

Лабораторная работа № 1

ПОСТРОЕНИЕ МОДЕЛЕЙ ПРЕДМЕТНОЙ ОБЛАСТИ С ИСПОЛЬЗОВАНИЕМ CASE-СРЕДСТВ ФИРМЫ PLATINUM TECHNOLOGY – BPWIN

Цель работы:

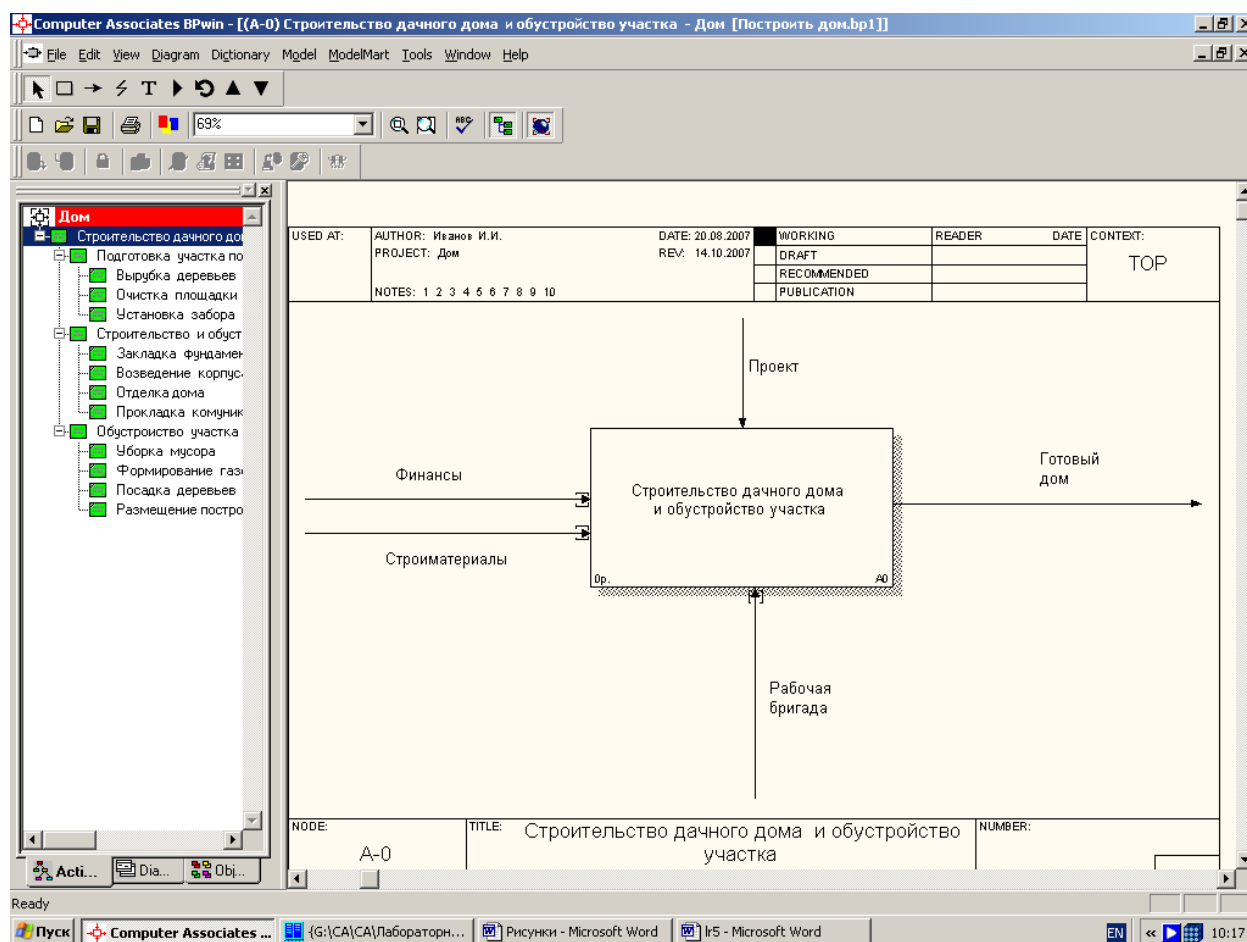
Создание в среде BPwin функциональной модели системы в нотации IDEF0.

Содержание работы:

Создание в среде BPwin новой модели в нотации IDEF0. Разработка контекстной диаграммы модели. Развитие модели. Декомпозиция контекстной диаграммы. Разработка функциональной модели системы с глубиной декомпозиции 3 уровня.

Методика выполнения работы:

1. Создадим новую модель.
2. Разработаем диаграмму верхнего уровня модели (контекстную).

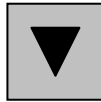


3. Определим функции, на которые может быть разложена функция, обозначенная на контекстной странице модели. Это:

- подготовка участка под строительство;
- строительство и обустройство дома;
- обустройство участка.

4. Создадим диаграмму декомпозиции первого уровня. Для этого:

- выделим функциональный блок на контекстной странице;
- на панели инструментов щелкнем по кнопке с изображением черного треугольника, направленного вершиной вниз (декомпозиция)



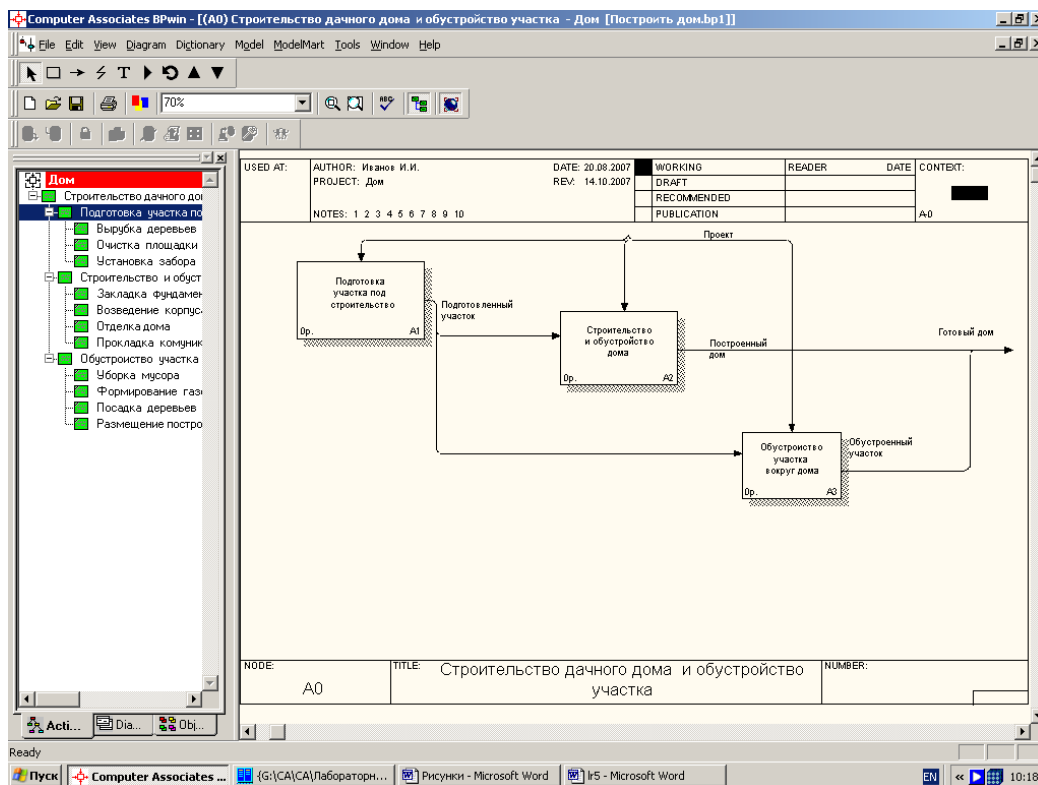
- в диалоговом окне укажем нотацию создаваемой диаграммы (IDEF0) и число функциональных блоков, которые она должна содержать (3 - по числу выделенных функций).

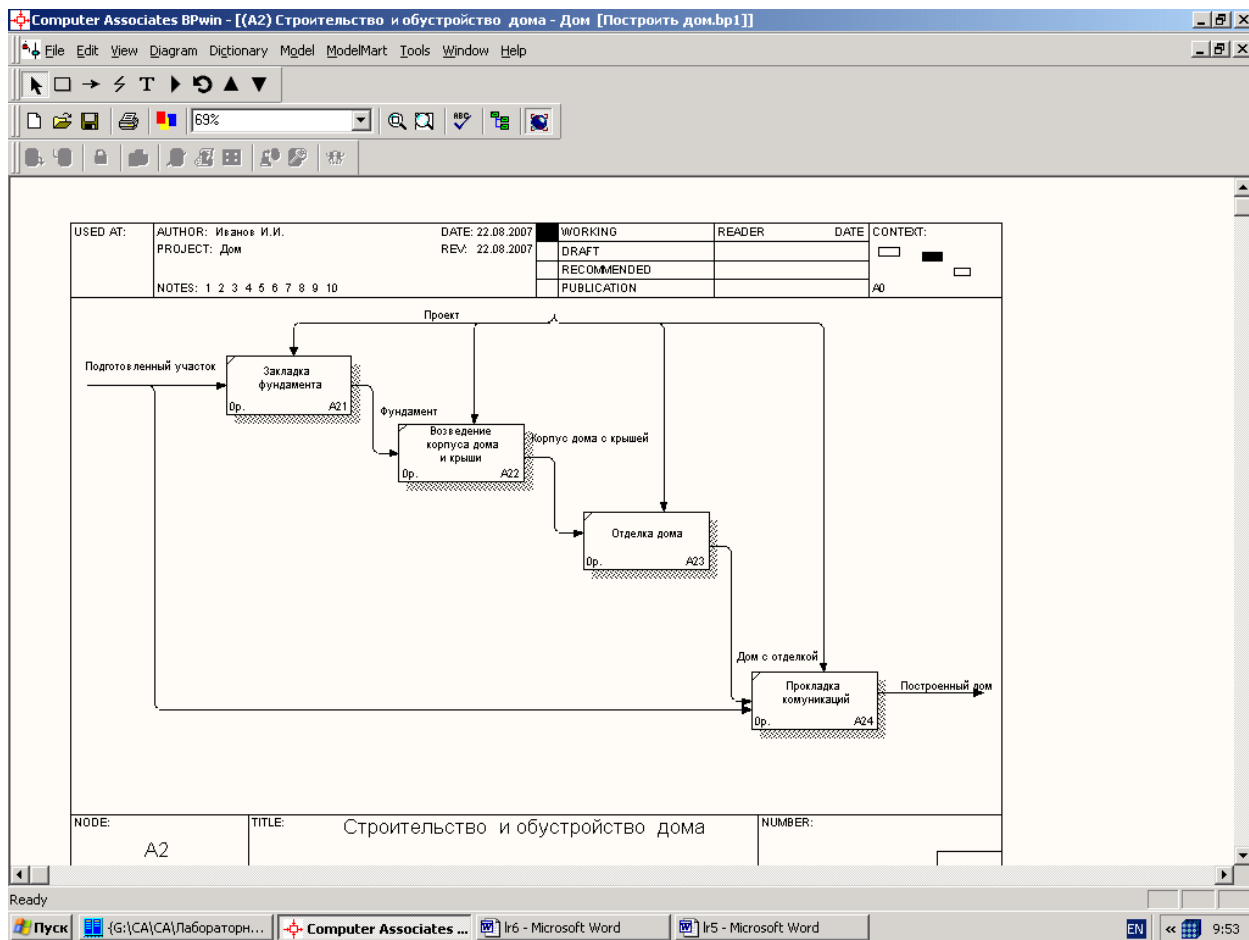
5. На диаграмме декомпозиции впишем названия выделенных функций в функциональные блоки.

