МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Методические указания
по выполнению лабораторных работ

по дисциплине «Корпоративные информационные системы»

для студентов направления подготовки
09.03.02 «Информационные системы и технологии»
Профиль «Прикладное программирование и интернет-технологии»

Ставрополь, 2026

ОГЛАВЛЕНИЕ

Введение

Основной целью и задачей дисциплины «Корпоративные информационные системы», является расширение знаний и приобретение навыков работы студентами, в следующих областях:

– основы построения интерфейсов;

– организация каналов связи, алгоритмы канального уровня МОС;

– основы использования кабельных сетей, алгоритмы физического уровня МОС ;

– основы использования беспроводных технологий;

– основы построения систем телеобработки;

– разработка и проектирование сетевых систем;

– разработка и проектирование систем телеобработки.

Методические указания к выполнению лабораторных работ по дисциплине «Корпоративные информационные системы» содержат 16 лабораторных работ.

Перечень планируемых результатов обучения по дисциплине (модулю), соотнесённых с планируемыми результатами освоения образовательной программы

Код, формулировка компетенции	Код, формулировка индикатора	Планируемые результаты обучения по дисциплине (модулю), характеризующие этапы формирования компетенций, индикаторов
ПК-2 – способен разрабатывать системы управления данными и знаниями	ПК-2$_{ИД-2.1}$ – Демонстрирует знания в области проектирования и разработки баз данных и приложений на основе баз данных	Демонстрирует понимание сложных концепций и методов проектирования баз данных и разработки приложений на основе баз данных. Навыки оптимизации структуры баз данных и производительности запросов. Обладает практическими навыками использования передовых технологий и методов работы с базами данных, таких как NoSQL, Big Data, облачные вычисления и т. д. Способность разрабатывать и реализовывать инновационные и сложные базы данных и приложения.
	ПК-2$_{ИД-2.2}$ – Разрабатывает подсистемы обработки данных на основе СУБД различного типа	Обладает всесторонними знаниями о различных аспектах работы СУБД, включая оптимизацию запросов, проектирование баз данных, обеспечение безопасности данных и масштабирование системы; способен создавать высокопроизводительные и надежные подсистемы обработки данных на основе любого типа СУБД

	ПК-2$_{ИД-2.3}$ – Применяет методы поиска, очистки и предварительной обработки данных различного формата	Способен проводить предварительную обработку данных различного формата с использованием более сложных методов. Способен применять регулярные выражения для поиска и замены данных, а также применять функции и формулы для преобразования и анализа данных. Способен использовать автоматическую обработку для массовой очистки данных.
	ПК-2$_{ИД-2.4}$ – Демонстрирует знания методов обработки и инструментария больших данных	Имеет глубокие знания о методах обработки и анализа больших данных. Владеет навыками разработки и оптимизации программ, работающих с большими объемами данных. Способен применять сложные алгоритмы и модели для анализа больших данных
	ПК-2$_{ИД-2.5}$ – Разрабатывает интеллектуальные системы, в том числе распределённые, обучающиеся на больших данных	Демонстрирует глубокое понимание принципов работы и оптимизации интеллектуальных систем. Имеет опыт работы с распределенными вычислениями и системами хранения данных. Способен разрабатывать сложные модели машинного обучения, такие как нейронные сети. Имеет опыт работы с высокопроизводительным вычислительным оборудованием, таким как GPU или TPU
ПК-6 – способен проектировать информационные подсистемы обработки данных	ПК-6$_{ИД-6.1}$ – Демонстрирует знания в области архитектуры систем управления базами данных, типологии баз данных, методов оптимизации баз данных	Обладает экспертными знаниями в области архитектуры СУБД, типологии баз данных и методов оптимизации баз данных. Способен проектировать и реализовывать сложные архитектуры СУБД, включая кластеризацию, распределенные системы и резервное копирование данных. Способен использовать продвинутые методы оптимизации баз данных, такие как горизонтальное и вертикальное шардинг, разделение данных и партиционирование. Способен исследовать и внедрять новые технологии и инструменты, связанные с архитектурой СУБД.
	ПК-6$_{ИД-6.2}$ – Применяет инструментальные средства для проектирования баз данных различного типа, в том числе распределенных	Обладает критическим мышлением и глубокими знаниями в области проектирования баз данных различного типа. Способен использовать инструменты и технологии, такие как NoSQL или реляционные базы данных, для работы с большими объемами данных и распределенной системой хранения данных. Способен применять передовые методы и подходы, такие как

		горизонтальное масштабирование или репликация, для создания и управления распределенными базами данных.
	ПК-6$_{ИД-6.3}$ – Владеет методикой оптимизации систем хранения данных на основе интерпретируемых критериев эффективности	Обладает глубокими знаниями в области оптимизации систем хранения данных и владеет несколькими методиками оптимизации. Способен анализировать и оптимизировать сложные системы хранения данных с использованием интерпретируемых критериев эффективности. Способен предлагать свои собственные методики оптимизации и модифицировать существующие.
	ПК-6$_{ИД-6.4}$ – Анализирует предметную область в рамках решения профессиональных задач и проектирует архитектуру систем обработки данных на основании типов данных, конфигурации потоков данных	Обладает глубоким знанием анализа предметной области и проектирования архитектуры систем обработки данных. Может проводить комплексный анализ данных, генерировать новые типы данных и выстраивать сложные конфигурации потоков данных. Умеет использовать передовые технологии и инструменты для обработки данных, такие как машинное обучение и искусственный интеллект. Может проектировать инновационные архитектуры систем обработки данных, учитывая потребности и требования бизнеса.
ПК-7 – способен осуществлять документирование всех стадий жизненного цикла информационных систем, в том числе интеллектуальных	ПК-7$_{ИД-7.1}$ – Демонстрирует знания методик автоматизации составления технической документации и разработки отчетности по утвержденным формам	Обладает высокими знаниями и умениями в методиках автоматизации составления технической документации и разработки отчетности. Способен применять различные стратегии и методы при автоматизации этих процессов, а также выполнять сложные и нетиповые задачи с использованием инструментов и программного обеспечения. Имеет опыт руководства в данной области.
	ПК-7$_{ИД-7.2}$ – Составляет техническую документацию на всех стадиях реализации проекта с использованием современных инструментальных средств	Обладает глубокими знаниями и опытом в составлении технической документации. Способен использовать широкий спектр инструментальных средств, включая специализированные программы для разработки технической документации, такие как CASE-системы и средства автоматизации процесса составления документов. Способен может использовать новейшие технологии и методы, такие как документация с применением средств виртуальной реальности или техники моделирования.

	ПК-7$_{ИД-7.3}$ – Применяет методы визуализации данных для обоснования эффективности проектов	Владеет основными методами визуализации данных и может использовать их в сочетании. Он может создавать сложные графики и диаграммы, используя несколько переменных и факторов. Способен применять методы многомерной визуализации данных, такие как облака точек, сетки Шулле и т. д. Способен создавать анимированные графики и визуализации для демонстрации динамики данных и трендов.

Лабораторная работа № 1

Интегрированная среда CASE-средства проектирования информационных систем ERwin 7.3

Цель и содержание: познакомить студентов с интерфейсом CASE-средства проектирования информационных систем ERwin 7.3.

Формируемые компетенции: ПК-2, ПК-6. ПК-7.

Теоретическое обоснование

ERwin- CASE-средство проектирования баз данных от фирмы ComputerAssociates. ERwincочетает графический интерфейс WindowsXP, инструменты для построения ER-диаграмм, редакторы для создания логического и физического описания модели данных и прозрачную поддержку ведущих реляционных СУБД. Для удобства изложения материала здесь и далее использована оригинальная терминология, принятая в ERwin.

ERwinне привязан к технологии какой-либо конкретной фирмы, поставляющей СУБД или средства разработки. Он поддерживает различные серверы баз данных и настольные СУБД, а также может обращаться к базе данных через интерфейс ODBC. Так, в текущей версии ERwinвстроена поддержка 23 СУБД, среди которых: Oracle; MicrosoftSQLServeri т.п. (см. рисунок 13.8). Заметим лишь, что речь идет только о реляционных СУБД.

ERwinможно использовать совместно с некоторыми популярными средствами разработки клиентских частей приложений: PowerBuilder, VisualBasic, Delphi. Кроме того, ERwinподдерживает работу в среде групповой разработки ModelMart, являющейся продуктом той же ComputerAssociates.

Процесс моделирования в ERwinбазируется на методологии проектирования реляционных баз данных IDEF1X. Данная методология была разработана для ВВС США и теперь широко используется в правительственных учреждениях и частных компаниях, как в самих США, так и далеко за их пределами. Она определяет стандарты терминологии и графического изображения типовых элементов на ER-диаграммах. Заметим, что некоторые обозначения могут несколько расходиться с традиционными, принятыми в

ER-модели, хотя в ERwinверсии 7.3 существует возможность выбора традиционной нотации. (При изложении материала использована нотация IDEF1X). Кроме того, существует ряд отличий, связанных с тем, что данная методология ориентирована на разработку реляционных БД. Но это не вносит заметных корректив в сам подход к разработке структуры БД, а жесткая стандартизация позволяет избежать такого недостатка ER-моделей, как воз-

можность различной трактовки.

Отображение модели данных в Erwin

Физическая и логическая модель данных

Erwin имеет два уровня представления модели - логический и физический.

Логический уровень - это абстрактный взгляд на данные, на нем данные представляются так, как выглядят в реальном мире, и могут называться так, как они называются в реальном мире, например «Постоянный клиент», «Отдел» или «Фамилия сотрудника». Объекты модели, представляемые на логическом уровне, называются сущностями и атрибутами (подробнее о сущностях и атрибутах будет рассказано ниже). Логическая модель данных может быть построена на основе другой логической модели, например на основе модели процессов (см. BPwin). Логическая модель данных является универсальной и никак не связана с конкретной реализацией системы управления базой данных (СУБД).

Физическая модель данных, напротив, зависит от конкретной СУБД, фактически являясь отображением системного каталога. В физической модели содержится информация обо всех объектах базы данных (БД). Поскольку стандартов на объекты БД не существует (например, нет стандарта на типы данных), физическая модель зависит от конкретной реализации СУБД. Следовательно, одной и той же логической модели могут соответствовать несколько разных физических моделей. Если в логической модели не имеет

значения, какой конкретно тип данных имеет атрибут, то в физической модели важно описать всю информацию о конкретных физических объектах -таблицах, колонках, индексах, процедурах и т. д. Разделение модели данных на логические и физические позволяет решить несколько важных задач.

Документирование модели. Многие СУБД имеют ограничение на именование объектов (например, ограничение на длину имени таблицы или запрет использования специальных символов - пробела и т. п.). Зачастую разработчики ИС имеют дело с нелокализованными версиями СУБД. Это означает, что объекты БД могут называться короткими словами, только латинскими символами и без использования специальных символов (т. е. нельзя назвать таблицу предложением - только одним словом). Кроме того, проектировщики БД нередко злоупотребляют «техническими» наименованиями, в результате таблица и колонки получают наименования типа **RTD_324** или **CUST_A12** и т. д. Полученную в результате структуру могут понять только специалисты (а чаще всего только авторы модели), ее невозможно обсуждать

с экспертами предметной области. Разделение модели на логическую и физическую позволяет решить эту проблему. На физическом уровне объекты БД могут называться так, как того требуют ограничения СУБД. На логическом уровне можно этим объектам дать синонимы - имена более понятные неспециалистам, в том числе на кириллице и с использованием специальных символов. Например, таблице **CUST_A12** может соответствовать сущность **Постоянный клиент**. Такое соответствие позволяет лучше задокументировать модель и дает возможность обсуждать структуру данных с экспертами предметной области.

Масштабирование. Создание модели данных, как правило, начинается с создания логической модели. После описания логической модели, проектировщик может выбрать необходимую СУБД и ERwinавтоматически создаст соответствующую физическую модель. На основе физической модели ERwinможет сгенерировать системный каталог СУБД или соответствующий SQL-скрипт. Этот процесс называется прямым проектированием (**ForwardEngineering**). Тем самым достигается масштабируемость модели (создав одну логическую модель данных, можно сгенерировать физические модели под любую поддерживаемую ERwinСУБД). С другой стороны, ERwinспособен по содержимому системного каталога или SQL-скрипту воссоздать физическую и логическую модель данных (**ReverseEngineering**). На основе полученной логической модели данных можно сгенерировать физическую модель для другой СУБД и затем сгенерировать ее системный каталог. Следовательно, ERwinпозволяет решить задачу по переносу структуры данных с одного сервера на другой. Например, можно перенести структуру данных с Oraclena Informix (или наоборот) или перенести структуру dbf-файлов в реляционную СУБД, тем самым, облегчив решение по переходу от файл-серверной к клиент-серверной информационной системы (ИС). Заметим, однако, что формальный перенос структуры «плоских» таблиц на реляционную СУБД обычно неэффективен. Для того чтобы извлечь выгоды от перехода на клиент-серверную технологию, структуру данных следует модифицировать. Методика прямого проектирования ИС будет рассмотрена ниже.

Для переключения между логической и физической моделью данных служит список выбора в левой части панели инструментов ERwin (рисунок 1.1).

Рисунок 1.1 - Переключение между логической и физической моделью

При переключении, если физической модели еще не существует, она будет создана автоматически.

Аппаратура и программное обеспечение.

Аппаратура. Для выполнения лабораторной работы необходимы:

- персональный суперкомпьютер i7-5930K/ 8D8192D4-2133/ K2200/ 2K40M;
- рабочие станции: монитор 29 + клавиатура + мышь. ПК MDT750/ i75930K/ 4D8192D42133";
- аппаратный сетевой обучающий комплекс мод.VDCN-1502.

Программное обеспечение. Для выполнения лабораторной работы необходима операционная система WINDOWS 7/8/10 и CASE-средство ERwin7.3.

Указания по технике безопасности

При выполнении лабораторной работы **запрещается**:

1. Самостоятельно производить ремонт персонального компьютера, а так же, установку и удаление имеющегося программного обеспечения.
2. Нарушать общепринятые правила техники безопасности при работе с электрооборудованием, в частности, касаться электрических розеток металлическими предметами и т.д.
3. Принимать пищу, напитки и сорить на рабочем месте пользователя персонального компьютера.

В случае неисправности персонального компьютера необходимо **немедленно** сообщить об этом обслуживающему персоналу лаборатории (системному администратору, оператору).

Методика и порядок выполнения работы

1. Пользуясь кнопкой **Start**(Пуск), запустите **ERwin**.
2. Для создания модели в Erwin необходимо выполнить **File>new** или нажать на соответствующий значок на панели инструментов. После чего появится окно (рисунок 1.2).

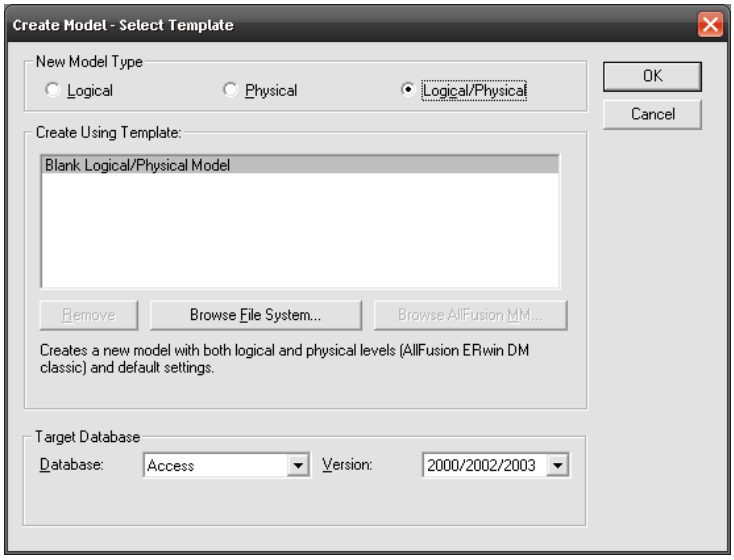


Рисунок 1.2 – Выбор шаблона модели

В рамке **NewModelType**установите опцию **Logical/Physical**, а в рамке **TargetDatabase**(Целевой сервер базы данных) установите в раскрывающемся окне списка систему управления базой данных (СУБД), **Access**и ее версию 2000.

3. Автоматически создается незаполненная модель данных **ERwin**(рисунок 1.3). Таким образом, вы получили доступ к интерфейсу среды **ERwin**. Знакомство с этим интерфейсом и составляет основное содержание лабораторного занятия № 1.

4. Познакомьтесь с интерфейсом среды **ERwin**. Рабочее пространство **ERwin**включает **DiagramWindow**(Окно диаграммы) и **ModelExplorer**(Браузер модели) (рисунок 1.4).

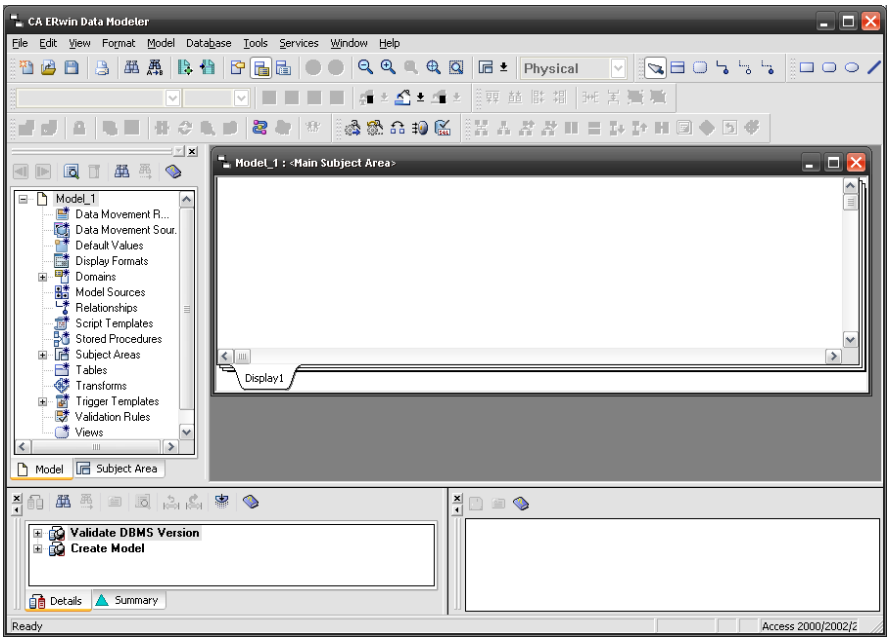


Рисунок 1.3 – Незаполненная модель данных **ERwin**

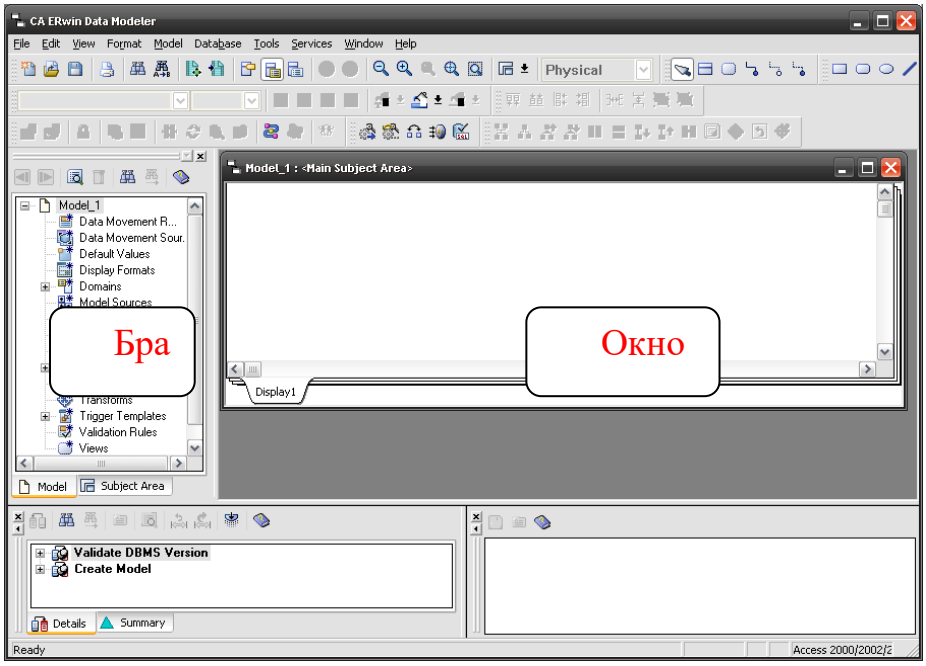


Рисунок 1.4 – Рабочее пространство ERwin.

5 . Изучитеглавноеменю**ERwin**(рисунок1.5).

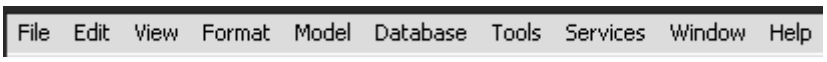


Рисунок 1.5 – Главное меню ERwin

Содержание пунктов главного меню ERwinраскрыто в таблице 1.1. Таблица 1.1 Таблица 1.1 –Содержание пунктов главного меню

Название пункта меню	Назначение пункта меню
File	Открытие, закрытие, сохранение модели
Edit	Редактирование модели
View	Вид модели
Format	Форматирование модели
Model	Свойства модели
ModelMart	Хранилище моделей
Tools	Инструментарий
Window	Окна
Help	Справка

6. Изучите содержание меню **File**(таблица 1.2).

Таблица 1.2 – Содержание меню **File**

Название подпункта меню	Назначение подпункта меню
New	Вызывает окно **ERwinTemplateSelection**, в котором вы можете выбрать шаблон, чтобы использовать как основание для создания новой модели
Open	Показывает окно **ERwinOpenFile**, в котором вы можете выбрать существующую модель, чтобы открыть ее.
Close	Вызывает окно **Closedialog,** в котором вы получаете возможность сохранить или отказаться от сохранения изменений, внесенных в модель
Save	Вызывает окно **SaveAs,** в котором вы получаете возможность сохранить модель под старым или новым именем (команда доступна только после того как модель будет хотя бы раз сохранена)
Save As	Вызывает окно **SaveAs,** в котором вы получаете возможность сохранить модель под новым именем (команда доступна всегда)
Save As New Model	Вызывает окно **SaveAs,** в котором вы получаете возможность сохранить модель под новым идентификационным номером (команда доступна всегда)
Import	Импорт
BPwin	Импорт модели из BPwinв ERwinили экспорт модели ERwinв BPwin
Designer 2000	Импорт модели из OracleDesigner/2000 в ERwinили экспорт модели ERwinв хранилище OracleDesigner/2000
Export	Экспорт
BPwin	Экспорт модели ERwinв BPwin
Designer	Экспорт модели ERwinв хранилище OracleDesigner/2000
Print	Вызывает окно **Printdialog,** в котором вы выбираете режим вывода на принтер активной модели ERwin
Print Setup	Вызывает окно **PrintSetup,** в котором вы выбираете установки принтера
Recent File	Список из четырех последних моделей ERwin, доступных для открытия
Exit	Выход из ERwin

7. Изучите содержание меню **Edit**(таблица 1.3).

Таблица 1.3 – Содержание меню **Edit**

Название подпункта	Назначение подпункта меню
Cut	Вырезание выделенных объектов из модели
Copy	Копирование выделенных объектов модели
Paste	Вставка содержимого буфера обмена в модель
Select All	Выделение всех объектов активной (текущей) модели
GoTo	Переход к указанному объекту модели

8. Изучите содержание меню **View**(таблица 1.4).

Таблица 1.4 – Содержание меню **View**

Название подпункта	Назначение подпункта меню
Redraw Diagram	Перерисовать диаграмму модели
Toolbars	Отображение или скрытие панелей инструментов
ModelExplorer	Отображение или скрытие проводника модели
Stored Display Tabs	Отображение или скрытие уровней вложенности модели
Status Bar	Отображение или скрытие строки состояния модели
Zoom	Управление масштабом отображения модели

9. Изучите содержание меню **Format**(таблица 1.5).

Таблица 1.5 – Содержание меню **Format**

Название подпункта	Назначение подпункта меню
Display Level	Уровень отображения модели
Entity Display	Уровень отображения сущностей модели
Relationship Display	Уровень отображения отношений между сущностями модели
Stored Display Tabs	Вызов диалогового окна StoredDisplaydля задания данных об авторе модели и установки параметров отображения модели

Название подпункта	Назначение подпункта меню
Preferences	Вызов диалогового окна FormatPreferencesдля задания предпочтительных режимов отображения сущностей и других параметров модели
Default Fonts &Colors	Вызов диалогового окна DefaultFonts&Colorsдля задания параметров шрифта и его цвета при отображении различных
Align or Space Evenly	Выравнивание или перемещение выделенных объектов модели в пределах окна
Show Shadows	Отображение или скрытие теней для сущностей модели
Show Page Grid	Отображение или скрытие координатной сетки для упрощения размещения сущностей

10. Изучите содержание меню Model (таблица 1.6).

Таблица 13.6 – Содержание меню **Model**

Название подпункта	Назначение подпункта меню
Subject Areas	Вызов диалогового окна SubjectAreas (подмножество модели) для выбора тематически общих сущностей модели
Entities	Вызов диалогового окна Entitiesдля задания параметров выделенной сущности модели
Attributes	Вызов диалогового окна Attributesдля задания параметров атрибутов выделенной сущности модели
Relationships	Вызов диалогового окна Relationshipsдля задания параметров отношения выделенной связи между сущностями модели
Key Groups	Вызов диалогового окна KeyGroupsдля задания параметров ключевых атрибутов выделенной сущности модели
Domain Dictionary	Вызов диалогового окна DomainDictionaryдля задания параметров доменов сущностей модели
Validation Rules	Вызов диалогового окна ValidationRulesдля задания параметров проверки корректности задания сущностей модели
Default Values	Вызов диалогового окна DefaultValuesдля задания параметров модели по умолчанию
UDP Dictionary	Вызов диалогового окна UserDefinedPropertyDictionaryдля задания словаря параметров модели, определяемых пользователем
Model Sources	Вызов диалогового окна ModelSourcesдля задания источников модели

Название подпункта	Назначение подпункта меню
Model Properties	Вызов диалогового окна ModelPropertiesдля задания данных о модели
Logical Mode	Отображение логической модели данных
Physical Model	Отображение физической модели данных

11. Изучите содержание меню **ModelMart**(таблица 1.7).

Таблица 1.7 – Содержание меню **ModelMart**

Название подпункта	Назначение подпункта меню
Open	Открыть хранилище моделей
Close	Закрыть хранилище моделей
Save	Сохранить модель в хранилище модели
Save As	Сохранить модель в хранилище модели под указанным именем
Lock	Закрыть модель, сохраненную в хранилище моделей, для редактирования
Merge	Разбить модель
Version	Версия модели
Change Control	Изменить параметры контроля состояния модели
Refresh	Перерисовать модель
Library	Вызов библиотеки моделей
Subject Areas	Работа с подмножеством модели
ModelMart Manager	Вызов менеджера хранилища моделей
Connection	Задание параметров соединения с хранилищем моделей
Session	Параметры сессии соединения с хранилищем моделей
модели	Задание параметров защиты модели
BP/ERSynchronizer	Задание параметров синхронизации моделей в средах BPwin и ERwin

12. Изучите содержание меню Tools (таблица 1.8).

Таблица 1.8 – Содержание меню **Tools**

Название подпункта меню	Назначение подпункта меню
Forward Engineer/Schema Generation	Вызов диалогового окна <Database>ServerSchemaGenerationReportдля выбора параметров генерации системного каталога для указанного целевого сервера СУБД и (или) сохранение сценария генерации системного каталога в формате

Название подпункта меню	Назначение подпункта меню
Reverse Engineer	Вызов диалогового окна ReverseEngineer- SetOptionsдля задания параметров генерации ERwinмодели на основе имеющегося SQLDDLили DL сценария или системного каталога СУБД
Add Model Source	Вызов диалогового окна мастера SourceModel, который позволяет задать сведенья об источниках (удобно при большом количестве источников)
Sync with Model Source	Вызов диалогового окна мастера SyncwithModelSource , который позволяет синхронизировать существующую модель ERwinc источниками модели при групповой работе над
Derive New Model	Вызов диалогового окна мастера DeriveNewModelкоторый позволяет создать новый вариант модели на основе же существующей
Split L/P Model	Разбиение логической и физической модели
Report Builder	Вызов диалогового окна мастера генерации отчета о модели
Data Browser	Вызов редактора отчетов о модели
Volumetrics	Вызов диалогового окна Volumetricsдля определения количественных характеристик параметров модели (размер таблиц и прочее)
Names	Вызов диалогового окна ModelNamingOptionsдля редактирования параметров имен, используемых в модели (максимальное количество символов и прочее)
Datatypes	Вызов диалогового окна ModelDatatypeOptionsдля использования стандартных типов файлов данных в модели
Add Ins	Вызов диалогового окна Add-InManager, которое позволяет подключить к ERwindополнительное программное обеспечение, например, ERwinExamineri др.

13. Изучите содержание меню Window (таблица 1.9).

Таблица 1.9 – Содержание меню **Window**

Название подпункта меню	Назначение подпункта меню
Cascade	Окна модели в рабочем пространство ERwinориентированы каскадом

Tile Horizontal	Окна модели в рабочем пространство ERwinориентированы с горизонтальной
Tile Vertical	Окна модели в рабочем пространство ERwinориентированы с вертикальной

14. Изучите содержание меню **Help**(таблица 1.10)

Таблица 1.10 – Содержание меню **Help**

Название подпункта	Назначение подпункта меню
Help Topics	Вызов окна ERwinOnlineсправочной системы
Tutorial	Вызов диалогового окна учебника по работе с ERwin
What's New	Вызов обзора новинок, включенных в текущую версию ERwinпо сравнению с предыдущей версией
How to Use Help	Вызов диалогового окна мастера HowtoUseHelp, в котором можно найти рекомендации по использованию справочной системы
About ERwin	Вызов диалогового окна справки о текущей версии ERwin

15. Изучите содержание элементов панели инструментов (таблица 1.11 – таблица 1.14).

Таблица 1.11 – Стандартная панель инструментов (Standard)

Кнопки	Назначение кнопок
	Создание, открытие, сохранение и печать модели
	Изменение уровня просмотра модели: уровень сущностей, уровень атрибутов и уровень определений
	Изменение масштаба просмотра модели
	Создание и переключение между подмножествами модели SubjectArea
	Отображение логической модели данных
	Отображение физической модели данных

Таблица 1.12 – Инструментарий (палитра инструментов - Toolbox)

Кнопки	Назначение кнопок
	Кнопка указателя (режим мыши) – в этом режиме можно устанавливать фокус на каком-либо объекте модели.

	Рекомендуется, всегда переходите в режим указателя мыши после редактирования объекта.
	Кнопка внесения сущности – для внесения сущности нужно щелкнуть левой кнопкой мыши по кнопке внесения сущностей и по свободному пространству на модели. Повторный щелчок приведет к внесению в модель новой сущности. Для редактирования сущностей или других объектов модели необходимо перейти в режим указателя
	Кнопка категории. Категория, или категориальная связь, - специальный тип связи между сущностями, которая будет рассмотрена ниже. Для установления категориальной связи нужно щелкнуть левой кнопкой мыши по кнопке категории, затем один раз щелкнуть по сущности - родовому предку, затем - по сущности-потомку
	Идентифицирующая связь между сущностями
	Связь между сущностями типа многие-ко-многим
	Неидентифицирующая связь между сущностями

Таблица 1.13 – Панель инструментов шрифт и цвет (FontandColor)

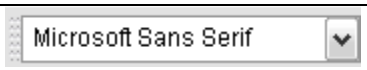
Кнопки	Назначение кнопок
	Выбор наименования шрифта
	Выбор размера шрифта
	Выбор стиля шрифта
	Выбор цвета символов
	Выбор цвета заливки
	Выбор цвета линий

Таблица 13.14 – Панель инструментов базы данных (Database)

Кнопки	Назначение кнопок
	Генерация схемы БД на основе логической модели данных (прямая задача)
	Реконструкция логической модели данных на основе схемы БД (обратная задача)
	Анализ адекватности результатов генерации схемы БД на основе

Кнопки	Назначение кнопок
	логической модели данных
	Выбор целевого сервера СУБД. Возможные варианты выбора показаны ниже на рисунке 1.6.

Для создания логических моделей данных в ERwinможно использовать две нотации: **IDEF1X** и **IE** (InformationEngineering). Методология **IDEF1X**была разработана для армии США и широко используется в государственных учреждениях США, финансовых и промышленных корпорациях. Методология **IE**, разработанная Мартином, Финкельштейном и другими авторами, используется преимущественно в промышленности. Переключение между нотациями можно с помощью выбора команд **Model►ModelProperties**. После того как откроется диалоговое окно **ModelProperties**(рисунок 1.7) следует раскрыть вкладку **Notation**и переключить радиокнопку во фрейме **LogicalNotation**в требуемое положение. В дальнейшем будем использовать нотацию **IDEF1X**, так как она получила большее признание.

