МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра информационных систем и технологий

УТВЕРЖДАЮ
Проректор по учебной работе
_____ О. Г. Локтионова
«___» _____ 2017г.

ДИАГРАММЫ КЛАССОВ ЯЗЫКА UML

Методические указания по выполнению
лабораторных работ по дисциплине
«Проектирование информационных систем»
для студентов направления подготовки бакалавров
09.03.02 Информационные системы
09.03.03 Прикладная информатика

Курск 2017

УДК 004.82 (075.8)
Составитель: Т.И.Лапина

Рецензент

Доктор технических наук, профессор *Р.А.Томакова*

Проектирование информационных систем: методические указания по выполнению лабораторных работ / Юго-Зап. гос. ун-т; сост.: Т. И. Лапина, Курск, 2017. 25 с.: ил. 27, табл. 1, Библиогр.: с. 5.

Содержат краткие теоретические сведения о методах разработки диаграмм классов при проектированию информационных систем на основе использования нотаций языка UML.

Методические указания соответствуют требованиям программ по направлениям подготовки бакалавров: 09.03.02 Информационные системы, 09.03.03 Прикладная информатика.

Предназначены для студентов направления подготовки бакалавров 09.03.02 Информационные системы, 09.03.03, Прикладная информатика дневной и заочной форм обучения.

Текст печатается в авторской редакции

Подписано в печать Формат 60x84 1/16.

Усл. печ. л. . Уч. – изд. л. .Тираж 100 экз. Заказ.

Бесплатно.

Юго - Западный государственный университет.
305040, г. Курск, ул. 50 лет Октября, 94.

Лабораторная работа №7

1 Цель работы

Приобретение навыков использования инструментальных сред для разработки диаграмм классов при проектированию информационных систем на основе использования нотаций языка UML

2 Основные теоретические положения

Для моделирования систем на логическом уровне используются следующие виды диаграмм:

диаграммы классов (class diagrams);

диаграммы поведения системы (behavior diagrams), которые включают:

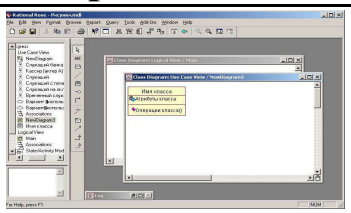
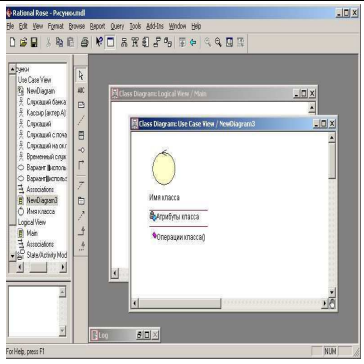
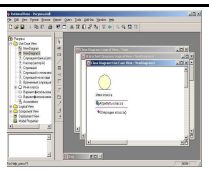
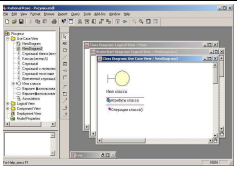
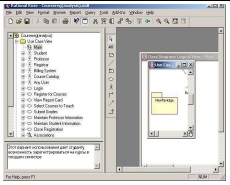
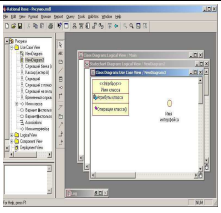
- ***диаграммы взаимодействия (interaction diagrams)*** - для моделирования процесса обмена сообщениями между объектами, к которым относятся:
  - ***диаграммы последовательности (sequence diagrams)***;
  - ***кооперативные диаграммы (collaboration diagrams)***;
- ***диаграммы состояний (statechart diagrams)*** - для моделирования поведения объектов системы при переходе из одного состояния в другое;
- ***диаграммы деятельностей (activity diagrams)*** - для моделирования поведения системы в рамках различных вариантов использования, или моделирования деятельностей.

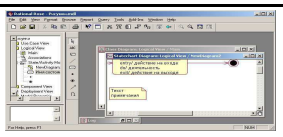
2.1 Диаграммы классов

Диаграмма классов ***(class diagrams)*** служит для представления статической структуры модели системы в терминах классов объектно-ориентированного проектирования. Диаграмма классов определяет типы классов системы и различного рода статические связи, которые существуют между ними. На диаграммах классов изображаются также атрибуты классов, операции классов и ограничения, которые накладываются на связи между классами.

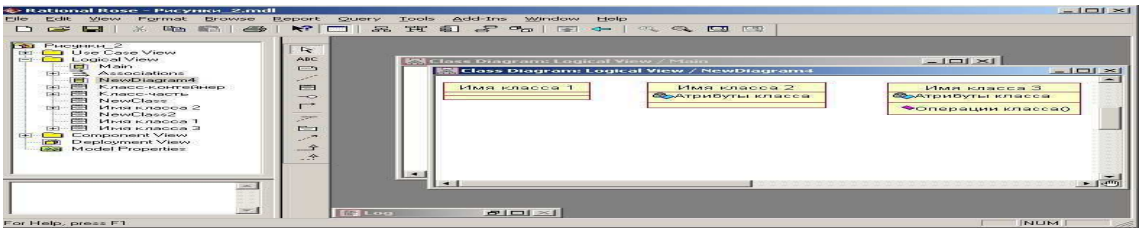
Основные элементы диаграммы классов приведены в табл. 1.

Таблица 1 – Основные элементы диаграммы классов

Графическое изображение	**Название**
	Класс (class)
	Управляющий класс (control class)
	Класс-сущность (entity class)
	Граничный класс (boundary class)
	Пакет (package)
	Интерфейс (Interface)
	Однонаправленная ассоциация, связь (unidirectional association)
	Обобщение (generalization relationship)
	Агрегирование (aggregation relationship)

	Композиция (composition relationship)
	Отношение зависимости (dependency or instantiates)
	Связь класса с ассоциацией (association class)
	Реализация (realize)
	Связь примечания (Anchor Note to Item)
	Примечание (note)
ABC	Надпись (text box)

Класс (***class***) в языке UML служит для обозначения множества объектов, которые обладают одинаковой структурой, поведением и отношениями с объектами из других классов. Графически класс изображается в виде прямоугольника, который дополнительно может быть разделен горизонтальными линиями на разделы или секции (рис. 1). В этих разделах могут указываться имя класса, атрибуты (переменные) и операции (методы).



а б в

Рисунок 1–Графическое изображение класса на диаграмме классов

Обязательным элементов обозначения класса является его имя. На начальных этапах разработки диаграммы отдельные классы могут обозначаться простым прямоугольником с указанием только имени соответствующего класса (рис. 1, а). По мере проработки отдельных компонентов диаграммы описания классов дополняются атрибутами (рис. 1, б) и операциями (рис. 1, в).

Предполагается, что окончательный вариант диаграммы содержит наиболее полное описание классов, которые состоят из трех разделов или секций. Иногда в обозначениях классов используется дополнительный четвертый раздел, в котором приводится семантическая информация справочного характера или явно указываются исключительные ситуации.

Даже если секция атрибутов и операций является пустой, в обозначении класса она выделяется горизонтальной линией, чтобы сразу отличить класс от других элементов языка UML. Примеры графического изображения классов на диаграмме классов приведены на рис. 2. В первом случае для класса "Прямоугольник" (рис. 2, а) указаны только его атрибуты — точки на координатной плоскости, которые определяют его расположение. Для класса "Окно" (рис. 2, б) указаны только его операции, секция атрибутов оставлена пустой. Для класса "Счет" (рис. 2, в) дополнительно изображена четвертая секция, в которой указано исключение — отказ от обработки просроченной кредитной карточки.