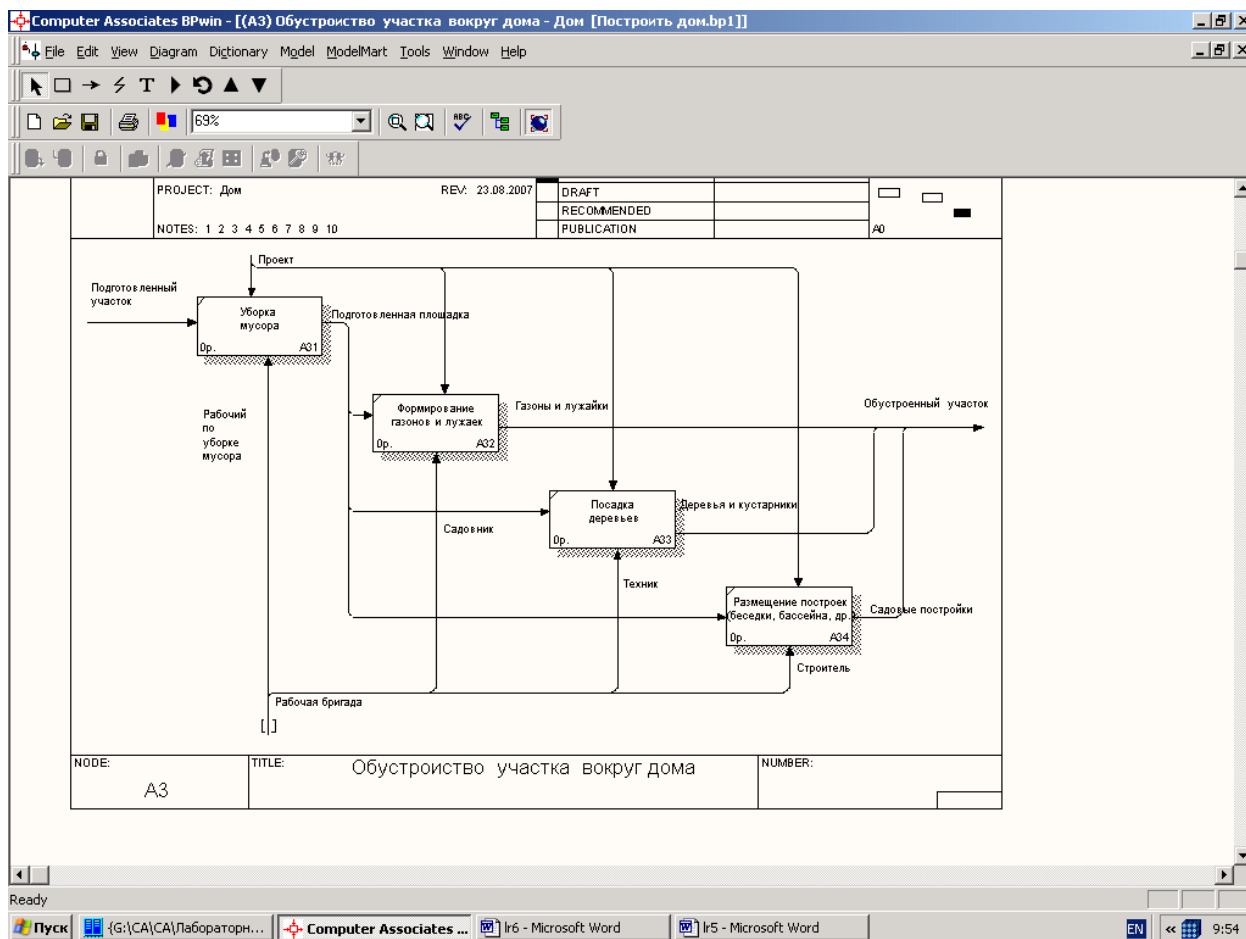
6. Соединим с функциональными блоками интерфейсные дуги, которые мигрировали на созданную диаграмму декомпозиции с контекстной диаграммы.

7. Создадим внутренние дуги для связи функциональных блоков между собой.

8. Аналогично создадим диаграммы декомпозиции для функциональных блоков A1, A2, A3.







Достигнутый результат.

В результате работы средствами редактора BPwin создана трехуровневая функциональная модель системы в нотации IDEF0.

Контрольное задание

Создайте средствами редактора BPwin трехуровневую функциональную модель в нотации IDEF0 системы по Вашему выбору. Для моделируемой системы в среде BPwin должна быть создана трехуровневая функциональная модель, содержащая кроме контекстной диаграммы, диаграммы двух уровней декомпозиции.

Задание:

1. Создайте новую модель.
2. Разработайте контекстную страницу модели.
3. Обдумайте, на какие функции может быть разложена главная функция системы, обозначенная Вами в функциональном блоке на контекстной странице модели. Помните, что число этих функций должно быть от 3 до 6.
4. Создайте диаграмму декомпозиции первого уровня. При создании диаграммы выберите в диалоговом окне нотацию диаграммы (IDEF0) и укажите, сколько функциональных блоков вы планируете разместить на диаграмме.
5. На диаграмме декомпозиции впишите названия выделенных функций в функциональные блоки. Помните о том, что функциональные блоки на диагонали должны быть расположены в порядке убывания их значимости или в соответствии с последовательностью выполнения работ.

6. Соедините интефейсные дуги, которые мигрировали с диаграммы верхнего уровня на созданную диаграмму декомпозиции в виде стрелок, с функциональными блоками в соответствии с их назначением.
7. Если в этом есть необходимость, сделайте разветвления дуг. Помните о том, что Вы можете оставить единое название для всех веток. В этом случае название располагается до разветвления стрелки. В случае, если ветки обозначают разные объекты, подпишите каждую ветку.
8. Создайте внутренние дуги, связывающие функциональные блоки между собой. Помните, что каждый функциональный блок обязательно должен иметь дуги Управления и Выхода. Дуги Механизма и Входа могут отсутствовать. Именуйте каждую дугу.
9. По описанной выше технологии создайте диаграммы декомпозиции для тех функциональных блоков, прояснить содержание которых требуется по логике модели.

Контрольные вопросы

1. Что такое бизнес-процесс?
2. Каковы основные компоненты функциональной модели?
3. Что представляют собой методологии функционального моделирования?
4. Что такое сценарии?
5. Какие виды сценариев Вы знаете?
6. В чем отличие серверных элементов управления от клиентских?
7. Какие технологии программирования серверных сценариев Вы знаете? В чем их отличие?

Содержание и оформление отчета

Отчет должен содержать: титульный лист, название и цель работы; вариант задания; скриншоты результатов работы; выводы по работе.

Лабораторная работа № 2

Методика построения ДМ предметной области для проектирования ИУС с использованием пакета Design/IDEF

1. Цель работы: изучение методики построения динамической модели предметной области

2. Задачи: получение навыка разработки динамических моделей в нотации IDEF3

3. Теоретическая часть

Основой модели IDEF3 служит так называемый сценарий процесса, который выделяет последовательность действий и подпроцессов анализируемой системы.

Как и в методе IDEF0, основной единицей модели IDEF3 является диаграмма. Другой важный компонент модели — действие, или в терминах IDEF3 "единица работы" (Unit of Work). Диаграммы IDEF3 отображают действие в виде прямоугольника. Действия именуются с использованием глаголов или отглагольных существительных, каждому из действий присваивается уникальный идентификационный номер. Этот номер не используется вновь даже в том случае, если в процессе построения модели действие удаляется. В диаграммах IDEF3 номер действия обычно предваряется номером его родителя.



Рисунок 1. Изображение и нумерация действия в диаграмме IDEF3

Существенные взаимоотношения между действиями изображаются с помощью связей. Все связи в IDEF3 являются однонаправленными, и хотя стрелка может начинаться или заканчиваться на любой стороне блока, обозначающего действие, диаграммы IDEF3 обычно организуются слева направо таким образом, что стрелки начинаются на правой и заканчиваются на левой стороне блоков.

Связь типа **"временное предшествование"** \longrightarrow показывает, что исходное действие должно полностью завершиться, прежде чем начнется выполнение конечного действия.

Связь типа **"объектный поток"** \longrightarrow используется в том случае, когда некоторый объект, являющийся результатом выполнения исходного действия, необходим для выполнения конечного действия. Наименования потоковых связей должны четко идентифицировать объект, который передается с их помощью. Исходное действие должно завершиться, прежде чем конечное действие может начать выполняться.

Связь типа **"нечеткое отношение"** \dashrightarrow используется для выделения отношений между действиями, которые невозможно описать с использованием связей предшествования или объектных связей. Одно из применений нечетких отношений — отображение взаимоотношений между параллельно выполняющимися действиями.

Завершение одного действия может инициировать начало выполнения сразу нескольких других действий или, наоборот, определенное действие может требовать завершения нескольких других действий до начала своего выполнения. Такие ситуации описываются с помощью **перекрестков (junction)**. Перекрестки (или соединения) используются для отображения логики взаимодействия стрелок при слиянии и разветвлении или для отображения множества событий, которые могут или должны быть завершены перед началом следующей работы. Перекрестки разбивают или соединяют внутренние потоки и используются для изображения ветвления процесса:

- разворачивающие соединения используются для разбиения потока. Завершение одного действия вызывает начало выполнения нескольких других;
- сворачивающие соединения объединяют потоки. Завершение одного или нескольких действий вызывает начало выполнения другого действия.

Соединения «И» инициируют выполнение конечных действий. Все действия, присоединенные к сворачивающему соединению "и", должны завершиться, прежде чем начнется выполнение следующего действия. Например, после обнаружения пожара инициируются включение пожарной сигнализации, вызов пожарной охраны, и начинается тушение пожара. Запись в журнал производится только тогда, когда все три перечисленных действия завершены.

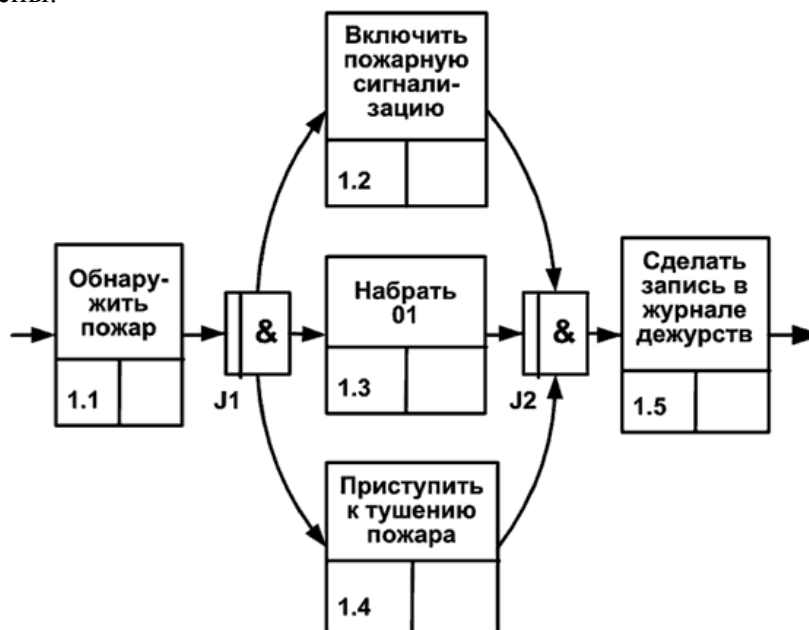


Рисунок 2. Соединения "и"

Соединение *"исключающее или"* означает, что вне зависимости от количества действий, связанных со сворачивающим или разворачивающим соединением, инициировано будет только одно из них, и поэтому только оно будет завершено перед тем, как любое действие, следующее за сворачивающим соединением, сможет начаться. Если правила активации соединения известны, они обязательно должны быть документированы либо в его описании, либо пометкой стрелок, исходящих из разворачивающего соединения. На рисунке 3 соединение "исключающее или" используется для отображения того факта, что студент не может одновременно быть направлен на лекции по двум разным курсам.