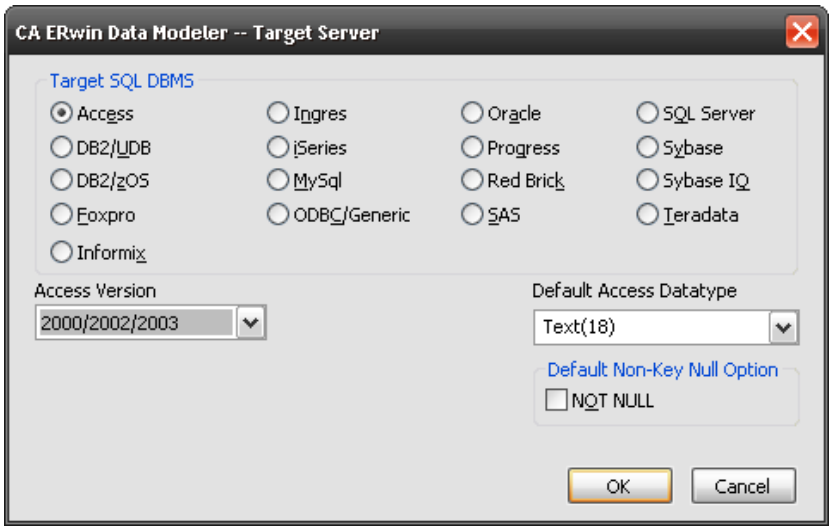


Рисунок 1.6–Выбор целевого сервера СУБД

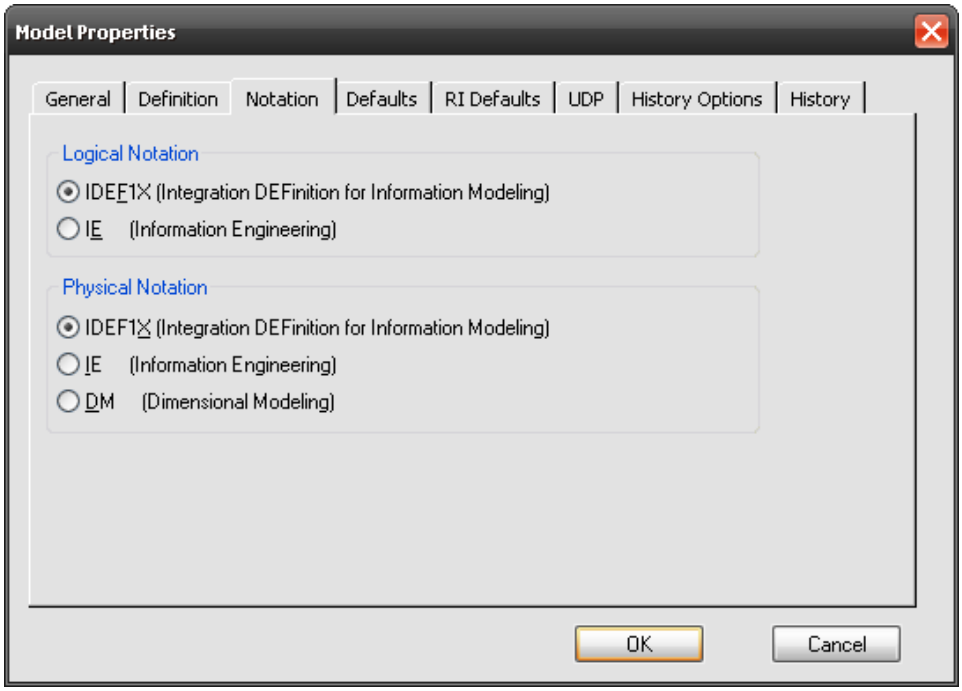


Рисунок 1.7 – Диалоговое окно ModelProperties

На рисунке 1.8 представлен вид фрагмента логической модели данных в нотации **IDEF1X**, а на рисунке 1.9представлен тот же фрагмент, но в нотации **IE**.

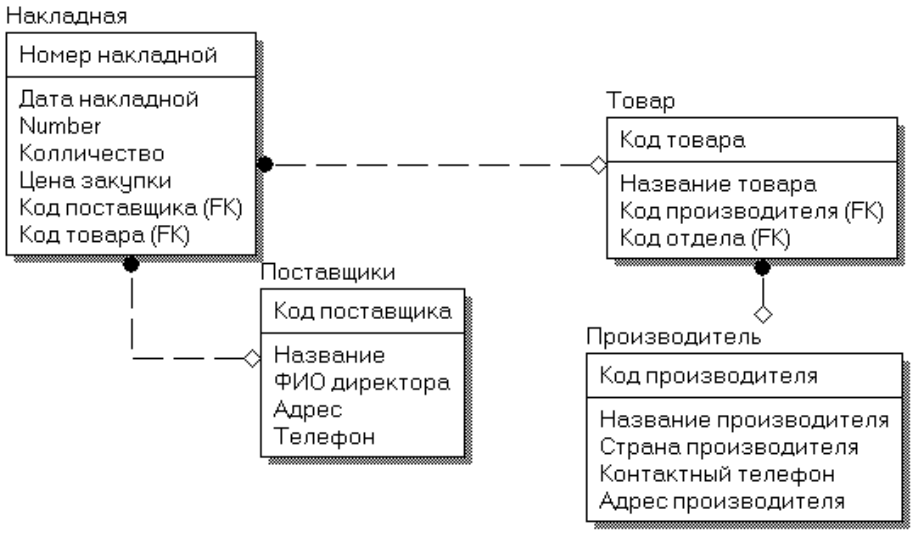


Рисунок 1.8 – Пример нотации IDEF1X

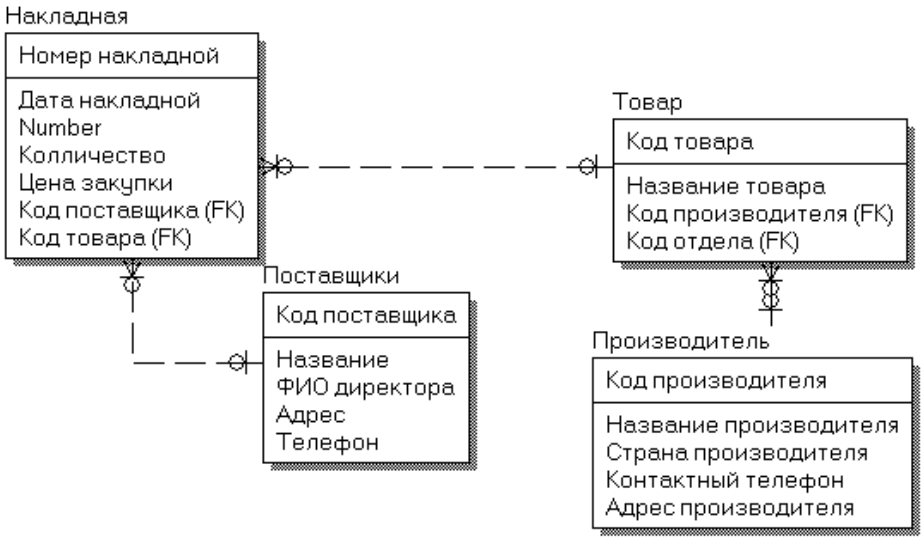


Рисунок 1.9 – Пример нотации IE

Как видно из рисунков 1.8 и 1.9, внешне две указанные нотации отличаются только обозначениями соответствующих связей между сущностями модели.

16. Выберите команды главного меню **View ►Toolbars**. Обратите внимание на то, как реагирует на это действие инструментальная панель (должно появиться ниспадающее меню). Убедитесь в том, что некоторые опции в этом меню отмечены галочками (являются активными). Измените перечень

активных опций в рассматриваемом меню и посмотрите, как это повлияет на

интерфейс среды Erwin.

Для того чтобы закрыть ниспадающее меню, кликните в любом месте на инструментальной панели. Потренируйтесь в этих действиях перед продолжением работы.

17. Потренируйтесь в переключении вида модели **Physical**и **Logical**. Переключитесь на физическую модель (выберите опцию **Physical**списка в инструментальной панели **ERwin**). Переключитесь на логическую модель (выберите опцию **Logical**списка в инструментальной панели **ERwin**). Обратите внимание, как влияют такие переключения на вид модели в окне редактирования. Например, логические названия (имена) заменяются физическими названиями (именами) и браузер независимых атрибутов переключается на браузер независимых столбцов (если браузер невидим, нажмите сочетание клавиш **Ctrl+B**).

18. Измените нотацию модели, перейдя от нотации **IDEF1X**к нотации **IE**. Измените нотацию модели, перейдя от нотации **IE**к нотации **IDEF1X**.

19. Закройте инструментальную среду **ERwin**. Для этого, используя главное меню **ERwin**, последовательно выполните следующие команды: **File►Exit.** и нажмите **OK**.

20. Выполните индивидуальное задание по пояснению назначения и выполнения настройки элементов интерфейса среды ERwindля указанного варианта.

Варианты заданий

Необходимо пояснить назначение и выполнить настройку элементов интерфейса среды Erwin для указанных вариантов (таблица 1.15). Номер варианта соответствует номеру рабочего места за компьютером.

Таблица 1.15 – Варианты индивидуальных заданий

Вариа	Содержание варианта
1	Главное меню ERwin. Пояснить назначение пунктов меню.
2	Меню File. Пояснить назначение пунктов меню. По указанию преподавателя выполните отдельные команды меню.
3	Меню Edit. Пояснить назначение пунктов меню. По указанию преподавателя выполните отдельные команды меню.
4	Меню View. Пояснить назначение пунктов меню. По указанию преподавателя выполните отдельные команды меню.
5	Меню Format. Пояснить назначение пунктов меню. По указанию преподавателя выполните отдельные команды меню.
6	Меню Model. Пояснить назначение пунктов меню. По указанию преподавателя выполните отдельные команды меню.
7	Меню ModelMart. Пояснить назначение пунктов меню. По указанию преподавателя выполните отдельные команды меню.
8	Меню Tools. Пояснить назначение пунктов меню. По указанию преподавателя выполните отдельные команды меню.
9	Меню Window. Пояснить назначение пунктов меню. По указанию преподавателя выполните отдельные команды меню.
10	Меню Help. Пояснить назначение пунктов меню. По указанию преподавателя выполните отдельные команды меню.
11	Стандартная панель инструментов (Standard). Пояснить назначение пунктов панель инструментов. По указанию преподавателя выполните действия с панелью инструментов.
12	Палитра инструментов (Toolbox). Пояснить назначение пунктов панель инструментов. По указанию преподавателя выполните действия с панелью инструментов.

Вариа	Содержание варианта
13	Панель инструментов шрифт и цвет (FontandColor). Пояснить назначение пунктов панель инструментов. По указанию преподавателя выполните действия с панелью инструментов
14	Панель инструментов базы данных (Database). Пояснить назначение пунктов панель инструментов. По указанию преподавателя выполните действия с панелью инструментов
15	Пояснить назначение браузера модели.
16	Уровни отображения модели. По указанию преподавателя откройте файл модели Emovies.er1 и измените и уровни ее

Содержание отчета и его форма

Отчет по лабораторной работе оформляется в виде текстового документа и должен включать:

1. Название лабораторной работы.
2. Цель лабораторной работы.
3. Формулировку индивидуального задания и результат его выполнения.
4. Краткие выводы по результатам выполнения лабораторной работы.

Вопросы для защиты работы

1. Что такое ERwinи зачем его используют?
2. Что такое физическая и логическая модель данных?
3. Как решаются в ERwinзадачи документирования модели?
4. Каковы свойства и общая характеристика системы меню ERwin?
5. ДайтехарактеристикуменюFile (Edit, View, Format, Model, Model Mart, Tools, Window, Help).
6. Каков порядок настройки интерфейса ERwin?
7. Каково назначение кнопок стандартной панели инструментов?
8. Каковы функции палитры инструментов?

Защита лабораторной работы

Оформленный, в соответствии с требованиями ГОСТов, отчет представляется студентом преподавателю для проверки и последующей защиты. Защита отчета по лабораторной работе производится студентом только индивидуально.

В ходе защите лабораторной работы студент отвечает на вопросы преподавателя (поясняет методику выполнения индивидуального задания, отвечает на контрольные вопросы и т.д.).

Отчет, оформленный с отступлениями от требований ГОСТов, небрежно и неаккуратно к защите не принимается.

Лабораторная работа № 2

Создание модели данных с помощью ERwin

Цель и содержание: привить студентам навыки анализа предметной области и создания модели данных в среде ERwin.

Формируемые компетенции: ПК-2, ПК-6. ПК-7.

Теоретическое обоснование

Как отмечалось в лабораторной работе № 1, среда ERwinимеет два уровня отображения модели данных: логический и физический.

На **логическом уровне** данные представляются так, как они выглядят в реальном мире. Объектами логического уровня являются сущности и атрибуты. Модель логического уровня является универсальной и не связана с конкретной базой данных.

На **физическом уровне** модель, напротив, зависит от конкретной реализации базы данных, выбираемой пользователем. Таким образом, одной логической модели может соответствовать несколько физических моделей. Кроме того, так как в физической модели речь идет уже о реально существующих в БД физических объектах, то обязательно должны быть определены типы данных атрибутов.

Фактически, 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.

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

Для непосредственной работы с элементами модели данных в среде **ERwin**имеется палитра инструментов (**ERwinToolbox**), представляющая собой «плавающее» окно (рисунок 2.1). При необходимости палитру инструментов можно убирать с экрана и вызывать нажатием комбинации клавиш **CTRL+T.**

Рисунок 2.1 – Палитра инструментов

В методологии **IDEF1X**различают зависимые и независимые сущности, которые могут быть ассоциированы связью различного типа.

Идентифицирующая связь - это связь между двумя сущностями, при которой экземпляр дочерней сущности идентифицируется через связь с материнской сущностью и не может существовать без нее.

Неидентифицирующая связь - это связь между независимыми сущностями.

Зависимая сущность изображается прямоугольником со скругленными углами. При установке идентифицирующей связи дочерняя сущность автоматически преобразуется в зависимую сущность.

Выполним анализ предметной области (обследование предприятия). В ходе обследования кратко рассмотрим предприятие ООО «КУРС», для которого в последующих лабораторных работах (лабораторные работы № 2- № 8), с помощью CASE-средства **ERwin**, будет разрабатываться информационная подсистема [2].

Направление деятельности ООО «КУРС» - обучение работе на компьютере. Для этого на предприятии оборудован специализированный класс (рисунок 14.2).

Рисунок 2.2 – Учебный класс - один из элементов предметной области

Предприятие расширяется и готовит еще несколько учебных классов.

Таким образом, в проектируемом приложении должна быть предусмотрена возможность одновременной работы с несколькими учебными классами (рисунок 2.3).

Рисунок 2.3 - Группа учебных классов

Учебный класс характеризуется следующими параметрами:

1. **Адрес.** Полный почтовый адрес класса. Пример адреса: «355026», Ставропольский край, г. Ставрополь, переулок Зоотехнический 15, оф.74, ООО «КУРС».
2. **Номер корпуса.** Часть классов может находиться в учебных комплексах, состоящих из нескольких корпусов. Номер корпуса может содержать

произвольные символы. Пример: «12-Б», «2/4».

3. **Номер комнаты.** Произвольная символьная информация, как и номер корпуса.

4. **Телефон.** Номер телефона, установленного в классе. Может содержать цифры и символы «-», «(», «)».

Учебный класс представляет собой совокупность учебных мест (рисунок 2.4).

Рисунок 14.4 – Состав учебного класса

Оптимальное число учебных мест в классе - около десяти. Это связанос принятой на предприятии индивидуально-групповой методикой обучения.

Кроме того, к учебному классу может относиться дополнительное оборудование, например сетевое - кабели, хабы, разъемы. Класс снабжен также хотя бы одним сетевым принтером, который может быть подключен к любому рабочему месту, но обслуживает весь класс.

Каждое учебное место представляет собой персональный компьютер. В классе учебному месту присвоен номер, который не обязательно должен быть цифровым. Цель его в том, чтобы преподаватели и учащиеся могли отличать одно учебное место от другого. Например, преподаватель при составлении расписания может сказать учащемуся: «В четверг вы занимаетесь на учебном месте 3А».

Кроме того, учебное место имеет несколько характеристик, связанных с тем, что компьютеры объединены в локальную сеть. Это могут быть:

1. **Имя рабочей станции.** Строковая характеристика, задается при установке операционной системы на компьютер.

2. **IP-адрес.** В качестве основного сетевого протокола на предприятии принят протокол **TCP/IP**, причем динамическое присвоение адресов прокси-сервером может не использоваться. В этом случае необходимо хранить информацию об IP-адресе учебного места. IP-адрес - строка вида «999.999.999.999», состоящая из трехзначных цифровых групп, разделенных точками.

Учебный класс на протяжении длительного времени комплектовался компьютерами, приобретавшимися в разное время для нужд других отделов предприятия и попадавшими в класс по мере их модернизации. Для поддержания учебных мест в рабочем состоянии имеется некоторый запас запасных частей и комплектующих (рисунок 14.5), которые по мере надобности устанавливаются вместо вышедших из строя узлов и деталей. Разрабатываемоеприложение должно обеспечивать учет комплектующих -

составных частей

учебных мест, а также учет запасных частей.

Комплектующие учебных мест (единицы оборудования) могут быть самыми разными. В принципе, в состав учебного места можно включить и части мебели - стол и стул. Таким образом, целесообразно рассматривать отдельно, как сущность, тип оборудования, который будет характеризовать каждый экземпляр.

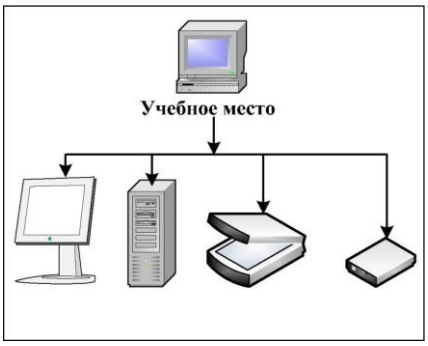


Рисунок 2.5 – Состав учебного места

Каждая единица оборудования, может обладать инвентарным номером и некоторым набором присущих данному типу заводских характеристик. Например, для монитора это будет фирма-изготовитель, размер экрана по диагонали, а для накопителя на жестком диске (винчестера) - фирма-изготовитель и емкость.

Единица оборудования может находиться в резерве, то есть не быть установленной в учебное место. Кроме того, она может быть вообще неисправна и, подлежать списанию, то есть обладает некоторой характеристикой «исправность».

Если единица оборудования установлена в учебное место, то необходимо хранить дату установки.

Непосредственно работой с учащимися в классе занимаются преподаватели. За каждым учебным классом закреплено несколько преподавателей, работающих посменно, так как занятия в учебном классе продолжаются около двенадцати часов Обычно в учебном классе находятся один или два преподавателя (рисунок 2.6)

Все учащиеся обучаются по независимым индивидуальным программам, которые составляются преподавателем в соответствии с пожеланиями учащегося. Существует несколько типов занятий:

1. **Теоретическое занятие.** В этом случае учащийся работает с одним из уроков компьютерного мультимедийного курса, выполняя по ходу урока различные задания. Встретив затруднение, он обращается к преподавателю, который находится здесь же. Преподаватель может сесть рядом с учащимся и подробнее объяснить непонятное место, показать необходимые действия.

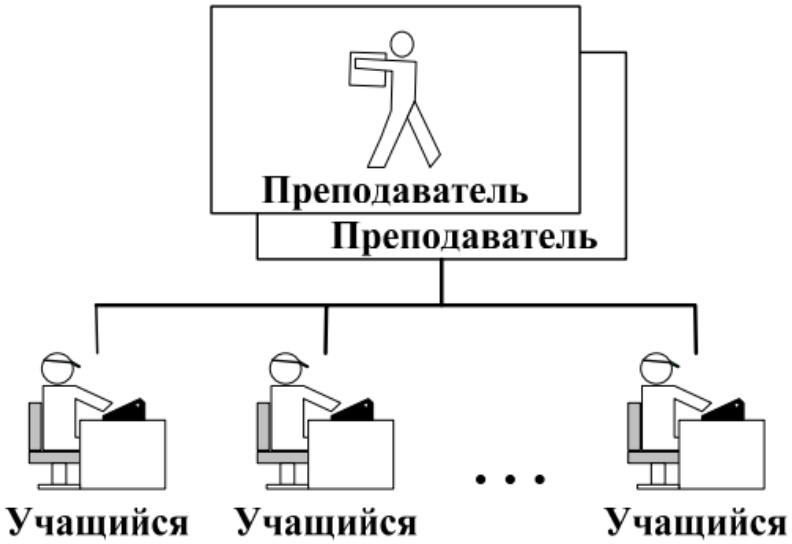


Рисунок 2.6 – Преподаватели и учащиеся

2. **Практическое занятие.** Преподаватель дает учащемуся задание, рассчитанное на целое занятие. Учащийся выполняет задание, обращаясь в случае затруднения к преподавателю.

3. **Контрольное занятие.** Учащийся выполняет выданное задание без всякой помощи со стороны преподавателя. За выполненное задание может ставиться оценка.

4. Занятие длится один академический час (45 минут).

На предприятии разработано несколько десятков типовых учебныхкурсов, каждый из которых состоит из 20 - 30 тем (рисунок 2.7). Это могут быть курсы общего характера, например «Работа в операционной системе WindowsXP» или «Работа в текстовом редакторе Word», а также специальные курсы, посвященные, к примеру, бухгалтерским или издательским программам. Каждая тема рассчитана на одно занятие. Темы могут объединяться в разделы по иерархическому принципу.

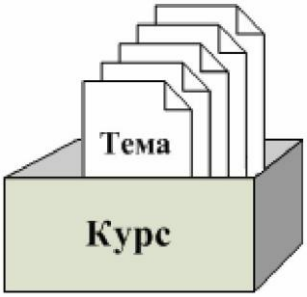


Рисунок 2.7 – Типовой учебный курс

Перед началом обучения для каждого учащегося один из преподавателей составляет индивидуальную программу обучения. В качестве основы индивидуальной программы принимается один или несколько типовых курсов, однако она может дополняться любыми темами, в зависимости от пожеланий учащегося и уровня его знаний. Например, учащийся сообщает, что не имеет

никаких знаний о компьютерах, но хотел бы научиться пользоваться бухгалтерской программой СУБД MicrosoftAccess. На основании имеющегося опыта преподаватель рекомендует сначала пройти общин курс знакомства с операционной системой (ОС) WindowsXP, составляющий 30 часов, затем по 25 часов знакомство с офисными приложениями MicrosoftExceli Word, а затем 30 часов собственно курса работы с СУБД MicrosoftAccess. Кроме того, учащийся изъявил желание научиться работать с электронной почтой и программами отсылки факсов. Преподаватель отвел на это 5 часов. Таким образом, общее количество занятий составит 115 часов (рисунок 2.8)

Учебный день класса разбивается на уроки, и хотя все учащиеся занимаются по индивидуальным программам, начало, и конец их занятий привязывается к началу и концу уроков.

На рисунке 2.9 показано размещение занятий по учебным дням.

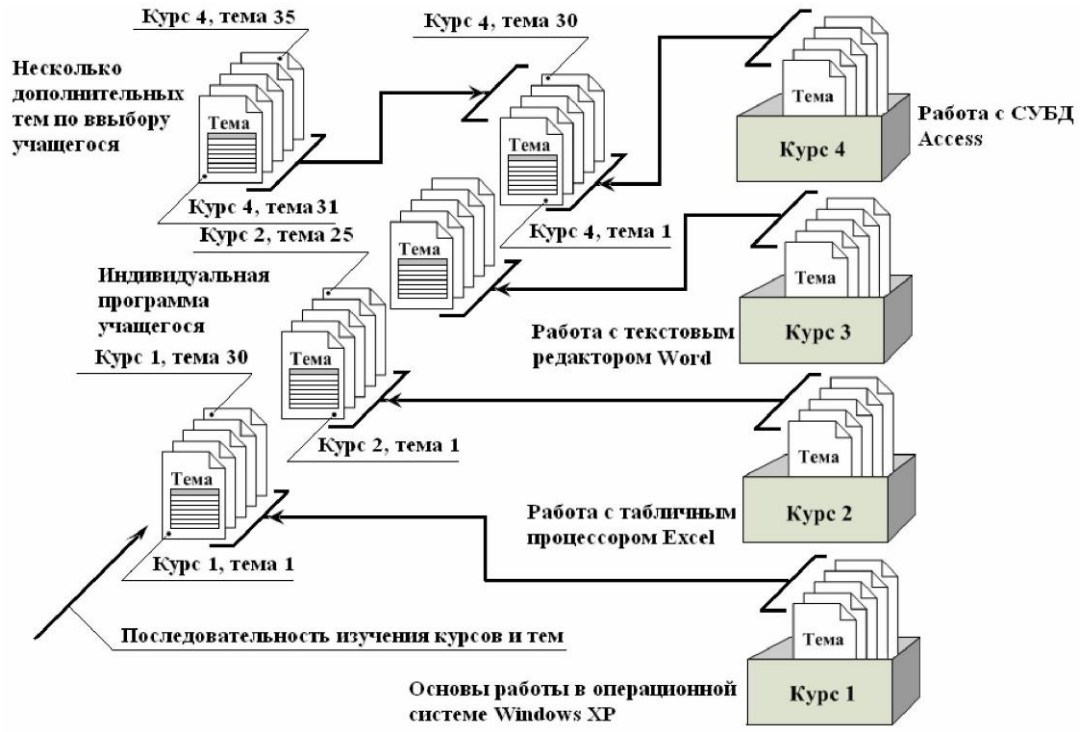


Рисунок 2.8 – Составление индивидуальной программы обучения

Планирование занятий проводится на сетке, столбцами которой являются учебные места, а строками - академические часы. Ячейка сетки соответствует занятию, которое может быть проведено на данном учебном месте во время данного академического часа.

В то же время занятие может быть, и не использовано, если учебное место в течение данного академического часа никем не занято. Используемые занятия отмечены на схеме установленными в ячейки темами занятий. Например, учащийся 1 занимается на первом учебном месте во время первого академического часа учебного дня 1, а учащийся 2 занимается на

второмучебном месте во время второго академического часа. Пустые ячейки на схеме означают неиспользуемые занятия.

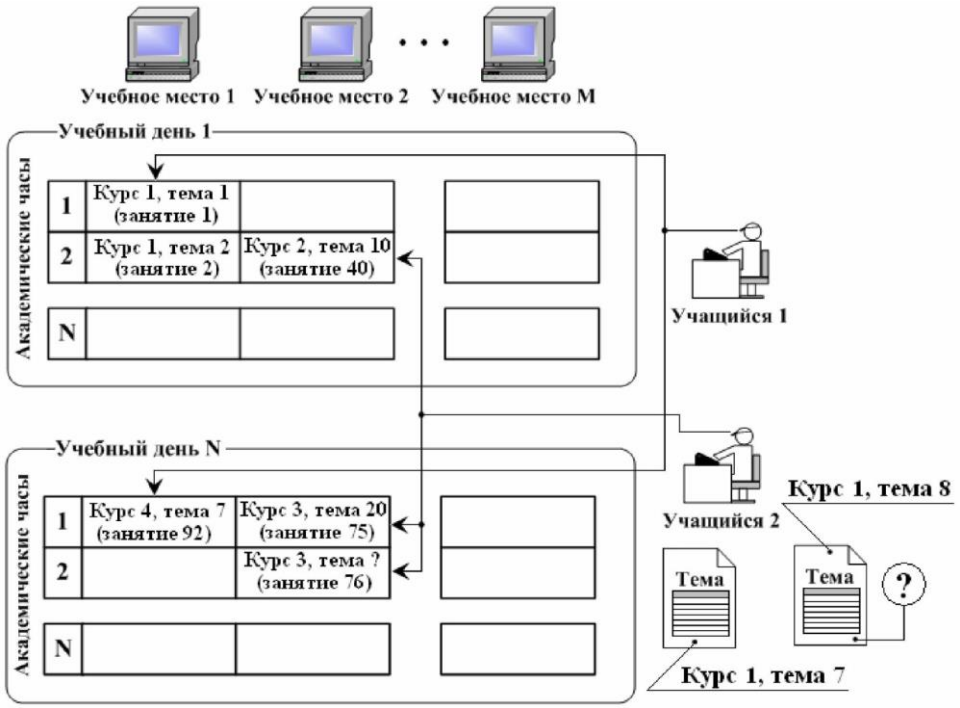


Рисунок 2.9 – Размещение занятий в планировочной сетке

Кроме того, возможны следующие ситуации:

1. Занятие зарезервировано за определенным учащимся, однако тема занятия не определена. На схеме это занятие 76 с неизвестной темой (Курс 3, Тема?**)** второго учащегося, которое должно проводиться на втором учебном месте во время академического часа 2 дня N.
2. Занятие спланировано, но не размещено в сетке уроков. Это занятие по курсу 1, теме 7 учащегося 2. На схеме оно показано стоящим вне сетки.
3. Учащемуся назначена тема для изучения (Курс 1, Тема 8), но занятие для нее не спланировано и не зарезервировано в сетке. На схеме это тема, находящаяся рядом с учащимся 2.

Рассмотрим процесс планирования занятий подробнее. Выше уже было сказано о том, что для каждого учащегося составляется индивидуальная программа занятий на основе базовых курсов. Однако, независимо от индивидуальной программы, учащемуся планируется определенное количество занятий, причем оно может быть больше, чем количество тем в программе. После этого темы расставляются по плановым занятиям, а плановые занятия привязываются к сетке академических часов.

На рисунке 2.10 показано, как темы индивидуальной программы расставляются по плановым занятиям, а плановые занятия устанавливаются в сетке. Таким образом, к академическим часам оказываются привязаны не темы индивидуальной программы, а плановые занятия - «ящички», в которые «вкладываются» темы.

2. Перенос занятия. Такая операция может производиться во многих случаях, например, неявка учащегося, неисправность учебного места (компьютера), отсутствие питания в электросети и т.п. При этом «ящичек» - плановое занятие перемещается в другую ячейку сетки (рисунок 2.12).

Такая схема работы удобна тем, что позволяет обрабатывать следующие ситуации:

1. Планирование занятий без указания их тематики, то есть резервирование учебных мест на определенное количество академических часов, например. В этом случае схема резервирования будет выглядеть так, как показано на рисунке 2.11.

3. Перемещение темы индивидуальной программы. Необходимость в подобной операции возникает, например, если учащийся не успевает усваивать материал. Предположим, учащемуся поставили в программу тридцать занятий по ознакомлению с операционной системой WindowsXP. Через три занятия стало ясно, что для усвоения материала в полной мере требуется, положим, еще одно занятие. По плану, после первых трех занятий знакомства с операционной системой WindowsXPстоит первое занятие курса 2 работы с офисным приложением MicrosoftExcel. Тогда, по согласованию с учащимся, преподаватель может вставить в программу дополнительное занятие по ОС WindowsXP. При этом плановые занятия («ящички») остаются на своих местах в сетке, а темы («листки») перемещаются каждый на один шаг в соседний «ящичек». Последняя тема вообще покидает сетку и становится вне плана (рисунок 2.13).

Такая операция показана на предыдущем рисунке. Обратная ситуация возникает, если учащийся усваивает материал быстрее, чем запланировано. В этом случае некоторые темы можно выбросить из плана, причем темы, следующие за ними, сместятся вперед. Последнее плановое занятие окажетсябез темы, как показано на рисунке 2.14, и преподаватель может либо совсем удалить его, сократив курс, либо поместить в него какую-нибудь новую тему.

После анализа предметной области, следующим этапом разработки является проектирование. Теперь, на основании обследования предприятия, составим модель данных.

Для проектирования базы данных необходим некоторый общий подход, который изначально гарантировал нам надежность хранения данных и простоту манипуляции ими. Такой подход назовем моделью данных, и необходимость его проиллюстрируем следующим примером.

Пусть нам необходимо хранить в базе данные о классах, учебных местах и комплектующих, их которых состоят учебные места. Для этого используем таблицу 2.14.