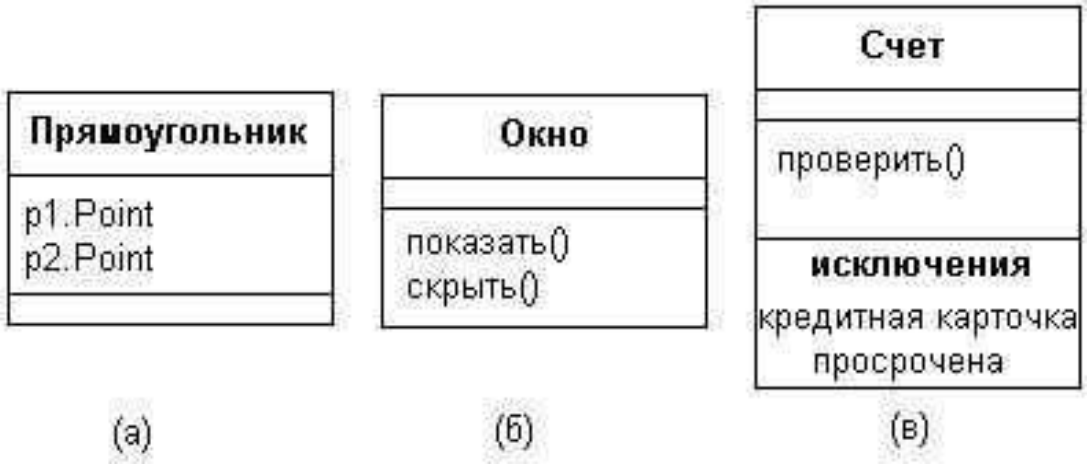


Рисунок 2– Примеры графического изображения классов на диаграмме

Стереотипы классов

Стереотипы - это механизм, позволяющий разделять классы на категории. В языке UML основными стереотипами являются: ***Boundary*** (граница), ***Entity*** (сущность) и ***Control*** (управление).

Граничные классы (***boundary classes***) - это классы, которые расположены на границе системы и окружающей среды. Они включают все формы, отчеты, интерфейсы с аппаратурой (такой, как принтеры

или сканеры) и интерфейсы с другими системами.

Для того чтобы найти граничные классы, надо исследовать диаграммы вариантов использования. Каждому взаимодействию между действующим лицом и вариантом использования должен соответствовать по крайней мере один граничный класс. Именно такой класс позволяет действующему лицу взаимодействовать с системой.

Классы-сущности (***entity classes***) отражают основные понятия (абстракции) предметной области и, как правило, содержат хранимую информацию. Обычно для каждого класса-сущности создают таблицу в базе данных.

Управляющие классы (***control classes***) отвечают за координацию действий других классов. Обычно у каждого варианта использования имеется один управляющий класс, контролирующий последовательность событий этого варианта использования. Управляющий класс отвечает за координацию, но сам не несет в себе никакой функциональности - остальные классы не посылают ему большого количества сообщений. Вместо этого он сам посылает множество сообщений. Управляющий класс просто делегирует ответственность другим классам, по этой причине его часто называют классом-менеджером.

В системе могут быть и другие управляющие классы, общие для нескольких вариантов использования. Например класс ***SecurityManager*** (менеджер безопасности), отвечающий за контроль событий, связанных с безопасностью. Класс ***Transaction-Manager*** (менеджер транзакций) занимается координацией сообщений, относящихся к транзакциям с базой данных. Могут быть и другие менеджеры для работы с другими элементами функционирования системы, такими, как разделение ресурсов, распределенная обработка данных или обработка ошибок.

Помимо упомянутых выше стереотипов можно создавать и свои собственные.

Атрибуты

Атрибут (***attribute***) класса - это элемент информации, связанный с классом. Атрибут служит для представления отдельного свойства или признака, который является общим для всех объектов данного класса. Например, у класса ***Company*** (Компания) могут быть атрибуты ***Name*** (Название), ***Address*** (Адрес) и ***NumberOfEmployees*** (Число служащих).

Атрибуты записываются во второй сверху секции прямоугольника класса.

Операции

Операция (***operation***) – это некоторый сервис, который предоставляет каждый экземпляр или объект класса по требованию своих клиентов (других объектов, в том числе и экземпляров данного класса). Операции реализуют связанное с классом поведение. Операция включает три части - имя, параметры и тип возвращаемого значения. Параметры - это аргументы, получаемые операцией «на входе». Тип возвращаемого значения относится к результату действия операции. Операции записываются в третьей секции прямоугольника класса.

Механизм пакетов

Пакеты применяют, чтобы сгруппировать классы, обладающие некоторой общностью. Существует несколько наиболее распространенных подходов к группировке. Во-первых, можно группировать их по стереотипу. В таком случае получается один пакет с классами-сущностями, один с граничными классами, один с управляющими классами и т.д. Этот подход может быть полезен с точки зрения размещения готовой системы, поскольку все находящиеся на клиентских машинах компоненты с граничными классами уже оказываются в одном пакете.

Другой подход заключается в объединении классов по их функциональности. Например, в пакете ***Security*** (безопасность) содержатся все классы, отвечающие за безопасность приложения. В таком случае другие пакеты могут называться ***Employee Maintenance*** (Работа с сотрудниками), ***Reporting*** (Подготовка отчетов) и ***Error Handling*** (Обработка ошибок). Преимущество этого подхода заключается в возможности повторного использования.

Механизм пакетов применим к любым элементам модели, а не только к классам. Если для группировки классов не применять некоторые эвристики, то она становится весьма произвольной. Одна из них, которая в основном используется в UML, - это зависимость. Зависимость между двумя пакетами существует в том случае, если между любыми двумя классами в пакетах существует любая зависимость. Таким образом, ***диаграмма пакетов*** (рис. 13) представляет собой диаграмму, содержащую пакеты классов и

зависимости между ними. Строго говоря, пакеты и зависимости являются элементами диаграммы классов, т.е. диаграмма пакетов - это форма диаграммы классов.

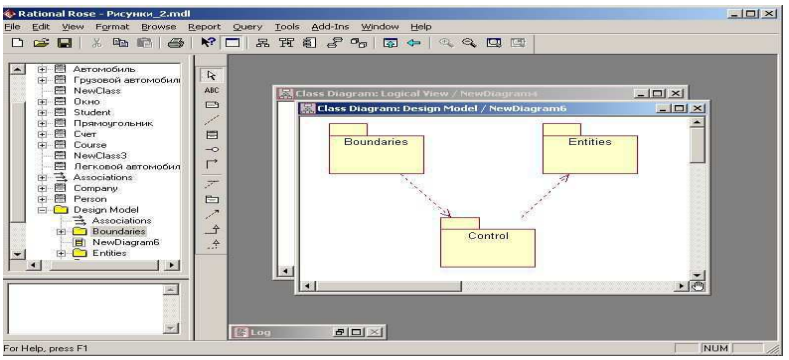


Рисунок 13–Диаграмма пакетов

Зависимость между двумя элементами имеет место в том случае, если изменения в определении одного элемента могут повлечь за собой изменения в другом. Что касается классов, то причины для зависимостей могут быть самыми разными: один класс посылает сообщение другому; один класс включает часть данных другого класса; один класс использует другой в качестве параметра операции. Если класс меняет свой интерфейс, то любое сообщение, которое он посылает, может утратить свою силу.

Пакеты не дают ответа на вопрос, каким образом можно уменьшить количество зависимостей в вашей системе, однако они помогают выделить эти зависимости, а после того поработать над снижением их количества. Диаграммы пакетов можно считать основным средством управления общей структурой системы.

Пакеты являются жизненно необходимым средством для больших проектов. Их следует использовать в тех случаях, когда диаграмма классов, охватывающая всю систему в целом и размещенная на единственном листе бумаги формата А4, становится нечитаемой.

Интерфейсы.

Интерфейс (***interface***) является специальным случаем класса, у которого имеются только операции и отсутствуют атрибуты. Для его изображения используется специальный графический символ — прямоугольник класса со стереотипом ***<<interface>>*** (рис. 14). При этом секция атрибутов у прямоугольника отсутствует, а указывается только секция операций.

а) б)

Рисунок 14– Пример графического изображения интерфейса на диаграмме классов

Пример:

Диаграмма классов для варианта использования «Снять деньги со счета» показана на рис. 15.

На этой диаграмме классов показаны связи между классами, реализующими вариант использования «***Снять деньги со счета***». В этом процессе задействованы четыре класса: ***Card Reader*** (устройство для чтения карточек), ***Account*** (счет), ***ATM Screen*** (экран АТМ) и ***Cash Dispenser*** (кассовый аппарат). Каждый класс на диаграмме выглядит в виде прямоугольника, разделенного на три части. В первой содержится имя класса, во второй - его атрибуты. В последней части содержатся операции класса, отражающие его поведение (действия, выполняемые классом).

Связывающие классы линии отражают взаимодействие между классами. Так, класс ***Account*** связан с классом ***ATM Screen***, потому что они непосредственно сообщаются и взаимодействуют друг с другом. Класс ***Card Reader*** не связан с классом ***Cash Dispenser***, поскольку они не сообщаются друг с другом непосредственно.