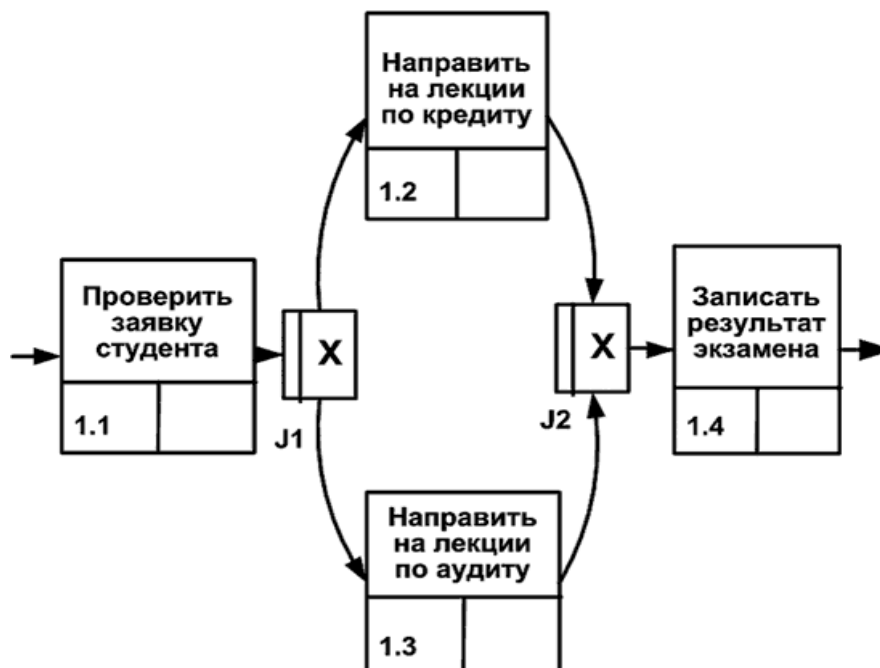


Рисунок 3. Соединение "исключающее "или""

Соединение **"ИЛИ"** предназначено для описания ситуаций, которые не могут быть описаны двумя предыдущими типами соединений. Аналогично связи нечеткого отношения соединение "или" в основном определяется и описывается непосредственно аналитиком. На рисунке 4 соединение J2 может активизировать проверку данных чека и/или проверку суммы наличных. Проверка чека инициируется, если покупатель желает расплатиться чеком, проверка суммы наличных — при оплате наличными. И то, и другое действие инициируются при частичной оплате, как чеком, так и наличными.

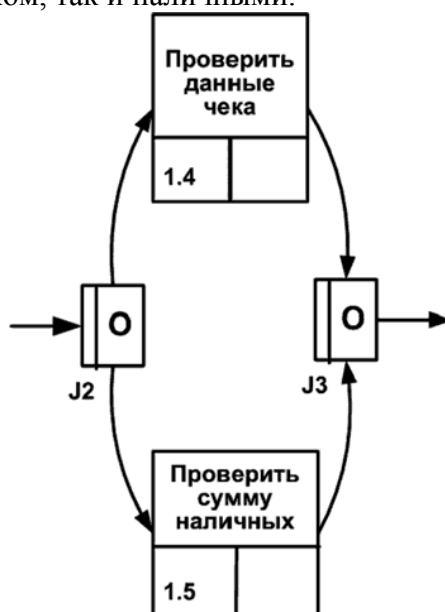


Рисунок 4. Соединения "или"

В рассмотренных примерах все действия выполнялись асинхронно, т.е. они не инициировались одновременно. Однако существуют случаи, когда время начала или окончания параллельно выполняемых действий должно быть одинаковым, т.е. действия должны выполняться синхронно. Для моделирования такого поведения системы используются различные виды синхронных соединений, которые обозначаются двумя двойными вертикальными линиями внутри прямоугольника.

Существует 5 типов перекрестков:

Наименование	Сворачивающий перекресток	Разворачивающий перекресток
--------------	---------------------------	-----------------------------

Асинхронный «И»	Все предшествующие процессы должны быть завершены	Все следующие процессы должны быть запущены
Синхронный «И»	Все предшествующие процессы завершены одновременно	Все следующие процессы запускаются одновременно
Асинхронный «ИЛИ»	Один или несколько предшествующих процессов должны быть завершены	Один или несколько следующих процессов должны быть запущены
Синхронный «ИЛИ»	Один или несколько предшествующих процессов завершены одновременно	Один или несколько следующих процессов запускаются одновременно
Исключающее «ИЛИ»	Только один предшествующий процесс завершен	Только один следующий процесс запускается

Все соединения на диаграммах должны быть парными, из чего следует, что любое разворачивающее соединение имеет парное себе сворачивающее. Однако типы соединений не обязательно должны совпадать.

Соединения могут комбинироваться для создания более сложных ветвлений. Комбинации соединений следует использовать с осторожностью, поскольку перегруженные ветвлением диаграммы могут оказаться сложными для восприятия.

Действия в IDEF3 могут быть декомпозированы или разложены на составляющие для более детального анализа. Метод IDEF3 позволяет декомпозировать действие несколько раз, что обеспечивает документирование альтернативных потоков процесса в одной модели.

Еще одним элементом диаграммы IDEF3 является *указатель*. Указатель выражает некую идею, концепцию или данные, которые нельзя связать со стрелкой, перекрестком или работой. Указатели должны быть связаны с единицами работ или перекрестками пунктирными линиями.

Тип указателя	Назначение
Объект (OBJECT)	Описывает участие в работе важного объекта, не рассматриваемого в качестве основного объекта в модели в целом.
Ссылка (GOTO)	Инструмент циклического перехода (в повторяющейся последовательности работ), возможно на текущей диаграмме, но не обязательно. Если все работы цикла присутствуют на текущей диаграмме, цикл может также изображаться стрелкой, возвращающейся на стартовую работу. Может ссылаться на перекресток.
Единица действия (UOB-Unit of behavior)	Применяется, когда необходимо подчеркнуть множественное использование работы, но без заикливания. Обычно этот тип ссылки не используется для моделирования автоматически запускающихся работ. Например, если действие «Подсчет наличных» запускается несколько раз, в первый раз оно создается как действие, а последующие его появления на диаграмме оформляются указателями UOB.
Заметка (NOTE)	Используется для документирования важной информации, относящейся к графическим объектам на диаграмме.
Уточнение (ELAB - Elaboration)	Используется для усовершенствования графиков или их более детального описания. Обычно употребляется для детального описания разветвления и слияния стрелок на перекрестках.

Моделирование потоков данных

Диаграммы потоков данных (Data Flow Diagrams — DFD) предназначены для демонстрации того, как каждый процесс преобразует свои входные данные в выходные, а также выявить отношения между этими процессами.

Основными компонентами диаграмм потоков данных являются:

- внешние сущности;
- функциональные блоки;
- потоки данных;
- хранилища данных.

Внешняя сущность представляет собой материальный объект или физическое лицо, являющиеся источником или приемником информации, например, заказчики, персонал, поставщики, клиенты, склад. Определение некоторого объекта или системы в качестве внешней сущности указывает на то, что она находится за пределами границ анализируемой системы.

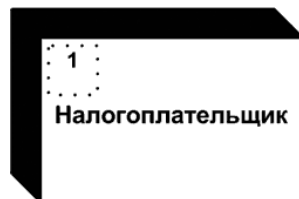


Рисунок 5. Графическое представление внешней сущности.

Функциональный блок моделирует некоторую функцию или процесс, который преобразует входные потоки данных в выходные в соответствии с определенным алгоритмом. Физически процесс может быть реализован различными способами: это может быть подразделение организации, выполняющее обработку входных документов, или программа, аппаратно реализованное логическое устройство и т.д.



Рисунок 6. Графическое изображение процесса в виде функционального блока.

Функциональные блоки нотации DFD имеют входы и выходы, но не имеют управления и механизма исполнения. В некоторых интерпретациях нотации DFD механизмы исполнения моделируются как ресурсы и изображаются в нижней части функционального блока.

Поток данных определяет информацию, передаваемую через некоторое соединение от источника к приемнику. Реальный поток данных может быть информацией, передаваемой по кабелю между двумя устройствами, пересылаемыми по почте письмами, магнитными лентами или дискетами, переносимыми с одного компьютера на другой и т.д.

Поток данных на диаграмме изображается линией, оканчивающейся стрелкой, которая показывает направление потока. Каждый поток данных имеет имя, отражающее его содержание. В DFD используются также двунаправленные стрелки, которые нужны для отображения взаимодействия между блоками по типу «запрос – ответ».



Рисунок 7. Поток данных

Хранилище данных — это абстрактное устройство для хранения информации, которую можно в любой момент поместить в хранилище и через некоторое время извлечь, причем способы помещения и извлечения могут быть любыми. В отличие от потоков данных, описывающих объекты в движении, хранилища данных изображают объекты в

покое. В материальных системах хранилища данных изображаются там, где объекты ожидают обработки, например, в очереди. В системах обработки информации хранилища данных являются механизмом, который позволяет сохранить данные для последующих процессов. Хранилище данных может быть реализовано физически в виде микрофиши, ящика в картотеке, таблицы в оперативной памяти, файла на магнитном носителе и т.д.

Хранилище данных в общем случае является прообразом будущей базы данных, и описание хранящихся в нем данных должно соответствовать модели данных.



Рисунок 8. Графическое изображение хранилища данных

Построение DFD-диаграмм.

Первым шагом при построении иерархии DFD является построение контекстных диаграмм. Обычно при проектировании относительно простых систем строится единственная контекстная диаграмма со звездообразной топологией, в центре которой находится так называемый главный процесс, соединенный с приемниками и источниками информации, посредством которых с системой взаимодействуют пользователи и другие внешние системы. Перед построением контекстной DFD необходимо проанализировать внешние события (внешние сущности), оказывающие влияние на функционирование системы. Количество потоков на контекстной диаграмме должно быть по возможности небольшим, поскольку каждый из них может быть в дальнейшем разбит на несколько потоков на следующих уровнях диаграммы.

Для проверки контекстной диаграммы можно составить список событий. Список событий должен состоять из описаний действий внешних сущностей (событий) и соответствующих реакций системы на события. Каждое событие должно соответствовать одному или более потокам данных: входные потоки интерпретируются как воздействия, а выходные потоки — как реакции системы на входные потоки.

Для сложных систем (десять и более сущностей) строится иерархия контекстных диаграмм. При этом контекстная диаграмма верхнего уровня содержит не единственный главный процесс, а набор подсистем, соединенных потоками данных. Контекстные диаграммы следующего уровня детализируют контекст и структуру подсистем.

4. Задание

1. На основе разработанной на предыдущей лабораторной работе функциональной модели построить диаграмму IDEF3.

2. На основе диаграммы IDEF3 создать сценарий, более подробно описывающий какой-либо подпроцесс обработки документов.

3. Провести сравнение двух диаграмм – сценария IDEF3 и диаграммы IDEF3, выбрав в меню команду **Tools\Visual Diagram Compare...**

4. Построить диаграмму DFD.

5. Построить организационную диаграмму, пользуясь положением об отделах предприятия «Уралтранснефтепродукт».

5.1. Заполнить словарь данными, необходимыми для построения организационной диаграммы – выделить ролевые группы и роли в организационной структуре. Для этого выбрать в меню команду **Dictionary/Role Group...** для описания ролевых групп и команду **Dictionary/Role...** для описания ролей сотрудников.

5.2. Для построения организационной диаграммы выбрать в меню команду **Diagram/Add Organization Chart...**

5.3. Полученная в результате простейшая двухуровневая организационная структура может быть развернута как по вертикали, так и по горизонтали. Для этого нужно выделить блок на организационной диаграмме и нажать правую кнопку мыши. При этом

команды *Edit subordinate list...* и *Add subordinates...* добавляют следующий уровень иерархии, а команды *Add sibling on left* и *Add sibling on right* добавляют блоки организационной диаграммы на том же уровне иерархии.

6. Сформировать отчет в виде HTML-документа. Для этого выполнить команду Tools/report Builder.

5. Контрольные вопросы

1. Понятие динамической модели.
2. Теория сетей Петри.
3. Методология динамического моделирования IDEF/CPN. Правила трансформирования IDEF0 модели в IDEF/CPN модель. Методология IDEF3.

Содержание и оформление отчета

Отчет должен содержать: титульный лист, название и цель работы; вариант задания; скриншоты результатов работы; выводы по работе.

Лабораторная работа № 3

ПОСТРОЕНИЕ МОДЕЛЕЙ ПРЕДМЕТНОЙ ОБЛАСТИ С ИСПОЛЬЗОВАНИЕМ CASE-СРЕДСТВ ФИРМЫ PLATINUM TECHNOLOGY – ERWIN

Цель работы:

- овладение навыками работы в Erwin;
- построение логической модели заданной предметной области.

Задание:

Построить логическую информационную модель поставки товаров в соответствии с договорами средствами Erwin.

Методика выполнения работы

1. Знакомство с пользовательским интерфейсом

- Загрузите программу Erwin.
- В появившемся диалоговом окне установите переключатель **Create a New Model**.

На экране появится диалог **Create Model – Select Template**, где необходимо выбрать уровень моделирования.

Erwin имеет два уровня моделирования: логический и физический. На *логическом* уровне данные представляются так, как они выглядят в реальном мире. Объектами логического уровня являются сущности и атрибуты.

На *физическом* уровне модель зависит от конкретной реализации базы данных, выбираемой пользователем. При переходе модели на физический уровень производится трансформация сущностей в таблицы, а атрибутов в поля, поэтому все имена и описания физической модели должны соответствовать принятым для выбранной СУБД соглашениям.