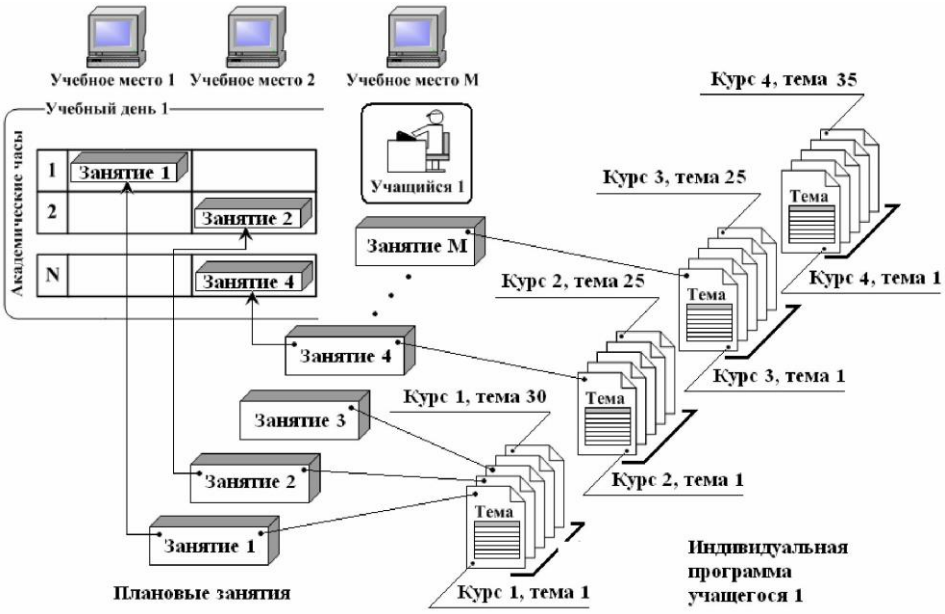


Рисунок 2.10 – Расстановка тем и плановых занятий

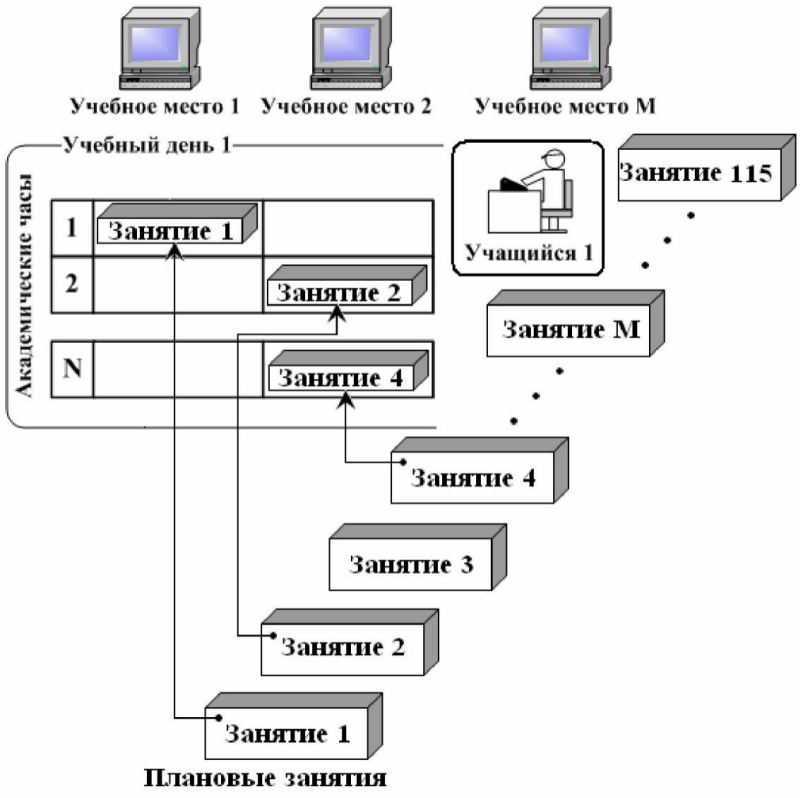


Рисунок 2.11 - Резервирование без назначения темы занятия

Рисунок 2.12 - Перенос занятия

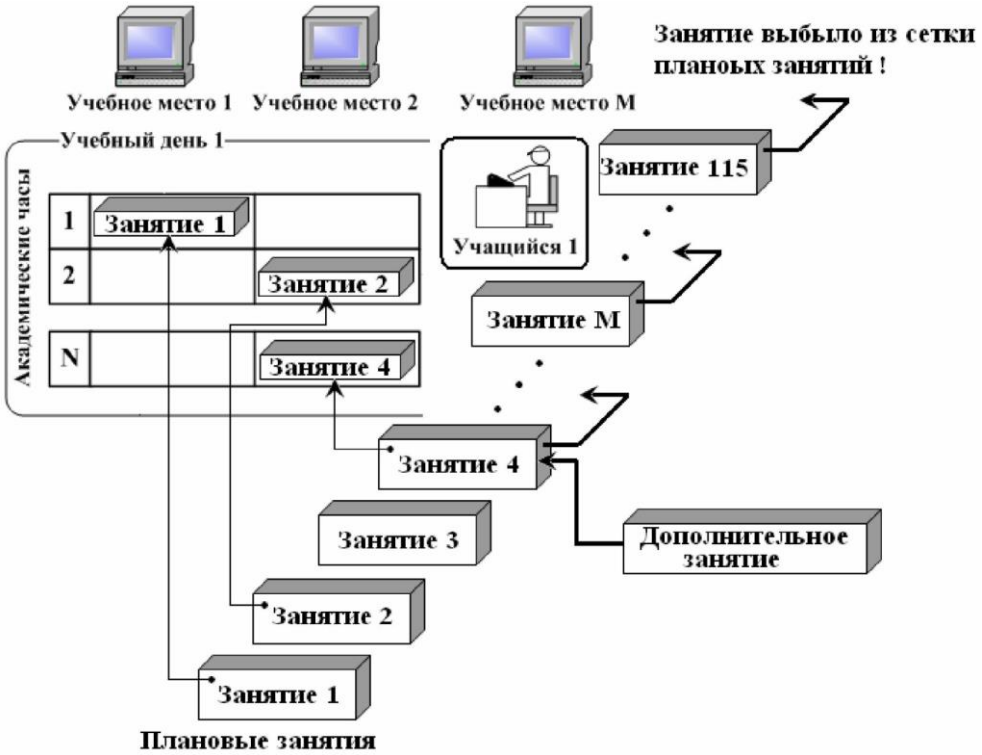


Рисунок 2.13 - Сдвиг тем занятий

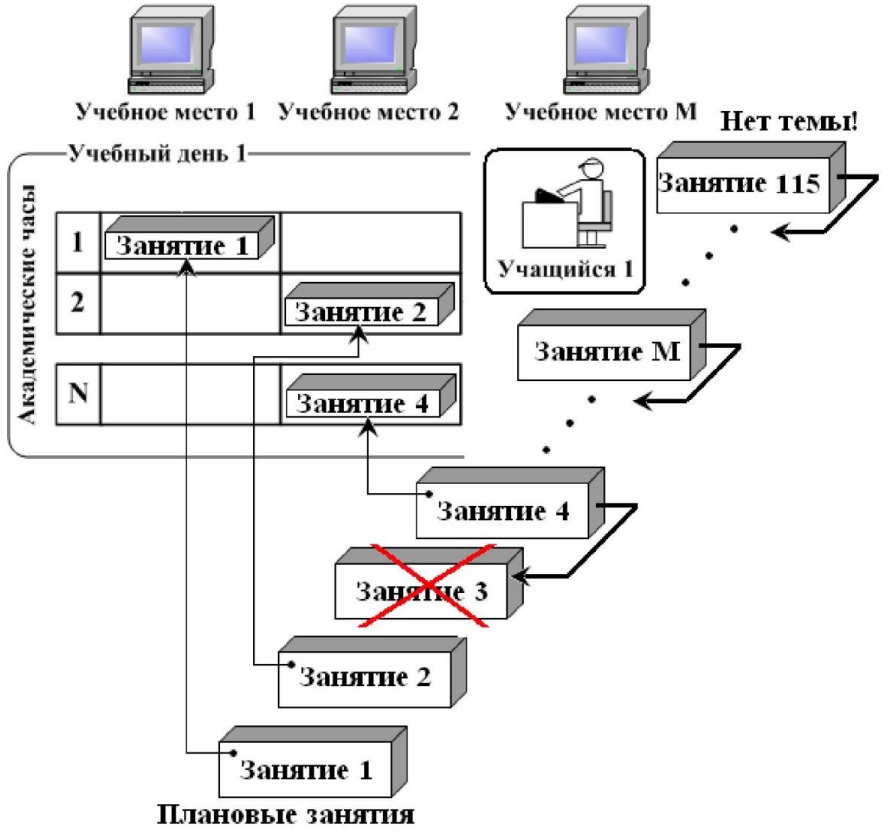


Таблица 21 – Пример таблицы

Адрес класса	Номер учебного	Инвентарный номер	Наименование комплектующей части	Дата установки
пер. Зоотехни-	1	124688	LCD 17" LG 1730 S	10. 11. 2004

Здесь у нас имеется одно учебное место, заданное номером 1, и находящееся в учебном классе по адресу «пер. Зоотехнический 15». Предположим, по каким-либо причинам в классе расформировывается место номер 1. Следовательно, для отражения в базе данных текущей ситуации нужно удалить запись с номером рабочего места 1:

пер. Зоотехни-	1	124688	LCD 17" LG 1730 S	10. 11. 2004

Однако при этом будет удалена информация о классе - адрес, а также и
о мониторе.

Ситуация, которая возникает в этом случае, называется аномалией уда-

ления. Удаление устаревших данных о рабочем месте приводит к удалению достоверных данных, то есть искажению информации.

Существует также аномалия добавления. Например, в системный блок компьютера рабочего места № 1 устанавливается еще одна плата расширения, например, сетевая плата. При этом кроме данных об этой плате необходимо будет вводить и адрес учебного класса.

Как видно из этого примера, модель данных должна обеспечивать независимость данных о классе, учебном месте и комплектующих.

Структура БД определяется положенной в ее основу моделью данных. Существуют различные модели баз данных, например, реляционная модель, иерархическая модель, сетевая модель и другие. Наиболее распространенной в настоящее время является реляционная модель данных.

Реляционная структура данных была разработана в конце 60-х годов рядом исследователей, из которых наиболее значимый вклад внес сотрудник фирмы IBMд-р Эдгар Кодд. При реляционном подходе данные представляются в виде двумерных таблиц - наиболее естественном для человек. В то же время, для обработки данных Кодд предложил использовать аппарат теории множеств - объединение, пересечение, разность, декартово произведение.

Напомним основные понятия реляционных баз данных.

Тип данных - это понятие имеет такой же смысл, как и в языках программирования. Все существующие современные базы данных поддерживают специальные типы данных, предназначенные для хранения данных целого типа, дробного с плавающей точкой, символов и строк, календарных дат. У многих серверов баз данных реализованы и другие типы, например, у сервера InterBaseимеется специальный тип данных для хранения крупных массивов бинарной информации (BLOB).

Домен - это потенциальное множество значений простого типа данных, он имеет сходство с подтипом данных в некоторых языках программирования. Домен определяется двумя элементами - типом данных и логическим выражением, которое применяется к данным. Если результат этого выражения равен значению «истина», то экземпляр данных принадлежит домену

Отношение - это двумерная таблица особого вида, состоящая из заголовка и тела

Заголовок - это фиксированное множество атрибутов, каждый из которых определен на каком-то домене, причем между атрибутами и определяющими доменами существует взаимно однозначное соответствие.

В таблице 14.2, содержащей информацию об оборудовании, заголовком является множество атрибутов «Код оборудования»,

«Инвентарный номер», «Наименование» и «Дата установки». Каждый из атрибутов определен на своем домене. Например, код оборудования может быть любым неотрицательным целым числом, то есть его домен представляет собой тип данных «целый», а логическое условие – n> 0. Заголовок является неизменным во времени, в отличие от тела отношения.

Таблица 14.2 - Оборудование и его атрибуты

Код оборудования	Инвентарный номер	Наименование	Дата установки
1	124688	LCD 17" LG 1730	10.11.2004
2	021638	JNC **ПО**	24.02.2004
3	340345	BTC	10.11.2004
4	459812	NEC 52x	10.11.2004
5	348120	Logitech	10.11.2004
6	184658	SEAGETE 60 Гб	10.11.2004
7	38578	ACORP 56 INT	12.12.2004
8	135578	Samsung FDD 3.5"	10.11.2004

Тело отношения - это совокупность **кортежей,** каждый из которых представляет собой пару «атрибут - значение».

Мощностью отношения называется число его кортежей, а **степенью отношения** - число атрибутов.

Степень отношения является для данного отношения величиной постоянной, тогда как мощность отношения изменяется во времени. Мощность отношения еще называют кардинальным числом.

В приведенной выше таблице 14.2 мощность отношения - 8, а степень отношения – 4. Приведенные выше понятия являются теоретическими и используются при разработке языковых средств и программных систем реляционных СУБД. В повседневной работе вместо них используются их неформальные эквиваленты:

а) отношение - таблица;

б) атрибут - колонка или поле;

в) кортеж - запись или строка.

Таким образом, степень отношения - это число колонок в таблице, а кардинальное число - количество строк.

Так как отношение представляет собой множество, а в классической теории множеств по определению множество не может содержать совпадающих элементов, то у отношения не может быть двух одинаковых кортежей. Поэтому для данного отношения всегда существует набор атрибутов, однозначно идентифицирующих кортеж. Такой набор атрибутов называется **ключом**.

Ключ должен удовлетворять следующим требованиям:

а) должен быть уникальным;

б) должен быть минимальным, то есть удаление любого атрибута из ключа ведет к нарушению уникальности.

Как правило, число атрибутов в ключе меньше степени отношения, однако, в крайнем случае, ключ может содержать все атрибуты, так как комбинация всех атрибутов удовлетворяет условию уникальности. Обычно отношение имеет несколько ключей. Из всех ключей отношения (их еще называют «возможными ключами») один выбирается в качестве первичного ключа. При выборе первичного ключа предпочтение обычно отдается ключу с наименьшим числом атрибутов. Нецелесообразно также использовать ключи с длинными строковыми значениями

На практике в качестве первичного ключа часто применяют специальный числовой атрибут - автоинкрементное поле, значение которого может генерироваться триггером или специальными средствами, определенными в механизме СУБД.

Описанные в данной главе основные понятия не относятся к какой-либо конкретной реализации базы данных, а являются общими для них всех. Таким образом, эти понятия являются основой определенной общей модели, которая называется **реляционной моделью данных.**

Основатель реляционного подхода Дейт установил, что реляционная модель состоит из трех частей:

а) структурной;

б) манипуляционной;

в) целостной.

В структурной части модели фиксируются отношения, как единственная структура данных, используемая в реляционной модели.

В манипуляционной части фиксируются два базовых механизма манипулирования реляционными базами - реляционная алгебра и реляционное исчисление.

Под целостной частью понимают некий механизм обеспечения неразрушаемости данных. Целостная часть заключает в себе два основных требования целостности реляционных баз данных - целостность сущностей и целостность по ссылкам.

Требование целостности сущностей состоит в том, что любой кортеж любого отношения должен быть отличим от любого другого кортежа этогоотношения, то есть другими словами, любое отношение должно обладать

первичным ключом. Это требование должно выполняться, если

выполняются

базовые свойства отношений.

Требование целостности по ссылкам - это ограничение, налагаемое на реляционную базу данных вследствие того, что отношения в базе данных ассоциированы друг с другом **связями.**

Объекты реального мира могут представляться в базе данных несколькими отношениями. Например, в описанном выше примере анализа предметной области такими отношениями могут являться отношения «Учебное место» и «Единица оборудования».

На схеме, изображенной на рисунке 2.15, отношение «Единица оборудования» содержит атрибут «Код учебного места». Понятно, что этот атрибут не является свойством объекта «Единица оборудования», а служит для воспроизведения полного экземпляра объекта «Учебное место».

Рисунок 2.15 - Внешний ключ

В отношении «Учебное место» атрибут «Код учебного места» является первичным ключом. В отношении «Единица оборудования» значение атрибута «Код учебного места» должно соответствовать значению первичногоключа в одном из кортежей. Такой атрибут называется внешним ключом.

Требование целостности по ссылкам состоит в том, что для каждого значения внешнего, ключа, появляющегося в ссылающемся отношении, в отношении, на которое ведет ссылка, должен найтись кортеж с таким же значением первичного ключа, либо значение внешнего ключа должно быть неопределенным (то есть ни на что не указывать). Для нашего примера это означает, что если для единицы оборудования указан номер отдела, то этот отдел должен существовать,

В связи с этим существует три подхода при удалении кортежа, на который имеются ссылки:

1. Запрет. СУБД не допускает удаление кортежа, пока существуют ссылающиеся на него кортежи. В нашем примере это значит, что СУБД не позволит удалить запись таблицы «Учебное место» с кодом 6, пока существуют кортежи с таким значением кода в отношении «Единица оборудования».

2. Установка неопределенного значения.

3. Удаление кортежей, ссылающихся на данный кортеж. Этот вид действий называется каскадным удалением.

Выше мы выяснили, что для исключения аномалий обновления данных необходимо, чтобы схема БД была построена из отношений, данные которых независимы.

Для решения этой задачи Кодд предложил аппарат нормализации отношений. Нормализация - это процесс проверки и преобразования отношений и атрибутов таким образом, чтобы устранить аномалии хранения данных. Процесс нормализации сводится к последовательному приведению структуры данных к нормальным формам, каждая из которых представляет собой совокупность формализованных требований к организации данных.

Известно шесть нормальных форм:

а) первая нормальная форма (1NF);

б) вторая нормальная форма (2NF),

в) третья нормальная форма (3NF),

г) нормальная форма Бонса-Кодда (усиленная 3NF, BCNF),

д) четвертая нормальная форма (4NF);

е) пятая нормальная форма (5NF).

Нормальные формы основаны на понятии функциональной зависимости.

Функциональная зависимость (ФЗ) - это такая связь между атрибутами**В**и **А** одного и того же отношения, когда каждому значению **А** соответствует только одно значение **В**.

Атрибут**А**называется детерминантом. Детерминанты могут быть составными, то есть могут представлять собой не единичные атрибуты, а группы, состоящие из двух и более атрибутов.

Существует три вида функциональных зависимостей: полная, частичная и транзитивная.

Полной функциональной зависимостью называется такая зависимость, в которой атрибут**В**не зависит от подмножества атрибутов детерминанта **А**.

Частичными называют функциональные зависимости, в которых атрибут**В**зависит только от части составного детерминанта **А**.

Транзитивными называют функциональные зависимости, в которых для атрибутов**А**, **В** и **С** выполняются условия **А** - **В** и **В** - **С**, но отсутствует

обратная зависимость.

Аномалии порождают частичные и транзитивные функциональные зависимости. Поэтому, нормализация отношений - это преобразование исходного отношения к их совокупности так, что каждое из полученных отношений содержит только одну полную функциональную зависимость.

Теоретически процесс нормализации должен начинаться с универсального отношения, содержащего все атрибуты предметной области. Его можно представить, как одну огромную таблицу. Понятно, что данные в такой таблице будут многократно повторяться.

Каждая нормальная форма полностью удовлетворяет требованиям предыдущей нормальной формы и выдвигает некоторые дополнительные требования.

Первая нормальная форма (1NF) Отношение находится в первой нормальной форме тогда и только тогда, когда все его атрибуты содержат атомарные значения. Наиболее распространенным нарушением этого требования является атрибут типа «адрес». Его значение содержит название города, области, улицы, которые являются различными по смыслу, и в силу требований первой нормальной формы должны быть разнесены в различные атрибуты.

Кроме того, отношения не должны содержать повторяющихся групп. Например, у приведенного на рисунке 2.16 отношения «Класс» может потребоваться хранение не одного, а двух номеров телефонов.

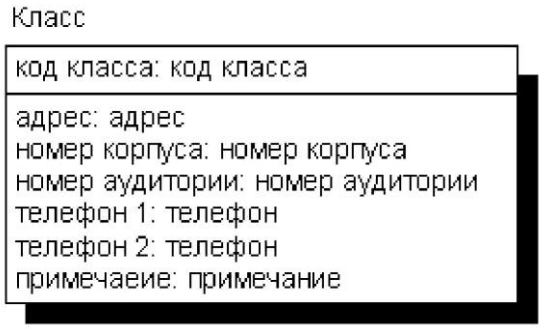


Рисунок 2.16 - Пример несоответствия 1NF

Добавление еще одного атрибута не решает проблемы, ведь если появится третий телефон, информацию об этом негде будет хранить

Вторая нормальная форма (2NF). Для соответствия второй нормальной форме отношение не должно содержать частичных функциональных зависимостей, то есть каждый неключевой атрибут должен зависеть только от всего составного ключа, а не от его частей. Например, в приведенном ниже отношении наименование комплектующей части зависит только от инвентарногономера, но не от адреса класса и номера учебного места.

Адрес класса	Номер учебного	Инвентарный номер	Наименование комплектующей части	Дата установки
пер. Зоотехни-	1	124688	Монитор SyncMaster	10. 11. 2004

Третья нормальная форма (3NF) требует исключения транзитивных зависимостей. На практике это означает, что не должно быть зависимостей между неключевыми атрибутами одного отношения.

Нормальная форма Байса-Кодда (BCNF) требует чтобы каждый детерминант являлся возможным ключом.

Четвертая нормальная форма (4NF) В отношении не должно быть многозначных зависимостей между атрибутами. В следующем примере преподаватель обучает нескольких студентов по различным предметным курсам. По одному курсу могут учиться несколько студентов, каждый студент может обучаться по нескольким курсам. Таким образом, между атрибутами «Курс обучения» и «Студент» имеет место многозначная зависимость, которая может стать причиной аномалии добавление в базу данных нового студента потребует добавления нескольких записей, по числу изучаемых им курсов (рисунок 2.17).

Рисунок 2.17 - Пример несоответствия 4NF

Для приведения к четвертой нормальной форме необходимо выполнить декомпозицию данного отношения на два, и разнести атрибуты «Курс обучения» и «Студент» в разные отношения.

Пятая нормальная форма на практике не используется.

Соблюдение требований нормализации зачастую ведет к понижению производительности конкретной информационной системы. После нормализации растет число отношений, поэтому при выполнении запросов к такой БД требуется больше объединений (JOIN), поглощающих ресурсы системы. В целях повышения производительности при переходе на физический уровень производится сознательный отход от нормальных форм, «денормализа-ция». Процесс денормализации не имеет четко сформулированных правил, конкретные решения зависят от специфики

задачи и предметной области.

Реляционная модель сама по себе неудобна для работы при проектировании базы данных. Вместо нее используются различные семантические модели, наиболее распространенной из которых является так называемая ER-модель (от английского «entity-relation») или модель «сущность-связь».

Модель была предложена Питером Ченом в 1976 г. и представляет собой ряд графических диаграмм, включающих небольшое число разнородных компонентов. Чен предложил интерпретировать реляционные структуры данных, как набор сущностей и связей между ними. Работа Чека, а также работы Хаммера и Мак-Леода позволили разработать модель «сущность-связь» или ER-модель. Основными компонентами ER-модели являются сущность, связь и атрибут.

Сущность - это любой реальный или представляемый объект, информацию о котором необходимо хранить в базе данных. На диаграммах ER-модели сущность обычно изображают прямоугольником, внутри которого проставляется имя сущности. Сущность должна иметь наименование с четким смысловым значением, и именоваться существительным в единственном числе. Необходимо различать тип сущности и экземпляр сущности. Имя сущности присваивается типу, а не экземпляру Экземпляр сущности - это конкретная вещь в наборе однородных предметов, идей, событий и т. д.

На рисунке 2.18 изображена сущность «Учащийся». «Учащийся» является названием типа сущности, а экземплярами сущности будут конкретные учащиеся - Иванов, Петров, Сидоров и т.д. Экземпляр сущности примерно соответствует кортежу в реляционной модели, поэтому каждый экземпляр сущности должен быть отличим от любого другого экземпляра той же сущности.

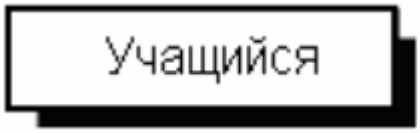


Рисунок 2.18 - Сущность «Учащийся»

Связь - это ассоциация двух или более сущностей. На графической диаграмме связь изображается линией, соединяющей две сущности. Связь всегда существует только между двумя разными сущностями или между сущностью и ею самой. Последняя связь называется рекурсивной.

Существуют различные стандартные методологии разработки диаграмм «сущность-связь», например IDEF1X, IE, DM, в каждой из которых принята своя нотация для изображения сущностей и связей.

В общем случае над каждым концом связи проставляют степень связи

(1 или букву М, обозначающую слово «много») В зависимости от степени связи различают следующие ее виды:

1. Один к одному. Каждому экземпляру сущности А соответствует 0 и1 экземпляр сущности В (рисунок 2.19).

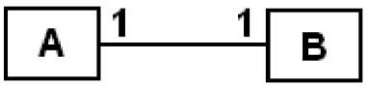


Рисунок 2.19 - Связь «Один к одному»

2. **Один ко многим**. Одному представителю сущности**А**соответствуют 0, 1 или несколько представителей сущности **В** (рисунок2.20)

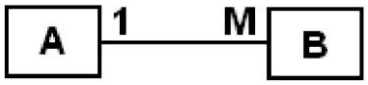


Рисунок 2.20 - Связь «Один ко многим»

3. **Многие ко многим**. Каждой сущности понятия А соответствует несколько (или 0) сущностей понятия В и наоборот (рисунок 2.21).

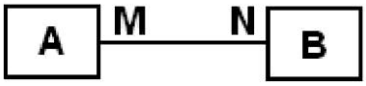


Рисунок 2.21 - Связь «Многие ко многим»

Помимо степени связи (называемой еще ассоциативностью) у связей имеются и другие свойства, например время существования (постоянные, долговременные и кратковременные связи) и избирательность (обязательные, необязательные, возможные и условные связи).

Атрибут (свойство) - именованная характеристика сущности. Он представляет собой элементарную единицу структуры понятия, которая служит для уточнения, идентификации, классификации, числовой характеристики или выражения состояния сущности.

Например, атрибуты сущности «Учащийся» - код, фамилия, имя, адрес, возраст, пол и т.п.

Набор атрибутов сущности в принципе бесконечен, при разработке информационной системы он зависит от потребностей пользователя и решаемых им задач. При составлении инфологической модели стараются обычно, как можно более полно описать свойства понятий, учитывая не только текущие задачи, но и возможные будущие потребности пользователей.

На ER-диаграммах атрибуты отображаются по-разному в зависимости от разновидности модели. Например, имена атрибутов могут быть записаны

малыми буквами внутри прямоугольника, изображающего сущность, под именем сущности. В других случаях атрибуты могут быть вынесены за прямоугольник сущности и связаны с ним линиями.

Ключевые атрибуты, как правило, выделяют. Иногда атрибут помещают в овал.

К более сложным элементам ER-диаграмм относятся **подтипы** сущностей. Сущность может быть расщеплена на два или более взаимно исключающих подтипа, каждый из которых включает общие атрибуты и/или связи. Эти общие атрибуты и/или связи явно определяются один раз на более высоком уровне. В подтипах могут определяться собственные атрибуты и/или связи. В принципе подтипизация может продолжаться и на более низких уровнях, но опыт показывает, что в большинстве случаев оказывается достаточно двух-трех уровней.

Сущность, на основе которой определяются подтипы, называется **супертипом**. Подтипы должны образовывать полное множество, то есть любой экземпляр супертипа должен относиться к некоторому подтипу. К примеру, для сущности «Единица оборудования» можно определить подтипы «Монитор», «Сетевая плата» и т. д.

При разработке ER-диаграммы не требуется декомпозиции универсального отношения и приведения его ко второй и третьей нормальной форме Определение сущностей - это, по сути, приведение универсального отношения к 3NF.

Теперь построим информационную модель этого предприятия при помощи программы ERwin. В процессе создания модели она пройдет несколько уровней детализации. Для каждого из этих уровней мы создадим отдельное хранимое отображение.

Аппаратура и программное обеспечение.

Аппаратура. Для выполнения лабораторной работы необходимы:

- персональный суперкомпьютер i7-5930K/ 8D8192D4-2133/ K2200/ 2K40M;

- рабочие станции: монитор 29 + клавиатура + мышь. ПК MDT750/ i75930K/ 4D8192D42133";

- аппаратный сетевой обучающий комплекс мод.VDCN-1502.

Программное обеспечение. Для выполнения лабораторной работы необходима операционная система WINDOWS 7/8/10 и CASE-средство ERwin7.3.

Указания по технике безопасности

При выполнении лабораторной работы **запрещается**:

1. Самостоятельно производить ремонт персонального компьютера, атак же, установку и удаление имеющегося программного обеспечения.

2. Нарушать общепринятые правила техники безопасности при работе с электрооборудованием, в частности, касаться электрических розеток металлическими предметами и т.д.

3. Принимать пищу, напитки и сорить на рабочем месте пользователя персонального компьютера.

В случае неисправности персонального компьютера необходимо **немедленно** сообщить об этом обслуживающему персоналу лаборатории (системному администратору, оператору).

Методика и порядок выполнения работы

1. Запустите программу Erwin и создайте модель (см. лабораторную работу № 1).

2. В диалоговом окне программы (рисунок 2.23).Выберите тип создаваемой модели **Logical/Physical**и целевой сервер базыданных (БД) **Access**(см. рисунок 2.23) и нажмите **ОК**.

3.Откроется основное окно программы **ERwin**(рисунок 2.24).

В верхней части окна находится титульная строка, в которой указано название программы, наименование модели, наименование подмножества(**SubjectArea**) и хранимого отображения (**StoredDisplay**).

По умолчанию, при запуске программы исходная модель получает имя **Model 1**, ее рабочая область - имя **MainSubjectArea**(главное подмножество объектов) и сохраняется, как хранимое Отображение **Display1**.

4. Начнем проектирование логической модели с уровня сущностей. Выберите пункт главного меню **Format► StoredDisplay**. На экране появится окно редактора хранимых отображений (рисунок 2.25).

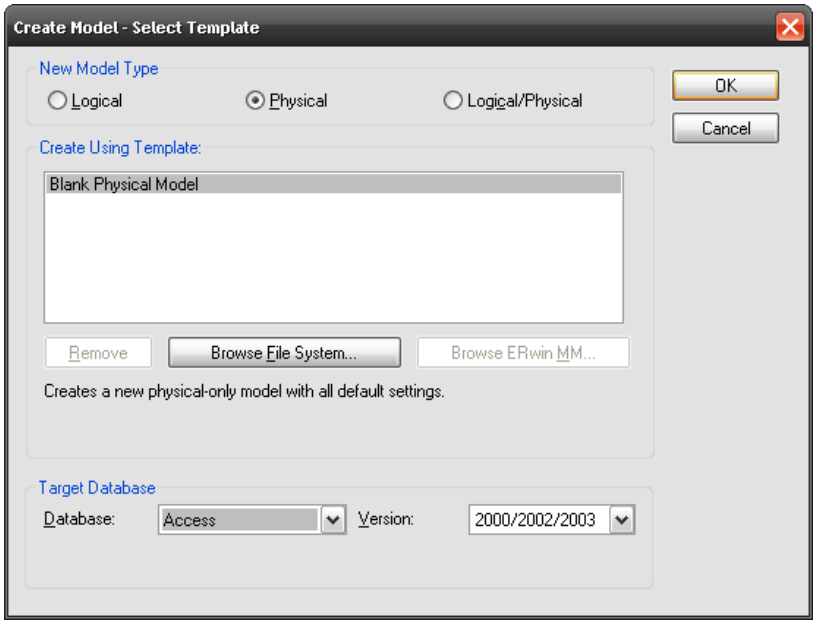


Рисунок 2.23 - Второе диалоговое окно программы ERwin.

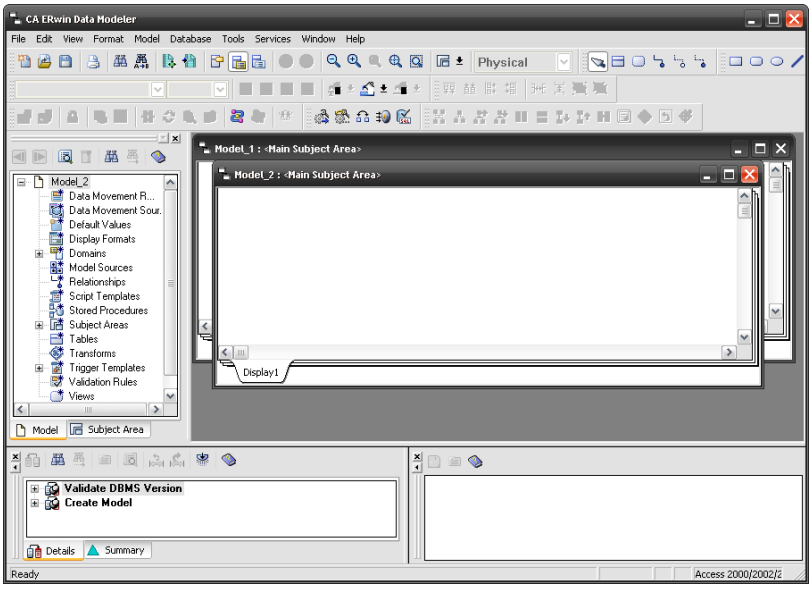


Рисунок 2.24 - Основное окно программы ERwin

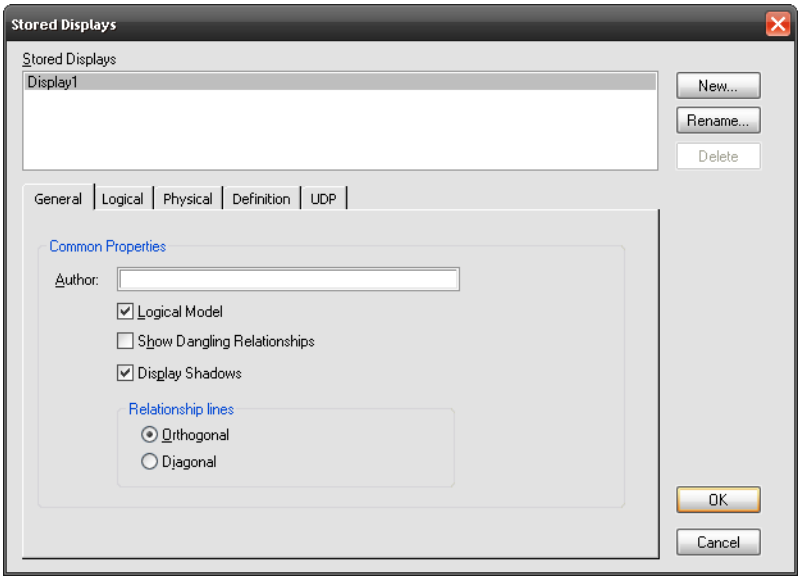


Рисунок 2.25 - Редактор хранимых отображений

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.

Вкладка «General» - общие свойства отображения. В поле «Author» (автор) введите с клавиатуры свое имя. Это имя автора хранимого отображения. Включите «галочки»:

- **LogicalModel**(логическая модель) - хранимое отображение будет использоваться только на логическом уровне модели.

- **DisplayShadows**(покалывать тени) - прямоугольники сущности,будут изображаться на экране с «тенью». Размер тени в пикселях задается вменю **Option ►Preferences**на странице **DisplayOptions**.