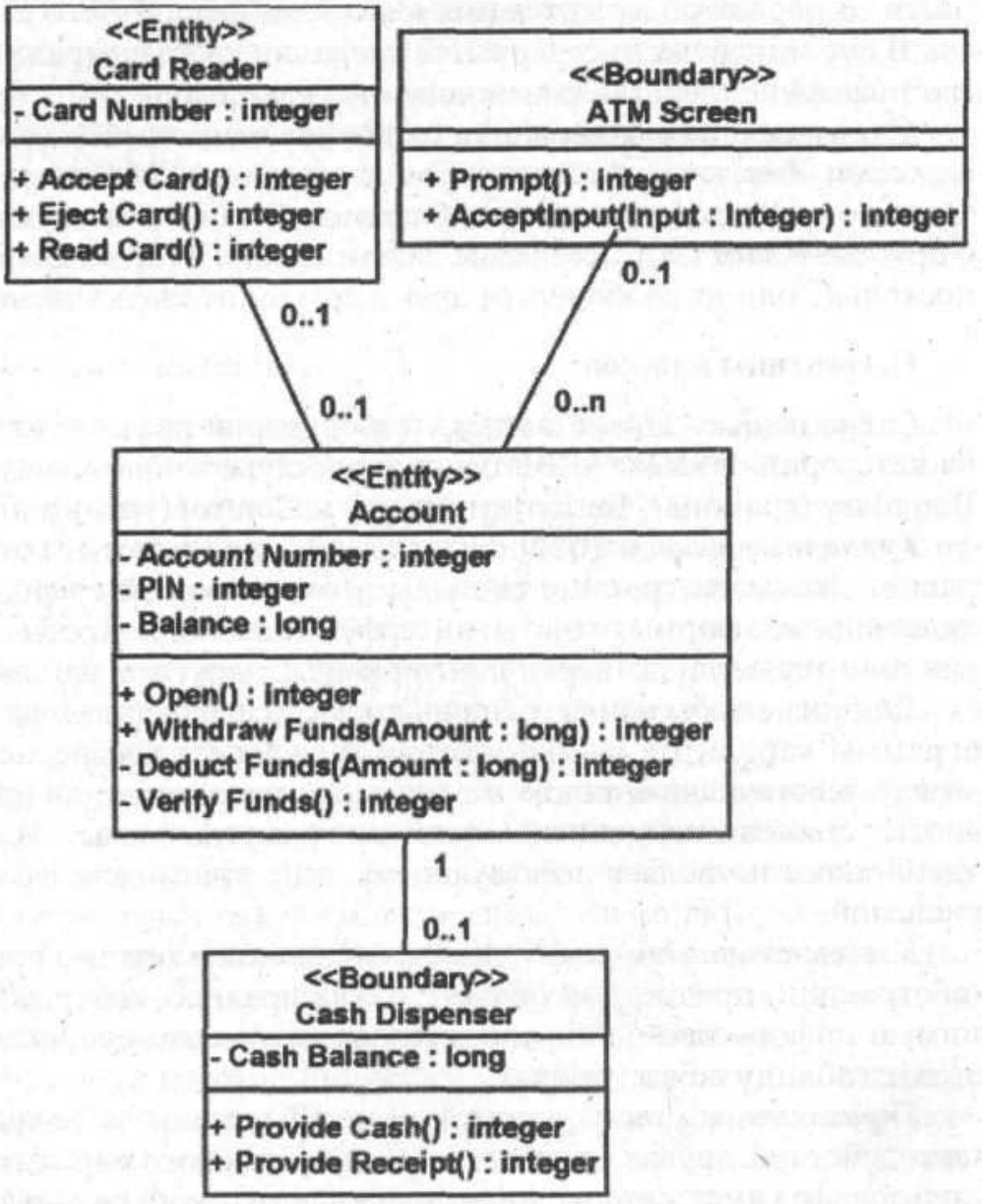


Рисунок 15– Диаграмма классов для варианта использования «Снять деньги со счета»

3 Порядок выполнения работы

Принятие соглашений по моделированию включает:

- используемые диаграммы и элементы модели;
- правила их применения;
- соглашения по именованию элементов;
- организацию модели (пакеты).

Пример соглашений моделирования

- Имена вариантов использования должны быть короткими глагольными фразами.
- Для каждого варианта использования должен быть создан пакет ***Use-Case Realization***, включающий:
  - по крайней мере одну реализацию варианта использования;
  - диаграмму «***View Of Participating Classes***» (VOPC).
- Имена классов должны быть существительными, соответствующими по возможности понятиям предметной области.
- Имена классов должны начинаться с заглавной буквы.
- Имена атрибутов и операций должны начинаться со строчной буквы.
- Составные имена должны быть сплошными, без подчеркиваний, каждое отдельное слово должно начинаться с заглавной буквы.

Идентификация ключевых абстракций.

Заключается в предварительном определении классов системы (классов анализа). Источники - знание предметной области, требования к системе, глоссарий. Классы анализа для системы регистрации показаны на рис. 16.

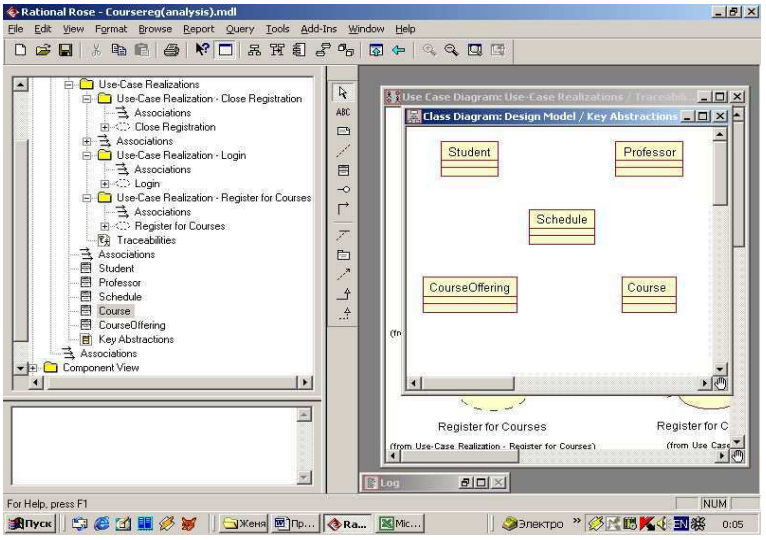


Рисунок 16– Классы анализа для системы регистрации

Упражнение 1. Создание структуры модели и классов анализа в соответствии с требованиями архитектурного анализа

Структура логического представления браузера должна иметь следующий вид (рис. 17).

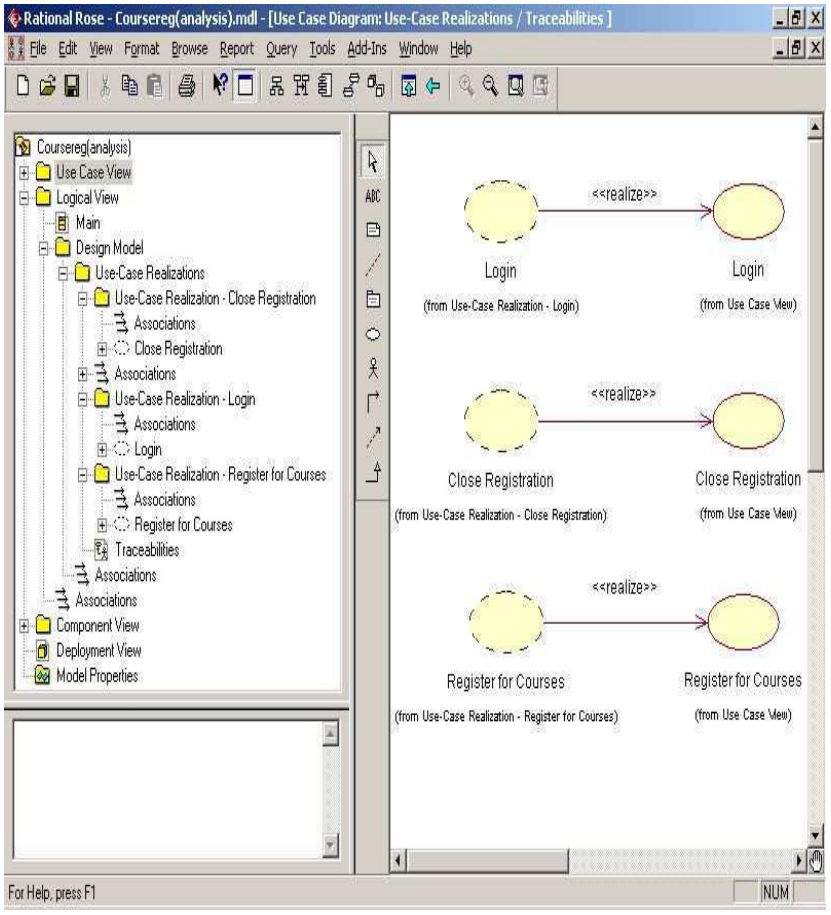


Рисунок 17– Структура логического представления браузера

Создание пакетов:

1. Щелкните правой кнопкой мыши по логическому представлению (***Logical View***) браузера.
2. Выберите пункт ***New > Package*** в открывшемся меню.
3. Назовите новый пакет ***Design Model***.

