Дважды щелкнуть на графическом изображении нужного класса. Откроется диалоговое окно свойств **Class Specification** соответствующего класса, в котором выбрать вкладку **Attributes**. В рабочем поле вкладки **Attributes** щелкнуть правой кнопкой мыши и выполнить команду контекстного меню **Insert**.

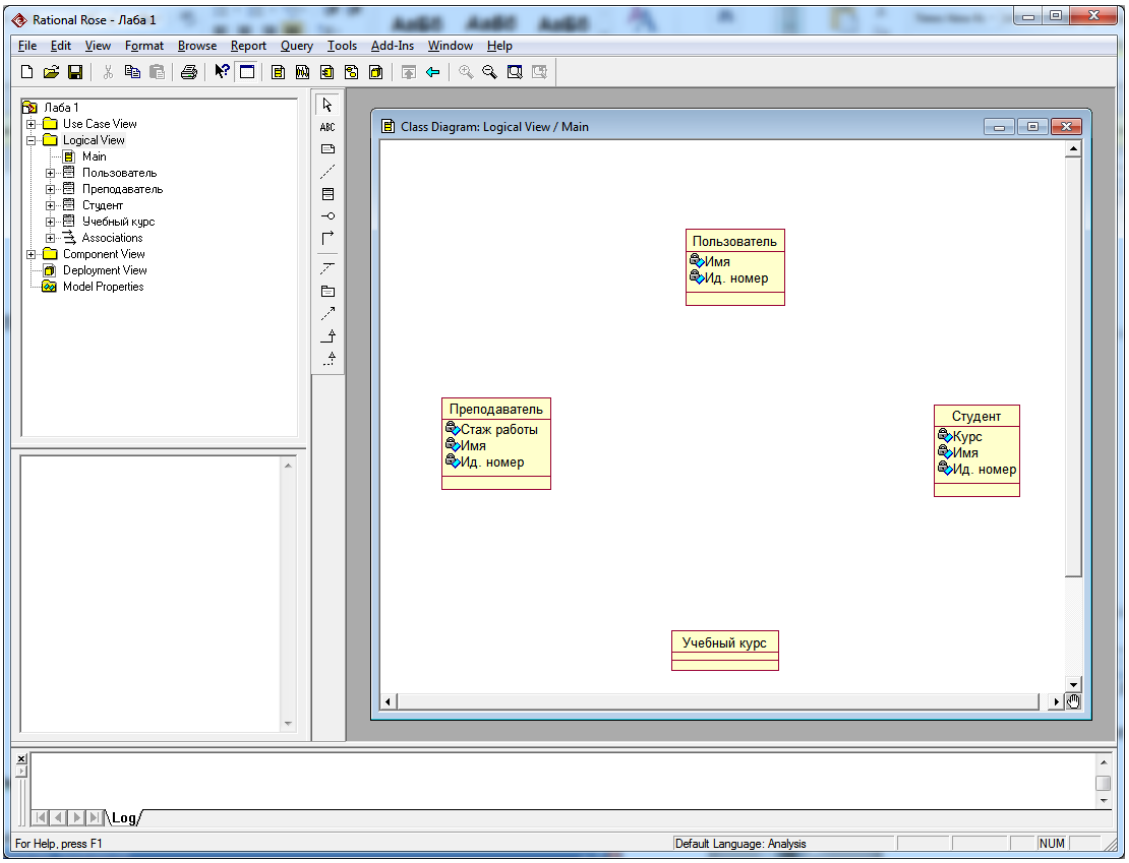

Рисунок 4.11 – Классы с атрибутами в окне диаграммы классов

2.4. Установить взаимосвязи между классами. Для этого:

2.4.1. С помощью инструмента **Unidirectional Association**  специальной панели инструментов построить взаимосвязь типа ассоциация между классами **Преподаватель** и **Учебный курс**. В диалоговом окне спецификации свойств ассоциации **Association Specification** (открывается двойным щелчком мыши на линии взаимосвязи) перейти на вкладку **Role B Detail**, отключить флажок **Navigable**, чтобы убрать направленность отношения, а в раскрывающемся списке **Multiplicity** выбрать кратность ассоциации равную 1. Далее перейти на вкладку **Role A Detail**, отключить флажок **Navigable**, в раскрывающемся списке **Multiplicity** выбрать кратность ассоциации "один ко многим" (1.. n), и вместо n поставить 5. Таким образом, мы указали кратность ассоциации: со стороны **Преподавателя** 1, а со стороны **Учебного курса** – диапазон кратности от 0 до 5. Это означает, что один преподаватель может вести от 0 до 5 учебных курсов.