Внизу, в рамке **Relationshipslines**(линии связей) устанавливается способ изображения линий связи между сущностями. В режиме **Orthogonal**(ортогональный) линии связей прокладываются отрезками, параллельными осям XY, в диагональном режиме линии связей могут проводиться под произвольным углом..

По умолчанию, на ER-диаграммах изображаются связи в ортогональном режиме, оставьте эту установку без изменений.

5. Перейдите на закладку **Logical**(логический уровень), изображенную на рисунок 2.26.

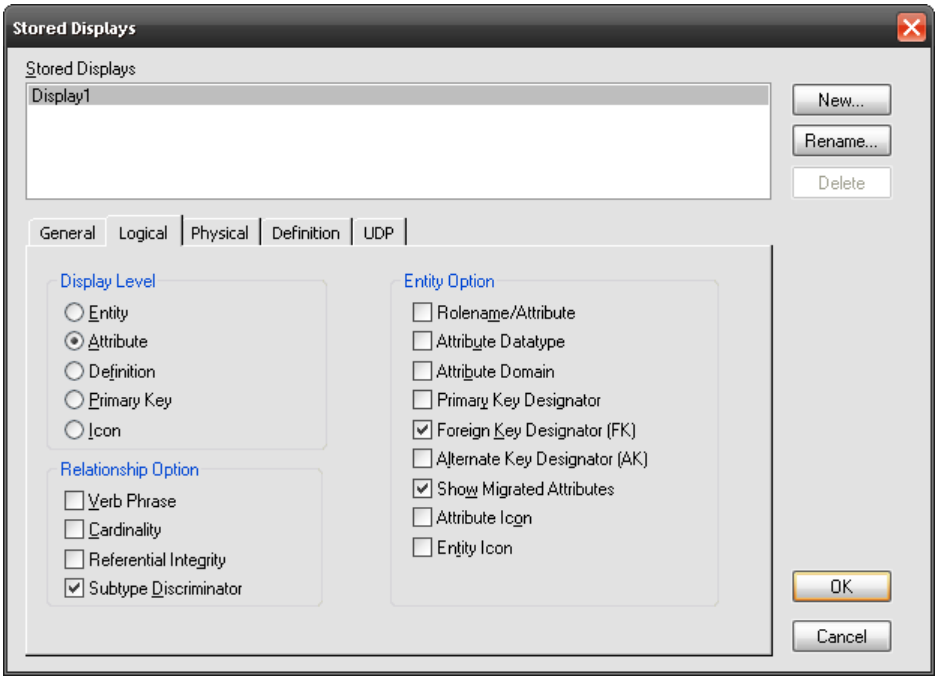


Рисунок 2.26 - Закладка «Логический уровень»

6. Установите переключатель **DisplayLevel**(уровень отображения) в положение **Entity**(сущность). Тем самым задается, что на экране будут показаны только сущности, без атрибутов.

7. Установите флажок **VerbPhrase**(глагольная фраза), чтобы на схеме отображались глагольные фразы, именующие связи между сущностями.

Остальные флажки на данной странице оставьте без изменений.

9. Перейдите на закладку **Physical**(физический уровень), изображенную на рисунок 2.27.

На вкладке «Physical» задаются свойства отображения физического уровня модели. Так как пока мы будем заниматься логическим уровнем, оставьте эти установки без изменений (см. рисунок 2.27).

9. Перейдите на закладку **Definition**(Описание), изображенную на рисунок 14.28. На вкладке **Definition**можно ввести описание хранимого отображения.

Текст вводится в мемо-поле, но при необходимости можно перейти в

полноэкранный встроенный тестовый редактор **ERwin**, нажав кнопку  При вводе и редактировании текста поддерживаются функции копирования, вырезания и вставки с использованием буфера обмена.

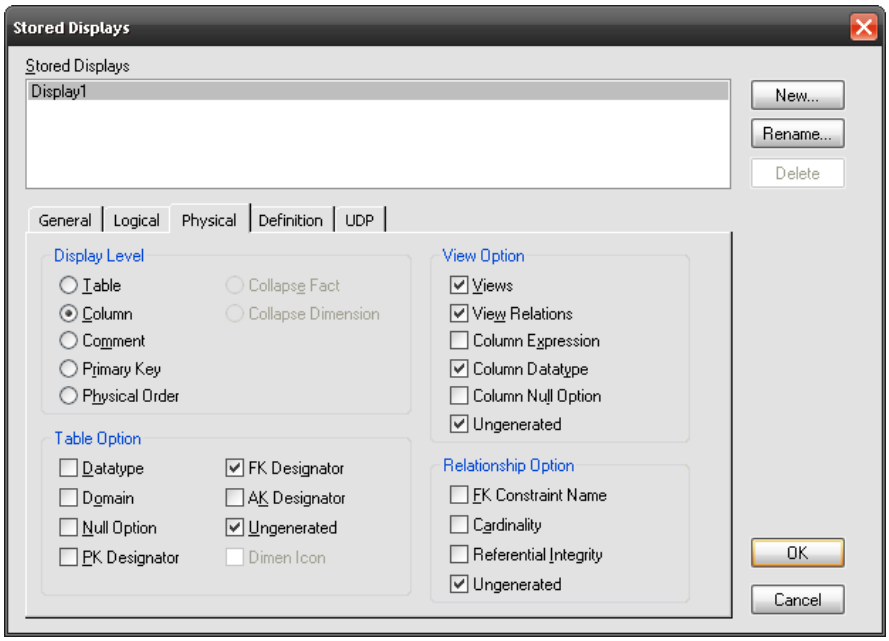


Рисунок 2.27 - Закладка «Физический уровень»

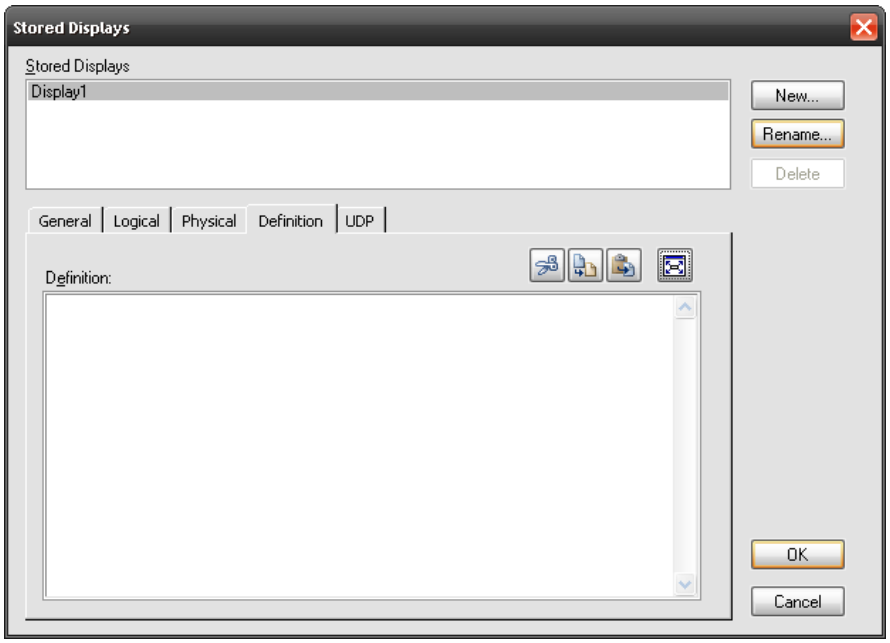


Рисунок 2.28 - Закладка Definition

10. Перейдите на закладку **UDP** (Параметры, определяемые пользователем), изображенную на рисунке 2.29. На вкладке **UDP** реализован механизм «свойств, определяемых пользователем». Суть этого механизма в том, что каждому из объектов модели может быть присвоено любое число дополнительных свойств, которые создает сам пользователь.

Эти свойства могут быть различных типов: числовой - целый и дробный, текст, дата и список. К значениям свойств можно обращаться при генерации отчетов и сценариев. Применение пользовательских свойств мы рассмотрим позднее.

11. То, что мы задавали с помощью редактора отображений, относитсяк отображению «Display 1». Переименуйте его, нажав на кнопку «Rename». В

появившемся диалоге введите имя отображения «Уровень сущностей» (рисунок 2.30).

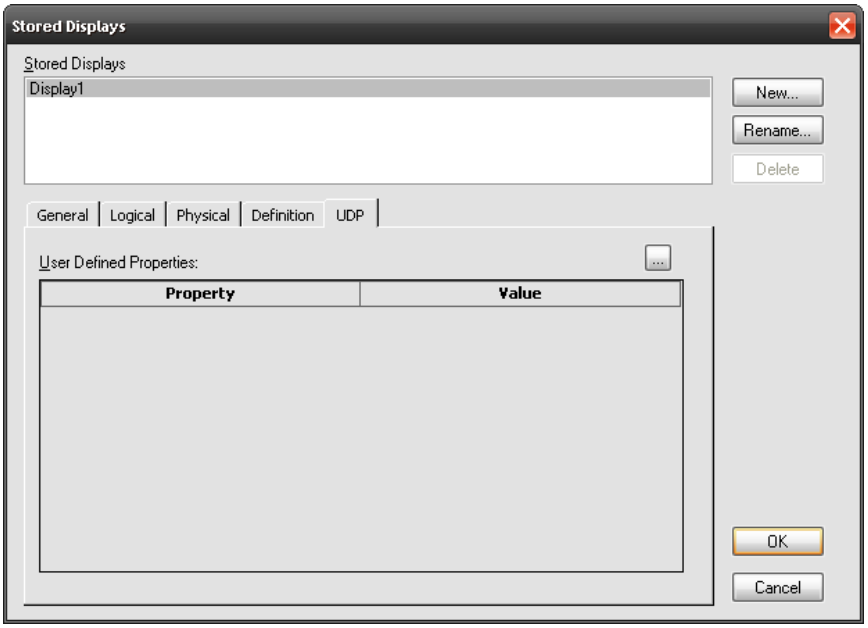


Рисунок 2.29 - Закладка **UDP**

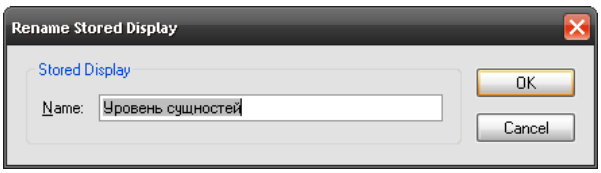


Рисунок 2.30 – Переименование в редакторе отображений

Нажмите кнопку **ОК**и еще раз **ОК**. Это название появится в титульной строке, а также на закладке в нижней части экрана.

12. Снова выберите пункт главного меню **Edit►** **StoredDisplay**и создайте еще одно хранимое отображение под названием «Уровень атрибутов».

Для этого нажмите кнопку **New**рядом со списком отображений в верхней части редактора и введите это название с клавиатуры (рисунок 2.31).

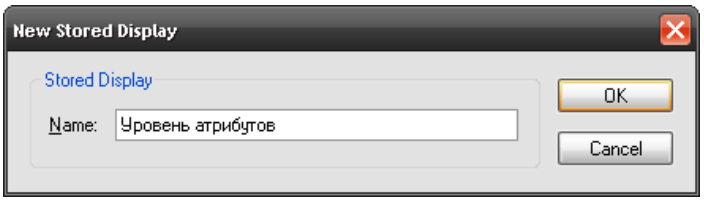


Рисунок 2.31 – Переименование в редакторе отображений

13. Выделите отображение «Уровень атрибутов» в списке и на странице **Logical**установите переключатели, как показано на рисунке 2.32.

В результате появятся две закладки хранимых отображений (рисунок 2.33).

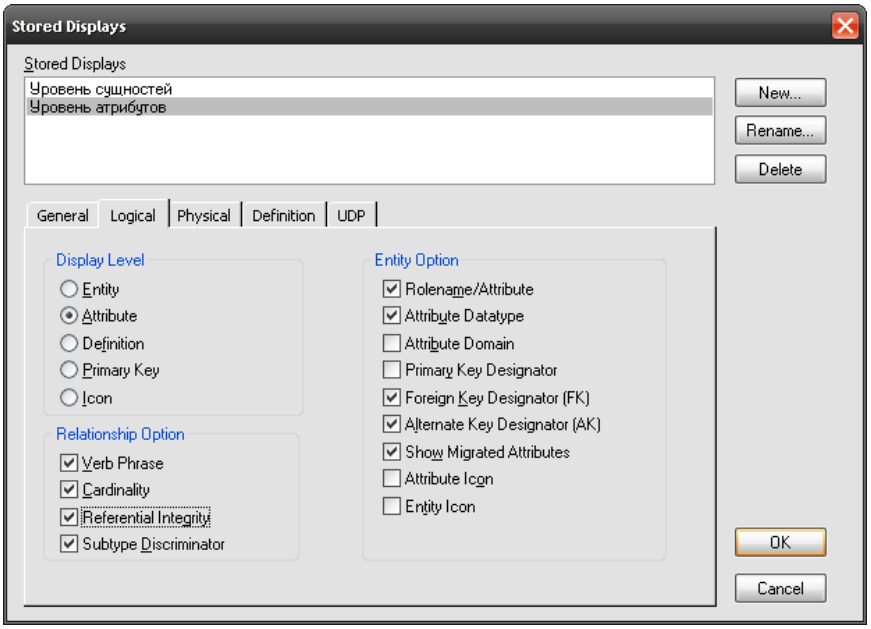


Рисунок 2.32 – Установка свойств уровня атрибутов

Рисунок 2.33 – Закладки хранимых отображений

14. Начиная разработку ER-диаграммы, необходимо определить ее некоторые общие свойства. Выберите пункт меню **Model►ModelProperties...** Введите в поля **Name**и **Author**(рисунок 2.34) название диаграммы и фамилию. В поле **Type**указывается тип модели (**Logical/Physical**) и версия сервера (**InterBase**), для которого разрабатывается диаграмма. Нажмите кнопку **ОК**.

15. Перейдите на закладку «Уровень сущностей». Для построения логической модели, прежде всего, необходимо определить набор сущностей и задать связи между ними. Этим мы займемся на лабораторной работе № 3.

16. Сохраните проект, например, под именем Lab_2.er1.

17. 17. Выполните индивидуальное задание по анализу и формальному описанию информационной системы для указанной предметной области (см.

таблицу 2.3).

18. Повторив пункты 1÷16 настоящих указаний, выполните индивидуальное задание по созданию каркаса проекта заданной информационной подсистемы в среде ERwin.

Варианты заданий

Необходимо разработать в среде ERwininформационную подсистему для указанных предметных областей по вариантам (см. таблицу 2.3). Номервариантасоответствует номеру рабочего места за компьютером.

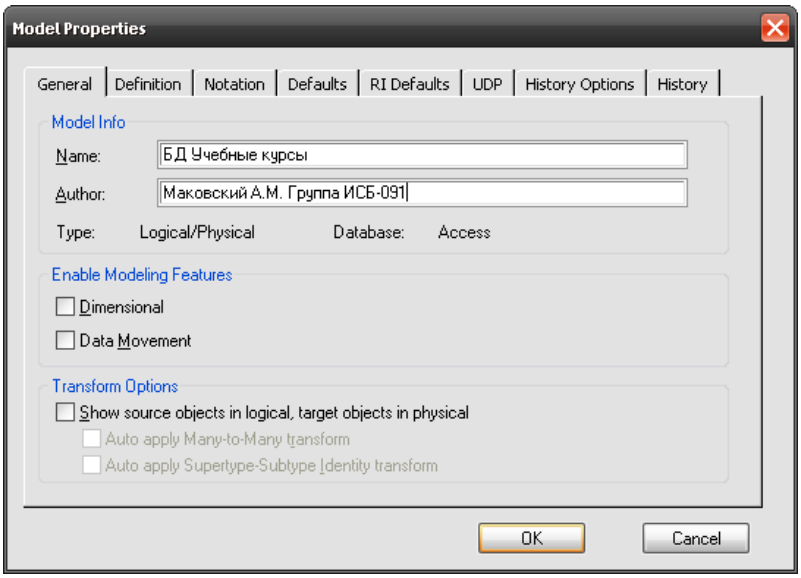


Рисунок 2.34 – Редактор свойств ER-диаграммы

Таблица 2.3 – Варианты индивидуальных заданий

Вариант	Содержание варианта
1	Разработать в среде Erwin информационную подсистему для страховой компании.
2	Разработать в среде Erwin информационную подсистему для пункта проката видеофильмов.
3	Разработать в среде Erwin информационную подсистему для начисления сдельной заработной платы.
4	Разработать в среде Erwin информационную подсистему для учета транспортных перевозок.

Вариант	Содержание варианта
5	Разработать в среде Erwin информационную подсистему для кассы автостанции.
6	Разработать в среде Erwin информационную подсистему для учета заявок клиентов торговой фирмы.
7	Разработать в среде Erwin информационную подсистему для приемной комиссии университета.
8	Разработать в среде ERwinинформационную подсистему для учета коммунальных платежей.
9	Разработать в среде ERwinинформационную подсистему для учебной библиотеки университета.
10	Разработать в среде ERwinинформационную подсистему для деканата университета.
11	Разработать в среде ERwinинформационную подсистему для общежития университета.
12	Разработать в среде ERwinинформационную подсистему для пункта обмена валюты.
13	Разработать в среде ERwinинформационную подсистему для учета производственного травматизма.
14	Разработать в среде ERwinинформационную подсистему для учета компьютерного парка университета.
15	Разработать в среде ERwinинформационную подсистему для пункта продаж оператора сотовой связи.
16	Разработать в среде ERwinинформационную подсистему для регистратуры поликлиники.

Содержание отчета и его форма

Отчет по лабораторной работе оформляется в виде текстового документа и должен включать:

1. Название лабораторной работы.
2. Цель лабораторной работы.
3. Формулировку индивидуального задания и результат его выполнения.
4. Краткие выводы по результатам выполнения лабораторной работы.

Вопросы для защиты работы

1. Что такое идентифицирующая связь?
2. Что такое неидентифицирующая связь?
3. Что такое физическая и логическая модель данных?
4. Что такое домен?
5. Что такое отношение?
6. Что такое заголовок?
7. Что такое тело отношения?
8. Что такое мощность отношения?
9. Что такое сущность?

10. Как производится проектирование базы данных?
11. Дайте характеристику модели данных.
12. Каково назначение нормализации отношений?
13. Что такое диаграмма «сущность-связь»?
14. Какова методика выполнения индивидуального задания?
15. Какие выводы можно сделать по результатам выполнения индивидуального задания?

Защита лабораторной работы

Оформленный, в соответствии с требованиями ГОСТов, отчет представляется студентом преподавателю для проверки и последующей защиты. Защита отчета по лабораторной работе производится студентом только индивидуально.

В ходе защите лабораторной работы студент отвечает на вопросы преподавателя (поясняет методику выполнения индивидуального задания, отвечает на контрольные вопросы и т.д.).

Отчет, оформленный с отступлениями от требований ГОСТов, небрежно и неаккуратно к защите не принимается.

Лабораторная работа № 3

Определение набора сущностей и задания их атрибутов в ERwin

Цель и содержание: привить студентам навыки определения набора сущностей и задания их атрибутов в ERwin.

Формируемые компетенции: ПК-2, ПК-6. ПК-7.

Теоретическое обоснование

Как отмечалось ранее в лабораторной работе № 2, для построения логической модели данных в ERwin, прежде всего, необходимо определить набор сущностей и задать связи между ними. По результатам обследования предметной области можно обозначить несколько объектных областей (таблица 3.1).

Таблица 3.1 – Сущности по предметным областям

Материальное обеспечение процесса обучения	
Класс	Учебный класс, в котором проводятся занятия
Учебное место	Входит в учебный класс и существует только в его составе
Единица оборудования	Составная часть учебного места. Может существовать отдельно от учебного места
Тип оборудования	Тип, к которому относится единица оборудования
Методическое обеспечение процесса обучения	
Типовой курс	Стандартный типовой курс обучения, может содержать 0 или более тем
Тема типового курса	Составляющая часть типового курса
Индивидуальный план	Программа обучения, составляется из тем различных типовых курсов
Плановое занятие	«Держатель» темы. В описании предметной области изображался «ящичком»
Учебный процесс	
Учебный день	Календарный день, в который проводится обучение. Содержит академические часы
Академический час	Урок, занятие. Принадлежит определенному учебному дню, имеет время начала и время окончания
Занятие в классе	Сущность, связывающая «плановое занятие» и «академический час»
Персонал и учащиеся	
Преподаватель	Лицо, которое проводит занятие
Учащийся	Лицо, с которым проводят занятие

Рассмотрим сущности, входящие в объектную область «Материальное обеспечение процесса обучения» и методику задания их атрибутов в ERwin.

Аппаратура и программное обеспечение

Аппаратура. Для выполнения лабораторной работы необходимы:

- персональный суперкомпьютер i7-5930K/ 8D8192D4-2133/ K2200/ 2K40M;
- рабочие станции: монитор 29 + клавиатура + мышь. ПК MDT750/ i75930K/ 4D8192D42133";
- аппаратный сетевой обучающий комплекс мод.VDCN-1502.

Программное обеспечение. Для выполнения лабораторной работы необходима операционная система WINDOWS 7/8/10 и CASE-средство ERwin7.3.

Указания по технике безопасности

При выполнении лабораторной работы **запрещается**:

1. Самостоятельно производить ремонт персонального компьютера, а так же, установку и удаление имеющегося программного обеспечения.
2. Нарушать общепринятые правила техники безопасности при работе с электрооборудованием, в частности, касаться электрических розеток металлическими предметами и т.д.
3. Принимать пищу, напитки и сорить на рабочем месте пользователя персонального компьютера.

В случае неисправности персонального компьютера необходимо **немедленно** сообщить об этом обслуживающему персоналу лаборатории (системному администратору, оператору).

Методика и порядок выполнения работы

1. Откройте проект, созданный вами на лабораторной работе № 2.

2. Выделите в палитре инструментов кнопку сущности  ;щелкнув по ней указателем мыши. Затем щелкните по чистой области диаграммы. На поле диаграммы появится прямоугольник, изображающий новую сущность, с автоматически сгенерированным именем**E/1**. Разумеется, это имя нас не может устраивать, поэтому изменим его на имя «Класс». Это можно сделать сразу же после вставки сущности в диаграмму, так как ее имя в этот момент находится в режиме редактирования (рисунок 3.1).

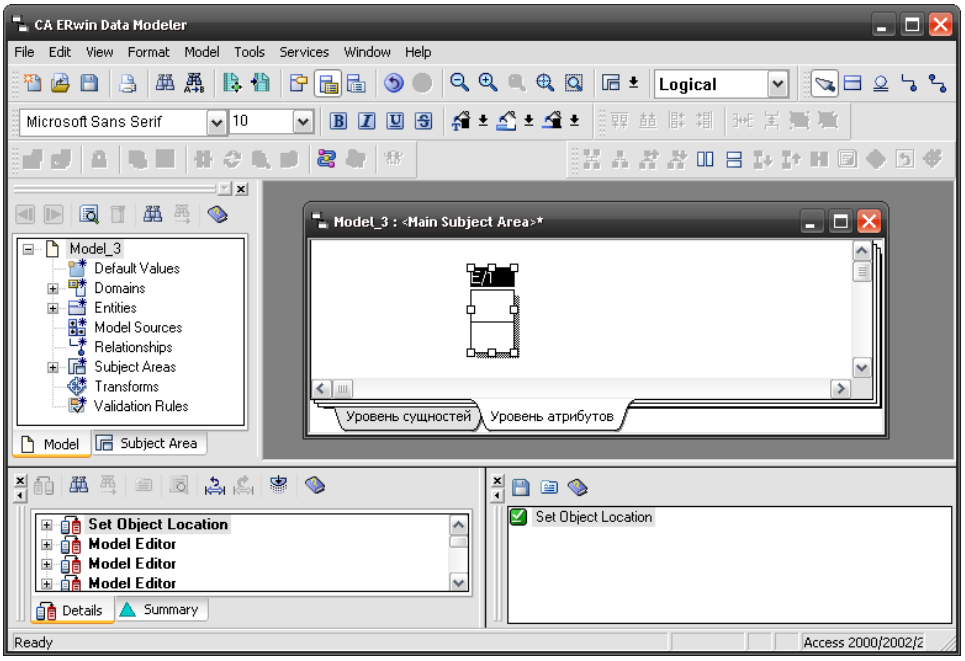


Рисунок 3.1 - Редактирование имени сущности

3. Введите с клавиатуры имя сущности «Класс» и нажмите клавишу **Enter**, Точно таким же образом дополните диаграмму еще тремя сущностями: «Учебное место», «Тип оборудования» и «Единица оборудования» (рисунок 3.2).

Добавленные в диаграмму сущности можно перемещать и удалять. Для перемещения выберите в палитре инструментов «стрелку» и выделите сущность, которую хотите переместить или удалить, щелкнув «стрелкой» по ее прямоугольнику на ER-диаграмме. У выбранной сущности название выделяется подсветкой - становится инверсным, и ее можно «перетаскивать» мышью. Для удаления выделенной сущности необходимо нажать клавишу **Del**.

При этом на экране появится диалог, требующий подтверждения операции(рисунок 3.3).

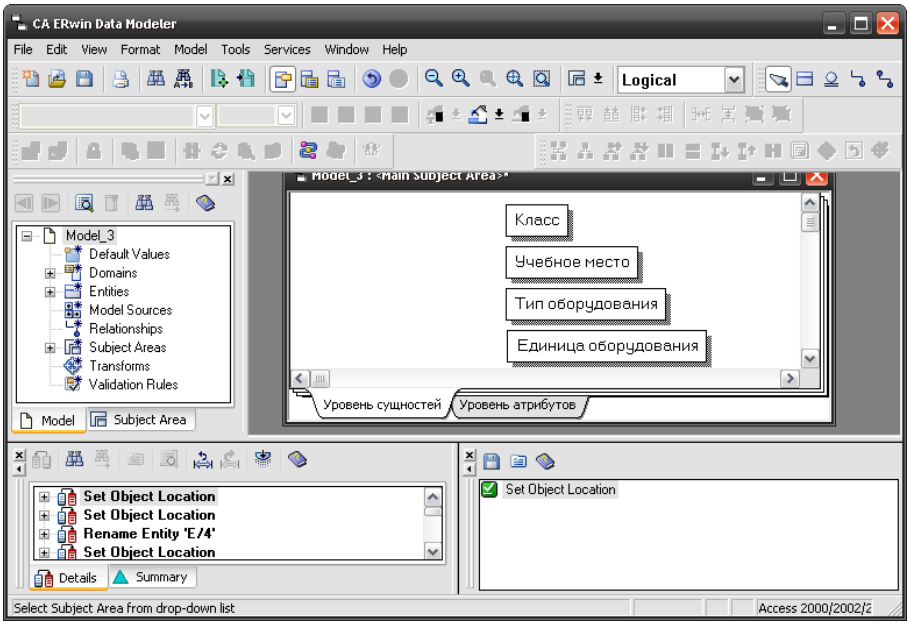


Рисунок 3.2–Сущности объектной области «Материальное обеспечение»

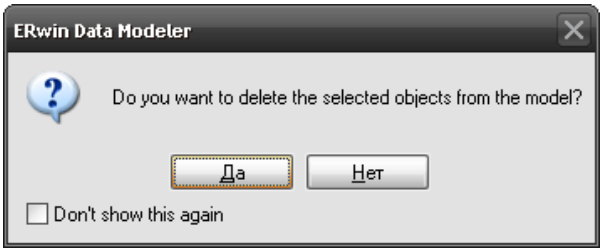


Рисунок 3.3 –Диалог удаления сущности

Здесь можно отказаться от операции или подтвердить ее. Каждая из сущностей, как и всякий объект ER-диаграммы, обладает контекстным меню.Для вызова контекстного меню (рисунок 3.4) необходимо щелкнуть по прямоугольнику сущности правой кнопкой мыши.

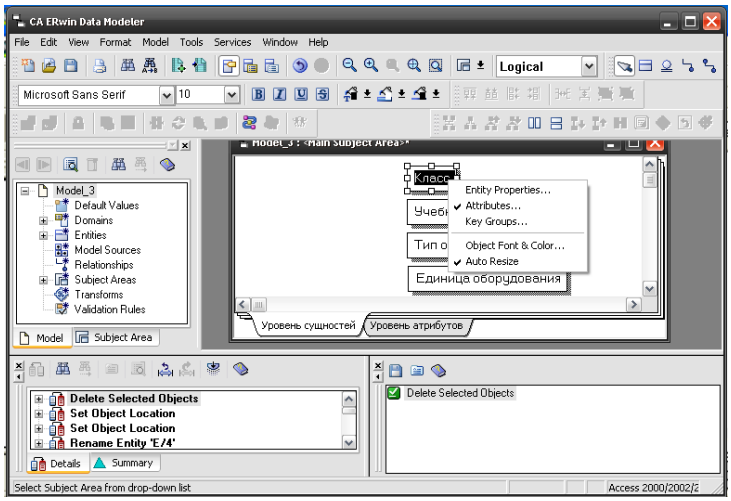


Рисунок 3.4 –Контекстное меню сущности

Первый пункт в контекстном меню **EntityProperties... (**Свойства сущности …**)** вызывает редактор, позволяющий изменять свойства выбранной

сущности. В верхней части окна этого редактора (рисунок 3.5) находится список всех сущностей (Entity), имеющихся на диаграмме.

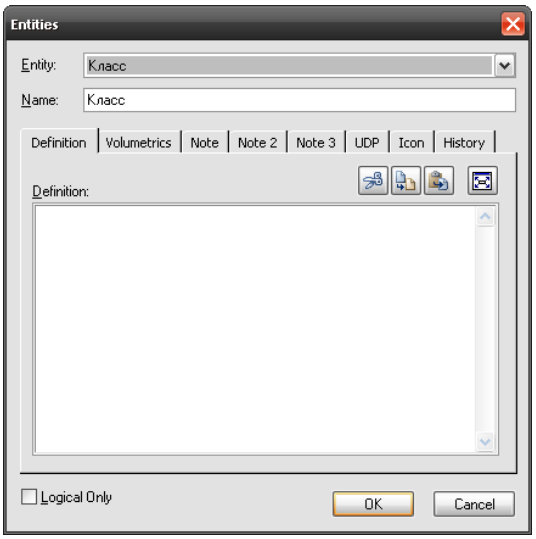


Рисунок 3.5 –Редактор сущности

С его помощью можно выбрать сущность, свойства которой необходимо посмотреть или изменить. По умолчанию, выбранной являете выделенная на диаграмме сущность, по которой щелкнули мышью. Далее имеется поле **Name**(имя), в котором высвечивается имя сущности. Имя можно редактировать.

Ниже в окне редактора (см. рисунок 3.5)находится ряд закладок:

- **Definition**(определение) - на этой странице вводится определение сущности Текст определения обычно зависит от стандартов, принятых на предприятии, и должен облегчать восприятие модели.

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

- **UDP**(определяемые пользователем свойства) – механизм пользовательских свойств совпадает с описанным выше, при определении хранимых отображений диаграммы.

- **Icon**(иконка) –для наглядности каждой сущности может быть присвоена иконка, которая выводится рядом с се названием (рисунок 3.6). Помимо маленькой иконки (**SmallIcon**), в целях презентации можно присвоить сущности и более крупный рисунок (**LargeIcon**).

Как большая, так и маленькая иконки выбираются из выпадающего списка. Для внесения в этот список новой иконки необходимо нажать находящуюся справа кнопку с многоточием, затем нажать кнопку **Import**и в появившемся диалоге выбрать файл с рисунком.

Чтобы перейти в режим показа иконок, щелкните правой кнопкой мыши по свободному полю диаграммы и» в контекстном меню выберите пункт **DisplayLevel►Entities►EntityIcon**(рисунок 3.7).

Если иконка не назначена, то рядом с именем будет выводиться иконка

по умолчанию, всегда присутствующая в списке. В отличие от добавленных пользователем иконок, иконку по умолчанию нельзя удалить из списка. Встандартной поставке ERwinимеется библиотека иконок. Выберите для каждой сущности свою иконку и включите режим **EntityIcon**.

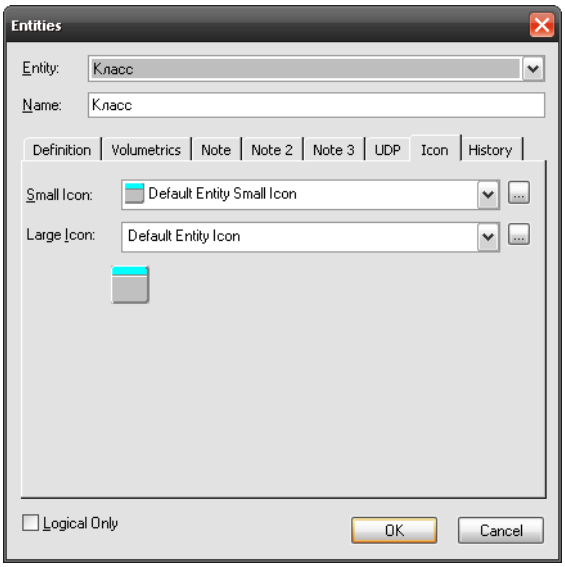


Рисунок 3.6 - Закладки редактора сущности

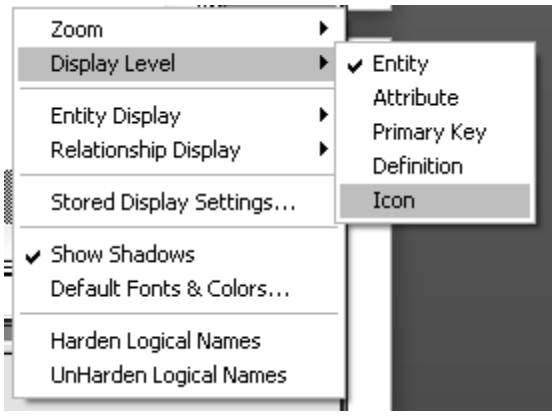


Рисунок 3.7 - Выбор режима показа иконок в контекстном меню

Определив сущности, необходимо внести в схему и атрибуты этих сущностей. Рассмотрим, какие атрибуты могут нам понадобиться при работе с каждой из сущностей.