Создание классов анализа и соответствующей диаграммы ***Key Abstractions***:

1. Щелкните правой кнопкой мыши по пакету ***Design Model***.

2. Выберите пункт ***New > Class*** в открывшемся меню. Новый класс под названием ***NewClass*** появится в браузере.

3. Выделите его и введите имя ***Student***.

4. Создайте аналогичным образом классы ***Professor***, ***Schedule***, ***Course*** и ***CourseOffering***.

5. Щелкните правой кнопкой мыши по пакету ***Design Model***.

6. Выберите пункт ***New > Class Diagram*** в открывшемся меню.

7. Назовите новую диаграмму классов ***Key Abstractions***.

8. Чтобы расположить вновь созданные классы на диаграмме классов, откройте ее и перетащите классы на открытую диаграмму мышью. Диаграмма классов должна выглядеть, как на рис. 16.

3.2 Идентификаторы классов

Идентификация классов для вариантов использования осуществляется на основе анализа потоков событий варианта использования. В потоках событий варианта использования выявляются классы трех типов:

- ***граничные классы*** (***Boundary***) - служат посредниками при взаимодействии внешних объектов с системой. Как правило, для каждой пары «действующее лицо - вариант использования» определяется один граничный класс. Типы граничных классов: пользовательский интерфейс (обмен информацией с пользователем, без деталей интерфейса - кнопок, списков, окон), системный интерфейс и аппаратный интерфейс (используемые протоколы, без деталей их реализации);
- ***классы-сущности*** (***Entity***) - представляют собой ключевые

абстракции (понятия) разрабатываемой системы. Источники выявления классов-сущностей: ключевые абстракции, созданные в процессе архитектурного анализа, глоссарий, описание потоков событий вариантов использования;

- ***управляющие классы*** (***Control***) - обеспечивают координацию поведения объектов в системе. Могут отсутствовать в некоторых вариантах использования, ограничивающихся простыми манипуляциями с хранимыми данными. Как правило, для каждого варианта использования определяется один управляющий класс. Примеры управляющих классов: менеджер транзакций, координатор ресурсов, обработчик ошибок.

Пример набора классов, участвующих в реализации варианта использования ***Register for Courses***, приведен на рис. 18.

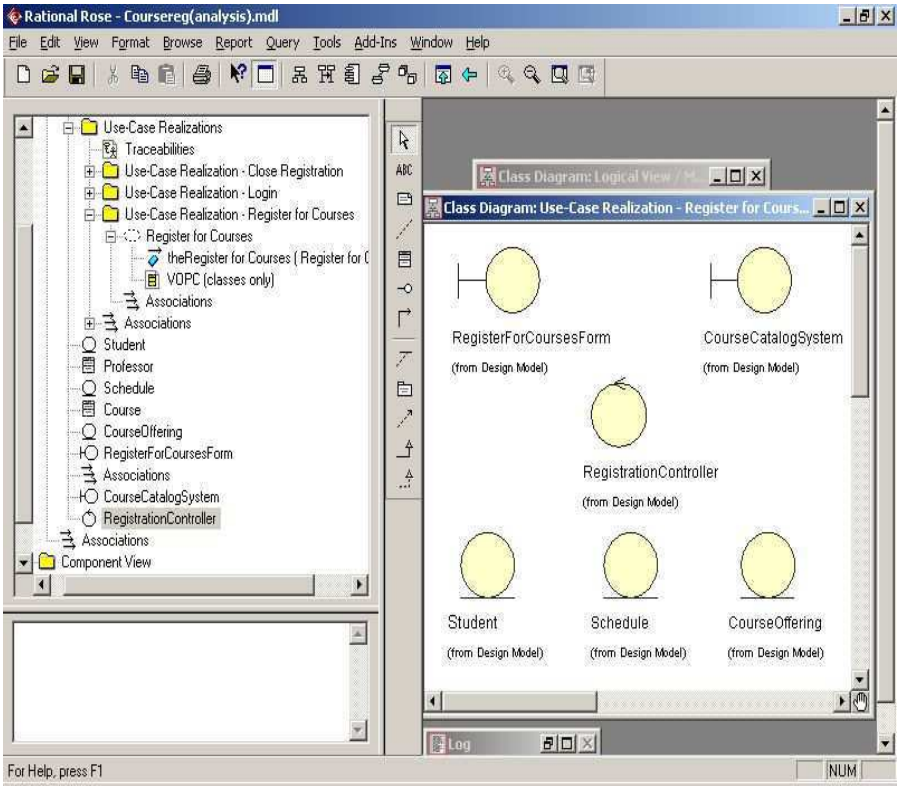


Рисунок 18– Классы, участвующие в реализации варианта использования
Register for Courses

Упражнение 2. Создание классов, участвующих в реализации варианта использования Register for Courses, и диаграммы классов «View Of Participating Classes» (VOPC)

1. Щелкните правой кнопкой мыши по пакету ***Design Model***.

2. Выберите, пункт ***New > Class*** в открывшемся меню. Новый класс под названием ***NewClass*** появится в браузере.

3. Выделите его и введите имя ***RegisterForCoursesForm***.

4. Щелкните правой кнопкой мыши по классу ***RegisterForCoursesForm***.

5. Выберите пункт ***Open Specification*** в открывшемся меню.

6. В поле стереотипа выберите ***Boundary*** и нажмите на кнопку ***OK***.

7. Создайте аналогичным образом классы ***CourseCatalogSystem*** со стереотипом ***Boundary*** и ***RegistrationController*** со стереотипом ***Control***.

8. Назначьте классам ***Schedule***, ***CourseOffering*** и ***Student*** стереотип ***Entity***.

9. Щелкните правой кнопкой мыши по кооперации ***Register for Courses*** в пакете ***Use-Case Realization - Register for Courses***.

10. Выберите пункт ***New > Class Diagram*** в открывшемся меню.

11. Назовите новую диаграмму классов ***VOPC*** (***classes only***).

12. Откройте ее и перетащите классы на открытую диаграмму в соответствии с рис. 18.

3.3. Создание атрибутов и связей между классами

Упражнение 3. Добавление атрибутов к классам

Настройка

1. В меню модели выберите пункт ***Tools > Options***.

2. Перейдите на вкладку ***Diagram***.

3. Убедитесь, что флажок ***Show All Attributes*** включен.

4. Убедитесь, что флажки ***Suppress Attributes*** и ***Suppress Operations*** не включены.

Добавление атрибутов

1. Щелкните правой кнопкой мыши по классу ***Student***.

2. Выберите пункт ***New Attribute*** в открывшемся меню.

3. Введите новый атрибут ***address***.

4. Нажмите клавишу <***Enter***>.

5. Повторите шаги 1-4, добавив атрибуты ***name*** и ***studentID***.