- Установите переключатель **Logical/Physical** для создания модели с логическим и физическим уровнями.

- В полях **DataBase** и **Version** указывается тип и версия сервера, для которого создается модель. Выберите в списке Access, 2000. Нажмите кнопку **OK**.

- На экране появится основное окно программы.

В верхней части окна находится титульная строка, в которой указано название программы, наименование модели, наименование подмножества (Subject Area) и хранимого

отображения (Stored Display). Основную часть пространства программы занимает рабочая область, в которой создается ER-диаграмма.

Для переключения между логическим и физическим уровнями на панели инструментов имеется список (рис 1.1).

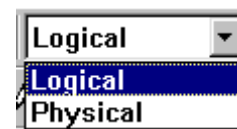


Рис. 1.1.

Помимо этого списка, на панели инструментов имеются кнопки (см. табл. 1.1).

Таблица 1.1.

Кнопки, расположенные на панели инструментов программы Erwin

Кнопка	Назначение
	Создание, открытие, сохранение и печать модели
	Вызов диалога Report Browser для генерации отчетов
	Изменение уровня просмотра модели: уровень сущностей, уровень атрибутов, уровень определений
	Изменение масштаба просмотра модели
	Генерация схемы БД, выравнивание схемы с моделью и выбор сервера (доступны только на уровне физической модели)
	Переключение между областями модели Subject Area

Для непосредственной работы с элементами модели в программе имеется палитра инструментов (Erwin Toolbox), представляющая собой «плавающее окошко» (рис. 1.2). При необходимости палитру инструментов можно убирать с экрана и вызывать нажатием комбинации клавиш «CTRL-T».

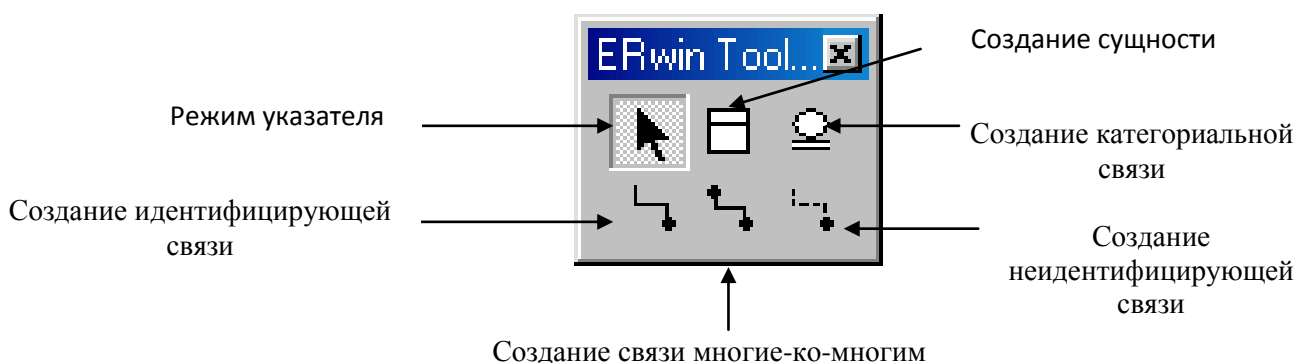


Рис. 1.2. Палитра инструментов на логическом уровне

2. Создание хранимых отображений

В процессе создания модели она пройдет несколько уровней детализации. Поэтому создадим две закладки хранимых изображений на уровне сущностей и на уровне атрибутов.

- Выберите пункт главного меню **FORMAT | Stored Display Settings**. На экране появится окно редактирования хранимых отображений (рис. 1.3).

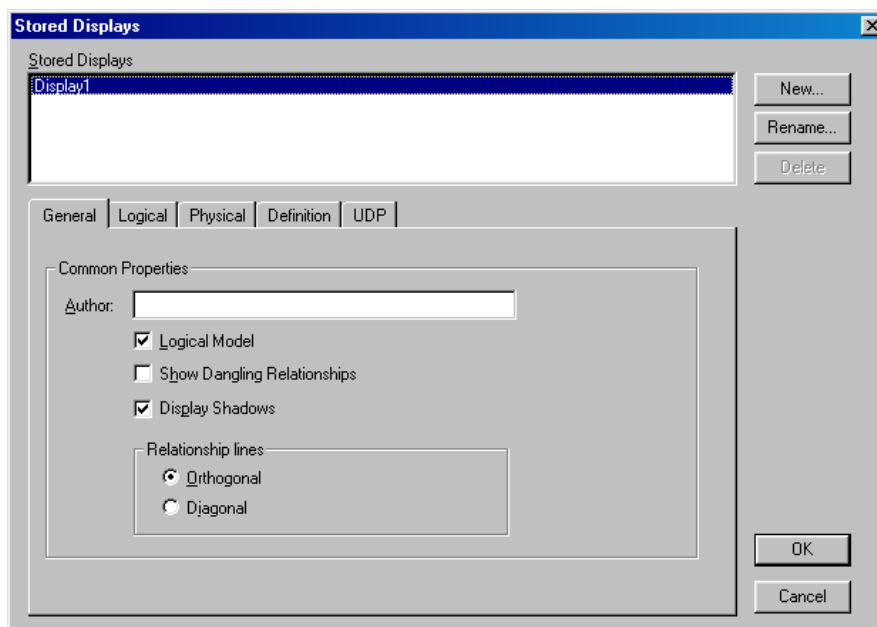


Рис. 1.3. Редактор хранимых отображений

В верхней части окна находится список хранимых отображений модели. В настоящее время он содержит только одно отображение, которое создается по умолчанию – Display1. В нижней части окна имеется несколько страниц с закладками для задания свойств отображения модели.

- На вкладке **General** в поле **Author** введите с клавиатуры свое имя.
- Установите опции **Logical Model** (логическая модель – хранимое отображение будет использоваться только на логическом уровне) и **Display Shadows** (показывать тени – прямоугольники сущности будут изображаться на экране с «тенью»).
- В рамке **Relationships lines** (линии связи) устанавливается способ изображения линий связи между сущностями. В режиме **Orthogonal** (ортогональный) линии связи прокладываются параллельно осям XY, в диагональном режиме (**Diagonal**) линии связи могут проводиться под произвольным углом. По умолчанию задан ортогональный режим изображения связей, оставьте эту установку без изменений.

- Перейдите на закладку **Logical** (логический уровень).
- Установите переключатель **Display Level** (уровень отображения) в положение **Entity** (сущность). Тем самым задается, что на экране будут показаны только сущности, без атрибутов.
- Установите флажок **Verb Phrase** (глагольная фраза), чтобы на диаграмме отображались глагольные фразы, именующие связи между сущностями. Остальные флажки на данной странице оставьте без изменений.
- Переименуйте отображение Display1, нажав на кнопку **Rename**. В появившемся диалоге введите имя отображения **Уровень сущностей**. Нажмите кнопку **OK** и еще раз **OK**. Это название появится в титульной строке, а также на закладке в нижней части экрана.
- Снова выберите пункт меню **FORMAT | Stored Display Settings** и создайте еще одно хранимое отображение под названием **Уровень атрибутов**. Для этого нажмите кнопку **New** и введите это название с клавиатуры.
- Выделите отображение **Уровень атрибутов** в списке и на странице **Logical** установите переключатели, как показано на рис. 1.4.

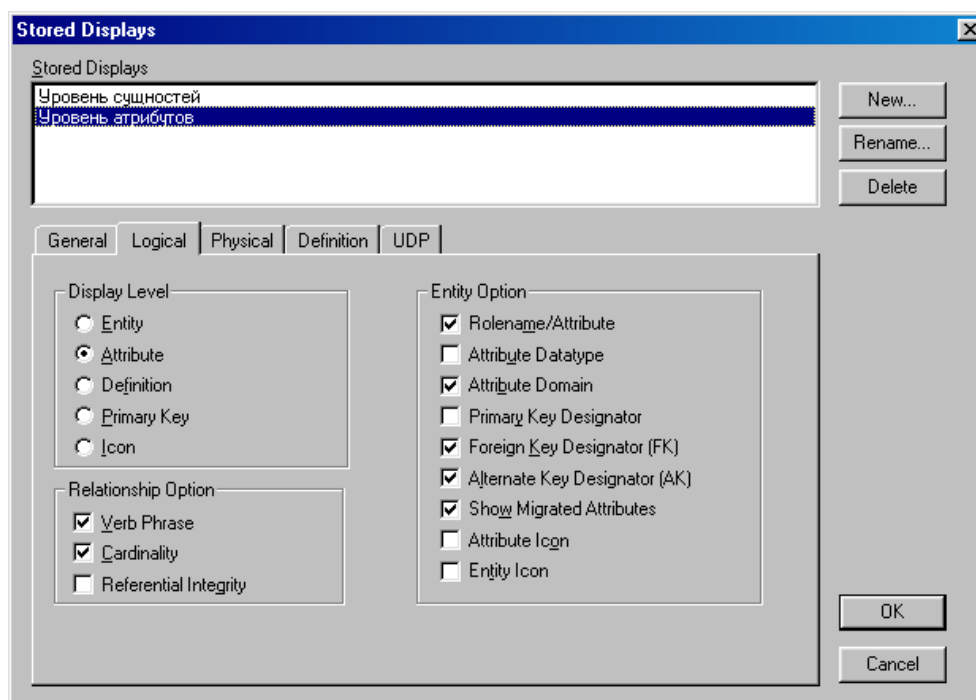


Рис. 1.4. Установка свойств уровня атрибутов

Теперь у диаграммы определены два отображения с разной степенью детализации, и для перехода из режима сущностей в режим атрибутов достаточно щелкнуть по соответствующей закладке в нижней части схемы (рис. 1.5).




Рис. 1.5. Закладки хранимых отображений

- Начиная разработку модели, необходимо выполнить настройку шрифтов. Для этого выберите пункт главного меню **FORMAT | Default Fonts & Colors**. В появившемся окне редактирования перейдите на вкладку **General** и в группе **All Fonts** в поле **Font** из списка выберите шрифт **Arial Cyr** и нажмите **OK**.

3. Внесение в модель сущностей

На данном этапе необходимо внести в модель следующие сущности, выявленные в результате анализа предметной области (поставка товара в соответствии с договорами): покупатель, договор, накладная, товар, склад.

- Выберите на панели инструментов (ERwin Toolbox) кнопку **Сущность** , щелкнув по ней указателем мыши. Затем щелкните мышкой по тому месту на диаграмме, где необходимо расположить новую сущность. На поле диаграммы появится прямоугольник, изображающий новую сущность, с автоматически сгенерированным именем «E/1».

- Введите с клавиатуры имя сущности «**Покупатель**» и нажмите **Enter**.
- Точно таким же образом вставьте в диаграмму еще четыре сущности: договор, накладная, товар, склад.

- Щелкнув правой кнопкой мыши по сущности и выбрав из контекстного меню пункт **Entity Properties**, можно вызвать редактор сущностей **Entities** (рис. 1.6), который позволяет изменять свойства выбранной сущности. Редактор сущностей также можно вызвать через главное меню: **Model | Entities**.

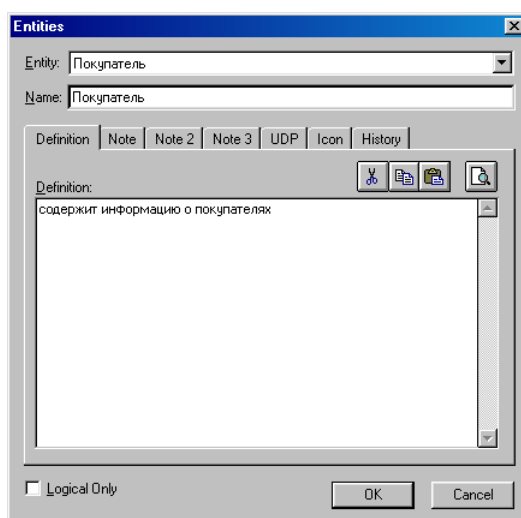


Рис. 1.6. Редактор сущности

В верхней части окна редактора находится список всех сущностей, имеющих на диаграмме. С его помощью можно выбрать сущность, свойства которой необходимо посмотреть или изменить. По умолчанию, выбранной является выделенная на диаграмме сущность, по которой щелкнули мышью. Далее имеется поле Name, в котором высвечивается имя сущности. Имя можно редактировать.

Ниже в окне редактора находится ряд закладок:

Definition (определение) – на этой странице вводится определение сущности.