Класс. При обследовании предметной области мы выяснили, что набор атрибутов зависит от конкретных потребностей предприятия и может включать в себя адрес, номер корпуса, номер аудитории, телефон и, возможно, некоторую произвольную информацию (примечание). Кроме того, необходимо обеспечить сущность уникальным ключом, поэтому добавим еще числовой атрибут «код класса». Этот атрибут должен быть невидимым для пользователя, поэтому его необходимо будет генерировать автоматически, о чем мы

позаботимся позднее. Вообще говоря, такой метод «искусственного» ключаприменяется очень широко на практике, и мы также будем использовать егов данном примере.

Учебное место. В качестве ключевого атрибута зададим код учебного места. Остальными атрибутами будут номер учебного места, имя рабочей станции, IP-адрес, примечание. Характеристики этих атрибутов уже рассмотрены при анализе предметной области.

Тип оборудования. Набор атрибутов здесь может быть достаточно велик. В данном примере мы ограничимся кодом типа оборудования в качестве ключа и наименованием.

Единица оборудования. Ключевым атрибутом будет код единицы оборудования, как и у предыдущих сущностей. Кроме того, в соответствии с анализом предметной области, назначим атрибуты инвентарный номер, техническая характеристика, признак исправности, дата установки и примечание.

Перечень сущностей, их атрибутов с характеристиками приведен в таблице 3.2.

Таблица 3.2 - Сущности, входящие в объектную область «Материальное обеспечение процесса обучения»

Сущность	Атрибут	Ключ	Тип
Класс	код класса		число
	адрес		строка
	номер корпуса		строка
	номер аудитории		строка
	телефон		строка
	примечание		строка
Учебное место	код учебного места		число
	номер учебного места		строка
	имя рабочей станции		строка
	ip-адрес		строка
	примечание		строка
Тип оборудования	код типа оборудования		число
	наименование типа		строка
Единица оборудования	код единицы оборудования		число
	инвентарный номер		строка
	техническая характеристика		строка
	признак исправности		число
	дата установки		дата
	примечание		строка

Каждый из атрибутов при преобразовании логической ER-диаграммы в схему базы данных становится колонкой таблицы, с которой связан определенный тип данных. ERwinпозволяет указать этот тип данных на диаграмме, однако удобнее определять тип данных атрибута не через простой тип данных, а через его подмножество – домен.

4. Для создания доменов, выберите пункт меню **Model ► Domain Dictionary…** (рисунок 3.8).

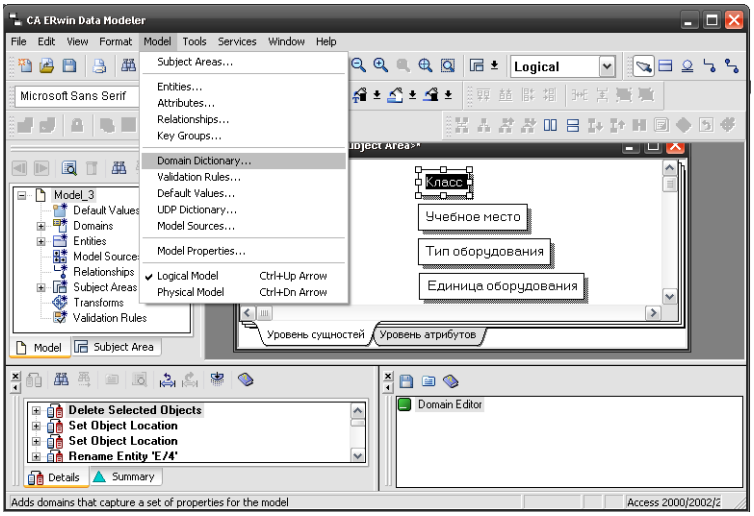


Рисунок3.8 – Выбор пункта меню **Model ► Domain Dictionary…**

Диалоговое окно **DomainDictionary**(Редактор словаря доменов), изображенное на рисунке 3.9, позволяет создавать и редактировать домены в двух режимах – логическом и физическом.

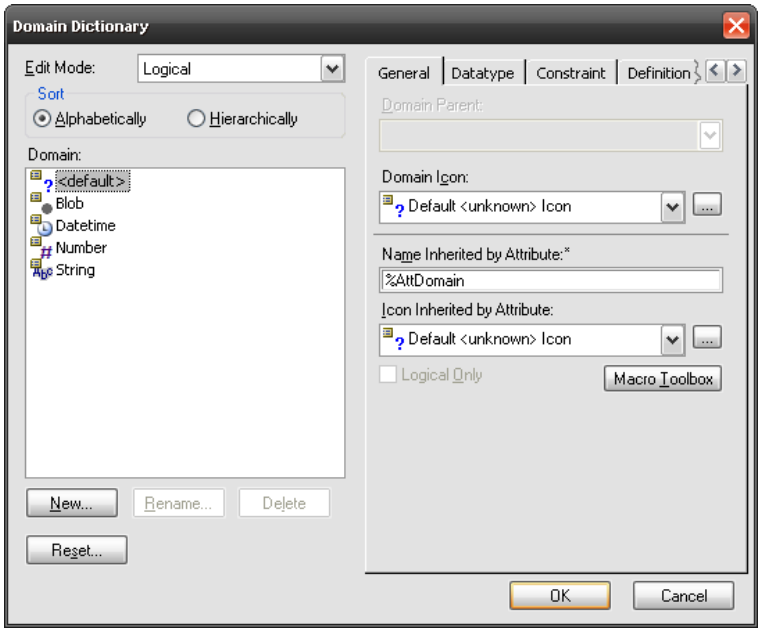


Рисунок 3.9 - Диалоговое окно редактора словаря доменов

Переключение режима производится при помощи выпадающего спи-

ска, находящегося наверху слева и снабженного меткой **EditMode**(Режим редактирования).

Ниже имеется переключатель, позволяющий установить порядок сортировки доменов в списке - алфавитный или иерархический.

Еще ниже находится собственно список доменов. По умолчанию он содержит пять базовых доменов, на основе которых разработчик может определить собственные домены. Базовые домены представляют основные типы данных, используемые в СУБД:

а) строковый (String);
б) числовой (Number);
в) дата/время (Datetime);
г) двоичный (blob).

Кроме того, имеется еще один базовый домен самого общего характера с неопределенным типом данных - «неизвестный» (<unknown>). Иерархически этот домен является родителем всех остальных, в чем легко убедиться, переключив вид сортировки с алфавитного на иерархический. При этом домены в списке расположатся в виде дерева, корнем которого будет <unknown> (рисунок 3.10)

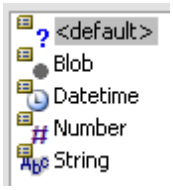


Рисунок 3.10 - Дерево базовых доменов

В нижней части диалогового окна редактора словаря доменов находятся кнопки (таблица 3.3).

Таблица 3.3 – Кнопки диалогового окна редактора словаря доменов

Кнопка	Назначение кнопки
New	Создание нового домена
Rename	Переименование домена
Delete	Удаление домена
Reset	Сброс свойств домена

В правой части расположены страницы с закладками, позволяющими редактировать свойства доменов. Набор страниц зависит от режима редактирования. В логическом режиме имеется четыре страницы - **General**(общие свойства), **Definition**(определение), **Note**(примечание) и **UDP**(пользовательские свойства). Последние три страницы не отличаются от аналогичных страниц, уже описанных для других объектов диаграммы. Страница **General**позволяет редактировать свойства двух типов - ненаследуемые и наследуемые. Ненаследуемые свойства относятся только к домену и не

передаются атрибутам, определяемым на базе этого домена. К ним относятся родительский домен (**Domainparent**) и иконка домена (**DomainIcon**). Напротив, наследуемые свойства передаются всем атрибутам, созданным на базе домена.К этим свойствам относятся наследуемое имя и иконка. В качестве наследуемого имени по умолчанию устанавливается **%AttDomain**-макроопределение, которое заменяется именем домена. Это означает, что при создании атрибута на базе данного домена его логическим именем будет имя домена.

Кроме того, при помощи специальных флажков на этой же странице вы можете задать:

- **Required**- атрибут, созданный на базе этого домена, является обязательным, то есть всегда должен содержать данные В физической модели это соответствует заданию для поля опции NOTNULL.
- **Logical**- домен должен быть виден только в логической модели.

5. Создайте домен для атрибута «код класса» Для этого нажмите накнопку**New**(Новый) и в появившемся диалоговом окне (рисунок 15.11) введите:

а) в поле **LogicalName**(логическое имя) - код класса;

б) в поле **PhysicalName**(физическое имя) - t_class_id;

в) в списке базовых доменов выберите числовой домен - number.

Физическое имя станет именем типа данных в таблице базы. По умолчанию ERwinгенерирует физическое имя из логического, заменяя пробелы символом подчеркивания.

В нашем случае он создал имя «код _класса», но так как сервера, как правило, не поддерживают имен, содержащих символы кириллицы, мы заменили его английским именем. Префикс «t_» означает, что это имя типа данных (домена). Подобного же подхода мы будем придерживаться и далее, то есть все имена доменов будут начинаться с «t_».

Нажмите кнопку **ОК**, и новый домен будет добавлен в список существующих доменов (рисунок 3.12).

5. Измените режим редактирования с логического на физический, выбрав в списке **EditMode**значение Physical (рисунок3.13).

Теперь в списке слева фигурируют физические имена доменов, а состави содержание страниц свойств изменились. В частности, на странице **General**

теперь находятся:

а) поле со списком **DomainParent**(родительский домен) - то же самое, что и в логическом режиме;

б) флажок **DOMAIN**. Если этот флажок установлен, то при

генерациифизической схемы базы данных в запросах **CREATETABLE**будут использованы домены, причем в разделе меню **Tools►ForwardEngineer/SchemaGeneration ...** имеется два флажка, позволяющие выбрать режим генерации.

в) поле**NameInheritedByColumn**(имя, наследуемоеколонкой). Как и для логических имен атрибутов, здесь вводится макроопределение, генерирующее имя колонки, которая создается на базе этого домена. ERwinобладает развитой системой макроопределений, с помощью которых можно существенно изменить и настроить поведение программы в различных ситуациях. Например, в данном случае, по умолчанию используется макроопределение **%CoIDomain**возвращающее физическое имя домена, то есть программа ERwinбудет генерировать имя колонки совпадающим с именем домена. В рассматриваемой схеме данных физическое имя домена начинается с префикса «t_», и мы не хотим, чтобы этот префикс попал в имя колонки. Для этого воспользуемся макросом **%Substr()**. Его синтаксис похож на синтаксис обычной функции нахождения подстроки, имеющейся во всех языках высокого уровня:

%Substr(<строка>,<начальная позиция>,<длина>).

Например, %Substr (macro, 1, 3) возвратит строку «mac».

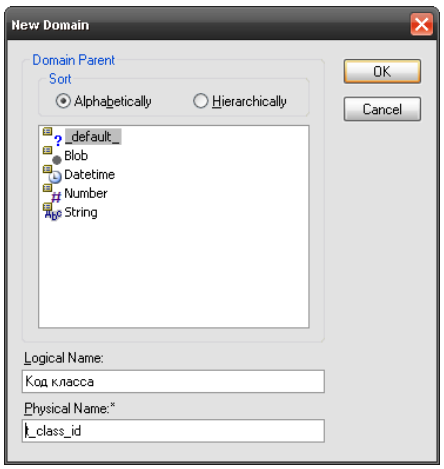


Рисунок 3.11 - Окно создания домена

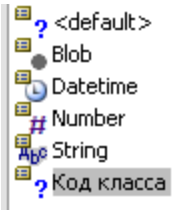


Рисунок 3.12 - Пример дерева с новым доменом

7. Введите в поле наследуемого имени %Substr(%ColDoinain,3,50). Та-

ким образом, имя колонки будет генерироваться равным имени домена, но начиная с третьего символа. В качестве длины строки мы проставили 50, так как в схеме заведомо не будет имен доменов длиннее 50 символов.

Следующая страница зависит от заданного в программе типа сервера, для которого будет генерироваться база данных. Тип сервера задавался нами на лабораторной работе № 2 (см. рисунок 2.2). В нашем случае был выбран сервер **Access**, поэтому на соответствующей закладке редактора словаря доменов проставлено **Access**(рисунок 3.15).

Для выбранного в списке домена здесь выбираются:

в) тип данных (**AccessDatatype**) - физический тип данных, определенный для выбранного сервера, в данном случае для сервера **Access**. В этом же поле проставляется и размерность, если это необходимо ФлажокNullOptionпозволяет задать домены, у которых должно быть предопределено свойство **NOTNULL**или **NULL**.

г) задается средняя ширина поля и процент записей, имеющих в этом поле **NULL**. Эти данные используются при оценке объемов таблиц и базы данных в целом (пункт меню **Edit► Volumetrics ...**).

д) два поля со списком задают правило валидации (проверки допусти мости значения) и значение по умолчанию. Оставьте пока содержимое этой страницы без изменений.

е) вкладка **Comment**. Внесение комментария к атрибуту. Программа позволяет определить собственно комментарий к атрибуту, а также задать комментарий, наследуемый колонкой таблицы.

ж) вкладка **UDP**. Эта вкладка позволяет задать свойства, определяемые пользователем. Как и в предыдущем случае, можно отдельно задать свойства, наследуемые колонкой.

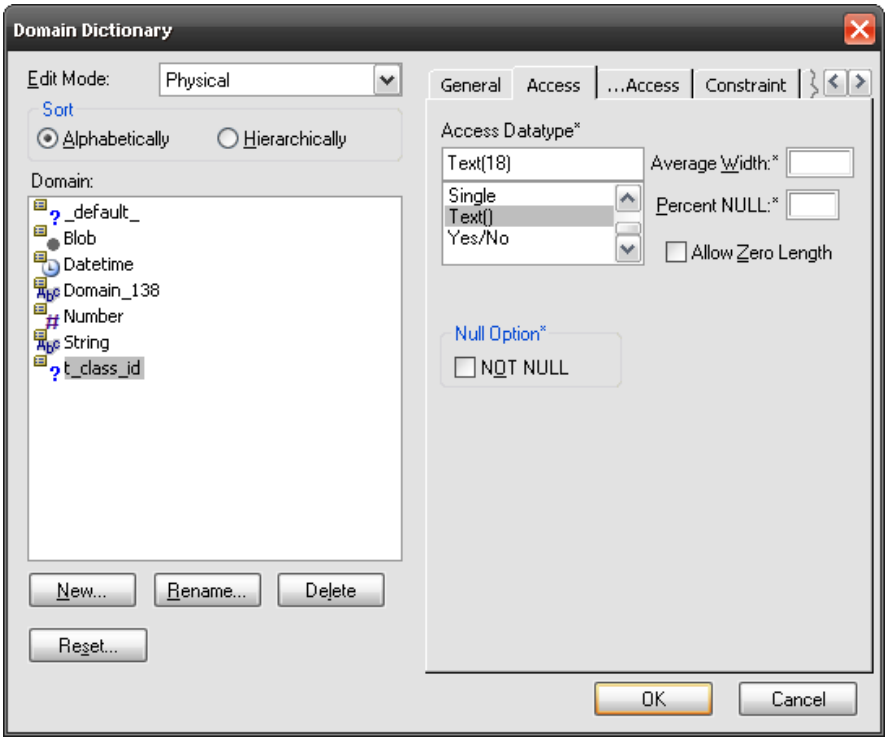


Рисунок 3.15 - Страница **Access**

Кроме этих страниц, имеется несколько страниц, специально предназначенных для генерации кодов клиентского приложения на **VisualBasic**и **PowerBuilder**.

8. Точно также создайте остальные домены, имена и базовые типы которых приведены в таблице 3.3.

Таблица 3.3 - Домены, используемые в объектной области «Материальноеобеспечение процесса обучения»

Логическое имя домена	Физическое имя домена	Родительский домен
код класса	t ctass id	Number
адрес	t address	String
номер корпуса	t corpus no	String
номер аудитории	t audit no	String
телефон	t telephone	String
примечание	t note	String
код учебного места	t workplace id	Number
номер учебного места	t workplace no	String
имя рабочей станции	t workplace name	String
ip-адрес	t ip address	String
код типа оборудования	t equip type id	Number
наименование типа	t equip name	String
код единицы оборудования	t equip id	Number
инвентарный номер	t inventory no	String

Логическое имя домена	Физическое имя домена	Родительский домен
техническая характеристика	t character	String
признак исправности	t malfuncf	Number
признак раздела	t is section	Number
дата установки	t insfall date	Datetime

9. Подготовив домены, мы можем перейти к заданию атрибутов сущностей на диаграмме. Для этого выделите сущность «класс», щелкнув по нему указателем мыши (в палитре инструментов должна быть, выбрана «стрелка»), а затем вызовите пункт меню **Model►Attribute ...** То же самое можно выполнить, выбрав пункт **Attibutes...** контекстного меню. При этом на экране появится окно редактора атрибутов (**Attributes**) (рисунок 3.16).

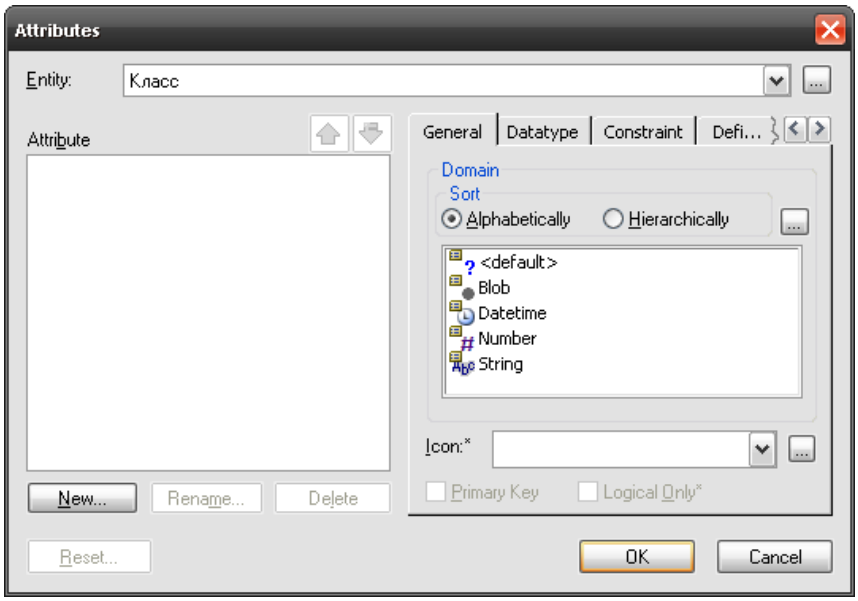


Рисунок 3.16 – Редактор атрибутов

Редактор атрибутов построен по тому же принципу, что и редактор сущностей. В верхней части диалогового окна находится выпадающийсписок, в котором можно выбрать сущность для редактирования. При вызове поумолчанию здесь проставляется имя сущности, выделенной на диаграмме.

Рядом имеется кнопка, вызывающая редактор сущностей.

Основная область окна редактора делится на две части - список атрибутов и страницы свойств. Для ввода нового атрибута нажмите кнопку **New**(Новый). Выберите в списке доменов домен «код класса». В поле **AttributeName**появится имя атрибута «код класса», а в поле **ColumnName-class_id**. Эти имена генерируются макроопределениями, которые мы задали для наследуемых имен домена - %AtlDomainи %Substr(%ColDommn, 3, 50).

После нажатия кнопки **ОК**атрибут появится в окне редактора.

Точно так же введите остальные атрибуты сущности «Класс». После этого выделите атрибут «код класса» и установите флажок **PrimaryKey**, так как «код класса» является первичным ключом сущности «Класс». Напротив имени этого атрибута в списке слева появится символ ключа. 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 (рисунок 3.17).

Нажмите кнопку **ОК**. Как вы помните, мы создали два хранимых экрана - «Уровень сущностей» и «Уровень атрибутов».

10. Шаг 10. 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 также в виде прямоугольников, однако имя сущности пишется над прямоугольником, а внутри него дается список атрибутов. Прямоугольник сущности делится на две части, в верхней из которых приводятся атрибуты первичного ключа, а в нижней - все остальные (рисунок 3.18). Пока на диаграмме определены только атрибуты сущности «Класс», поэтому прочие сущности являются незаполненными (пустыми).

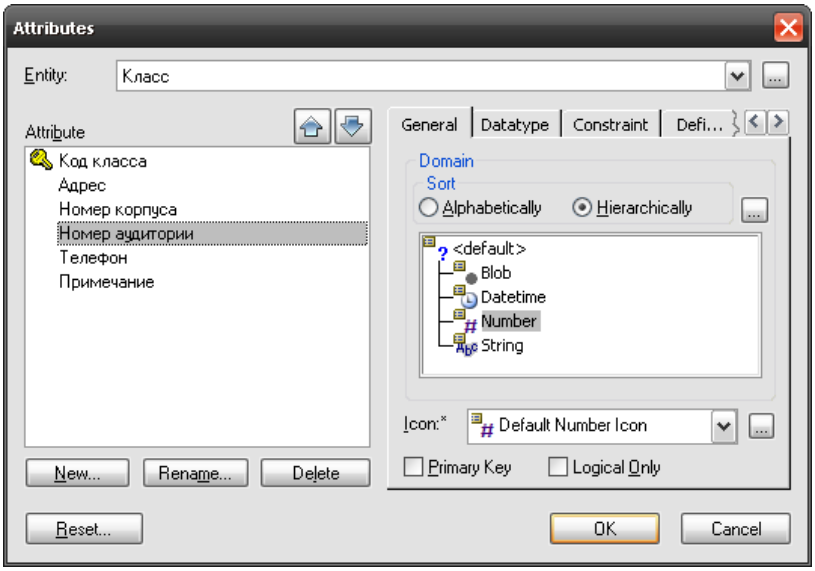


Рисунок 3.17 – Атрибуты сущности «Класс»

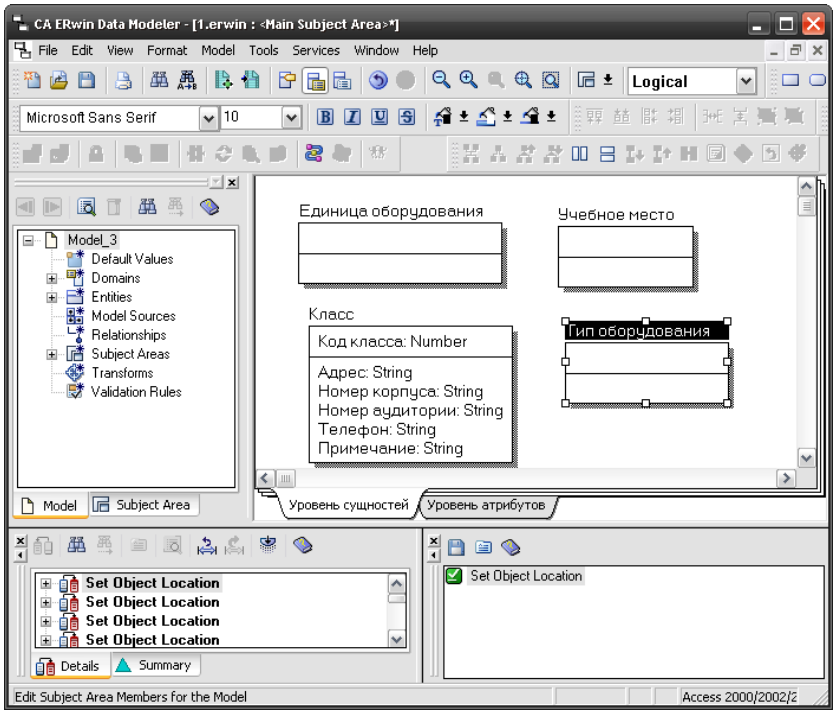


Рисунок 3.18 – Сущность «Класс» на уровне атрибутов

Определим атрибуты остальных сущностей на диаграмме. Описанный выше метод задания атрибутов не является единственным в ERwin. Значительно удобнее применение специального инструмента, который называется «Браузер (проводник) доменов» (**ModelExplorer**) вкладка **Domains**(**Dom…**). Он представляет собой тот же список доменов, который вводился в редакторе словаря доменов (**DomainDictionaryEditor**).

Основное преимущество браузера в том, что он позволяет задавать атрибут простым перетаскиванием мышью.

Для этого достаточно выделить нужное имя домена в списке, нажать левую кнопку мыши и, не отпуская ее, перенести указатель в прямоугольник сущности. Когда вы отпустите кнопку мыши, программа создаст атрибут от выбранного домена. После внесения атрибутов во все имеющиеся сущности диаграмма будет иметь вид, показанный на рисунке 3.19.

11. Вернитесь на вкладку «Уровень сущностей» и сохраните модель, например, под именем Lab_3.er1.

Итак, теперь диаграмма содержит четыре сущности. Методику определения связи между ними мы рассмотрим на лабораторной работе № 4.

12. Повторив пункты 1÷11 настоящих указаний, выполните индивидуальное задание по определению наборов сущностей и заданию их атрибутовв среде ERwindля указанной предметной области.

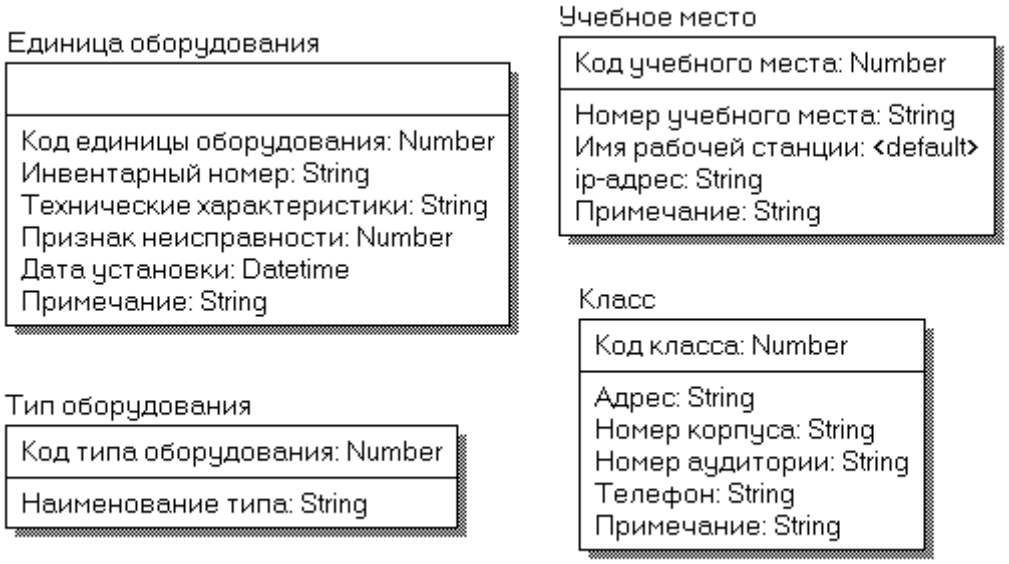


Рисунок 3.19 – Сущности объектной области «Материальное обеспечение»

Варианты заданий

Используя результаты выполнения лабораторной работы № 2, необходимо определить наборы сущностей и задать их атрибуты в среде ERwindля указанной предметной области. Номер варианта соответствует номеру рабочего места за компьютером и должен совпадать с номером варианта индивидуального задания к лабораторной работе № 2 (см. таблицу 2.3).

Содержание отчета и его форма

Отчет по лабораторной работе оформляется в виде текстового документа и должен включать:

1. Название лабораторной работы.
2. Цель лабораторной работы.
3. Результаты выполнения пунктов 1÷11 программы лабораторной работы.
4. Формулировку индивидуального задания и результат его выполнения (наборы сущностей и их атрибуты в среде ERwindля указанных предметных областей по вариантам);
5. Краткие выводы по результатам выполнения лабораторной работы.

Вопросы для защиты работы

1. Как создать и откорректировать сущность на диаграмме ERwin?
2. Как перемещать и удалять сущности на диаграмме ERwin?
3. Как пользоваться редактором сущностей?
4. Каково назначение и возможности редактора словаря доменов?
5. Каково назначение окна браузера доменов?

6. Какая разница между логическим и физическим именем домена?
7. Что означает макрос %Substr(%ColDoinain,3,50)?
8. Каково назначение и возможности редактора атрибутов?
9. Какие сущности были созданы при выполнении индивидуального задания?
10. Какие атрибуты сущностей были определены при выполнении индивидуального задания?
11. Какова методика выполнения индивидуального задания?
12. Какие основные выводы можно сделать по результатам выполнения индивидуального задания?

Защита лабораторной работы

Оформленный, в соответствии с требованиями ГОСТов, отчет представляется студентом преподавателю для проверки и последующей защиты. Защита отчета по лабораторной работе производится студентом только индивидуально.

В ходе защите лабораторной работы студент отвечает на вопросы преподавателя (поясняет методику выполнения индивидуального задания, отвечает на контрольные вопросы и т.д.).

Отчет, оформленный с отступлениями от требований ГОСТов, небрежно и неаккуратно к защите не принимается.

Лабораторная работа № 4

Определение связей между сущностями в ERwin

Цель и содержание: привить студентам навыки задания связей между сущностями модели данных в ERwin.

Формируемые компетенции: ПК-2, ПК-6. ПК-7.

Теоретическое обоснование

Как отмечалось ранее в лабораторной работе № 3, для построения логической модели данных, прежде всего, необходимо определить набор сущностей и задать связи между ними. На лабораторной работе № 3 была создана диаграмма, содержащая четыре сущности (рисунок 3.19). Методику определения связи между ними мы рассмотрим на текущей лабораторной работе.

Связь в ERwintрактуется как функциональная зависимость между двумя сущностями (в частности, возможна связь сущности с самой собой).

Если рассматривать диаграмму ERwinкак графическое представление правил предметной области, то сущности являются существительными, а связи - глаголами. Например, между сущностями ОТДЕЛ и СОТРУДНИК существует связь «состоит из» (ОТДЕЛ состоит из СОТРУДНИКОВ).

В ERwincсвязи между сущностями представлены пятью основными элементами информации:

а) тип связи;
б) родительская и дочерняя (зависимая) сущности;
в) мощность связи;
г) допустимость пустых (**null**) значений;
д) требования по обеспечению ссылочной целостности.

ERwinподдерживает следующие основные типы связей: идентифицирующая, неидентифицирующая, полная категория, неполная категория, «многие-ко-многим».

Связь называется **идентифицирующей**, если экземпляр дочерней сущности идентифицируется через ее связь с родительской сущностью. Атрибуты, составляющие первичный ключ родительской сущности, при этом входят в первичный ключ дочерней сущности. Дочерняя сущность при идентифицирующей связи всегда является зависимой.

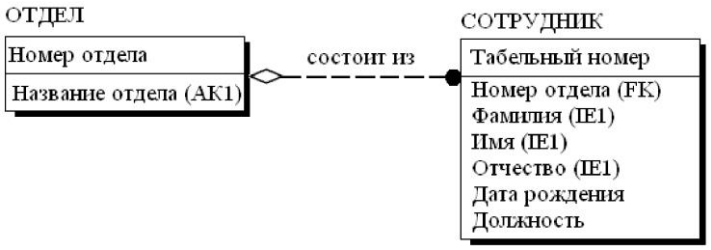
Связь называется **неидентифицирующей**, если экземпляр дочерней сущности идентифицируется иначе, чем через связь с родительской сущностью. Атрибуты, составляющие первичный ключ родительской сущности, при этом входят в состав неключевых атрибутов дочерней сущности.

Идентифицирующая связь изображается сплошной линией; неиденти-

фицирующая- пунктирной линией. Линии заканчиваются точкой со стороны дочерней сущности.

При определении связи происходит миграция атрибутов первичного ключа родительской сущности в соответствующую область атрибутов дочерней сущности. Поэтому такие атрибуты не вводятся вручную.

На рисунке 16.1 приведен пример неидентифицирующей связи. Первичный ключ сущности ОТДЕЛ «Номер отдела» мигрировал в область неключевых атрибутов (поскольку связь неидентифицирующая) сущности СОТРУДНИК. На диаграмме атрибуты, наследованные от родительской



сущности, помечаются символами FK, заключенными в скобки.

Рисунок 4.1 - Пример неидентифицирующей связи

Зависимая сущность может наследовать один и тот же атрибут от более чем одной родительской сущности или от одной и той же родительской сущности через несколько связей.

Поскольку атрибуты первичного ключа родительской сущности по умолчанию мигрируют со своими именами, ERwinсчитает, что в зависимой сущности атрибуты внешнего ключа появляются только один раз. Чтобы избежать этого ограничения, ERwinпозволяет ввести для них роли, т.е. новые имена, под которыми мигрирующие атрибуты будут представлены в дочерней сущности. В случае неоднократной миграции атрибута такое переименование необходимо. Например, при создании модели сделки по обмену валюты сущность СДЕЛКА (рисунок 4.2) должна иметь два различных атрибута для кодов проданной и купленной валюты. В данном случае первичный ключ сущности ВАЛЮТА («Код валюты») имеет две роли в дочерней сущности.