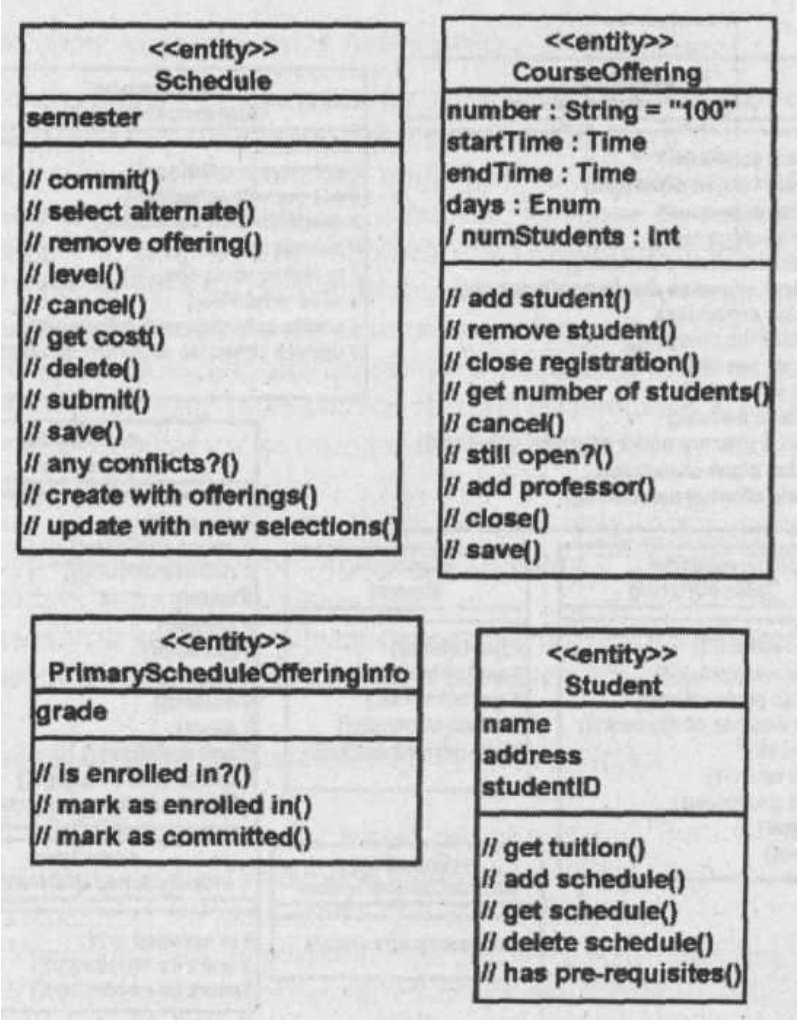
6. Добавьте атрибуты к классам ***CourseOffering***, ***Shedule*** и ***PrimaryScheduleOfferinglnfo***, как показано на рис. 19.

Связи между классами (ассоциации) определяются на основе диаграмм взаимодействия. Если два объекта взаимодействуют (обмениваются сообщениями), между ними должна существовать связь (путь взаимодействия). Для ассоциаций задаются множественность и, возможно, направление навигации. Могут использоваться множественные ассоциации, агрегации и классы ассоциаций.

Упражнение 4. Добавление связей

Добавим связи к классам, принимающим участие в варианте использования ***Register for Courses***. Для отображения связей между классами построим три новые диаграммы классов в кооперации ***Register for Courses*** пакета ***Use-Case Realization - Register for Courses*** (рис. 20- 22).

Добавлены два новых класса - подклассы ***FulltimeStudent*** (Студент очного отделения) и ***ParttimeStudent*** (Студент вечернего отделения).



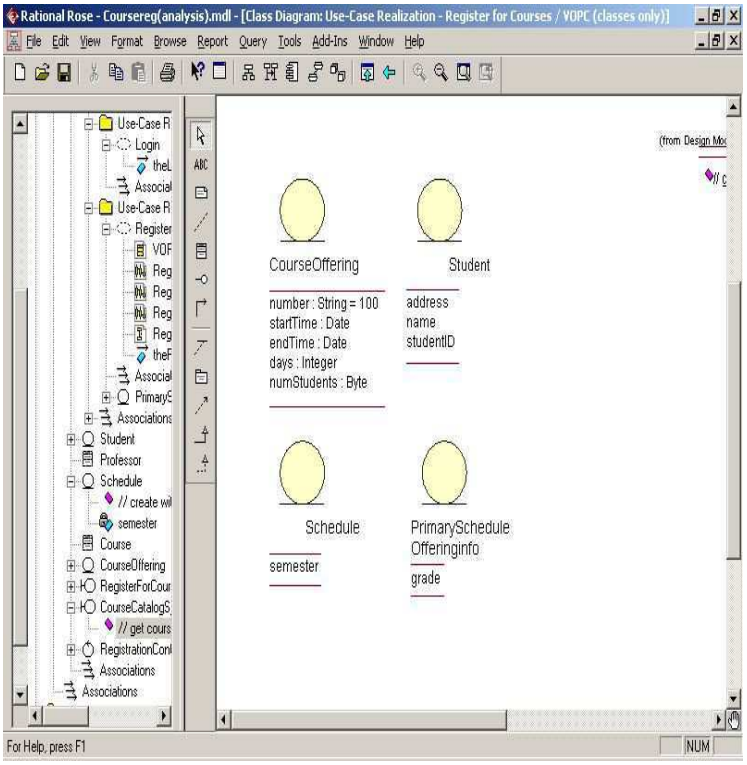
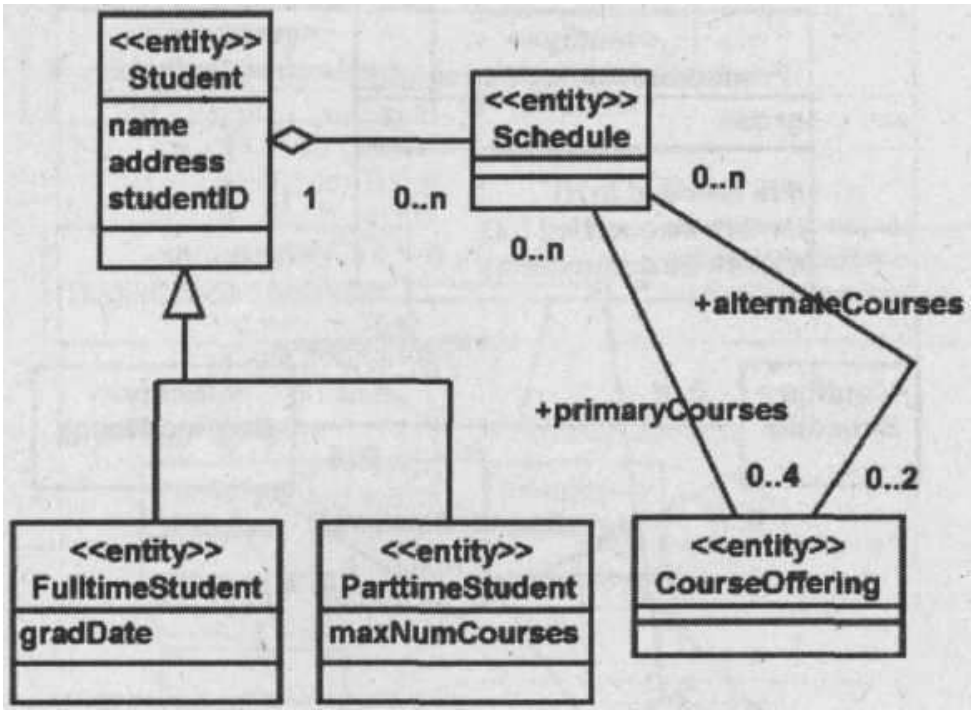


Рисунок 19–Классы с атрибутами

На данной диаграмме показаны классы ассоциаций, описывающие связи между классами ***Schedule*** и ***CourseOffering***, и добавлен суперкласс ***ScheduleOfferinglnfo***. Данные и операции, содержащиеся в этом классе (***status*** - курс включен в график или отменен), относятся как к основным, так и к альтернативным курсам, в то время как оценка (***grade***) и окончательное включение курса в график могут иметь место только для основных курсов.