2.4.2. Аналогично создать взаимосвязь между классами **Студент** и **Учебный курс**, считая, что в группе может быть от 3 до 10 студентов, которые могут быть слушателями от 0 до 4 курсов.

2.4.3. С помощью инструмента **Generalization**  специальной панели инструментов построить взаимосвязь типа обобщение между классами **Преподаватель**, **Студент** и **Пользователь**, учитывая, что в данном случае родитель – **Пользователь**.

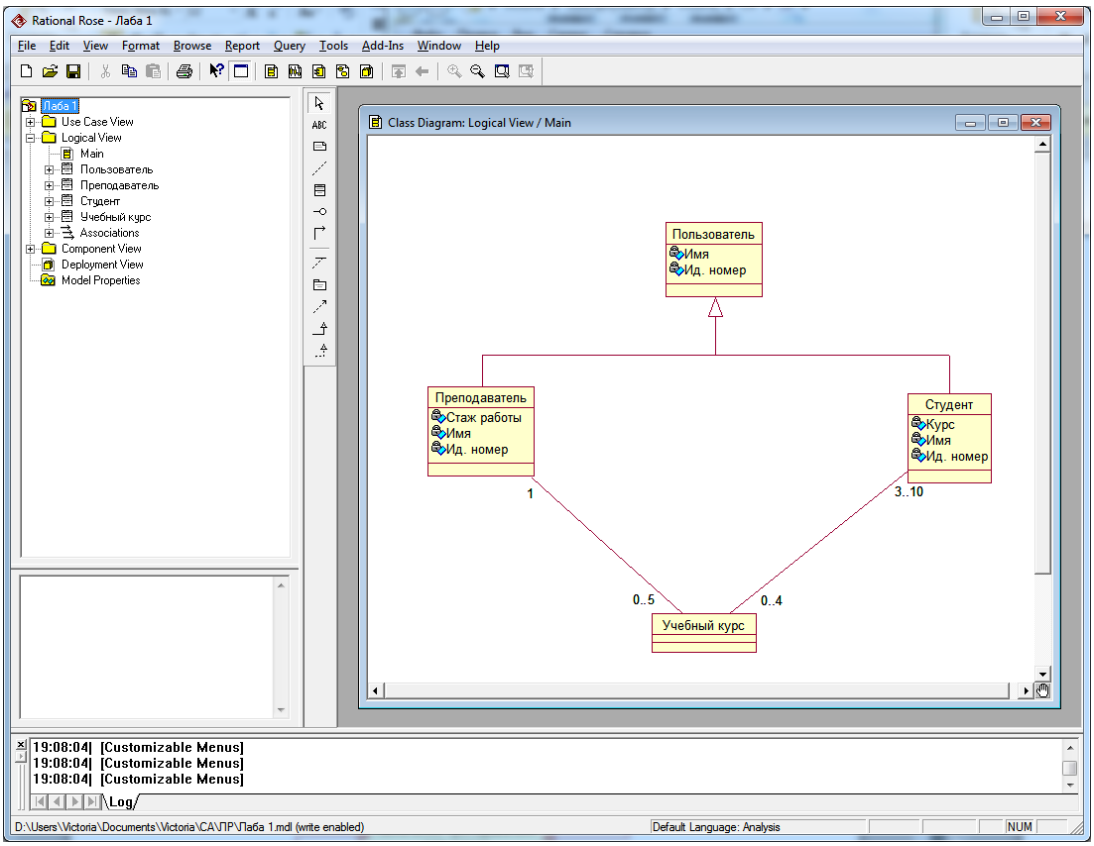

Рисунок 4.12 – Окончательный вариант простой диаграммы классов

Контрольные вопросы

1. Какое назначение программного средства Rational Rose?

2. Что такое CASE-средство?

3. Что такое UML?

4. Что называют диаграммами UML?

5. Какие типы UML диаграмм Вам известны?

6. Назовите основные элементы окна программы Rational Rose.

7. Какое назначение диаграммы прецедентов?

8. Что такое прецедент в UML?

9. Что представляет собой актер в UML?

10. Какое назначение диаграммы классов?

11. Что такое отношение ассоциации?

12. Как установить ассоциации между компонентами диаграммы?

13. Что такое кратность ассоциации?

14. Как задать кратность ассоциации?

15. Что представляет собой отношение обобщения?

16. Как установить отношение обобщения?

17. Опишите последовательность создания компонентов диаграммы прецедентов.

18. Опишите способы задания атрибутов компонентам диаграммы классов.