Рисунок 4.2 - Пример использования ролей

Ситуация, когда экземпляру одной сущности соответствует один или несколько экземпляров второй сущности, а экземпляру второй сущности соответствует один или несколько экземпляров первой сущности, отражается в логической модели связью многие-ко-многим между данными сущностями. На диаграмме связь изображается сплошной линией с точками на концах. Например, для заключения сделки в некоторой фирме клиент обращается к любому из свободных сотрудников этой фирмы. В то же время сотрудник фирмы может обслуживать нескольких клиентов. Поэтому тип связи междусущностями КЛИЕНТ и СОТРУДНИК должен быть многие-ко-многим (рисунок 4.3).

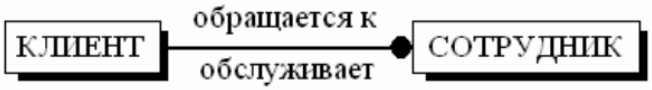


Рисунок 4.3 - Пример связи «многие-ко-многим»

Заметим, что связь типа многие-ко-многим возможна только на логическом уровне. Преобразование связи данного типа на физическом уровне будет рассмотрено в следующем пункте. Однако добавим, что связей многие-ко-многим рекомендуется избегать. В рассмотренном примере этого можно добиться, если ввести дополнительную сущность СДЕЛКА (рисунок 4.4).

Рисунок 4.4 - Пример устранения связи многие-ко-многим

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

Общая часть атрибутов сущностей-подтипов, включая первичный ключ, помещается в сущность-супертип. Различная часть (например, данные о почасовой оплате для временных работников или данные о зарплате и от-

пуске для штатных работников) помещается, в сущности-подтипы. В сущности - супертипе вводится атрибут-дискриминатор, позволяющий различать конкретные экземпляры сущности-подтипа.

В зависимости от того, все ли возможные сущности-подтипы включены в модель, категорийная связь является полной или неполной. В ERwinполная категория изображается окружностью с двумя подчеркиваниями, а неполная-окружностью с одним подчеркиванием.На рисунке 4.5 категорийная связь между сущностью СОТРУДНИК исущностями ПОСТОЯННЫЙ СОТРУДНИК и СОВМЕСТИТЕЛЬ являетсянеполной, если предположить, что существует еще один тип сотрудников -консультант.

Рисунок 4.5 - Пример неполной категории

Включение в модель сущности КОНСУЛЬТАНТ приводит к тому, что категорийная связь становится полной (рисунок 4.6).

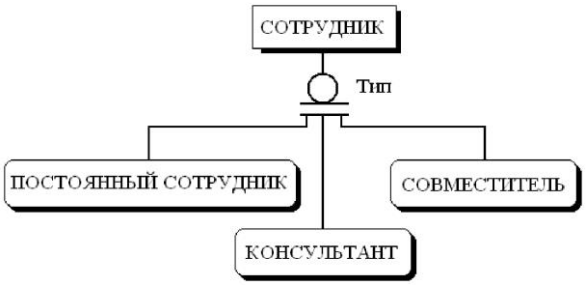


Рисунок 4.6 - Пример полной категории

Мощность связи представляет собой отношение количества экземпляров родительской сущности к соответствующему количеству экземпляров дочерней сущности. Мощность связи определяется только для идентифицирующих и неидентифицирующих связей и записывается как **1:n**. ERwin, в соответствии с методологией **IDEF1X**, предоставляет четыре варианта для параметра **n**, которые изображаются дополнительным символом у дочерней сущности (рисунок 4.7)

Рисунок 4.7 - Обозначение кратности связи

Допустимость пустых (**NULL**) значений в неидентифицирующих связях ERwinизображает пустым ромбиком на дуге связи со стороны родительской сущности.

В целях контроля ссылочной целостности (под ссылочной целостностью в ERwinпонимается обеспечение требования, чтобы значения внешнего ключа экземпляра дочерней сущности соответствовали значениям первичного ключа в родительской сущности) для каждой связи могут быть заданы требования по обработке операций **INSERT/UPDATE/DELETE**для родительской и дочерней сущности. ERwinпредоставляет следующие варианты обработки этих событий:

а) отсутствие проверки;

б) проверка допустимости;

в) запрет операции;

г) каскадное выполнение операции (**DELETE/UPDATE**);

д) установка пустого (**NULL**-значения) или заданного значения по умолчанию.

Аппаратура и программное обеспечение

Аппаратура. Для выполнения лабораторной работы необходимы:

- персональный суперкомпьютер i7-5930K/ 8D8192D4-2133/ K2200/ 2K40M;

- рабочие станции: монитор 29 + клавиатура + мышь. ПК MDT750/ i75930K/ 4D8192D42133";

- аппаратный сетевой обучающий комплекс мод.VDCN-1502.

Программное обеспечение. Для выполнения лабораторной работы необходима операционная система WINDOWS 7/8/10 и CASE-средство ERwin7.3.

Указания по технике безопасности

При выполнении лабораторной работы **запрещается**:

1. Самостоятельно производить ремонт персонального компьютера, а так же, установку и удаление имеющегося программного обеспечения.

2. Нарушать общепринятые правила техники безопасности при работе с электрооборудованием, в частности, касаться электрических розеток металлическими предметами и т.д.

3. Принимать пищу, напитки и сорить на рабочем месте пользователя

персонального компьютера.

В случае неисправности персонального компьютера необходимо **немедленно** сообщить об этом обслуживающему персоналу лаборатории (системному администратору, оператору).

Методика и порядок выполнения работы

1. Откройте проект, созданный вами на лабораторной работе №3,например, под именем Lab_3.er1.

Как уже упоминалось выше, различают зависимые и независимые сущности Мы задали на лабораторной работе № 3 для сущности «Учебное место» собственный уникальный ключ «код учебного места». Таким образом, сущность «Учебное место» является независимой сущностью и, связана с сущностью «Класс» неидентифицирующей связью.

2. Для того, чтобы проставить эту связь на диаграмме, щелкните указателем мыши по кнопке «NonIdentifyingRelationship»(неидентифицирующая связь) в палитре инструментов, затем щелкните по очереди по прямоугольникам сущностей «Класс» и «Учебное место» на диаграмме. Междуэтими сущностями появится пунктирная линия неидентифицирующей связи.

Посреди линии связи проставляется генерируемая по умолчанию глагольная

фраза - **R/1** (рисунок 4.8).

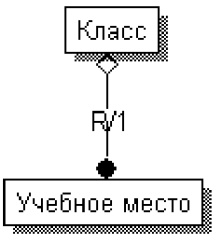


Рисунок 4.8 - Неидентифицирующая связь

3. Перейдите на уровень атрибутов и обратите внимание на то, что у сущности «Учебное место» добавился атрибут первичного ключа от сущности «Класс» и помечен буквами «FK». Говорят, что атрибут «мигрировал», aFK (foreignkey) означает, что атрибут является частью внешнего ключа (рисунок 4.9). Для идентифицирующей связи внешний ключ всегда входит в первичный ключ дочерней сущности, для неидентифицирующей не входит.Назначьте связи глагольную фразу. Для этого выделите связь, щелкнув по ней указателем мыши, затем нажмите правую кнопку мыши и в

Рисунок 4.9 - Миграция атрибутов

контекстном меню выберите пункт «RelationshipProperties… » (свойства отношений).

Общий вид окна редактора связей показан на рисунке 4.10.

Редактор связей похож на остальные редакторы объектов ER-диаграмм. В верхней части окна находится выпадающий список, содержащий полное название связи. В нашем случае осмысленная глагольная фраза для связи еще не определена, поэтому в этом поле значится «Класс R/1 Учебное место». Здесь же находятся две кнопки **New...** и **Delete**, с помощью которых можно добавить на схеме новую связь или удалить существующую. Как это сделать, мы рассмотрим чуть позже.

Далее в редакторе имеется пять страницы с закладками.

Первая закладка «General» (общие свойства). Здесь задаются основные свойства связи - глагольная фраза, тип и степень связи.

Область, озаглавленная **VerbPhrase**(глагольная фраза) содержит два

поля, в которых вводится глагольная фраза, характеризующая связь. ERwinпозволяет задать фразу для прямого и обратного направления связи: **Parent-To-Child**(Родительская-Дочерняя сущность) и **Child-To-Parent**(Дочерняя-Родительская сущность). Если для связи заданы прямая и обратная глагольные фразы, то они изображаются на диаграмме через символ «/». По умолчанию генерируется только прямая глагольная фраза (для направления **Parent-To-Child**). Щелкните указателем мыши по полю **Parent-To-Child**и введите с клавиатуры фразу «состоит из».

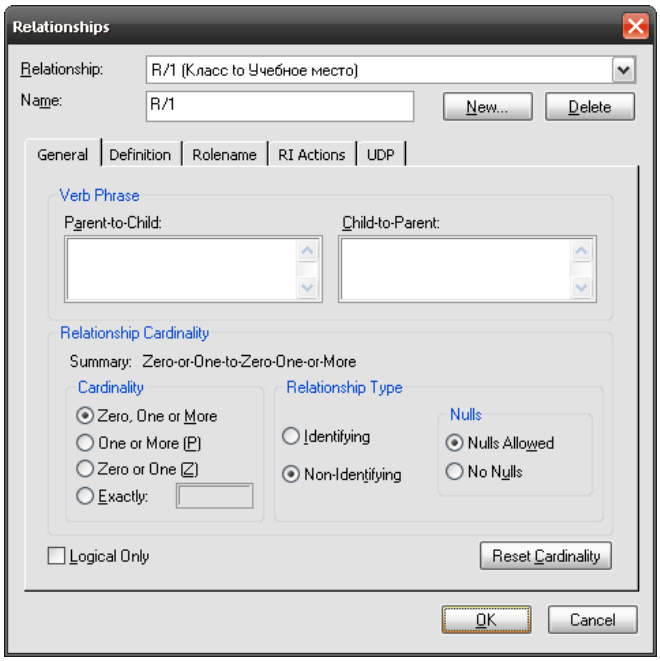


Рисунок 4.10 – Общий вид окна редактора связей

Степень связи задается в области, озаглавленной **Cardinality**. Переключатель позволяет выбрать одну из следующих степеней связи:

а) **Zero, OneorMore**(ноль, один или более) - каждый экземпляр родительской сущности связан с нулем, одним или более экземпляров дочернейсущности. Говоря «связан с нулем экземпляров», мы имеем в виду, что родительский экземпляр может быть не связан ни с одним экземпляром дочернейсущности.

б) **OneorMore (P)** (один или более) - каждый экземпляр родительской сущности связан с одним или более экземпляром дочерней сущности.

в) **ZeroorOne (Z)** (ноль или один) - каждый экземпляр родительскойсущности связан с нулем экземпляров или с одним экземпляром дочернейсущности.

г) **Exactly**(точно) - каждый экземпляр родительской сущности связанс заданным количеством экземпляров дочерней сущности. Рядом находитсяполе, где необходимо ввести это количество.

Тип связи выбирается в области **RelationshipType**. Связь может быть идентифицирующая (**Identifying**) и неидентифицирующая (**Non-Identifying**).

Кроме того, для неидентифицирующей связи задается обязательность (**Nulls**),которая показывает, может ли атрибут внешнего ключа принимать значение

NULLв таблице БД. Это свойство используется потом при генерации физической схемы базы данных.

В нашем примере, так как при анализе предметной области мы выяснили, что учебное место не может существовать отдельно от класса, установите этот переключатель в позицию **NoNulls**. Тем самым накладывается условие, что у существующего экземпляра рабочего места всегда должна быть ссылка на класс, в который оно входит.

Вторая закладка Definition(определение). На этой странице вводится определение связи. Текст определения связи, как и в случае сущности, зависит от стандартов, принятых на предприятии, и должен облегчать восприятие модели.

Третья закладка Rolename(Имя роли). Имя роли (**Rolename**) - это дополнительная характеристика, которая может присваиваться мигрирующему атрибуту первичного ключа (рисунок 4.11).

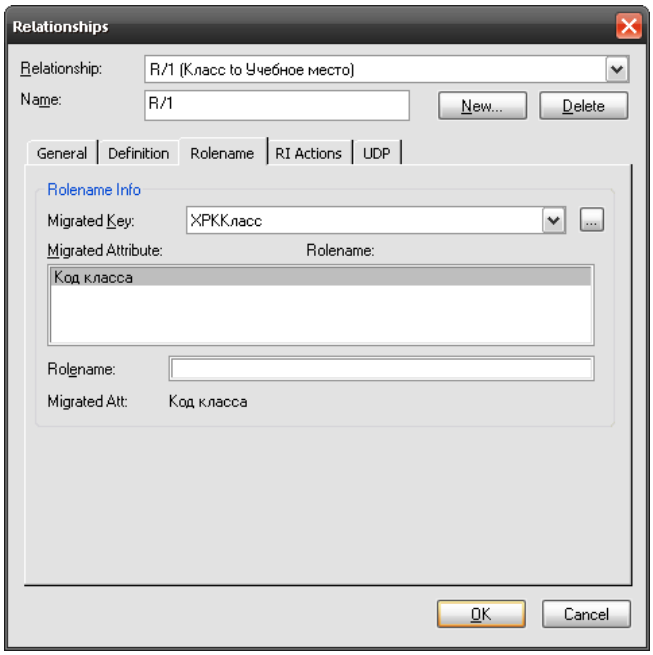


Рисунок 4.11 - Закладка **Rolename**(Имя роли)

Это особенно удобно в тех случаях, когда сущность связана со многими другими сущностями или сама с собой «циклической» связью. Тогда, если задать имя роли мигрирующему атрибуту, то в дочерней сущности он будет изображаться под этим именем. Например, если присвоить имя роли «включающий класс» атрибуту «код класса», набрав его в поле **Rolename**, то после окончания редактирования связи имя мигрировавшего атрибута «код класса» изменится на «включающий класс», а если включить режим **Rolename/Attribute**, то - «включающий класс, код класса». Режим показа ролей

переключается в контекстном меню диаграммы (рисунок 4.12).

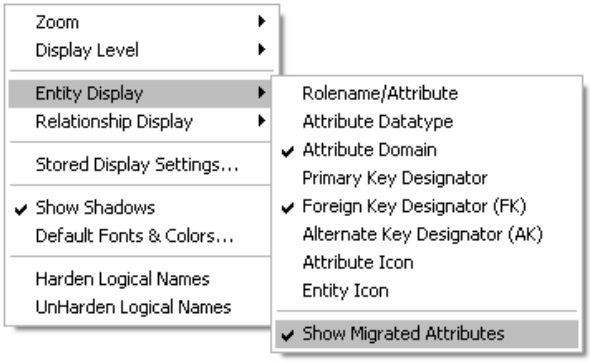


Рисунок 4.12 – Контекстное меню диаграммы для отображениямигрирующих атрибутов сущностей

Четвертая закладка «RI Actions» (Установки ссылочной целостности). Закладка предназначена для задания параметров ссылочной целостности проектируемой базы данных (рисунок 4.13).

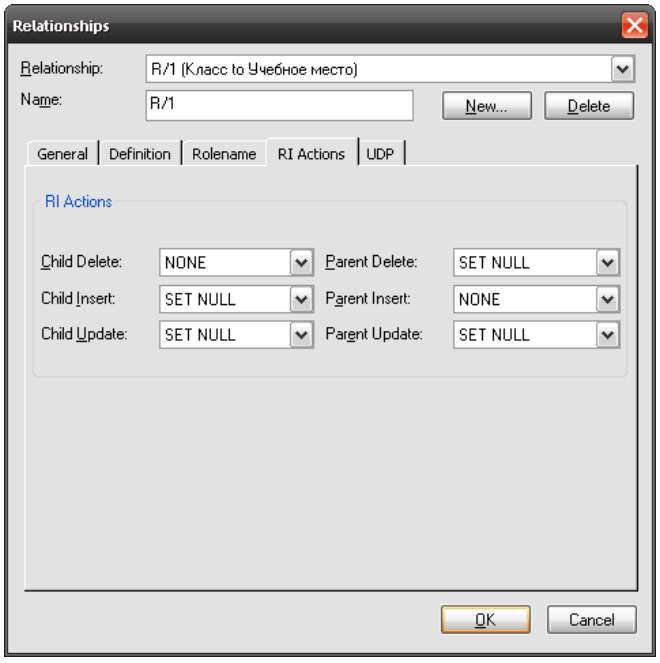


Рисунок 4.13 – Закладка **RIAction**(Установки ссылочной целостности)

Установки ссылочной целостности - это логические конструкции, которые выражают бизнес-правила использования данных. Они определяют, какие действия должна выполнить СУБД при удалении, вставке или изменении строки таблицы (экземпляра сущности). Заданные таким образом действия могут использоваться впоследствии при автоматической генерации триггеров, поддерживающих целостность данных.

Существуют следующие виды действий или правил, определяемых в логической модели (таблица 4.1).

Таблица 4.1 – Перечень действий, определяемых в логической модели

Действие	Свойства действия
RESTRICT	Запрет удаления, вставки или изменения экземпляра сущности
CASCADE	При удалении экземпляра родительской сущности удаление всех экземпляров дочерней сущности, ссылающихся на удаляемый родительский экземпляр
SET NULL	При удалении экземпляра родительской сущности атрибутам внешнего ключа всех экземпляров дочерней сущности, присваивается значение NULL
SET DEFAULT	То же самое, что и в предыдущем случае, только вместо значения NULLприсваивается значение по умолчанию
NONE	Никаких действий не предпринимается

Эти правила задаются на вставку, удаление и изменение экземпляра как родительской, так и дочерней сущности. Таким образом, каждая связь должна обладать набором из шести правил, которые вводятся в поля, объединенные общим заголовком **RIActions**. При добавлении связи в диаграмму ER-winпо умолчанию устанавливает для нее набор правил, которые можно редактировать в диалоге **ModelProperties**(Свойства модели) на вкладке **RIDefaults**, вызывающемся путем выбора из главного меню **Model**команды**ModelProperties**(рисунок 4.14).

Правила, присваиваемые связи по умолчанию, можно изменить, выбрав нужное значение из выпадающего списка (см. рисунок 4.14). При нажатии на кнопку **Rebind**(переназначить) новые установки умолчаний переносятся втекущую модель, если же просто выйти из диалога, не делая переназначения,

то измененные установки будут влиять только на новые модели.

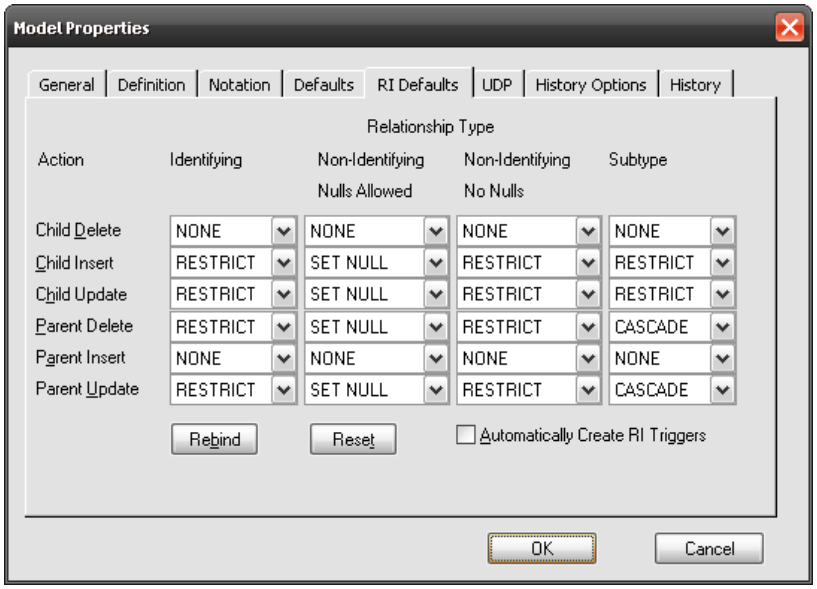


Рисунок 4.14 – Закладка **RIDefaults**диалогового окна **ModelProperties**

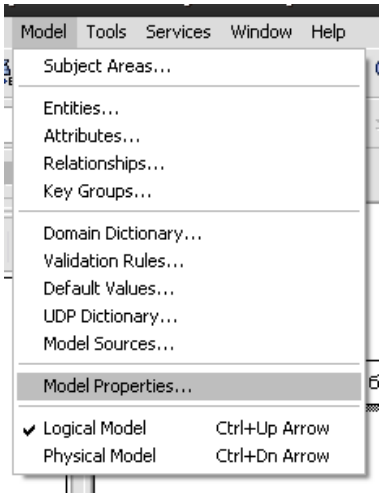


Рисунок 4.15 – Порядок вызова диалогового окна **ModelProperties**

Каждый тип связи имеет, в зависимости от вида действия, свой набор допустимых правил, приведенный в таблице 4.2.

Таблица 16.2 - Набор допустимых правил для различных типов связей

Вид действия	Тип связи (Relationship Type)			
	Идентифицирующая (Identifying)	Неидентифицирующая(NonIdentifying, Nulls Allowed)	Неидентифицирующая(NonIdentifying, No Nulls)	Категориальная связь (Subtype)
ChildDelete(удаление дочернего объекта)	RESTRICT, CASCADE, NONE	RESTRICT, CASCADE, NONE	RESTRICT, CASCADE, NONE SET NULL,	RESTRICT, CASCADE, NONE SET
ChildInsert(вставка дочернего объекта)	RESTRICT, CASCADE, NONE	RESTRICT, CASCADE, NONE	RESTRICT, CASCADE, NONE SET NULL,	RESTRICT, CASCADE, NONE SET
ChildUpdate (изменение	RESTRICT, CASCADE, NONE	RESTRICT, CASCADE, NONE	RESTRICT, CASCADE, NONE SET	RESTRICT, CASCADE, N6NE SET
ParentDelete (удаление родительского объ-	RESTRICT, CASCADE, NONE	RESTRICT, CASCADE, NONE	RESTRICT, CASCADE, NONE SET NULL, SET	RESTRICT, CASCADE, NONE SET DEFAULT

ParentInsert(вставка родитель-ского объ-	RESTRICT, CASCADE, NONE	RESTRICT, CASCADE, NONE	RESTRICT, CASCADE, NONE SET NULL,	RESTRICT, CASCADE. NONE SET
ParentUp-date (изменение родитель-	RESTRICT, CASCADE, NONE	RESTRICT, CASCADE, NONE	RESTRICT, CASCADE, NONE SET NULL, SET	RESTRICT, CASCADE, NONE SET DEFAULT

Установки для связи «Класс - Учебное место», присвоенные по умолчанию, запрещают вставку и изменение экземпляра дочерней сущности, а также удаление и изменение родительской сущности. Это означает, что недопускается удаление или изменение класса, если в нем имеются учебныеместа, а также ввод учебного места без указания класса или со ссылкой нанесуществующий класс. Тем самым мы выполнили условие, по которомуучебное место может существовать только в составе класса.

Пятая закладка UDP(Параметры, устанавливаемые пользователем). Закладка - **UDP**, как и у предыдущих объектов диаграммы, позволяет присвоить связи свой набор пользовательских свойств.

Итак, мы создали неидентифицнрующую связь между сущностями «Класс» и «Учебное место» с условием **NoNulls**. Очевидно, связь того же типа должна существовать между сущностями «Тип оборудования» и «Единица оборудования», так как единица оборудования обязательно должна иметь тип. Внесите эту связь в диаграмму, выполнив те же действия, что и в предыдущем случае. Вызовите редактор связей и измените глагольную фразу на «описывает», остальные установки связи оставьте неизменными. Обратите внимание, что атрибут «код типа оборудования» мигрировал в состав неключевых атрибутов сущности «Учебное место» (рисунок 4.16).

Рассмотрим теперь связь между сущностями «Учебное место» и «Единица оборудования». Как мы выяснили при обследовании предметной области, единицы оборудования образуют некий фонд комплектующих, часть из которых установлена в учебные места. Другая часть комплектующих может находиться на складе, быть неисправной и дожидаться списания и т. п., то есть существовать отдельно от учебного места. Таким образом, сущности «Учебное место» и «Единица оборудования» не зависят друг от друга, и должны быть ассоциированы неидентифицирующей связью.

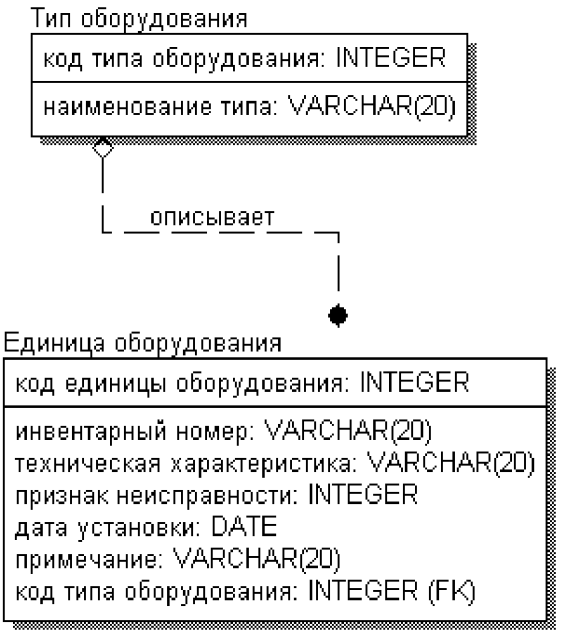


Рисунок 4.16 – Атрибут «код типа оборудования» мигрировал в состав неключевых атрибутов сущности «Учебное место»

Рассмотрим теперь связь между сущностями «Учебное место» и «Единица оборудования». Как мы выяснили при обследовании предметной области, единицы оборудования образуют некий фонд комплектующих, часть из которых установлена в учебные места. Другая часть комплектующих может находиться на складе, быть неисправной и дожидаться списания и т. п., то есть существовать отдельно от учебного места. Таким образом, сущности «Учебное место» и «Единица оборудования» не зависят друг от друга, и должны быть ассоциированы неидентифицирующей связью.

5. Выберите неидентифицнрующую связь в палитре инструментов и внесите ее в диаграмму, выбрав «Учебное место» в качестве родительской сущности, а «Единицу оборудования» - дочерней. В редакторе связи измените глагольную фразу **Parent-to-Child**на «состоит из». Неидентифицирующаясвязь имеет две разновидности - допускающая значения **NULL**(**NullsAllowed**) и не допускающая (**NoNulls**). По умолчанию выбирается разновидность **NullsAllowed**, оставьте это без изменений. Такая установка означает, что у экземпляра сущности «Единица оборудования» поля внешнего ключамогут иметь нулевое значение, то есть отсутствовать указание на экземпляр «Учебного места». Таким образом, единица оборудования может существовать «сама по себе».

После установки связей диаграмма на уровне сущностей будет иметь вид, показанный на рисунке 4.17.

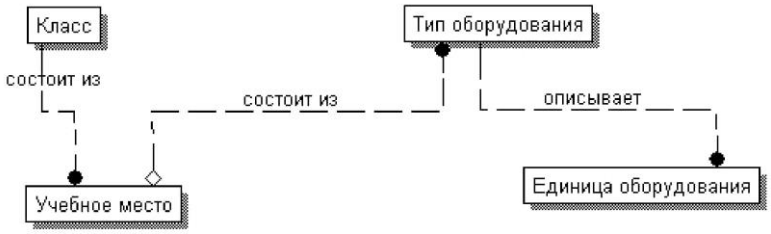


Рисунок 4.17 - ER-диаграмма после установки связей

В зависимости от типа, связи изображаются на диаграмме по-разному. Графическое изображение зависит и от принятой методологии построения диаграммы. Например, в нотации IDEF1X, используемой в этом примере, принято изображение связей, показанное на рисунке 4.18.

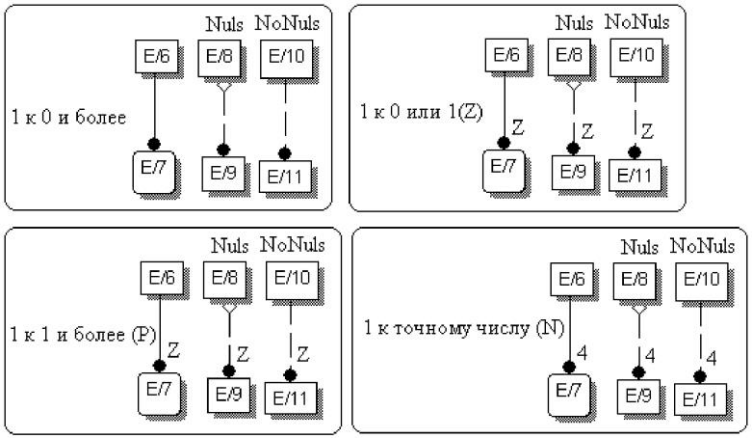


Рисунок 4.18 - Изображение связей в нотации **IDEF1X**Помимо типа связи, на диаграмме могут отображаться и установки ссылочной целостности - для этого необходимо выбрать в контекстном менюдиаграммыпункт**RelationshipDisplay**иподпункт**ReferentialIntegrity**(рисунок4.19).

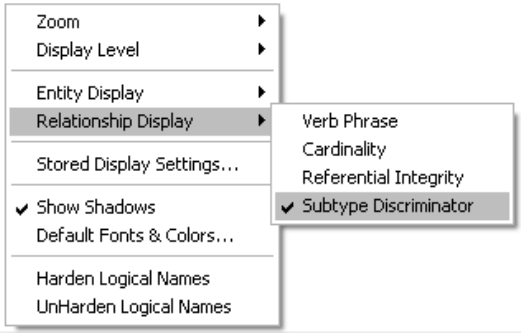


Рисунок 4 .19 – Выбор команд для отображения установок ссылочной целостности

Обозначение ссылочной целостности на схеме представляет собой две алфавитные группы, разделенные символом двоеточия «**:**». Первый символ обозначает действие, к которому относится правило целостности: **D**- удаление (**delete**), **I**- вставка (**insert**), **U**- изменение (**update**).

Вторыягруппаобозначаетправило: **R- RESTRICT**, **C- CASCADE**, **SN-**

SETNULL, **SD- SETDEFAULT**. Таким образом, запрет удаления обозначается **D:R**, а установка **NULL**при изменении - **U:SN**. Обозначения проставляются у родительского или дочернего конца связи, в зависимости от того, к какой сущности они относятся. С включенными установками ссылочной целостности диаграмма выглядит так, как показано на рисунке 4.20.

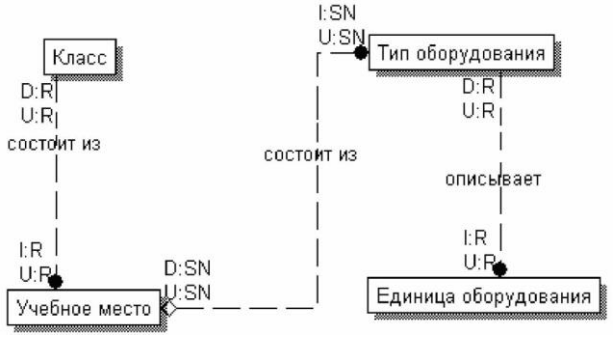


Рисунок 4.20 - ER-диаграмма с включенными установками ссылочной целостности

На диаграмме теперь определена одна из объектных областей, которых мы выделили четыре - материальное обеспечение процесса обучения.Рассмотрим другие объектные области на последующих упражнениях.

6. На вкладке «Уровень сущностей» сохраните модель, например, под именем Lab_4.er1.

7. Повторив пункты 1÷6 настоящих указаний, выполните индивидуальное задание по определению связей между сущностями в ERwindля указанной предметной области.

Варианты заданий

Используя результаты выполнения лабораторной работы № 3, необходимо выполнить определению связей между сущностями в ERwindля указанной предметной области. Номер варианта соответствует номеру рабочего места за компьютером и должен совпадать с номером варианта индивидуального задания к лабораторной работе № 2 (см. таблицу 2.3).

Содержание отчета и его форма

Отчет по лабораторной работе оформляется в виде текстового документа и должен включать:

1. Название лабораторной работы.
2. Цель лабораторной работы.
3. Формулировку индивидуального задания и результат его выполнения.
4. Краткие выводы по результатам выполнения лабораторной работы.

Вопросы для защиты работы

1. Как различают зависимые и независимые сущности на диаграмме

ERwin?

2. Какая связь между сущностями называется неидентифицирующей?
3. Что такое физическая и логическая модель данных?
4. Какая связь между сущностями называется идентифицирующей?Поясните смысл утверждения о том, что некоторый атрибут «мигрировал»?
5. Что обозначает символика «FK» на диаграмме ERwin?
6. Какими возможностями обладает редактора связей?
7. Каково изображение связей в нотации IDEF1X?
8. Что такое супертип?
9. Что такое подтип?
10. Каков набор допустимых действий или правил, определяемых в логической модели для идентифицирующей связи?
11. Каков набор допустимых действий или правил, определяемых в логической модели для неидентифицирующей связи?
12. Как производится обозначение ссылочной целостности на диаграмме ERwin?
13. Какие связи между сущностями были использованы при выполнении индивидуального задания?
14. Какова методика выполнения индивидуального задания?
15. Какие основные выводы можно сделать по результатам выполнения индивидуального задания?

Защита лабораторной работы

Оформленный, в соответствии с требованиями ГОСТов, отчет представляется студентом преподавателю для проверки и последующей защиты. Защита отчета по лабораторной работе производится студентом только индивидуально.

В ходе защите лабораторной работы студент отвечает на вопросы преподавателя (поясняет методику выполнения индивидуального задания, отвечает на контрольные вопросы и т.д.).

Отчет, оформленный с отступлениями от требований ГОСТов, небрежно и неаккуратно к защите не принимается.