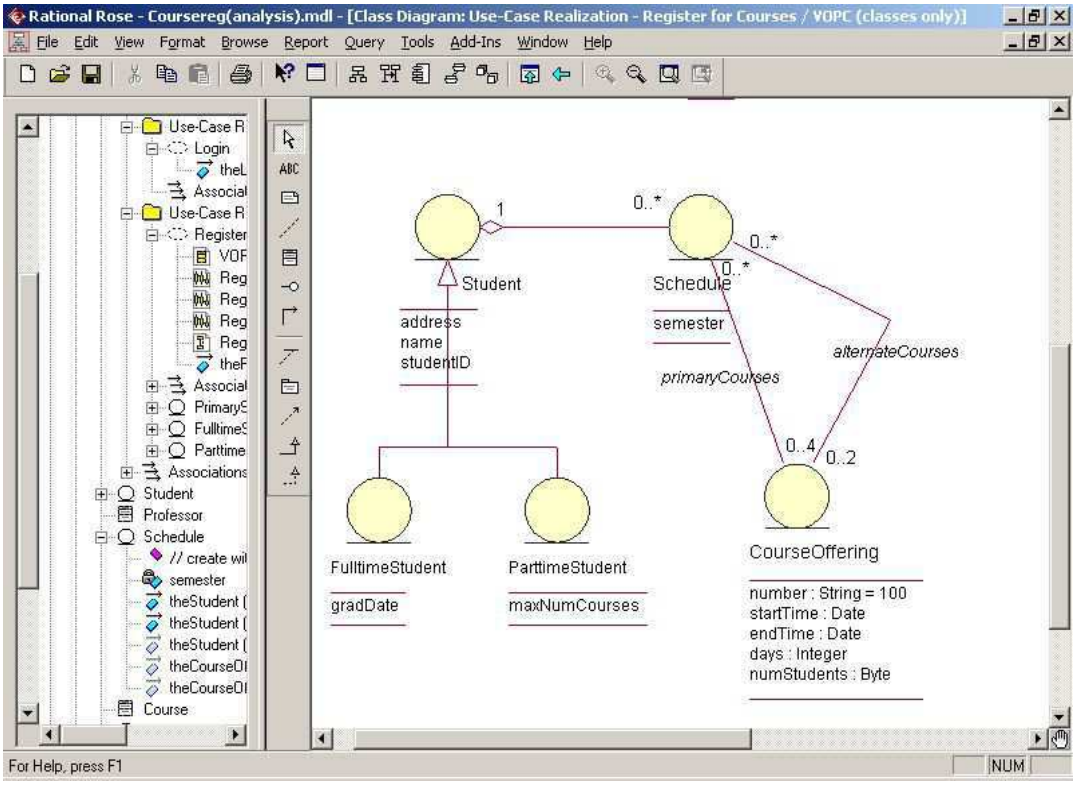


Рисунок 20– Диаграмма ***Entity Classes*** (классы-сущности)

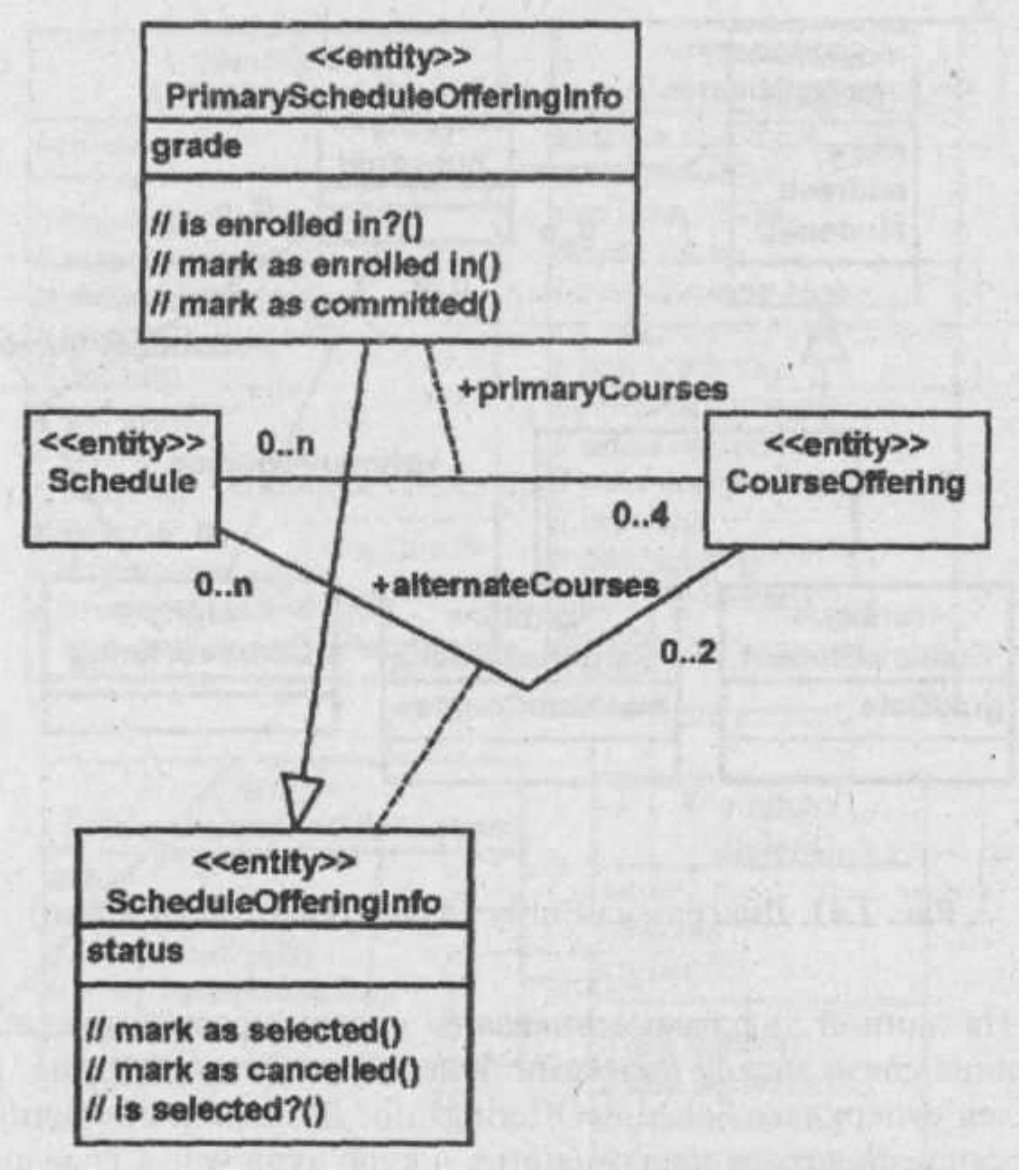


Рисунок 21– Диаграмма ***CourseOfferinglnfo***

Создание ассоциаций

Ассоциации создают непосредственно на диаграмме классов. Панель инструментов диаграммы классов содержит кнопки для создания как одно-, так и двунаправленных ассоциаций.

Для создания на диаграмме классов ассоциации сделайте следующее:

1. Нажмите на панели инструментов кнопку ***Association***.

2. Проведите мышью линию ассоциации от одного класса к другому.

С целью задать возможности навигации по ассоциации

(направления ассоциации) необходимо выполнить следующие действия:

1. Щелкните правой кнопкой мыши по связи с того конца, на котором хотите показать стрелку.

2. Выберите пункт ***Navigable*** в открывшемся меню.