Note, *Note2*, *Note3* (примечание) – используются для ввода произвольного текста, связанного с сущностью, например, образцы данных и запросы.

UDP – определяемые пользователем свойства.

Icon (иконка) – для наглядности каждой сущности может быть присвоена иконка, которая выводится рядом с ее названием.

- Для каждой сущности введите определение **Definition**.

4. Определение атрибутов сущностей

Определив сущности, необходимо внести в схему и атрибуты этих сущностей. В табл. 1.2 приведен перечень сущностей и их атрибутов с характеристиками для рассматриваемой предметной области.

Таблица 1.2.

Характеристика атрибутов сущностей

Тип сущности	Атрибут	Ключ	Тип данных
Покупатель	КОД_ПОК	PK	Number
	ИНН		<i>Number</i>
	НАИМ_ПОК		String
	АДРЕС_ПОК		String
	ТЕЛ		String
	НОМ_РСЧ		String

	Банк		String
Товар	КОД_ТОВ	PK	Number
	НАИМ_ТОВ		String
	ЕИ		String
	ЦЕНА		Number
	СТАВКА_НДС		Number
Склад	КОД_СК	<i>PK</i>	Number
	НАИМ_СК		String
	АДРЕС_СК		String
	ОТВ_ЛИЦО		String
Договор	НОМ_ДОГ	PK	Number
	ДАТА_ДОГ		Datetime
	СУММА_ДОГ		Number
Накладная	НОМ_НАКЛ	PK	Number
	ДАТА_ОТПР		Datetime
	СУММА_НАКЛ		Number

- Выделите сущность **Покупатель**, щелкнув по ней указателем мыши, а затем вызовите пункт меню **Model | Attributes**. То же самое можно выполнить, выбрав пункт **Attributes** контекстного меню. При этом на экране появится окно редактора атрибутов **Attributes**.

Редактор атрибутов построен по тому же принципу, что и редактор сущностей. В верхней части диалогового окна находится выпадающий список, в котором можно выбрать сущность для редактирования. Рядом имеется кнопка, вызывающая редактор сущностей.

- Для ввода нового атрибута нажмите кнопку **New**.
- В диалоге **New Attribute** в поле **Attribute Name** введите имя атрибута –**КОД_ПОК**, в поле **Column Name** необходимо указать имя соответствующей атрибуту в физической модели колонки. По умолчанию Erwin генерирует имя колонки из имени атрибута, заменяя пробелы символом подчеркивания. Поскольку СУБД Access, для которой мы создаем модель, допускает использование букв русского алфавита в идентификаторах колонок таблиц, подставляемое по умолчанию значение в **Column Name** мы оставляем без изменения.
- В группе Domain находится список доменов, представляющих основные типы данных, используемые в СУБД: строковый (string), числовой (number), время (datetime), двоичный (blob). Для атрибута КОД_ПОК выберите числовой домен – **Number**.
- После нажатия кнопки **ОК** атрибут появится в окне редактора.
- Выделите атрибут **КОД_ПОК** и установите на закладке **General** флажок **Primary Key**, так как данный атрибут является первичным ключом сущности Покупатель.
- Аналогичным образом введите остальные атрибуты сущности Покупатель в соответствии с табл. 1.2.

В результате окно редактора атрибутов будет выглядеть так, как показано на рис. 1.7.

Порядок следования атрибутов в списке можно изменять при помощи кнопок со стрелками, находящимися над окном списка. Для этого необходимо выбрать нужный атрибут в списке, нажать одну из этих кнопок, и атрибут сместится в списке в направлении стрелки, изображенной на кнопке.

- Нажмите кнопку **ОК**.

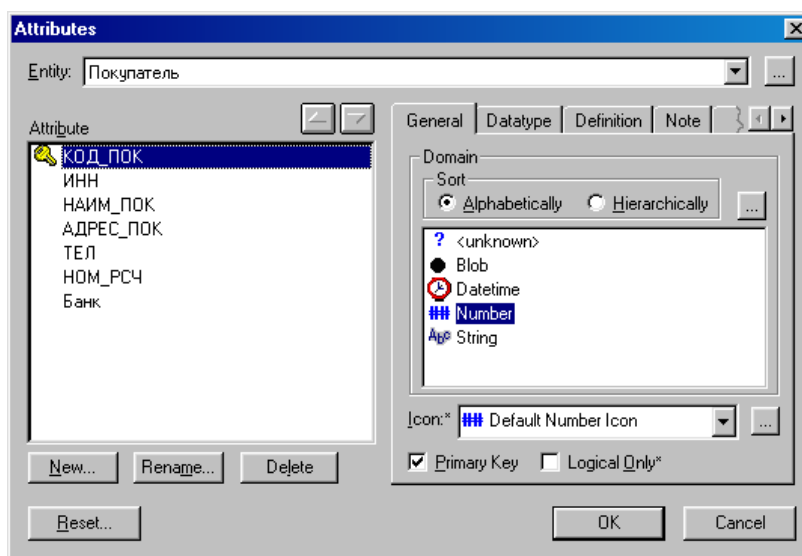


Рис. 1.7. Атрибуты сущности Покупатель

- Как вы помните, мы создали два хранимых отображения – «Уровень сущностей» и «Уровень атрибутов». До сих пор мы работали на уровне сущностей, где сущности изображались просто прямоугольниками с названием сущности внутри. Перейдите на вкладку «Уровень атрибутов». Сущности изображаются здесь в виде прямоугольников, однако имя сущности пишется над прямоугольником, а внутри дается список атрибутов. Прямоугольник сущности делится на две части. В верхней части приводятся атрибуты первичного ключа, а в нижней – все остальные.

Пока на диаграмме определены только атрибуты сущности Покупатель, поэтому прочие сущности пусты.

- Определите атрибуты остальных сущностей на диаграмме в соответствии с табл. 1.2.

5. Определение альтернативных ключей и инверсных входов

Альтернативный ключ (Alternate Key) – потенциальный ключ, не ставший первичным. Erwin позволяет выделять атрибуты потенциальных ключей и при генерации схемы БД генерировать по этим группам отдельные уникальные индексы.

Инверсный вход (Inversion Entry) – атрибут или группа атрибутов, которые не определяют экземпляр сущности уникальным образом, но часто используются для обращения к экземплярам сущности. Erwin генерирует неуникальный индекс для каждого инверсного входа.

В табл. 1.3. приведен перечень ключевых групп, определенных для рассматриваемой предметной области.

Таблица 1.3.

Ключевые группы

<i>Сущность</i>	Атрибуты ключевой группы	Имя ключевой группы	Тип ключевой группы
Покупатель	ИНН	ИНН	Альтернативный ключ
Покупатель	НАИМ_ПОК	НАИМ_ПОК	Инверсный вход
Товар	НАИМ_ТОВ	НАИМ_ТОВ	Инверсный вход
Склад	НАИМ_СК	НАИМ_СК	Инверсный вход

- Вызовите редактор ключевых групп **Key Groups**, щелкнув правой кнопкой мыши по сущности **Покупатель** и выбрав из контекстного меню пункт **Key Groups**. Редактор ключевых групп также можно вызвать через главное меню: **Model | Key Groups**.

Редактор ключевых групп содержит элементы управления:

Entity – поле с выпадающим списком, в котором следует выбрать сущность для редактирования.

Окно с перечнем ключевых групп. Каждая группа представлена отдельной строкой, включающей в себя имя (Key Group), тип (Type) и определение (Definition).

Кроме того, диалоговое окно редактора ключевых групп содержит следующие закладки:

- ✓ *Members (члены).* Задаются члены ключевых групп и их порядок следования в группе.

- ✓ *General (общие установки).* Переключатели, позволяющие задавать тип ключевой группы. Для первичного и внешнего ключа эти группы недоступны.

- ✓ *Definition (определение).* Произвольная текстовая информация, относящаяся к выбранной ключевой группе.

- ✓ *Note (примечание).* Примечание к выбранной группе.

- ✓ *UDP (пользовательские свойства).*

- Нажмите кнопку **New**.

- В окне **New Key Group** в поле **Key Group** введите имя ключевой группы – **ИНН**. В поле **Index** выводится генерируемое программой Erwin имя индекса. Оставьте его без изменений.

- Переключатель **Key Group Type** задает тип создаваемого ключа. Это может быть альтернативный ключ (Alternate Key) или инверсный вход (Inversion Entry). Выберите **Alternate Key** и нажмите **ОК**. Вновь введенный альтернативный ключ появится в перечне ключей.

- Перейдите на закладку **Members**. Новый ключ пока не содержит никаких атрибутов, поэтому правый список **Key Group Members** (члены ключевой группы) пуст. Выберите в левом списке атрибут **ИНН** и переместите его в правый список при помощи кнопки со стрелкой (см. рис. 1.8).

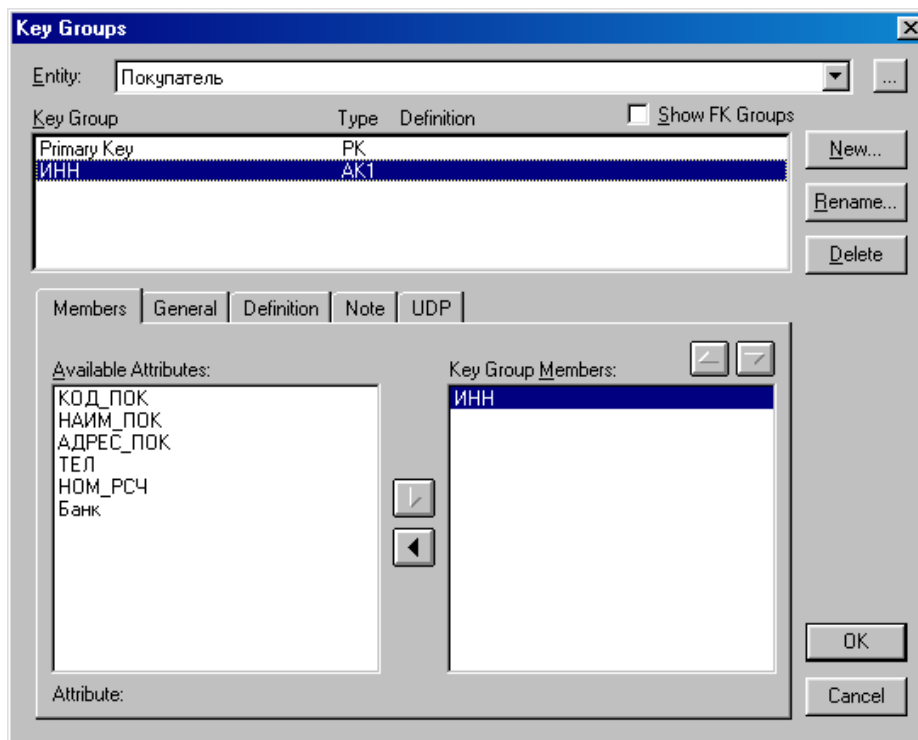


Рис. 1.8. Редактор ключевых групп

- Таким же образом создайте ключевые группы для инверсных входов, приведенных в табл. 1.3.

6. Установление связей между сущностями

Связь является логическим соотношением между сущностями. Связь имеет имя, мощность, тип.

Имя связи (Verb Phrase) – фраза, характеризующая отношение между родительской и дочерней сущностями. Для связи один-ко-многим достаточно указать имя, характеризующее отношение от родительской к дочерней сущности (Parent-to-Child). Для связи многие-ко-многим следует указывать имена как Parent-to-Child, так и Child-to-Parent.

Мощность связи (Cardinality) – служит для обозначения отношения числа экземпляров родительской сущности к числу экземпляров дочерней.

Различают четыре типа мощности:

- общий случай, когда одному экземпляру родительской сущности соответствуют 0, 1 или много экземпляров дочерней сущности (не помечается каким-либо символом);
- символом **P** помечается случай, когда одному экземпляру родительской сущности соответствуют 1 или много экземпляров дочерней сущности (исключено нулевое значение);
- символом **Z** помечается случай, когда одному экземпляру родительской сущности соответствуют 0 или 1 экземпляр дочерней сущности (исключены множественные значения);
- цифрой помечается случай, когда одному экземпляру родительской сущности соответствует заранее заданное число экземпляров дочерней сущности.

Различают **два типа связей**: идентифицирующая и неидентифицирующая.

Идентифицирующая связь устанавливается между независимой (родительский конец связи) и зависимой (дочерний конец связи) сущностями. Когда рисуется идентифицирующая связь, ERwin автоматически преобразует дочернюю сущность в зависимую. Зависимая сущность изображается прямоугольником со скругленными углами.