Лабораторная работа № 5

Определение атрибутов и связей между сущностями, входящими вобъектные области в ERwin

Цель и содержание: привить студентам навыки определения атрибутов и связей между сущностями, входящими в объектные области модели ERwin.

Формируемые компетенции: ПК-2, ПК-6. ПК-7.

Теоретическое обоснование

В ходе выполнения лабораторных работ № 3 и № 4 были определены атрибуты и установлены связи между сущностями, входящими в объектную область «Материальное обеспечение обучения». Методика определения атрибутов и связей между сущностями, входящими в другие объектные области аналогична. Рассмотрим порядок определения атрибутов и установления связей между сущностями, входящими в объектные области «Методическое обеспечение», «Учебный процесс», «Персонал и учащиеся» и «Занятия в классе». Подобное рассмотрение позволит, с одной стороны, получить законченную модель схемы данных для разрабатываемой информационной подсистемы предприятия, а с другой стороны, закрепить навык определения атрибутов и связей между сущностями в среде ERwin.

Аппаратура и программное обеспечение

Аппаратура. Для выполнения лабораторной работы необходимы:

- персональный суперкомпьютер i7-5930K/ 8D8192D4-2133/ K2200/ 2K40M;
- рабочие станции: монитор 29 + клавиатура + мышь. ПК MDT750/ i75930K/ 4D8192D42133";
- аппаратный сетевой обучающий комплекс мод.VDCN-1502.

Программное обеспечение. Для выполнения лабораторной работы необходима операционная система WINDOWS 7/8/10 и CASE-средство ERwin7.3.

Указания по технике безопасности

При выполнении лабораторной работы**запрещается**:

1. Самостоятельно производить ремонт персонального компьютера, а так же, установку и удаление имеющегося программного обеспечения.
2. Нарушать общепринятые правила техники безопасности при работе с электрооборудованием, в частности, касаться электрических розеток металлическими предметами и т.д.

3. Принимать пищу, напитки и сорить на рабочем месте пользователя персонального компьютера.

В случае неисправности персонального компьютера необходимо **немедленно** сообщить об этом обслуживающему персоналу лаборатории (системному администратору, оператору).

Методика и порядок выполнения работы

1. Откройте проект, созданный вами на лабораторной работе № 4, например, Lab_4.er1 и создайте сущности для объектной области «Методическое обеспечение». Сущности, относящиеся к этой области, приведены в таблице 5.1. Внесите их в диаграмму и задайте атрибуты, предварительно создав соответствующие домены. Рассмотрим связи между этими сущностями.

Рассмотрим более подробно сущность «Тема типового курса». Как было выяснено при анализе предметной области, темы не только могут объединяться в типовые курсы, но и в разделы этих типовых курсов. Каждая тема может входить в какой-нибудь раздел и/или быть заголовком раздела. Физически это сводится к тому, что каждый ее экземпляр должен содержать ссылку на тему - заголовок включающего раздела. Поэтому сущность «Тема типового курса» должна позволять строить иерархическую структуру из своих экземпляров, то есть быть ассоциирована «сама с собой». Связь такого типа называется циклической или «рыболовным крючком» (**fishhook**), и она является разновидностью неидентифицирующей связи. Чтобы задать эту связь, выберите в палитре инструментов «неидентифицирующую связь», затем щелкните по сущности «Тема типового курса» два раза, выбрав ее одновременно и родительской, и дочерней. Связь изображается в виде замкнутого контура (рисунок 5.1). Присвойте связи глагольную фразу «объединяет в раздел».

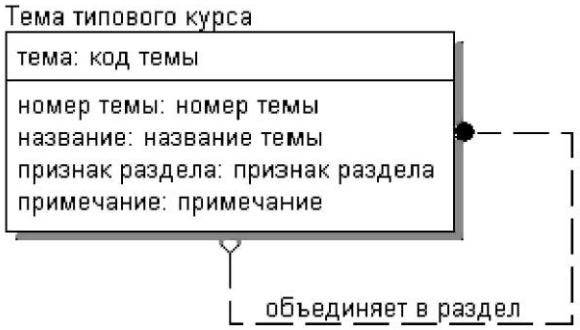
Таблица 5.1 - Сущности, входящие в объектную область«Методическое обеспечение»

Сущность	Атрибут	Ключ	Имя домена		Тип
			логическое	физическое	
Типовой курс	курс		код курса	tcoursid	число
	название		название курса	tcoursname	строка
	примечани		примечани	t note	стро

Тема типового курса	тема		код темы	tthemeid	число
	номер		номер	t theme no	числ
	название		название темы	tthemename	строка
	признак		признак	tissection	число
	примечани		примечани	tnote	стро
Индивидуальный план	план		код плана	t_pian_id	число
	дата составления		дата составления	tcompiledate	дата
	примечани		примечани	fnofe	стро
Плановое занятие	занятие		код планового занятия	tboxid	число
	номер п/п		номер планового занятия	tboxno	число

К сожалению, реализация сущности в том виде, как изображена на рисунке 5.1, нас не может устроить, так как у сущности не появился указатель на включающий тему раздел. Таким разделом должен был стать мигрировавший атрибут ключа «тема», однако в списке атрибутов он отсутствует, так

как не может появиться в нем под одним и тем же именем. В таких случаях необходимо обязательно задавать мигрирующему атрибуту имя роли. Войдите в редактор связи и на странице «Rolename/RI Actions»



задайте атрибуту «тема» имя роли «объединяет в раздел» (рисунок 5.2)

Рисунок 5.1 - Сущность «Тема типового курса»: Этап I- создание связи циклического типа

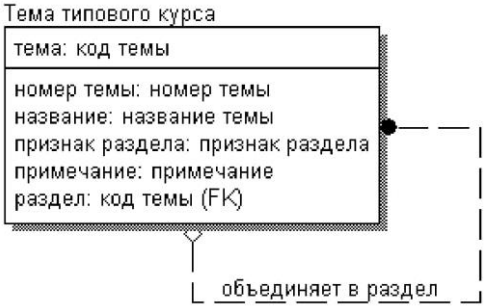


Рисунок 5.2 - Сущность «Тема типового курса»: Этап II - Присвоение роли мигрирующему атрибуту

2. Задайте остальные связи для сущностей этой области в соответствии с данными, приведенными в таблице 5.2.

Таблица 5.2 - Связи для сущностей, входящих в объектную область«Методическое обеспечение»

Родительская сущность	Дочерняя	Тип связи	Степень связи	Нулевые значения	Глагольная фраза
Индивидуальный план	Плановое занятие	Неидентифи-цирующая	0 или 1 к 0,	No NULLS	состоит из
Типовой курс	типового курса	Неидентифи-цирующая	0 или 1 к 0,	NULL	состоит из
Родительская сущность	Дочерняя сущность	Тип связи	Степень связи	Нулевые значения	Глагольная фраза
Тема типового курса	Плановое занятие	Неидентифи-цирующая	0 или 1 к 0,	NULL	предназначена для

После внесения всех элементов диаграмма области методического обеспечения должна выглядеть, как показано на рисунке 5.3.

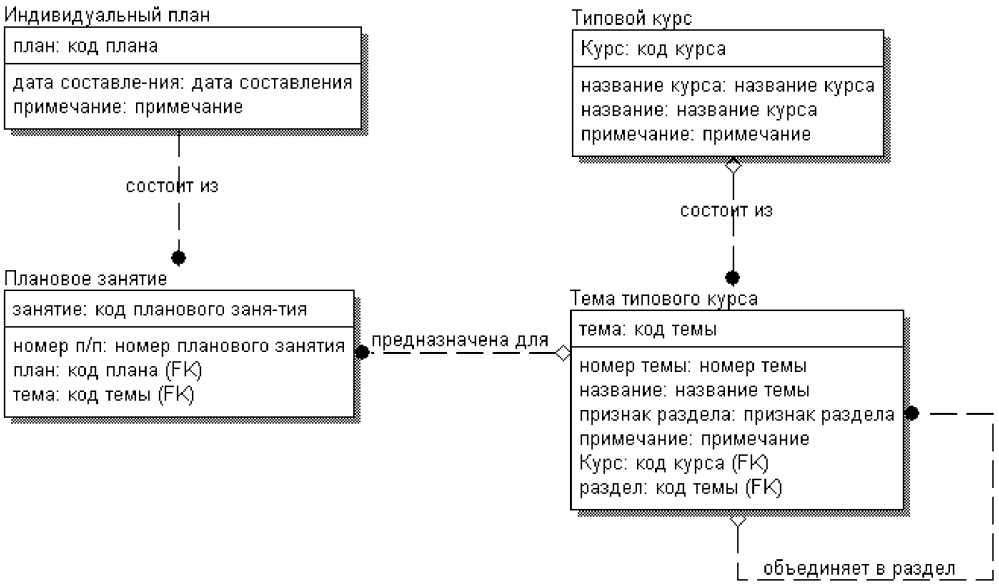


Рисунок 5.3 - Диаграмма объектной области «Методическое обеспечение»

3. Создайте сущности для объектной области «Учебный процесс». Сущности, относящиеся к этой области, приведены в таблице 5.3.

Атрибут «академ. час» в таблице 5.3 представляет собой порядковый номер занятия в пределах учебного дня, поэтому для описания экземпляра сущности обязательно необходимо указывать учебный день. Поэтому сущности «Учебный день» и «Академический час» связаны идентифицирующей связью, причем родителем является «Учебный день» (рисунок 5.4).

Таблица 17.3 - Сущности, входящие в объектную область «Учебный процесс»

Сущность	Атрибут	Ключ	Имя домена		Тип
			логическое	физическое	
Учебный день	учебный день		учебный день	t_workday	Дата
	примечание		примечание	t_note	строка
Академический час	академ. час		номер академ. часа	tworkhourid	число
	начало		время суток	t time	время
	конец		время суток	t time	время

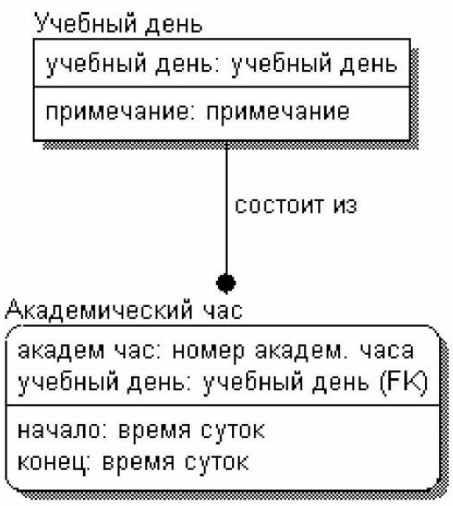


Рисунок 5.4 - Диаграмма объектной области «Учебный процесс»

4. Создайте сущности для объектной области «Персонал и учащиеся». Сущности, относящиеся к этой области, приведены в таблице 5.4.

В указанной предметной области персонал учебных курсов представлен только одной сущностью - «Преподаватель», однако вполне возможно, что со временем понадобится ввести в базу данных и другие виды работников, например, техников по обслуживанию учебных мест, методистов для составления программ и т.п.

Учащиеся представлены сущностью «Учащийся». Сущности «Преподаватель» и «Учащийся» должны иметь сходные атрибуты - фамилию, имя и отчество, в то же время каждая из этих сущностей обладает и своими, присущими только ей атрибутами. Например, для преподавателя это адрес, телефон, дата приема на работу, а для учащегося - дата начала обучения, датаокончания, дата выдачи удостоверения

Таблица 17.4 - Сущности, входящие в объектную область«Персонал и учащиеся»

Сущность	Атрибут	Ключ	Имя домена		Тип
			логическое	физическое	
Физическое лицо	код физического		код физического	t_person_id	число
	фамилия		фамилия	t last name	стро
	имя		имя	t first name	стро

	отчество		отчество	t middle name	стро
	категория		категория	t category	числ
Преподаватель	адрес		адрес	taddress	стро
	телефон		телефон	ttelephone	стро
	дата		дата приема	tenterdate	дата
	примечани		примечание	tnote	стро
Учащийся	дата		дата начала	tstartdate	дата
	дата окончания		дата окончания	tenddate	дата
	дата		дата выдачи	tcertifdate	дата
	примечани		примечание	tnote	стро

Для представления сущностей, имеющих общие характеристики, существует особый тип объединения, называемый иерархией наследования. Для образования иерархии необходимо наличие родового предка. В нашем случае это может быть сущность «Физическое лицо», которая и будет содержать атрибуты «фамилия», «имя» и «отчество». Специфическая для каждого типа информация будет расположена в сущностях «Преподаватель» и «Учащийся», которые называются категориальными сущностями или потомками.

Для того чтобы отличать одну категориальную сущность от другой, родовой предок должен содержать специальный атрибут – дискриминатор.

Сущности, составляющие иерархию наследования, ассоциируются связью категориального типа (subtyperelationship).

Введите в диаграмму сущности и атрибуты, приведенные в следующей таблице 5.4, так же, как это было сделано для предыдущих областей. Следует иметь в виду, что часть доменов уже определены в модели - адрес, телефон, примечание, поэтому их вводить не надо.

Обратите внимание, что у категориальных сущностей не задан первичный ключ - им станет атрибут внешнего ключа, который мигрирует из родового предка. После ввода сущностей следует определить связи между ними.

Это делается следующим образом:

а) выделите в палитре инструментов категориальную связь

б) щелкните указателем мыши по родовому предку - «Физическое лицо», а затем по одному из потомков - «Преподаватель».

в) повторно выделите в палитре инструментов категориальную

связь.Щелкните по символу связи на диаграмме, а затем по второму потомку -«Учащийся» (рисунок 5.5).

Иерархия категорий может быть полной или неполной.

В полной категории одному экземпляру родового предка соответствует экземпляр в одном из потомков. В неполной категории у родового предка могут существовать экземпляры, не имеющие соответствующих экземпляров в потомках.

Для редактирования категориальной связи необходимо выделитьсимвол связии в контекстном меню выбрать пункт **SubtypeRelationship...** Здесьследует выбрать в списке дискриминатор (рисунок 5.6), а также установить тип связи - полная (**Complete**) и неполная (**Incomplete**), Укажите дискриминатор «категория» и нажмите кнопку **ОК**. Имя дискриминатора появится в диаграмме рядом с символом связи (рисунок 5.7).

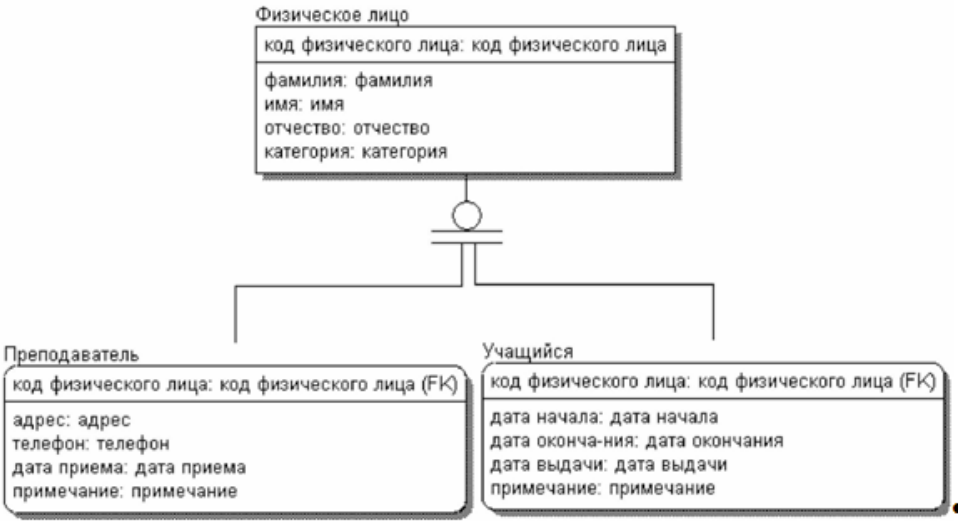


Рисунок 5.5 - Связь категориального типа

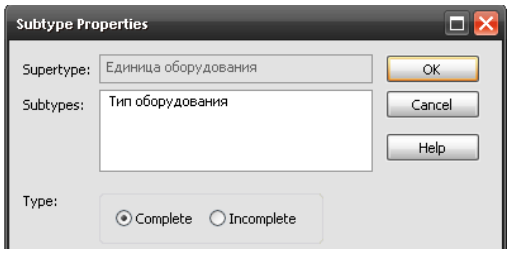


Рисунок 5.6 - Редактор свойств категориальной связи

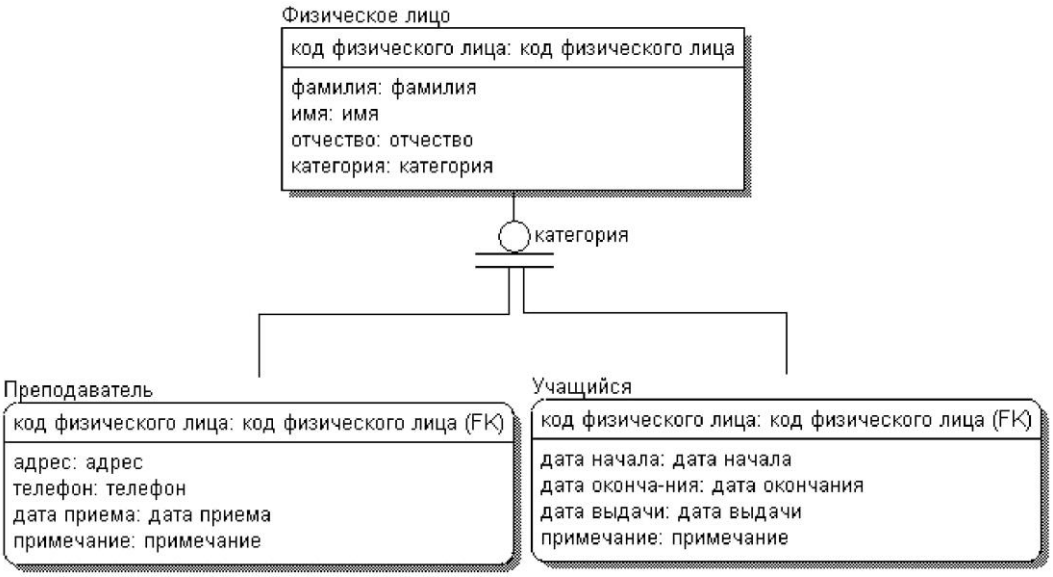


Рисунок 5.7 - Связь категориального типа с именем «категория»

Теперь на диаграмме (рисунок 5.8) отображены все объектные области, и она выглядит, как совокупность не связанных друг с другом групп сущностей.

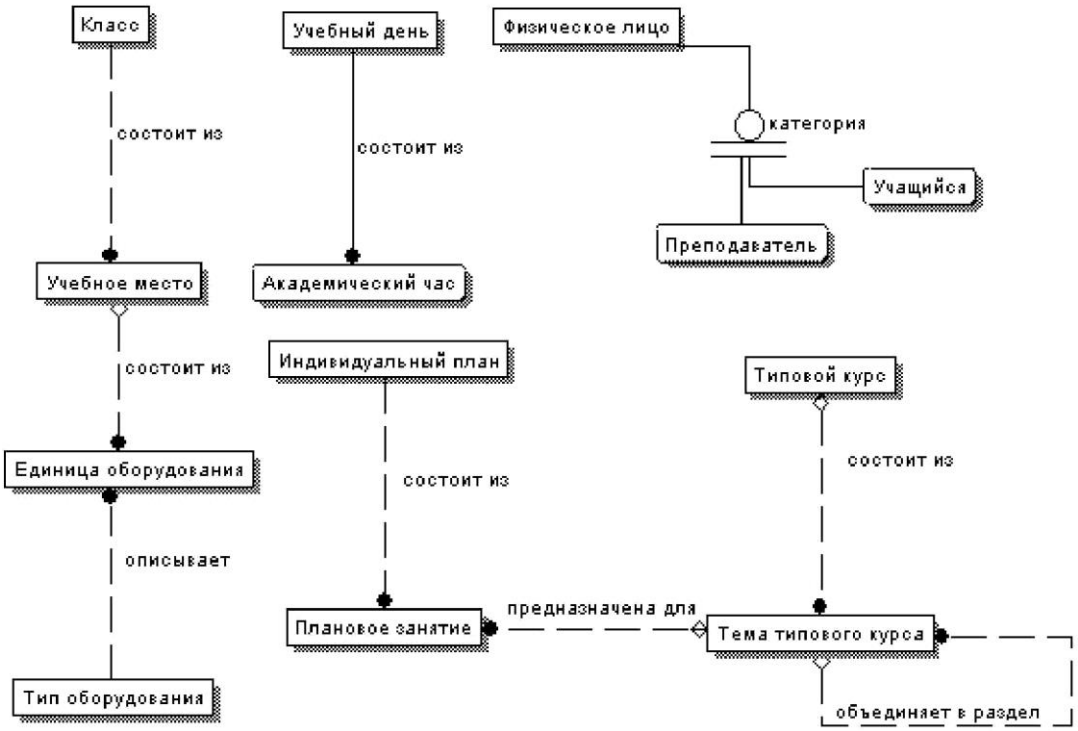


Рисунок 5.8 - ER-диаграмма после определения объектных областей

5. Анализ ER-диаграммы, представленной на рисунке 5.8, свидетельствует о том, что на ней недостает еще одной сущности, объединяющей все объектные группы. Эта сущность (ранее мы назвали ее «Занятие в классе») должна содержать информацию о том, что на некотором учебном месте в некоторый академический час, определенный преподаватель провел со студентом плановое занятие. Кроме того, необходимо также хранить информацию о том, проведено ли занятие, и если оно не было проведено, то по какой причине. Для этого введем еще одну сущность, которая будет называться «Причина отмены» (таблица 5.5) Таблица 5.5 - Сущности, входящие в объектную область «Занятие в классе»

Сущность	Атрибут	Ключ	Имя домена		Тип
			логическое	физическое	
Занятие в классе	статус		статус	t status	числ
	примечание		примечание	t note	стро
Причина отмены	код отмены		код отмены	tmisfireid	числ
	наименование		наименование причины	t misfire name	строка

Хотя «Занятие в классе» является ключевой связующей сущностью модели, собственных атрибутов у нее всего два - все остальные атрибуты представляют собой мигрировавшие ключи связанных с ней сущностей (таблица 17.6).

Таблица 17.6 - Связи для сущностей, входящих в объектную область «Занятие в классе»

Родительская сущность	Дочерняя	Тип связи	Степень связи	Нулевые	Глагольная фраза
Плановое занятие	Занятие в классе	Неидентифи-цирующая	0 или 1 к 0, 1 (Z)	NULL	запланировано для
Учебное место	Занятие в классе	Идентифицирующая	0 или 1 к 0, 1 или более	-	предоставляется для
Академический час	Занятие в классе	Идентифицирующая	0 или 1 к 0, 1 или более	-	задает время проведения
Физическое лицо	Занятие в классе	Неидентифи-цирующая	0 или 1 к 0, 1 или более	NULL allowed	провел

Родительская сущность	Дочерняя	Тип связи	Степень связи	Нулевые	Глагольная фраза
Физическое лицо	Индивидуальный план	Неидентифи-цирующая	0 или 1 к 0, 1 или более	NO NULLS	занимается по
Физическое лицо	Индивидуальный план	Неидентифи-цирующая	0 или 1 к 0, 1 или более	NULL allowed	составил
Причина отмены	Занятие в классе	Неидентифи-цирующая	0 или 1 к 0, 1 или более	NULL allowed	поясняет отмену

Обратите внимание, что сущности «Плановое занятие» и «Занятие вклассе» имеют степень связи «один-или-нуль-к-одному-или-нуль» (Z) Это является следствием того, что «Занятие в классе» представляет собой «клеточку» в расписании, в которую вписывается тема занятия, и больше одной темы в нее вписать нельзя.

«Физическое лицо» в диаграмме связано с «Индивидуальным планом» двумя связями - это учащийся, занимающийся по этому плану и преподаватель, который его составил, поэтому необходимо назначить мигрирующему ключу «код физ. лица» имя роли:

а) для неидентифицирующей связи (**NoNulls**) - «учащийся»;

б) для неидентифицирующей связи (**Nullsallowed**) - «составил».

Таким образом, составитель индивидуального плана может быть неуказан. Кроме того, для улучшения читаемости диаграммы назначьте у связи «Физическое лицо» - «Занятие в классе» мигрирующему атрибуту «код физ. лица» имя роли «провел». После введения этих изменений диаграмма должна выглядеть, как показано на рисунках 5.9 и 5.10. На рисунках 5.11 и 5.12 показана модель, приведенная на рисунке 5.10, но в более крупном масштабе (по отдельным листам).

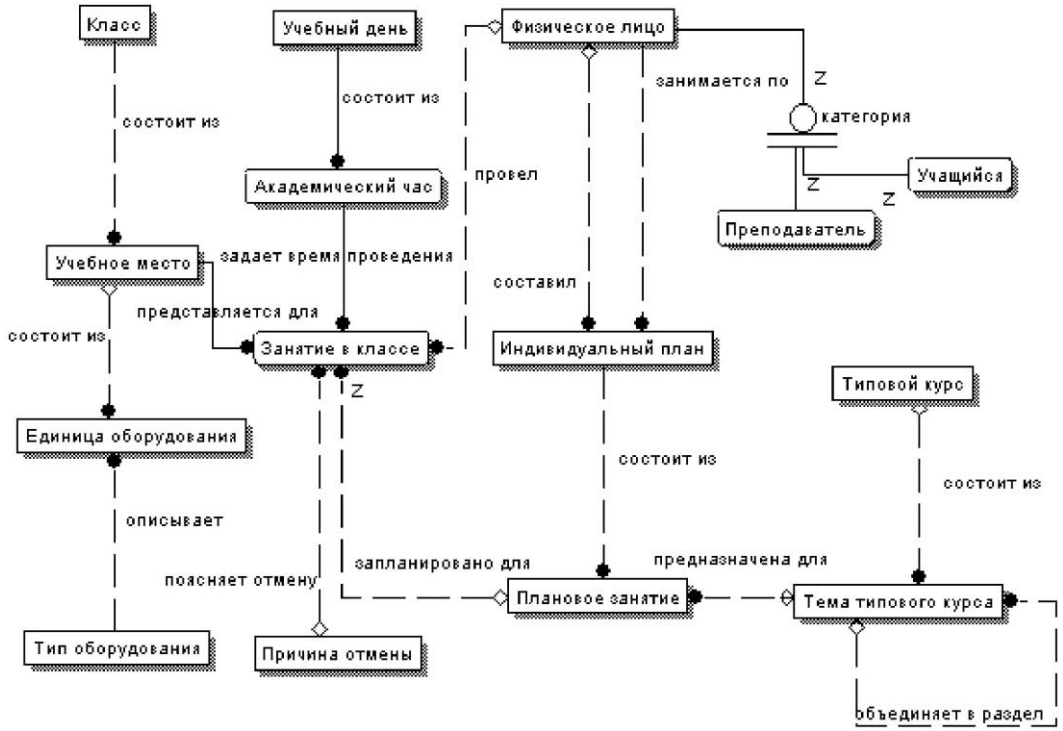


Рисунок 5.9 - ER-диаграмма после введения сущности «Занятие в классе»

На вкладке «Уровень сущностей» сохраните модель, например, под именем Lab_5.er1.

6. Выполните индивидуальное задание по определению связей между сущностями входящими в объектные области модели ERwin для указанной предметной области.121

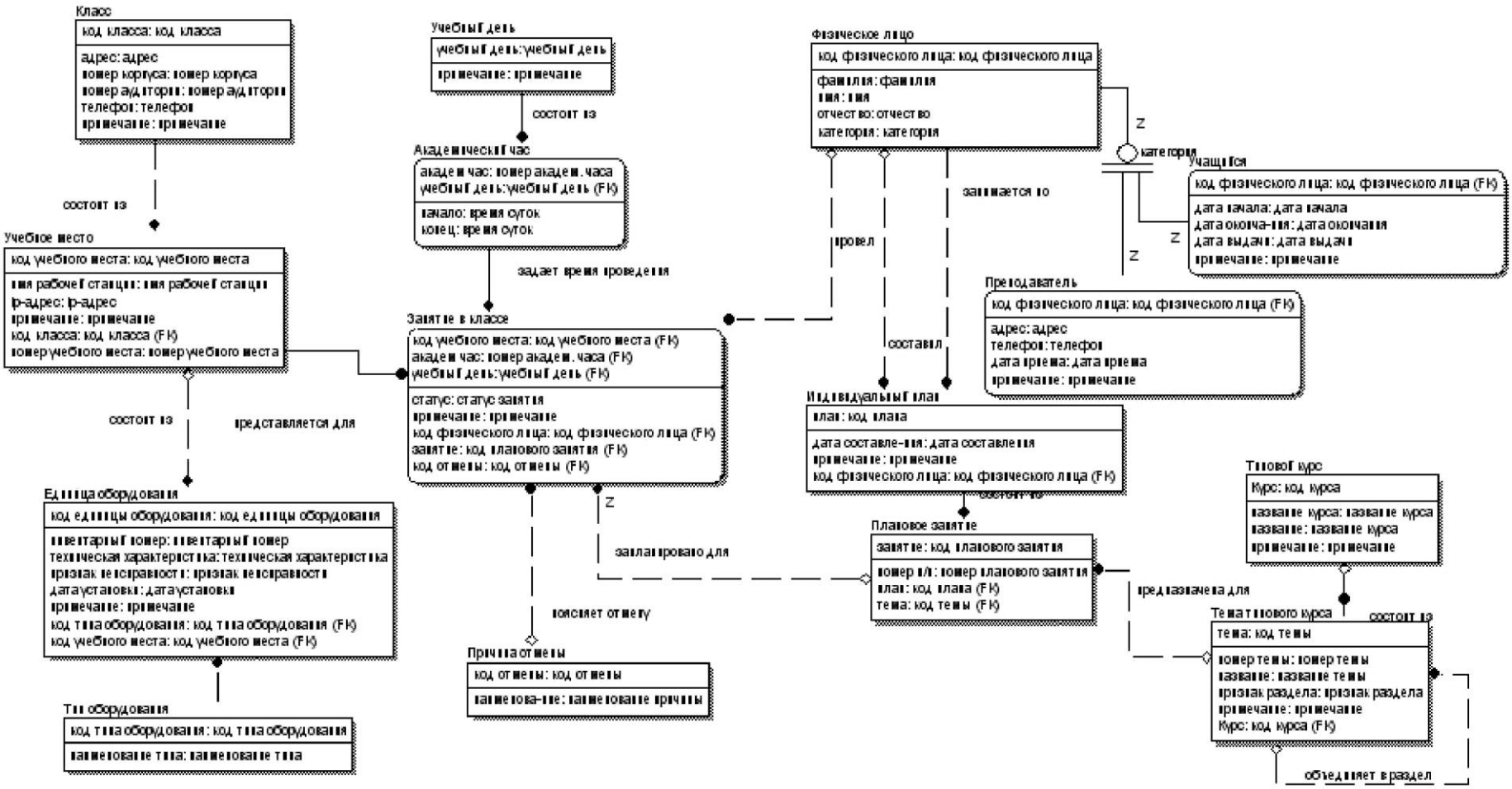


Рисунок 17.10 - ER-диаграмма после введения сущности «Занятие в классе». Уровень атрибутов