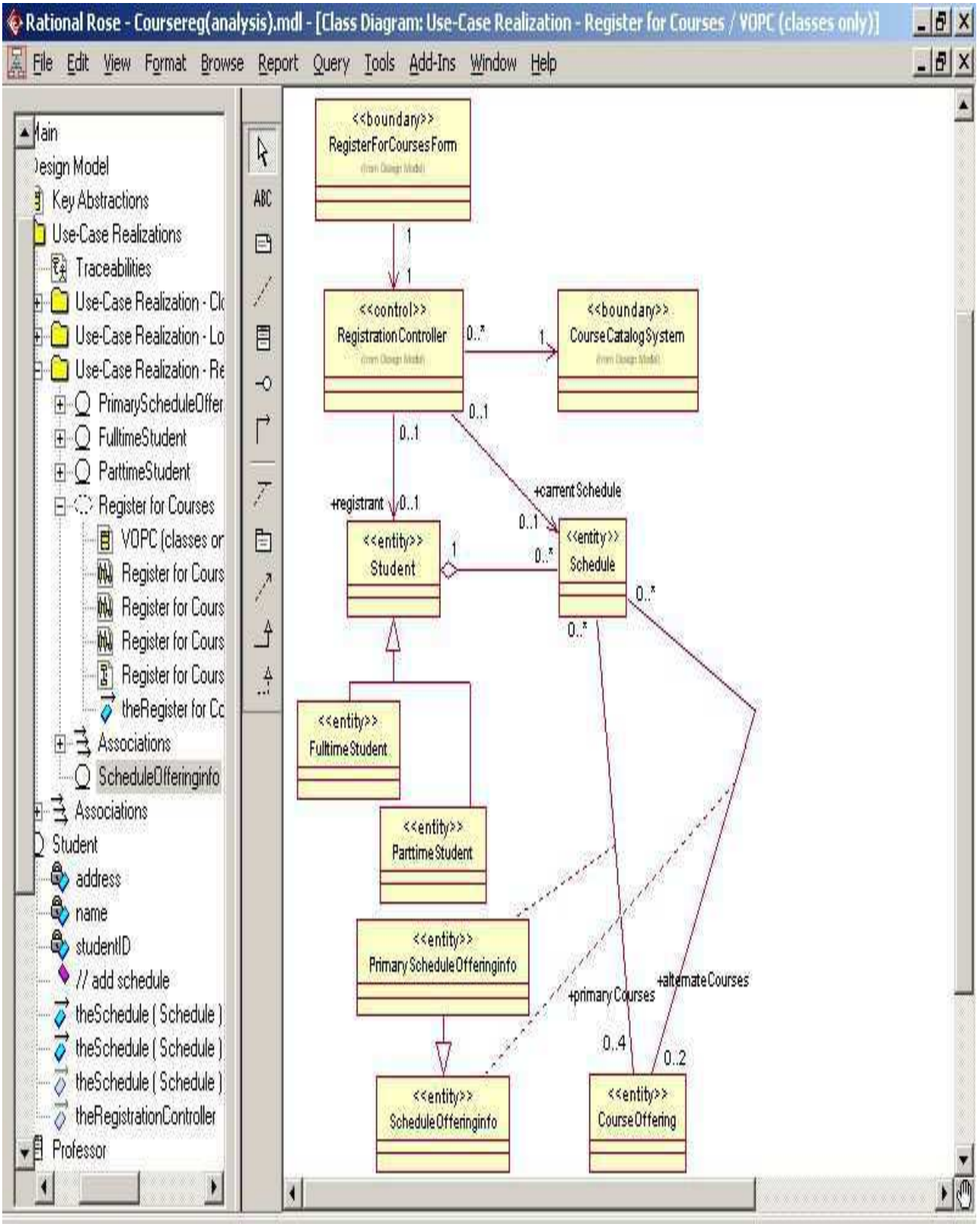


Рисунок 22– Полная диаграмма классов ***VOPC*** (без атрибутов и операций)

Для того чтобы создать рефлексивную ассоциацию:

1. На панели инструментов диаграммы нажмите кнопку ***Association***.

2. Проведите линию ассоциации от класса до какого-нибудь места вне класса.

3. Отпустите кнопку мыши.

4. Проведите линию ассоциации назад к классу.

Создание агрегаций

1. Нажмите кнопку ***Aggregation*** панели инструментов.

2. Проведите линию агрегации от класса-части к целому. Для того чтобы поместить на диаграмму классов рефлексивную агрегацию:

1. На панели инструментов диаграммы нажмите кнопку ***Aggregation***.

2. Проведите линию агрегации от класса до какого-нибудь места вне класса.

3. Отпустите кнопку мыши.

4. Проведите линию агрегации назад к классу.

Создание обобщений

При создании обобщения может потребоваться перенести некоторые атрибуты или операции из одного класса в другой. Если, например, понадобится перенести их из подкласса в суперкласс ***Employee***, в браузере для этого достаточно просто перетащить атрибуты или операции из одного класса в другой. Не забудьте удалить другую копию атрибута из второго подкласса, если он имеется.

Чтобы поместить обобщение на диаграмму классов:

1. Нажмите кнопку ***Generalization*** панели инструментов.

2. Проведите линию обобщения от подкласса к суперклассу.

Спецификации связей

Спецификации связей касаются имен ассоциаций, ролевых имен, множественности и классов ассоциаций. Для того чтобы задать множественность связи:

1. Щелкните правой кнопкой мыши на одном конце связи.
2. Выберите пункт ***Multiplicity*** в открывшемся меню.
3. Укажите нужную множественность.
4. Повторите то же самое для другого конца связи.

Для того чтобы задать имя связи:

1. Выделите нужную связь.
2. Введите ее имя.

Для того чтобы задать связи ролевое имя:

1. Щелкните правой кнопкой мыши на ассоциации с нужного конца.
2. Выберите пункт ***role Name*** в открывшемся меню.
3. Введите ролевое имя.

Для того чтобы задать элемент связи (класс ассоциаций):

1. Откройте окно спецификации требуемой связи.
2. Перейдите на вкладку ***Detail***.
3. Задайте элемент связи в поле ***Link Element***.

Рассмотрим пример ИС «Деканат».

Выделим сущности (классы-сущности). Определим связи, типы связей, их имена (рис. 23).

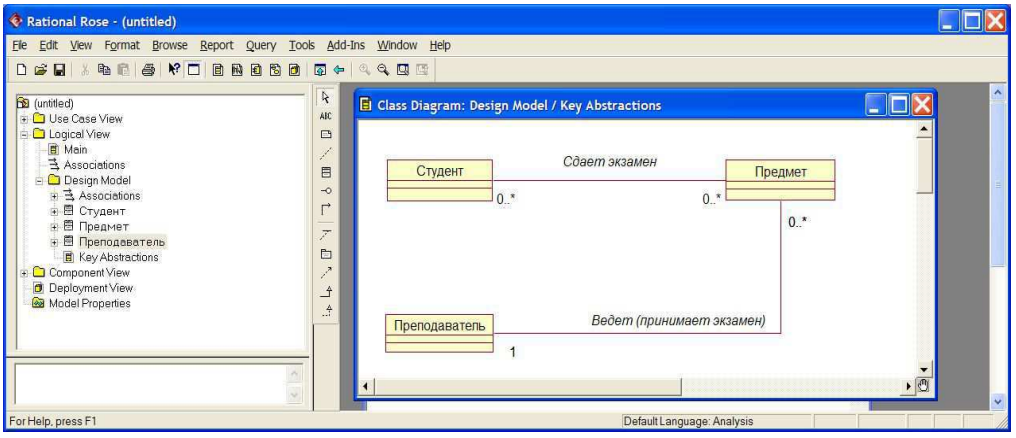


Рисунок 23–Инфологическая модель
(диаграмма классов ***Key Abstractions***)

Определяем атрибуты сущностей и связей. Поскольку связь СДАЕТ ЭКЗАМЕН имеет два атрибута необходимо ввести зависимую сущность с таким же именем (рис.24).

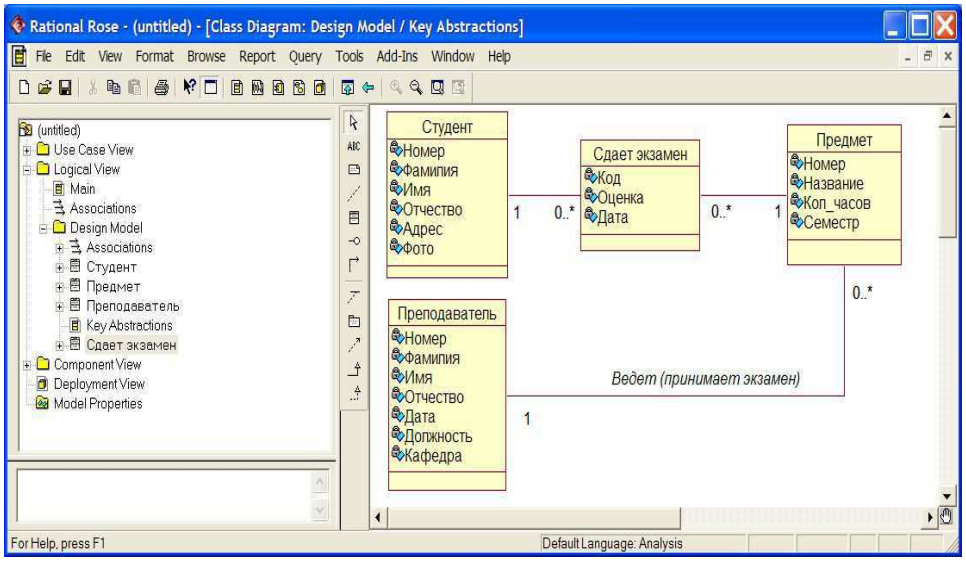


Рисунок 24– Инфологическая модель (диаграмма классов ***Key Abstractions***) –
второй шаг