Экземпляр зависимой сущности определяется только через отношение к родительской сущности. При установлении идентифицирующей связи атрибуты первичного ключа родительской сущности автоматически переносятся в состав первичного ключа дочерней сущности. В дочерней сущности новые атрибуты помечаются как внешние ключи – (FK).

При установлении неидентифицирующей связи дочерняя сущность остается независимой, а атрибуты первичного ключа родительской сущности мигрируют в состав неключевых компонентов дочерней. Неидентифицирующая связь служит для связи независимых сущностей.

Идентифицирующая связь показывается на диаграмме сплошной линией с жирной точкой на дочернем конце связи, неидентифицирующая – пунктирной.

Для неидентифицирующей связи можно указать обязательность (Nulls). В случае обязательной связи (No Nulls) при генерации схемы БД атрибут внешнего ключа получит признак NOT NULL, несмотря на то, что внешний ключ не войдет в состав первичного ключа дочерней сущности. В случае необязательной связи (Nulls Allowed) внешний ключ

может принимать значение NULL. Необязательная неидентифицирующая связь помечается прозрачным ромбом со стороны родительской сущности.

Атрибуты первичного ключа родительской сущности по умолчанию мигрируют со своими именами. ERwin позволяет ввести для них роли или функциональные имена (Rolename), т.е. новые имена, под которыми мигрирующие атрибуты будут представлены в дочерней сущности.

- Определим связи между сущностями в нашей разрабатываемой модели согласно табл. 1.4.

Таблица 1.4.

Характеристика связей для заданной предметной области

Родительская сущность	Дочерняя сущность	Тип связи	Мощность связи	Нулевые значения	Имя связи
Покупатель	Договор	Неидентифицирующая	0 или 1 к 1 или более	No NULLS	заклучает
Склад	Накладная	Идентифицирующая	0 или 1 к 1 или более	–	Выписывает
Договор	Накладная	Неидентифицирующая	0 или 1 к 1 или более	No NULLS	Составляется
Товар	Договор	<i>Многие-ко-многим</i>			Заказывается (Parent-to-Child), включает (Child-to-Parent)
Товар	Накладная	<i>Многие-ко-многим</i>			Отгружается (Parent-to-Child), включает (Child-to-Parent)

- Создадим связь между сущностями **Покупатель** и **Договор**. Для этого выберите в палитре инструментов кнопку «**Non-Identifying Relationship**» (неидентифицирующая связь).

- Затем щелкните сначала по родительской сущности – **Покупатель**, а потом по дочерней – **Договор**. Между сущностями появится пунктирная линия неидентифицирующей связи. Посреди линии связи проставляется генерируемая по умолчанию глагольная фраза – R/1.

- Перейдите на уровень атрибутов и обратите внимание на то, что у сущности Договор добавился атрибут первичного ключа КОД_ПОК от сущности Покупатель и помечен буквами «FK».

- Выделите связь, щелкнув по ней указателем мыши. Затем нажмите правую кнопку мыши и в контекстном меню выберите пункт **Relationship Properties** (редактор связей).

В верхней части редактора связей находится выпадающий список, содержащий полное название связи. В нашем случае осмысленная глагольная фраза для связи еще не определена, поэтому в этом поле значится «Покупатель R/1 Договор». Здесь же находятся две кнопки New и Delete, с помощью которых можно добавить на схеме новую связь или удалить существующую.

Кроме того, диалоговое окно редактора связей содержит следующие закладки:

- ✓ *General (общие свойства)*. Здесь задаются общие свойства связи – имя, тип и мощность связи.

- ✓ *Definition (определение)*. На этой странице вводится определение связи, облегчающее восприятие модели.

- ✓ *Rolename (Имя роли)* – вводятся функциональные имена (для мигрирующих атрибутов).

- ✓ *RI Actions (Установки ссылочной целостности)* – задаются правила ссылочной целостности.

- Перейдите на закладку **General**. В группе **Verb Phrase** в поле **Parent-to-Child** введите имя связи – **заключает**.

- В группе **Cardinality** (мощность связи) установите опцию **One or More (P)**.

- В группе **Relationship Type** (тип связи) установите опцию **Non-Identifying** (неидентифицирующая связь), а в группе **Nulls** (обязательность) включите флажок **No Nulls**, что означает недопустимость пустых значений внешних ключей.

- Задайте остальные связи для сущностей заданной предметной области в соответствии с табл. 1.4.

В результате логическая модель будет иметь вид, показанный на рис. 1.9.

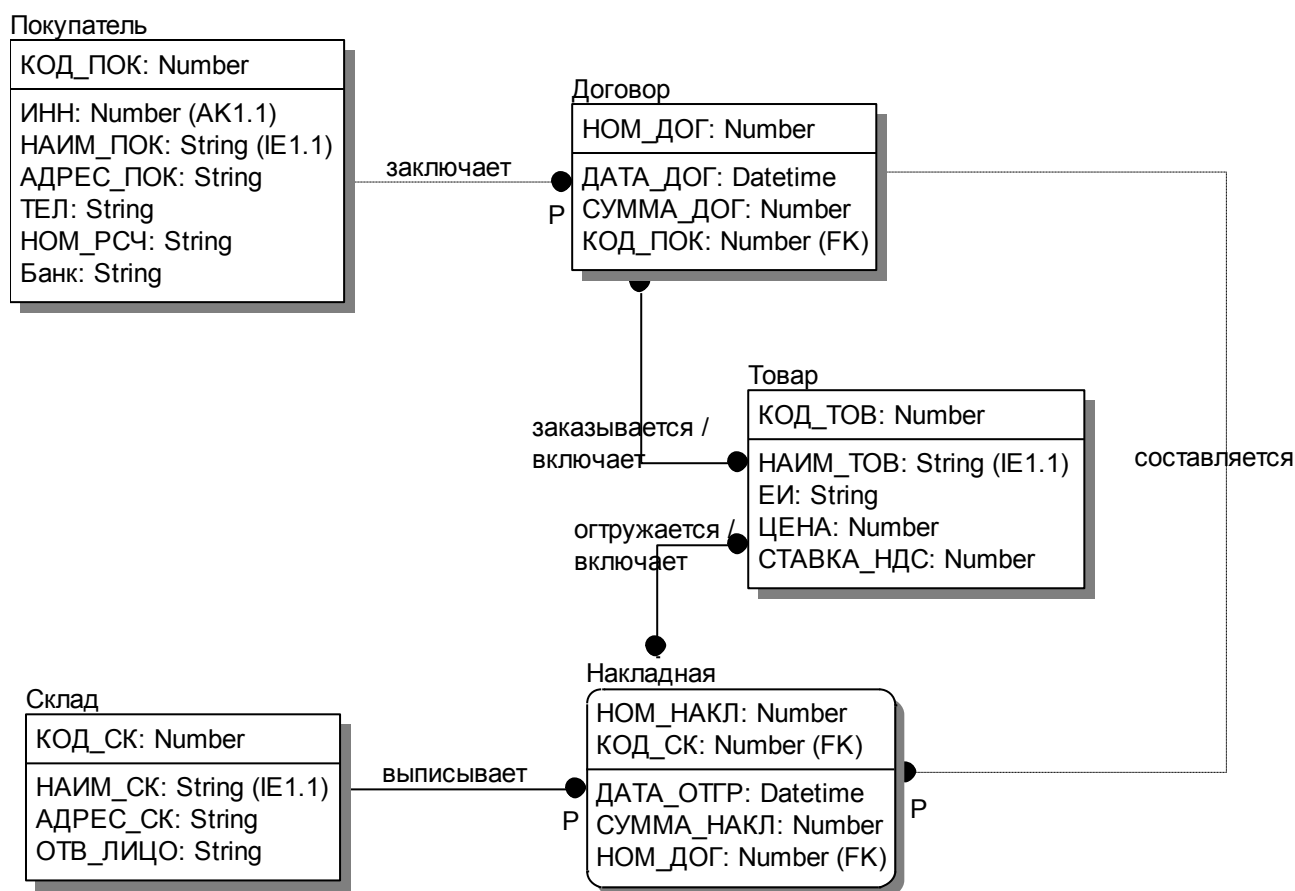


Рис. 1.9. Логическая модель

7. Установление категориальной связи

Некоторые сущности определяют целую категорию объектов одного типа. В ERwin в таком случае создается сущность для определения категории и сущности для каждого элемента категории, а затем вводится для них связь категоризации. Родительская сущность категории называется супертипом, а дочерние – подтипом.

В сущности-супертипе вводится атрибут-дискриминатор, позволяющий различать конкретные экземпляры сущности-подтипа.

В зависимости от того, все ли возможные сущности-подтипы включены в модель, категориальная связь является полной или неполной.

- Для создания категориальной связи необходимо, прежде всего, задать сущности и их атрибуты, не указывая первичный ключ для подтипов.
- Затем выбрать в палитре инструментов кнопку категориальной связи **Complete Sub-category** и щелкнуть сначала по родовому предку, а затем по первому потомку. После этого щелкнуть по символу категории, а потом по следующему потомку.

- Для редактирования категориальной связи необходимо выделить символ связи и в контекстном меню выбрать пункт **Subtype Relationship**.

- В окне редактора следует указать атрибут-дискриминатор (список **Discriminator Attribute Choice**), а также установить тип категории – полная (**Complete**) и неполная (**Incomplete**). Имя дискриминатора появится в диаграмме рядом с символом связи (см. рис. 1.10).

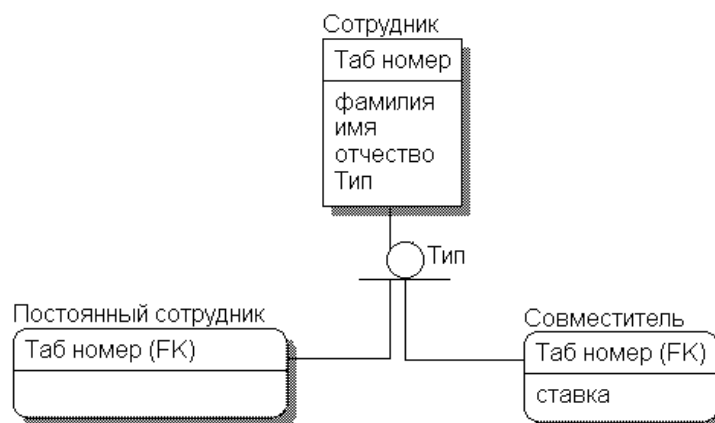


Рис. 1.10. Категориальная связь

Контрольное задание

Создайте средствами редактора ERwin информационную модель в нотации IDEF1X системы по Вашему выбору.

Контрольные вопросы

1. Что такое бизнес-процесс?
2. Каковы основные компоненты функциональной модели?
3. Что представляют собой методологии функционального моделирования?
4. Что такое сценарии?
5. Какие виды сценариев Вы знаете?
6. В чем отличие серверных элементов управления от клиентских?
7. Какие технологии программирования серверных сценариев Вы знаете? В чем их отличие?

Содержание и оформление отчета

Отчет должен содержать: титульный лист, название и цель работы; вариант задания; скриншоты результатов работы; выводы по работе.

Визуальное моделирование с использованием пакета Rational Rose

1. Цель работы

Целью работы является изучение методологии объектно-ориентированного анализа и проектирования программных систем в среде *Rational Software* на этапах создания динамических моделей программной системы посредством построения диаграмм поведения с использованием пакета *Rational Rose*.

2. Теоретические сведения

Для описания динамики системы используются диаграммы поведения (*Behavior diagrams*), которые подразделяются на:

- диаграммы состояний (*Statechart diagrams*);
- диаграммы активности (*Activity diagrams*);
- диаграммы взаимодействия (*Interaction diagrams*), состоящие из:
 - диаграмм последовательности (*Sequence diagrams*);
 - кооперативных диаграмм (*Collaboration diagrams*).

Диаграмма состояний может быть представлена в виде ориентированного графа состояний и переходов из одного состояния в другое. Вершины графа представляют собой состояния объектов, а дуги – переходы. Каждая дуга маркирована парой «условие-действие», где первое представляет условие перехода, а второе – результат.

Диаграмма активности отражает динамику проекта и представляет собой схемы потоков управления в системе от действия к действию, а также параллельные действия и альтернативные потоки.

Диаграммы взаимодействия отображают один из процессов обработки информации в рамках варианта использования. Поскольку в каждом варианте использования может быть несколько альтернативных потоков, то это значит, что для данного варианта использования можно разработать несколько диаграмм взаимодействия, отражающих один и тот же процесс при различных условиях (например, при нормальном ходе процесса функционирования и в случае возникновения ошибки). Диаграммы взаимодействия делятся на диаграммы последовательности и диаграммы кооперации. На диаграммах обоих типов представляется одна и та же информация, однако на диаграммах последовательности основное внимание уделяется управлению, а кооперативные диаграммы отображают потоки данных. Диаграммы последовательности упорядочены во времени, они помогают понять логическую последовательность событий. Кооперативные диаграммы показывают, как компоненты системы взаимодействуют друг с другом.

2.1. Разработка диаграммы состояний

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

Последовательность построения диаграммы состояний:

1. Щелкните правой кнопкой мыши по разделу *Use Case View* в списке браузера.
2. В появившемся меню выберите команду *New, Statechart Diagram* (создать новую, диаграмма состояний). В список будет добавлена новая диаграмма с именем *New Diagram*.

3. Введите название диаграммы (в данном случае: «Разработка учебной программы»).
4. Чтобы открыть диаграмму дважды щелкните по ней мышью в браузере.
5. Щелкните по кнопке *State* (состояние) на панели инструментов.
6. Щелкните по диаграмме, чтобы поместить на нее новое состояние.
7. Введите имя нового состояния (например, «Инициализация пользователя»).
8. Повторите этапы 5-7 для создания остальных состояний, указанных на рис. 14.
9. Щелкните по кнопке *State Transition* (состояние перехода) на панели инструментов. Переходы представляют собой смену исходного состояния последующим.
10. Щелкните по исходному состоянию на диаграмме и переместите линию перехода со стрелкой на последующее состояние.
11. Постройте направленные линии переходов для всех состояний (рис. 14).

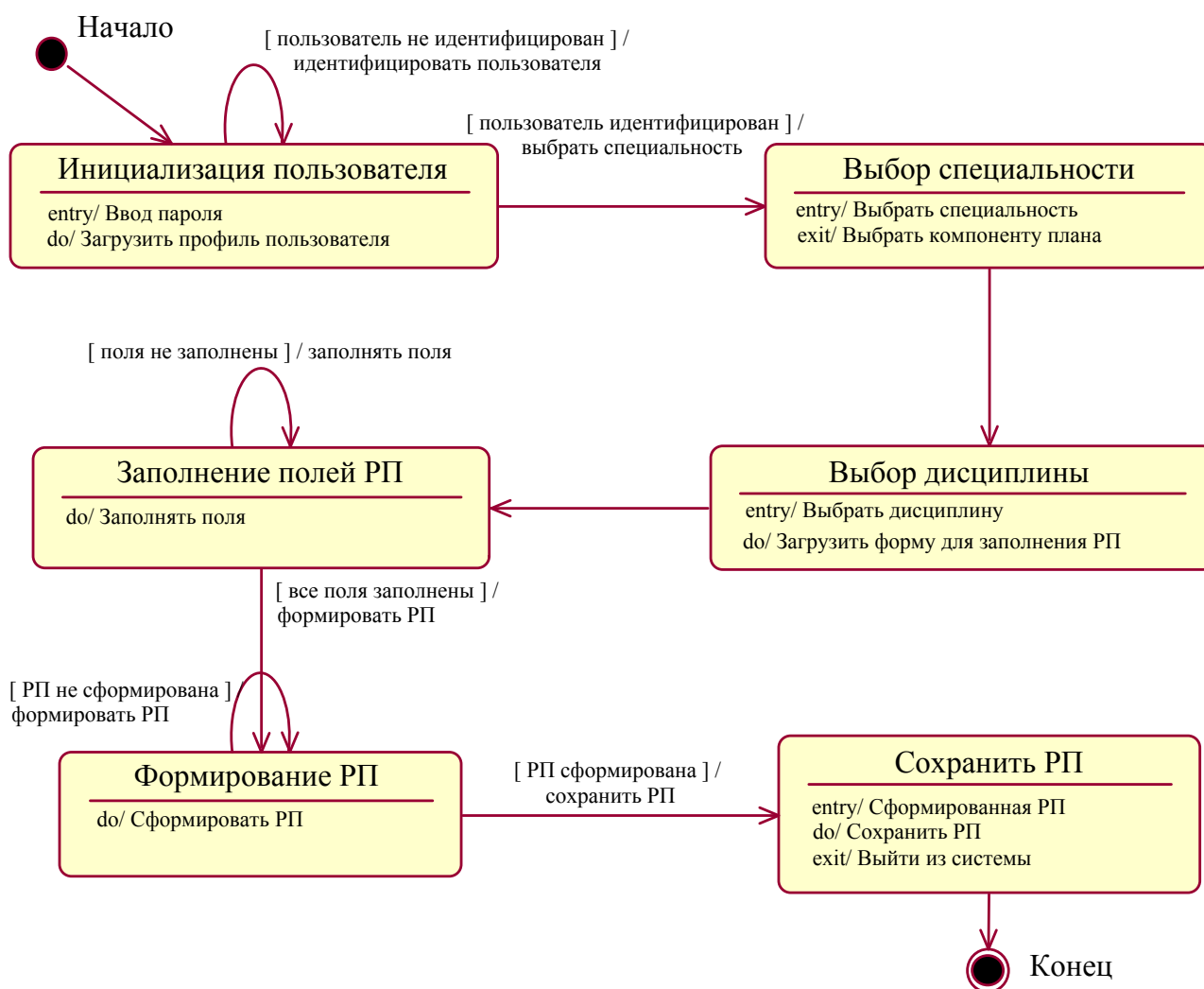


Рис. 14. Диаграмма состояний «Разработка учебной программы»

2.2. Разработка диаграммы активности

Диаграмма активности иллюстрирует действия, переходы между ними, элементы выбора и линии синхронизации. В языке *UML* действие изображается в виде прямоугольника с закругленными углами, переходы – в виде направленных стрелок, элементы выбора – в

виде ромбов, линии синхронизации – в виде горизонтальных или вертикальных линий. Пример диаграммы активности изображен на рис. 15.

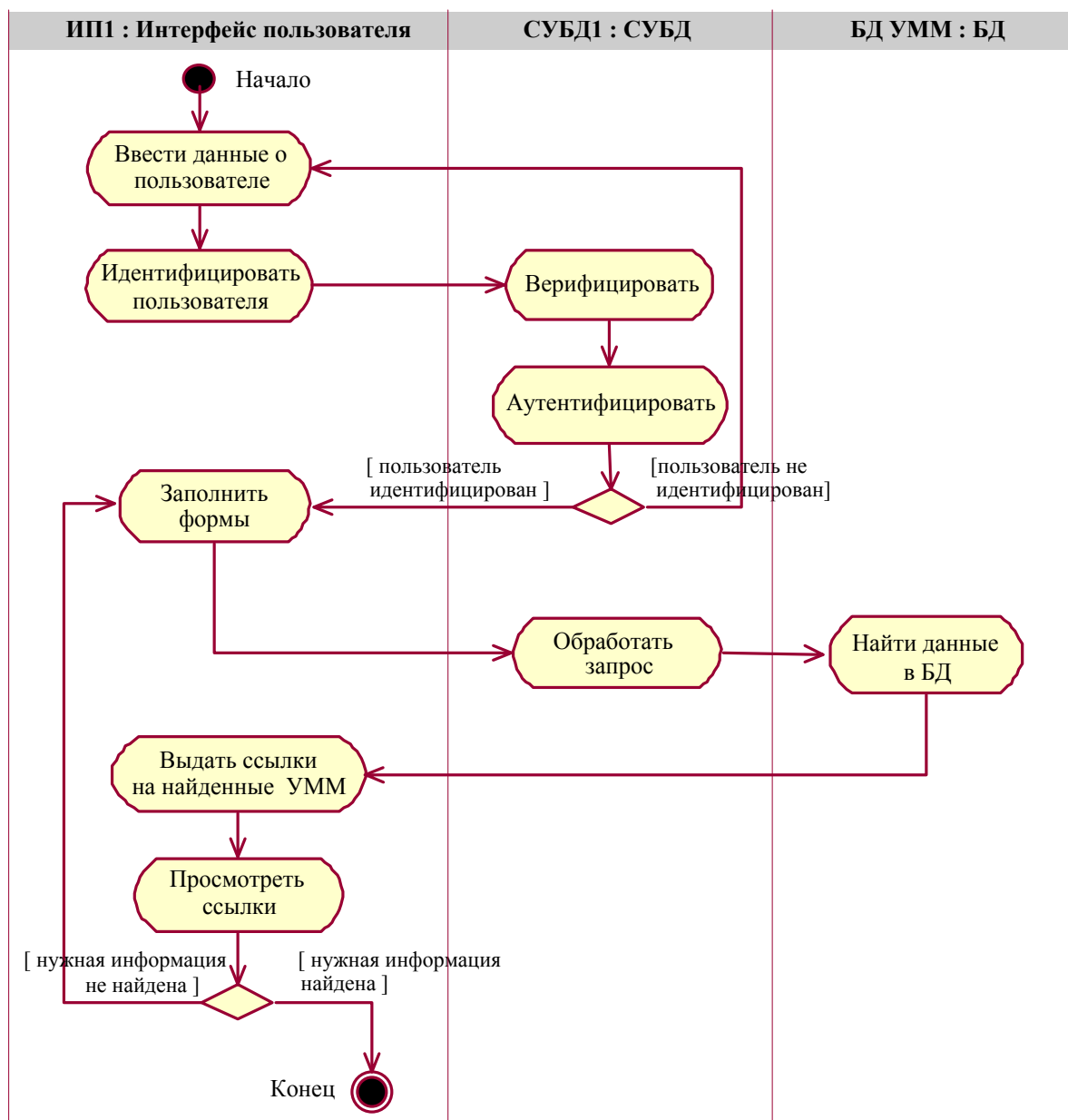


Рис. 15. Диаграмма активности «Поиск учебно-методических материалов»

Последовательность построения диаграммы активности:

1. Щелкните правой кнопкой мыши по разделу *Use Case View* в списке браузера.
2. В появившемся меню выберите команду *New, Activity Diagram* (создать новую, диаграмма активности). В список будет добавлена новая диаграмма с именем *New Diagram*.
3. Введите название диаграммы (в данном случае, «Поиск УММ»).
4. Чтобы открыть диаграмму дважды щелкните по ней мышью в браузере.
5. Щелкните по кнопке *Activity* (действие) на панели инструментов.
6. Щелкните по диаграмме действий, чтобы поместить элемент, изображающий действие, на диаграмму.
7. Введите имя нового действия (например, «Ввод данных о пользователе»).
8. Повторите этапы 5-7 для создания остальных действий, указанных на рис. 15.

9. Щелкните по кнопке *State Transition* (состояние перехода) на панели инструментов. Переходы используются для изображения пути потока управления от действия к действию.

10. Щелкните по начальному действию на диаграмме и переместите стрелку перехода на последующее действие.

11. Создайте стрелки перехода для всех действий, указанных на рис. 15.

Для построения элементов выбора:

1. Щелкните по кнопке *Decision* (элемент выбора) на панели инструментов.

2. Щелкните по диаграмме действий, чтобы поместить элемент выбора на диаграмму.

3. Щелкните по кнопке *State Transition* (состояние перехода) на панели инструментов.

4. Щелкните по начальному действию на диаграмме и переместите стрелку перехода на элемент выбора.

5. Создайте условные переходы от элемента выбора и назовите их (например, «Пользователь идентифицирован» и «Пользователь не идентифицирован»).

6. Повторите этапы 1-6 для задания следующего элемента выбора.

Линии синхронизации используются для отображения действий процесса, выполняемых параллельно.

Линии синхронизации строятся следующим образом:

1. Щелкните по кнопке *Horizontal Synchronization* (горизонтальная линия синхронизации) на панели инструментов.

2. Щелкните по диаграмме действий, чтобы поместить на нее линию синхронизации.

3. Щелкните по кнопке *State Transition* (состояние перехода) на панели инструментов, добавьте необходимые входящие и исходящие линии переходов и линии синхронизации, указанные на рис. 15.

Для явного задания участника реализации того или иного процесса на диаграмме активности используются так называемые плавательные дорожки (*Swimlanes*) или секции.

Задание секций:

1. Щелкните по кнопке *Swimlanes* (плавательная дорожка) на панели инструментов.