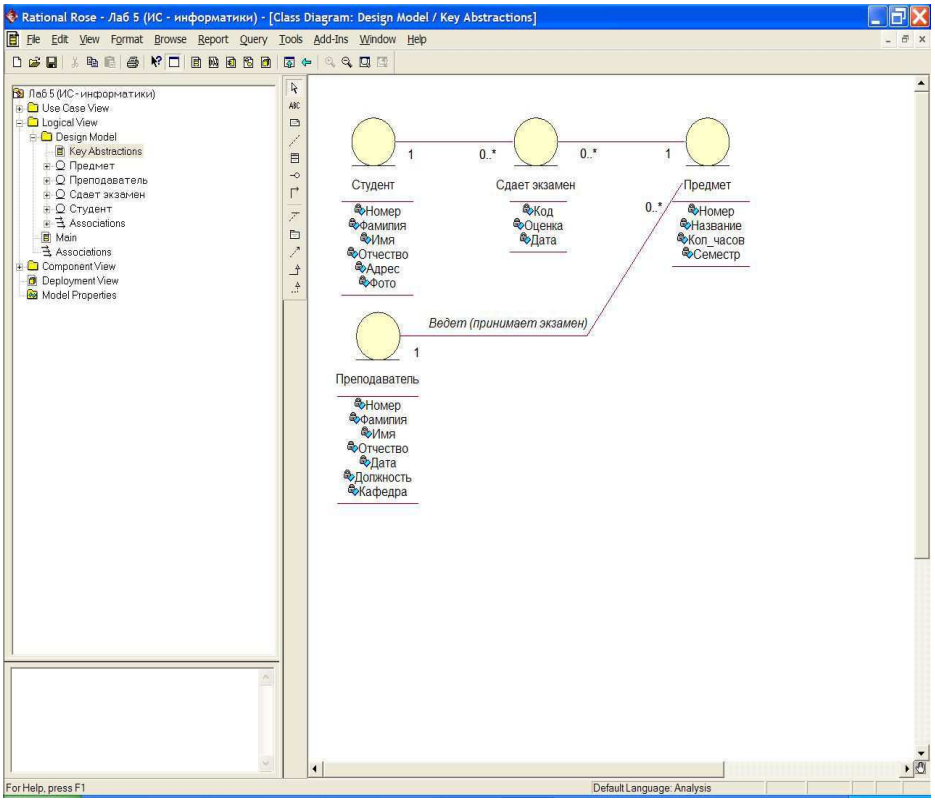


Рисунок 25– Инфологическая модель (диаграмма классов ***Key Abstractions***) –
другое графическое представление (отображение стереотипа)

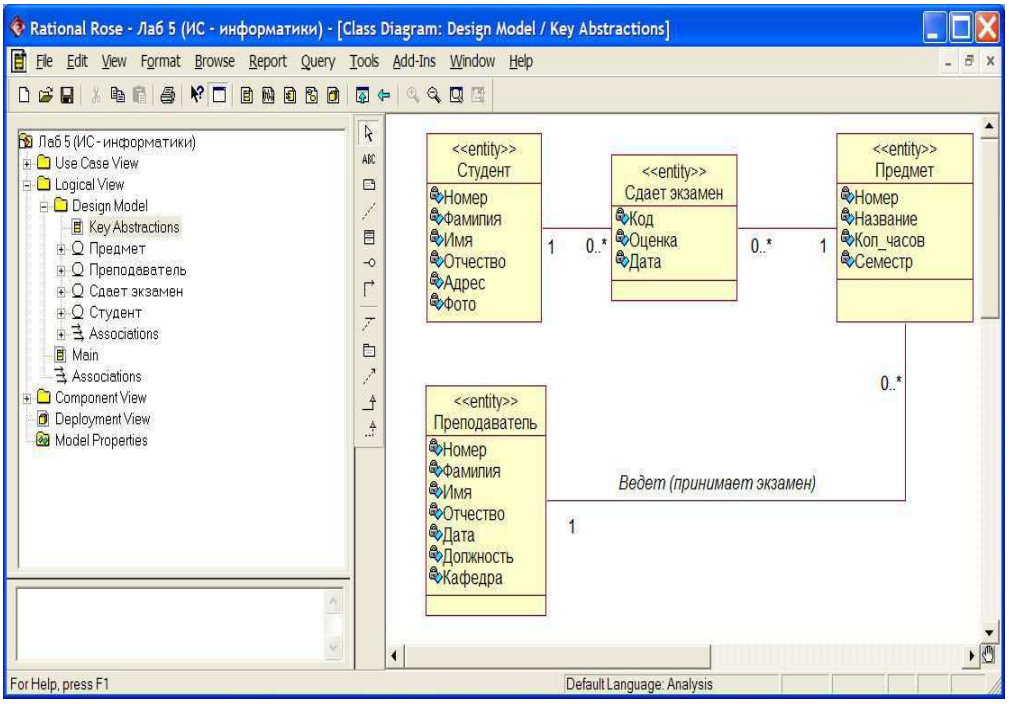


Рисунок 26– Инфологическая модель (диаграмма классов ***Key Abstractions***) –
другое графическое представление (отображение стереотипа)

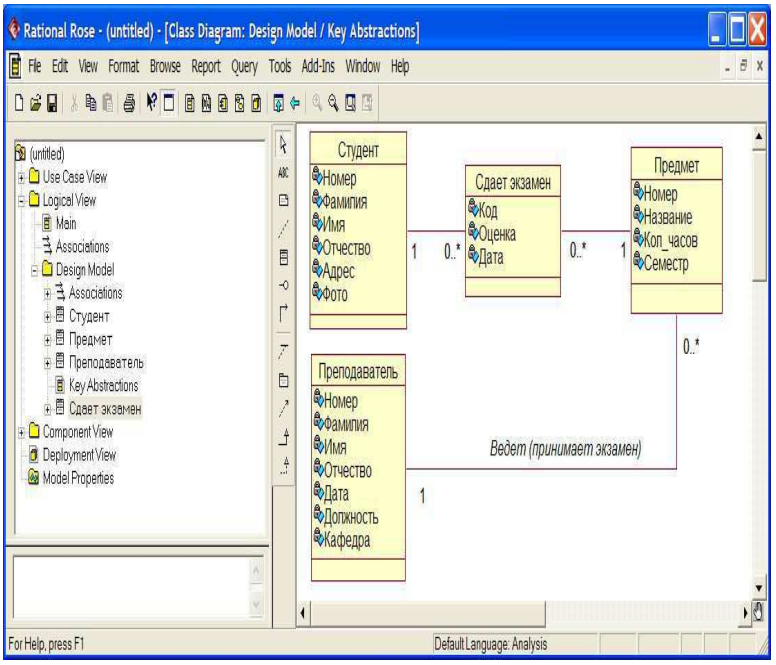


Рисунок 27– Структура логического представления браузера

4. Порядок выполнение работы

1. Изучить раздел 1 и 2.
2. Изучить раздел 3. Выполнить упражнения 1 – 4.

3. Разработать инфологическую модель системы в виде диаграммы классов в соответствии с вариантом задания.
4. Создать разработанную в п. 3 модель в среде Rational Rose.
5.

5. Содержание отчета

В качестве отчета о выполненной работе предъявите преподавателю:

1. на экране в среде Rational Rose: упражнения 1 – 2 (раздел 3) и инфологическую модель в виде диаграммы классов по индивидуальному заданию;
2. отчет в печатном виде содержащий:
   - задание,
   - диаграмму вариантов использования,
   - диаграмму классов.

Контрольные вопросы

1. Каково назначение диаграмм классов?
2. Для чего используется диаграмма классов на стадии анализа?
3. Для чего используется диаграмма классов на стадии проектирования?
4. Назовите основные компоненты диаграмм классов.
5. Назовите основные типы статических связей между классами.
6. Что представляет собой ассоциация?
7. В чем смысл множественности ассоциаций?;
8. В чем отличие атрибутов от ассоциаций?
9. Что такое признак видимости?
10. Что представляет собой операция класса?
11. В чем смысл обобщения?
12. Каково назначение ограничений на диаграммах классов?

Библиографический список

1. Вендров, А. М. Проектирование программного обеспечения экономических информационных систем [Текст]: учебник для студ. вуз. / А. М. Вендров. - М.: Финансы и статистика, 2000. - 352 с.
2. Леоненков, А.В. Самоучитель UML [Текст] / А. Леоненков. - СПб. : БХВ-Петербург, 2001. - 304 с. : ил. - ISBN 5-94157-008-2.
3. Смирнова, Г. Н. Проектирование экономических информационных систем [Текст] : учебник / А. А. Сорокин, Ю. Ф. Тельнов. - М. : Финансы и статистика, 2003. - 512 с. - ISBN 5-279-02295-0.
4. Торрес, Р. Дж. Практическое руководство по проектированию и разработке пользовательского интерфейса [Текст] / Р. Дж. Торрес. - М. : Вильямс, 2002. - 400 с. - ISBN 5-8459-0367-X.
5. Меняев, М. Ф. Управление проектами MS Project [Текст] : учебное пособие / М. Ф. Меняев. - М. : Омега-Л, 2005. - 276 с. с. : ил. - ISBN 5-98119-367-0.