2. Щелкните по диаграмме действий, чтобы создать на ней новую секцию. Дважды щелкните по названию секции и введите нужное имя. Для изменения размеров секции переместите ее границу с помощью мыши. Переместите все необходимые действия и переходы в секцию.

2.3. Разработка диаграмм взаимодействия

Диаграммы взаимодействия содержат объекты, классы или то и другое вместе и сообщения. Объект – это абстракция чего-либо в домене прикладной области или в выполняемой системе. Объект инкапсулирует данные и поведение, которые отличаются от традиционного разделения на функции и данные. Данные объекта представляются атрибутами, а его поведение – операциями. Значения атрибутов изменяются во времени, но сами атрибуты неизменны. Однотипные объекты (имеющие одинаковые атрибуты и операции) объединяются в классы. Класс – это некая сущность, представляющая собой как бы схему объекта. Иными словами, класс определяет данные и поведение, которыми должен

обладать объект. Класс – более общий термин, являющийся, по существу, шаблоном для объектов. Объекты создаются для реализации функциональных возможностей, заложенных в варианте использования.

Сообщение (*Message*) – это связь между объектами, в которой один из них (клиент) требует от другого (сервера) выполнения каких-то действий. При генерации кода сообщения транслируются в вызовы функций. С помощью сообщения один объект или класс запрашивает у другого выполнения какой-то конкретной функции.

На диаграмме последовательности взаимодействие изображается в виде двухмерной схемы (в формате графа или сети). По вертикали проходит временная ось, где течение времени происходит сверху вниз. По горизонтали указываются роли классификатора, которые представляют отдельные объекты кооперации. У каждой роли классификатора есть «линия жизни», идущая сверху вниз. Тот период времени, в течение которого объект существует, изображается на диаграмме вертикальной пунктирной линией. Во время вызова процедуры определенного объекта (активизации) его линия жизни изображается в виде полого прямоугольника, который замещает собой часть линии жизни объекта.

Сообщение выглядит на диаграмме последовательности как линия со стрелкой, идущая от линии жизни одного объекта к линии жизни другого объекта. Стрелки организованы согласно временной последовательности сообщений.

Активацией называется выполнение процедуры, включающее в себя время ожидания выполнения всех вложенных процедур. Активным называется объект, которому принадлежит стек активаций (и вызовов операций). У каждого активного объекта есть свой собственный поток управления, который выполняется параллельно с другими активными объектами. Объекты, вызываемые пассивными объектами, называется пассивными. Они получают управление только на время вызова, а потом возвращают его.

Диаграмма кооперации – это диаграмма классов, на которой отображаются не просто классификаторы и ассоциации, а роли классификатора и роли в ассоциации. Роли классификатора и роли в ассоциации описывают конфигурацию объектов и связей, которые могут образоваться при выполнении кооперации в реальной системе.

Последовательность построения диаграмм взаимодействия:

1. Создайте диаграмму последовательности для выбранного прецедента (контекстное меню в браузере – *New, Sequence Diagram* (создать новую, диаграмма последовательности) или в контекстном меню на рабочем столе *Sub Diagram, New Sequence Diagram* (поддиаграмма, новая диаграмма последовательности)). В данном случае – это диаграмма добавления учебной дисциплины (рис. 16).

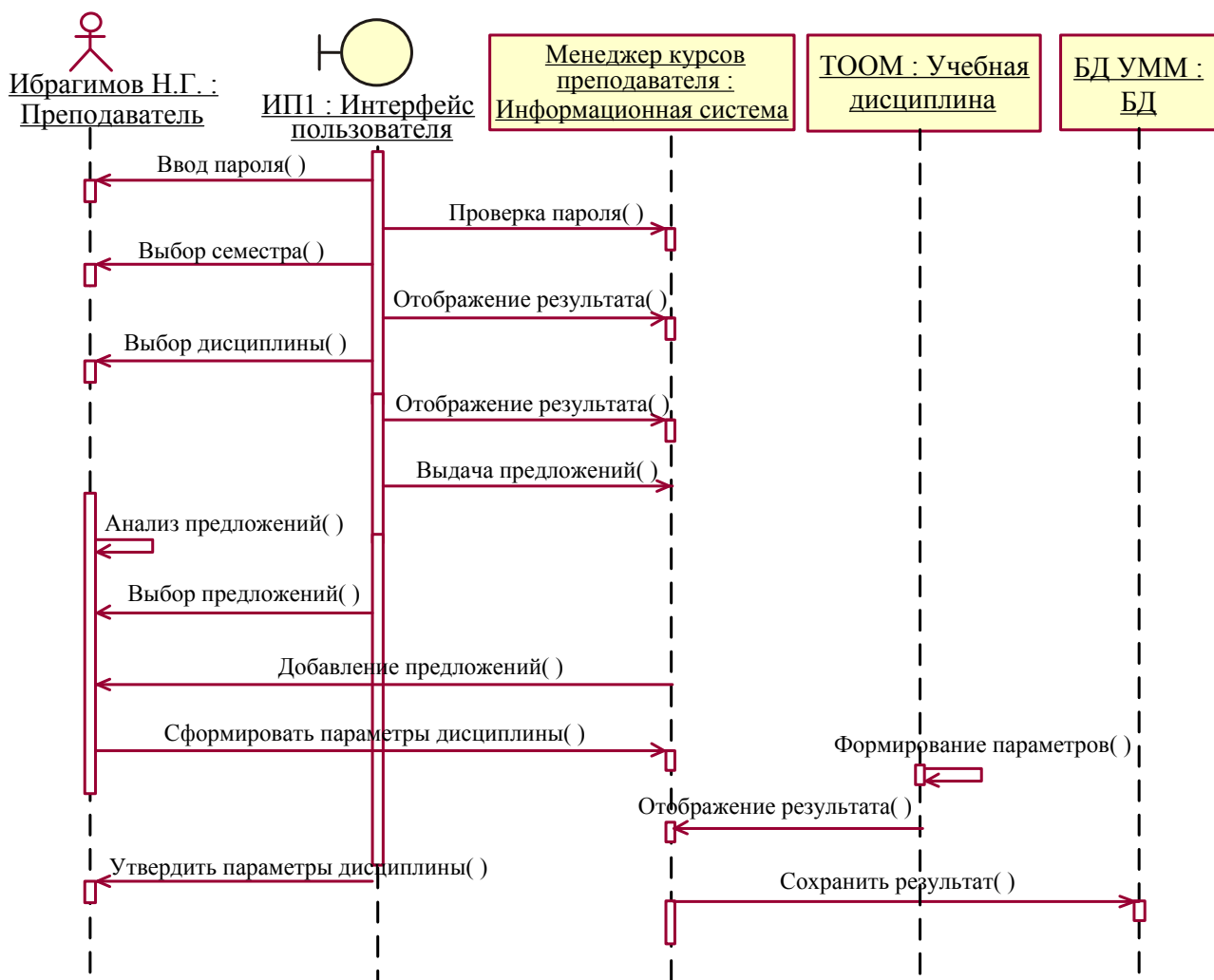


Рис. 16. Диаграмма последовательности «Добавление учебной дисциплины»

2. Создайте объекты, переместив их непосредственно на рабочий стол из строки инструментов (*Object*), или перенесите из окна браузера уже существующие. В данном случае это объекты-сущности «Преподаватель», «Интерфейс пользователя», «Учебная дисциплина», «БД» и управляющий класс «Информационная система».

3. Добавьте необходимые сообщения (*Message*) из строки инструментов. Вызовите окно спецификации для сообщения, щелкнув по линии мышкой. Задайте имя и свойства.

4. Задайте свойства объектов (*Open Specification* в контекстном меню выбранного элемента).

5. Диаграмму коопераций (рис. 17) можно создать из диаграммы последовательности, если она отображает тот же процесс (*Browse, Create Collaboration Diagram* (просмотр, создать диаграмму кооперации)).

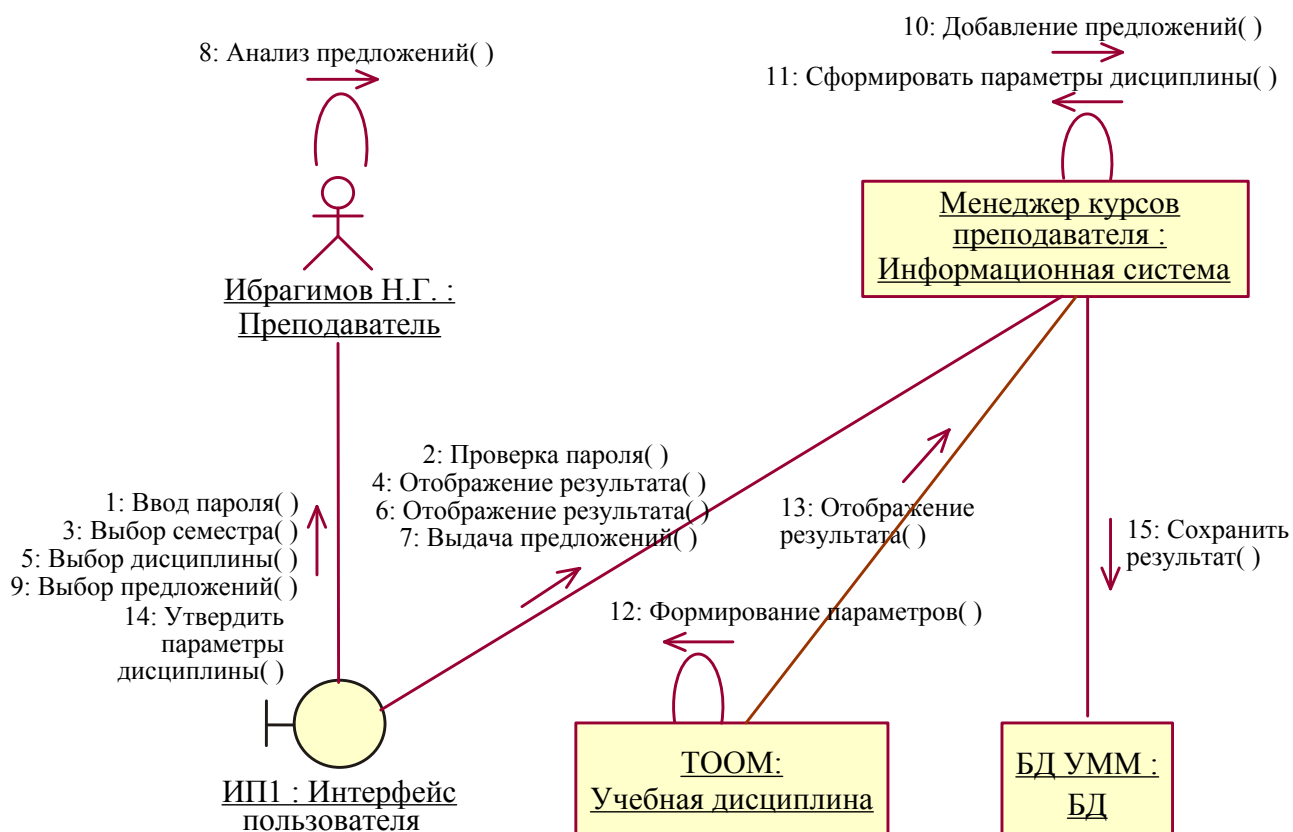


Рис. 17. Диаграмма кооперации «Добавление учебной дисциплины»

С помощью диаграмм взаимодействия проектировщики и разработчики системы могут определить классы, которые нужно создать, связи между ними, а также операции и ответственности (*Responsibilities*) каждого класса.

3. Порядок выполнения работы

1. Загрузите файл со структурным описанием проектируемой системы.
2. Выберите существующий прецедент системы и разработайте для него диаграмму последовательности и диаграмму коопераций.
3. Разработайте диаграмму состояний и диаграмму активности для любого объекта вашей системы, имеющего нетривиальное поведение.

4. Требования к отчету

Отчет к лабораторной работе должен содержать:

- титульный лист;
- описание динамики поведения разрабатываемой программной системы;
- построенные динамические диаграммы;
- ответы на контрольные вопросы.

5. Контрольные вопросы

1. Что такое *UML*?

2. Какими типами диаграмм задаются динамические модели?
3. Перечислите основные элементы диаграммы состояний.
4. Перечислите основные элементы диаграммы активности.
5. Что такое *Swimlanes*?
6. Чем диаграмма последовательности отличается от диаграммы кооперации?

6. Содержание и оформление отчета

Отчет должен содержать: титульный лист, название и цель работы; вариант задания; скриншоты результатов работы; выводы по работе.