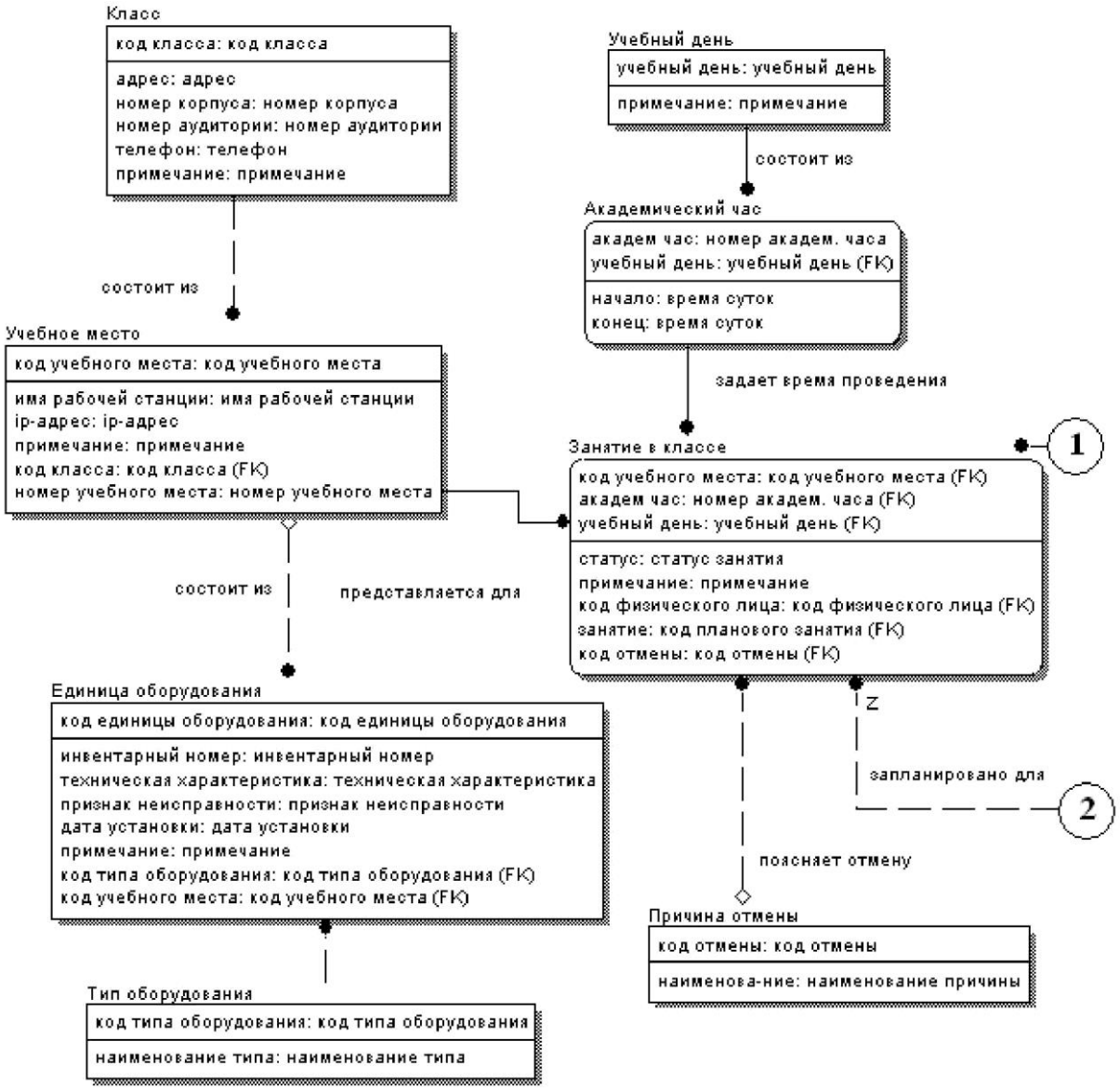
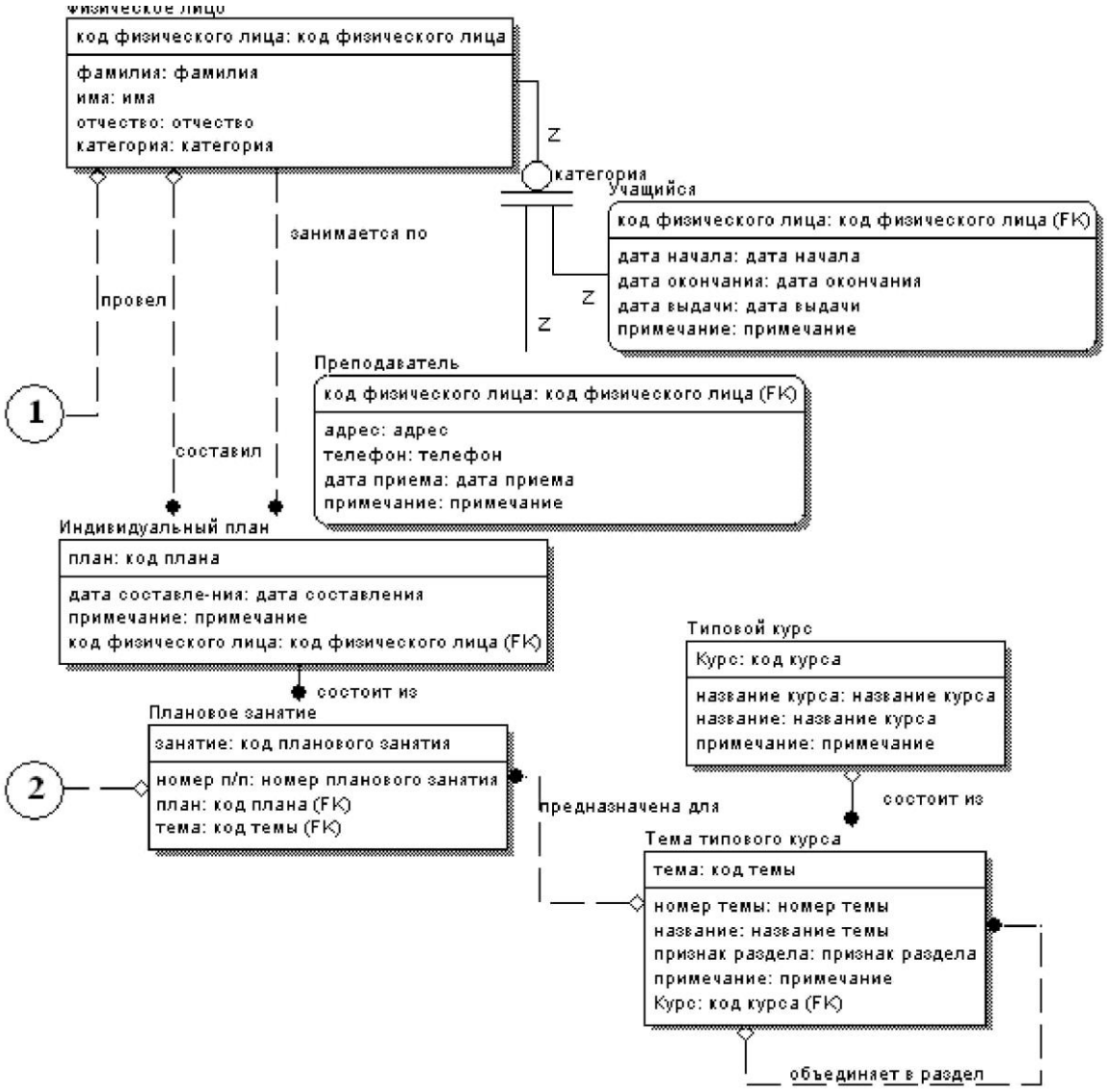


Рисунок5.11 –ER-диаграмма после введения сущности «Занятие в классе». Уровень атрибутов (лист 1)

Варианты заданий

Используя результаты выполнения лабораторной работы № 4, необходимо определить атрибуту и установить связей между сущностями, входящими в объектные области модели ERwinдля указанной предметной области. Номер варианта соответствует номеру рабочего места за компьютером и должен совпадать с номером варианта индивидуального задания к лабораторной работе № 2 (см. таблицу 2.3).

Рисунок 5.12 –ER-диаграмма после введения сущности «Занятие в классе».



Уровень атрибутов (лист 2)

Содержание отчета и его форма

Отчет по лабораторной работе оформляется в виде текстового документа и должен включать:

1. Название лабораторной работы.
2. Цель лабораторной работы.
3. Формулировку индивидуального задания и результат его выполнения.
4. Краткие выводы по результатам выполнения лабораторной работы.

Вопросы для защиты работы

1. Какой может быть иерархия категорий сущностей на ER-диаграмме?
2. Каковы возможности редактора свойств категориальной связи?
3. Что такое физическая и логическая модель данных?
4. Какая связь между сущностями называется идентифицирующей?
5. Какая связь между сущностями называется неидентифицирующей?
6. Какая связь между сущностями называется связью категориального типа?
7. Какая связь между сущностями называется циклической или «рыболовным крючком»?
8. Поясните смысл утверждения о том, что некоторый атрибут «мигрировал»?
9. Что обозначает символика «FK» на диаграмме ERwin?
10. Какими возможностями обладает редактора связей?
11. Какова методика выполнения индивидуального задания?
12. Какие основные выводы можно сделать по результатам выполнения индивидуального задания?

Защита лабораторной работы

Оформленный, в соответствии с требованиями ГОСТов, отчет представляется студентом преподавателю для проверки и последующей защиты. Защита отчета по лабораторной работе производится студентом только индивидуально.

В ходе защите лабораторной работы студент отвечает на вопросы преподавателя (поясняет методику выполнения индивидуального задания, отвечает на контрольные вопросы и т.д.).

Отчет, оформленный с отступлениями от требований ГОСТов, небрежно и неаккуратно к защите не принимается.

Лабораторная работа № 6

Разделение модели на подмножества в ERwin

Цель и содержание: привить студентам навыки разделения модели данных на подмножества в ERwin.

Формируемые компетенции: ПК-2, ПК-6. ПК-7.

Теоретическое обоснование

В уходе выполнения лабораторной работы № 5 были определены атрибуты и установлены связи между сущностями, входящими в объектные области «Методическое обеспечение», «Учебный процесс», «Персонал и учащиеся» и «Занятия в классе». Как видно из рисунков 5.9 - 5.12, ER-диаграмма после введения сущности «Занятие в классе» получилась достаточно громоздкой, особенно на уровне атрибутов. С такой моделью трудно работать (модель пришлось делить на два листа, см. рисунки 5.11 5.12). В целях упрощения работы со сложными моделями CASE-средство ERwinпозволяет разбивать модель на подмножества (SubjectArea). В данный момент модель имеет только одно подмножество - главное (MainSubjectArea), для которого создано два хранимых отображения - «Уровень сущностей» и «Уровень атрибутов». Так как модель становится труднообозримой, особенно на уровне атрибутов (или колонок), то ее можно разбить на несколько подмножеств - объектных областей, каждая из которых будет содержать часть сущностей модели.

Для этого следует воспользоваться редактором объектных областей (**SubjectAreaEditor**), выбрав в меню пункт **Model ► SubjectAreas...**

Аппаратура и программное обеспечение

Аппаратура. Для выполнения лабораторной работы необходимы:

- персональный суперкомпьютер i7-5930K/ 8D8192D4-2133/ K2200/ 2K40M;
- рабочие станции: монитор 29 + клавиатура + мышь. ПК MDT750/ i75930K/ 4D8192D42133";
- аппаратный сетевой обучающий комплекс мод.VDCN-1502.

Программное обеспечение. Для выполнения лабораторной работы необходима операционная система WINDOWS 7/8/10 и CASE-средство ERwin7.3.

Указания по технике безопасности

При выполнении лабораторной работы **запрещается**:

1. Самостоятельно производить ремонт персонального компьютера, а так же, установку и удаление имеющегося программного обеспечения;

2. Нарушать общепринятые правила техники безопасности при работе с электрооборудованием, в частности, касаться электрических розеток металлическими предметами и т.д.;

3. Принимать пищу, напитки и сорить на рабочем месте пользователя персонального компьютера.

В случае неисправности персонального компьютера необходимо **немедленно** сообщить об этом обслуживающему персоналу лаборатории (системному администратору, оператору).

Методика и порядок выполнения работы

Откройте проект, созданный вами на лабораторной работе №5, например, Lab_5.er1 и сразу сохраните его под новым именем, например, Lab_6.er1.

Выберите в главном меню ERwin пункт Model ► SubjectAreas... (рисунок 6.1).

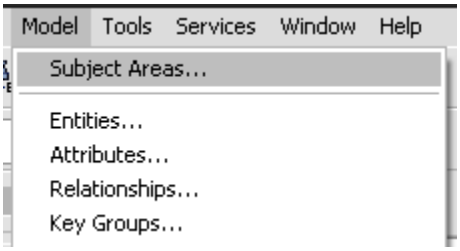


Рисунок 6.1 – Выбор в меню ERwin пункта Model ► SubjectAreas...

Диалоговое окно редактора (рисунок 6.2) содержит в верхней части

список заданных в модели подмножеств. По умолчанию в нем имеется одно

главное подмножество - **MainSubjectArea**, которое не может быть удалено.

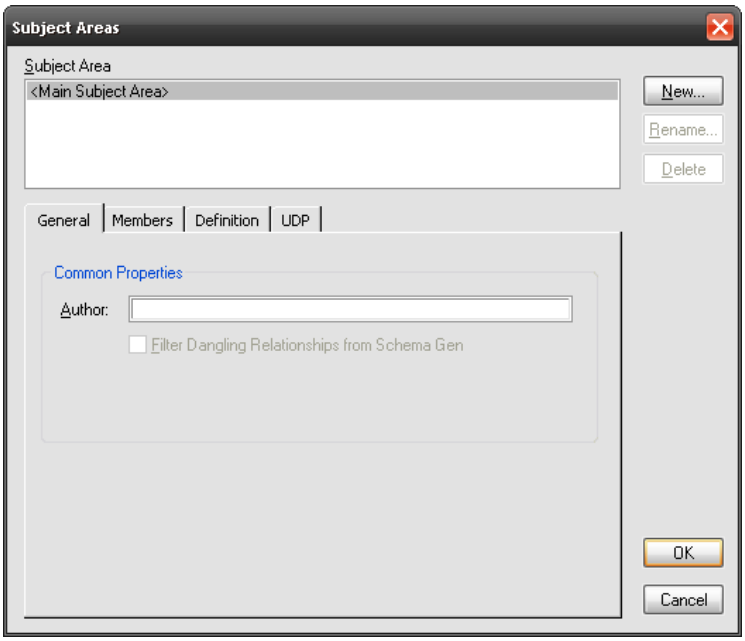


Рисунок 6.2 – Редактор подмножеств модели Остальные подмножества могут добавляться и удаляться по усмотрению разработчика.

Для этого справа от списка имеется три кнопки:

- **New**- вызывает диалог, в котором следует задать имя нового подмножества. Вновь введенное подмножество добавляется в список.
- **Rename** - позволяет переименовать подмножество.
- **Delete**- удаляет подмножество из списка, а после выхода из редактора нажатием «ОК» - из модели.

3. Добавьте в список четыре новых подмножества:

- Материальное обеспечение (рисунок 6.3).
- Методическое обеспечение.
- Учебный процесс.
- Физические лица.

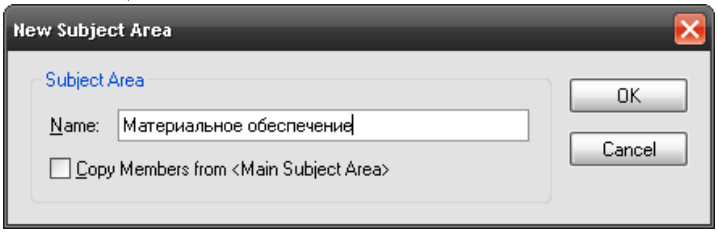


Рисунок 6.3 – Добавление в список нового подмножества «Материальное обеспечение»

Редактор подмножеств модели, после выполнения третьего пункта, примет вид представленный на рисунке 6.4.

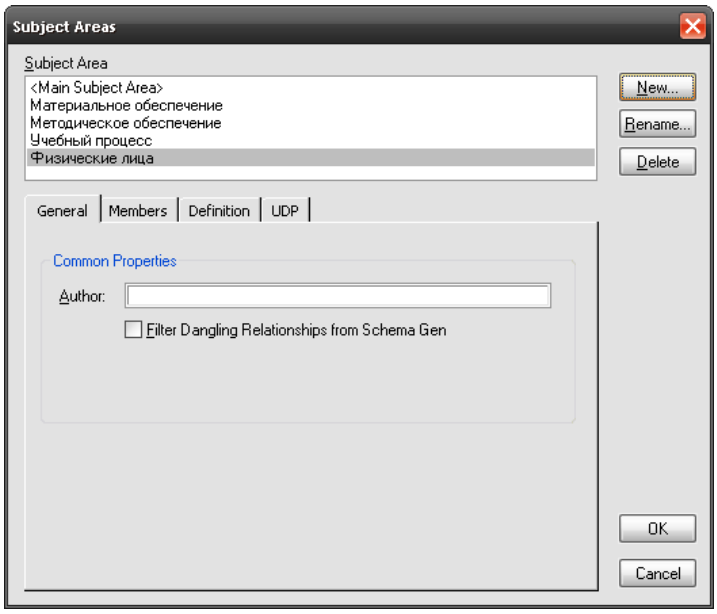


Рисунок 6.4 – Редактор подмножеств модели после добавления в список четыре новых подмножеств

В нижней части окна редактора имеется несколько страниц (вкладок), позволяющих редактировать свойства подмножеств.

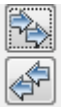
General (общие свойства). В поле **Author**вводится имя автора, ведущего данное подмножество модели (см. рисунок 6.4). Кроме того, на этой же странице имеется флаг **FilterDanglingRelationshipsfromSchemaGen**(Отфильтровать висячие связи при генерации схемы). Суть этой опции заключа-

ется в следующем: когда создается подмножество модели, мы включаем внего часть таблиц и других объектов главного подмножества, чтобы можнобыло работать с меньшими группами таблиц. По умолчанию, когда ERwinгенерирует схему для текущего подмножества, он формирует ссылки на таблицы, которые не вошли в это подмножество, особенно в запросах, связанных с внешними ключами и триггерами. Установка данного флага отключает генерацию этих ссылок.

Members(Члены). Выбор объектов, которые следует включить в заданное подмножество. Для этого имеется два списка **AvailableEntities**(Доступные сущности) и **IncludedEntities**(Включенные Сущности). В зависимости от переключателя **DisplayNames**(Показывать Имена) в нижней части страницы, списки содержат логические или физические имена сущностей (рисунок 6.5).

Перемещение объектов между списками производится при помощи кнопок, приведенных в таблице 6.1.

Таблица 6.1 – Кнопки перемещения объектов между списками

Кноп	Назначение кнопки
 	Перенос выделенного объекта.
	Перенос всех объектов, содержащихся в соответствующем списке
 	Перенос выбранного объекта «гнездом», вместе со связанными с ним объектами. При этом появляется диалог (рисунок 6.6), позволяющий задать режим отбора объектов: 1) все, входящие в иерархию связи; 2) объекты, входящие в иерархию не больше заданного уровня. Режим выбирается как для объектов, находящихся выше по иерархии(Ancestors), так и для потомков (Descendants) например, если выбрать сущность «Класс» и установить для потомков уровень 1 , то будут выбраны сущности «Класс» и «Учебное место». Если для сущности «Класс» установить уровень 2, то в выборку попадут уже сущности «Класс», «Учебное место», «Единица оборудования» и «Занятие в классе»

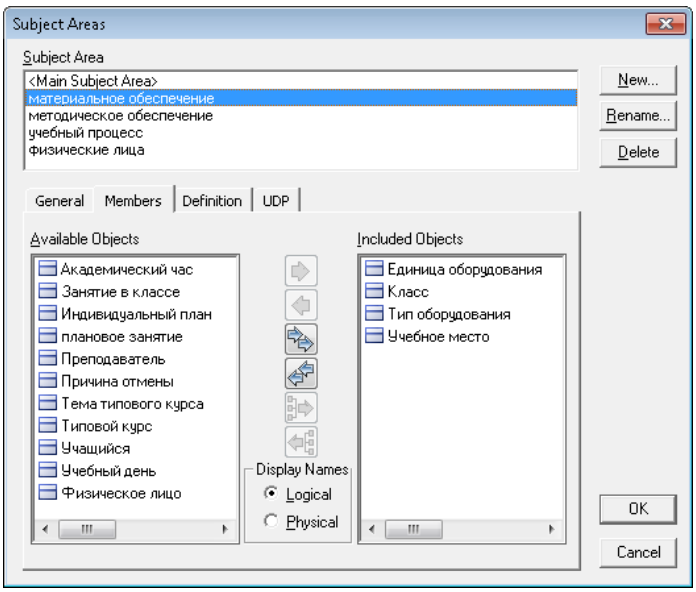


Рисунок 6.5 – Выбор объектов подмножества модели

Рисунок 6.6 – Диалог отбора объектов

Definition (Определение**)**. Позволяет ввести текстовый комментарий к подмножеству модели.

UDP(Пользовательские свойства). Подключение к подмножеству модели специально заданных пользователем свойств.

4. Задайте для введенных подмножеств сущности, приведенные в таблице

6.2. Таблица 6.2 –Подмножества и входящие в них объекты

Подмножество мелели	Входящие объекты
Материальное обеспечение	Класс, Учебное место. Единица оборудования. Тип оборудования
Методическое обеспечение	Типовой курс, Тема типового курса, Плановое занятие, Индивидуальный
Учебный процесс	Учебный день, Академический час
Физические лица	Физическое лицо, Учащийся,

После нажатия кнопки **ОК**заданные подмножества будут созданы в модели. Переключение между подмножествами легко осуществляется при помощи выпадающего списка на панели кнопок (рисунок 6.7).

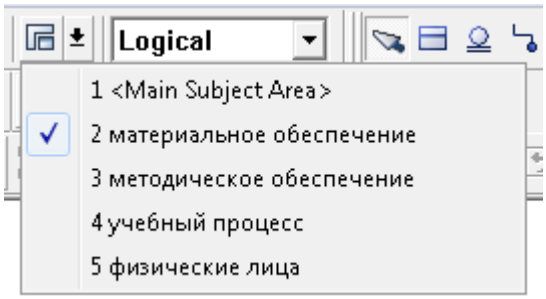
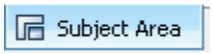


Рисунок 6.7 – Выпадающий список на панели кнопок отбора подмножествами модели

Для удобства можно создать для каждого подмножества хранимые отображения «Уровень сущностей» и «Уровень атрибутов», как это было сделано ранее для главного подмножества.

Подмножество модели отображаются в проводнике модели (рисунок 6.8) вкладка 

Если сделать правый клик на названии подмножестве модели в проводнике модели, то откроется контекстное меню (рисунок 6.9).

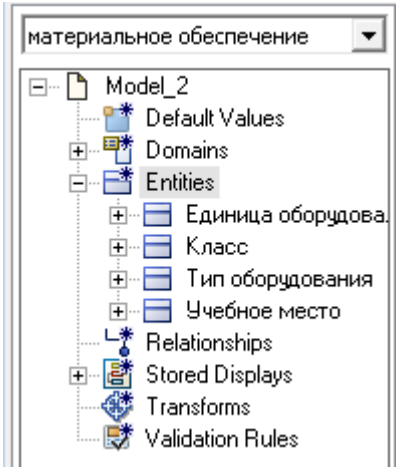


Рисунок 6.8 – Подмножество модели отображаются в проводнике модели

Если в этом контекстном меню выбрать команду **GoTo**, то можно перейти к подмножеству модели, которое будет отображено в окне дизайнера модели.

5. Сохраните модель, например, под именем Lab_6.er1.
6. Выполните индивидуальное задание по разделению модели на подмножества в ERwinдля указанной предметной области.

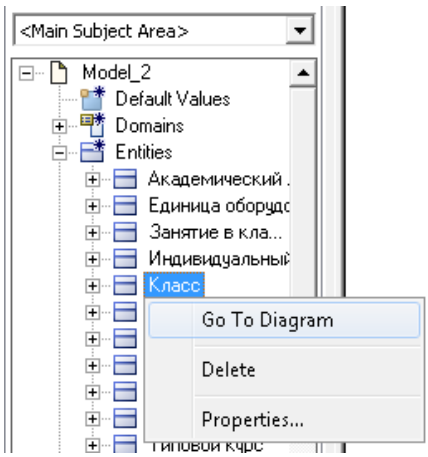


Рисунок 6.9 – Контекстное меню подмножества модели

Варианты заданий

Используя результаты выполнения лабораторной работы № 5, необходимо выполнить разделение модели на подмножества в ERwindля указанной предметной области. Количество подмножеств модели должно быть неменее двух. Номер варианта соответствует номеру рабочего места за компьютером и должен совпадать с номером варианта индивидуального задания к лабораторной работе № 2 (см. таблицу 2.3).

Содержание отчета и его форма

Отчет по лабораторной работе оформляется в виде текстового документа и должен включать:

1. Название лабораторной работы.
2. Цель лабораторной работы.
3. Формулировку индивидуального задания и результат его выполнения.
4. Краткие выводы по результатам выполнения лабораторной работы.

Контрольные вопросы

1. Что такое подмножество модели на диаграмме ERwin?
2. Как создать подмножество модели?
3. Каковы свойства редактора подмножеств модели?
4. Как осуществить переключение между подмножествами модели?
5. Какие подмножества модели были использованы при выполнении индивидуального задания?

Защита лабораторной работы

Оформленный, в соответствии с требованиями ГОСТов, отчет представляется студентом преподавателю для проверки и последующей защиты. Защита отчета по лабораторной работе производится студентом только индивидуально.

В ходе защите лабораторной работы студент отвечает на вопросы преподавателя (поясняет методику выполнения индивидуального задания, отвечает на контрольные вопросы и т.д.).

Отчет, оформленный с отступлениями от требований ГОСТов, небрежно и неаккуратно к защите не принимается.

Лабораторная работа № 7

Построение диаграмм SADT (IDEF0). Типовой вариант

Цель и содержание: Лабораторные работы №7-10 ориентированы на изучение CASE-средства RAMUS. С помощью данного средства проектирования необходимо разработать модель "Как есть (As Is)" для выбранной предметной области.

Процесс создания диаграмм начинается с этапа изучения предметной области, краткое описание которой приведено у вас в варианте. Все вопросы, не освещенные в задании, вы либо придумываете, либо уточняете у преподавателя.

Формируемые компетенции: ПК-2, ПК-6. ПК-7.

Теоретическое обоснование

Методология структурного анализа и проектирования ПО определяет шаги работы, которые должны быть выполнены, их последовательность, правила распределения и назначения операций и методов.

В настоящее время успешно используются такие методологии, как SADT (Structure Analysis and Design Technique), структурный системный анализ Гейна-Сарсона, структурный анализ и проектирование Йодана/Де Марко, развитие систем Джексона и другие.

Перечисленные структурные методологии жестко регламентируют фазы анализа требований и проектирования спецификаций и отражают подход к разработке ПО с позиций рецептов "кулинарной книги".

В настоящее время все большую популярность приобретают инженерные методы реорганизации предприятий на основе современных информационных технологий. Понятия «бизнес-процессы», «процессно-стоимостной подход», «структурный системный анализ», «функциональное моделирование», «информационное моделирование », «реинжиниринг» и многие другие, с ними связанные, уверенно входят в лексикон аналитиков всех уровней. Соответственно расширяется спектр компьютеризованных инструментальных методов анализа экономических процессов и бизнес-процессов в частности.

Перечень специальностей, где активно используются новейшие технологии, первоначально ориентированный в основном на специальности «Информационные системы в экономике», «Менеджмент», постоянно расширяется. В этом списке в данный момент уже нашли себе место такие специальности, как «Бухгалтерский учет и аудит», «Финансы и кредит», «Мировая экономика», «Антикризисное управление », «Маркетинг» и т.д.

Аппаратура и программное обеспечение

Аппаратура. Для выполнения лабораторной работы необходимы:

- персональный суперкомпьютер i7-5930K/ 8D8192D4-2133/ K2200/ 2K40M;

- рабочие станции: монитор 29 + клавиатура + мышь. ПК MDT750/ i75930K/ 4D8192D42133";

- аппаратный сетевой обучающий комплекс мод.VDCN-1502.

Программное обеспечение. Для выполнения лабораторной работы необходима операционная система WINDOWS 7/8/10 и CASE-средство ERwin7.3.

Указания по технике безопасности

При выполнении лабораторной работы **запрещается**:

4. Самостоятельно производить ремонт персонального компьютера, а так же, установку и удаление имеющегося программного обеспечения;
5. Нарушать общепринятые правила техники безопасности при работе с электрооборудованием, в частности, касаться электрических розеток металлическими предметами и т.д.;
6. Принимать пищу, напитки и сорить на рабочем месте пользователя персонального компьютера.

В случае неисправности персонального компьютера необходимо **немедленно** сообщить об этом обслуживающему персоналу лаборатории (системному администратору, оператору).

Методика и порядок выполнения работы

- ознакомиться с вариантами индивидуальных задания и закрепить за собой один из предложенных или предложить свой вариант;
- повторить основные принципы методологии IDEF0;
- повторить основные принципы работы с системой RAMUS;
- сформировать контекстную диаграмму системы, указав в качестве названия название проектируемой системы и указав глобальные потоки для системы;
- провести декомпозицию системы на 3-4 основные функции
- связать глобальные потоки с выделенными функциями;
- провести связи между функциями;
- посчитать количественные характеристики для построенной диаграммы;
- выполнить декомпозицию следующего уровня для двух блоков, повторив пункты 5-8 для каждого блока.

На рис.1 изображен процесс моделирования в SADT, описанный с помощью SADT-диаграммы. Диаграмма отражает тот факт, что процесс моделирования в SADT является итеративной последовательностью шагов, приводящих к точному описанию системы. Высокая эффективность этого процесса обусловлена его организацией, в основе которой лежит разделение функций, выполняемых участниками создания

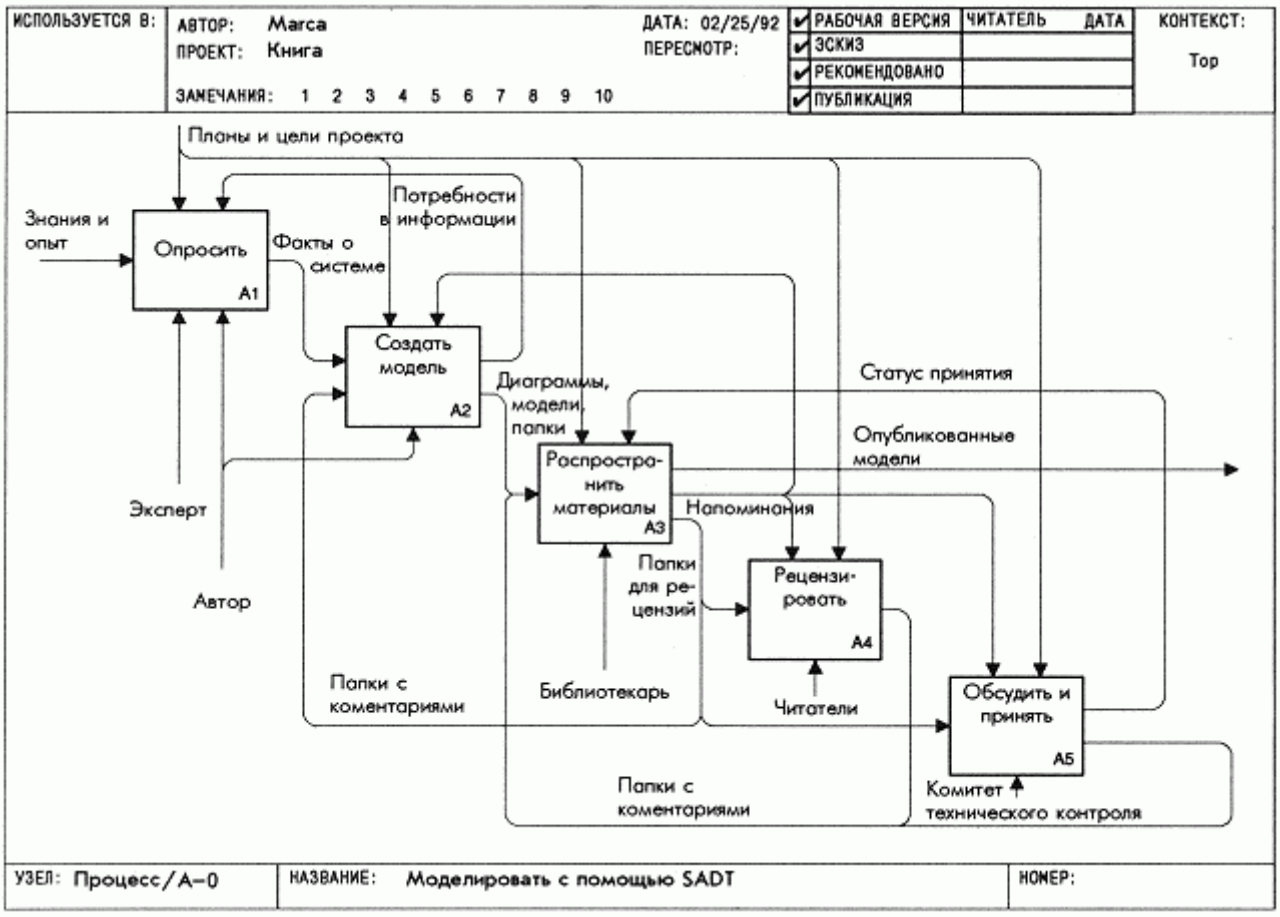


Рис. 1. Процесс создания SADT-модели

SADT-проектов: эксперты являются источниками информации, авторы создают диаграммы и модели, библиотекарь координирует обмен письменной информацией, читатели рецензируют и утверждают модели, а Комитет технического контроля принимает и утверждает модель. В данной главе представлен общий обзор процесса моделирования. Более детально его отдельные шаги обсуждаются в главах 5 и 6, а также в частях II и III.

1. Получение знаний в процессе опроса

В процессе моделирования сведения об изучаемой системе получают с помощью испытанной методики сбора информации - опросов или интервью. Для получения наиболее полной информации SADT предлагает использовать различные ее источники (например, читать документы, опрашивать людей, наблюдать за работой системы). Независимо от конкретного источника информации методология SADT рекомендует руководствоваться определенной целью при его использовании. Это означает, что вы должны определить свои потребности в информации прежде, чем выбрать очередной источник. Во время опроса графический язык SADT используется как средство для заметок, которые служат основой для построения диаграмм.

2. Документирование полученных знаний

Создание модели (блок 2 на рис. 1) - это второй важный этап в процессе моделирования, на котором аналитик документирует полученные им знания о данной проблемной области, представляя их в виде одной или нескольких SADT-диаграмм. Процесс создания модели осуществляется с помощью специального метода детализации ограниченного субъекта. Коротко говоря, в SADT автор вначале анализирует объекты, входящие в систему, а затем использует полученные знания для анализа функций системы. На основе этого анализа создается диаграмма, в которой объединяются сходные

объекты и функции. Этот конкретный путь проведения анализа системы и документирования его результатов является уникальной особенностью методологии SADT.

3. Корректность модели проверяется в процессе итеративного рецензирования

Моделирование в SADT - инженерная дисциплина. Это означает, что модели создаются исходя из действительной ситуации и что эти модели проходят через серию последовательных улучшений до тех пор, пока они в точности не будут представлять реальный мир. Одной из основных компонент методологии SADT является итеративное рецензирование, в процессе которого автор и эксперт многократно совещаются (устно и письменно) относительно достоверности создаваемой модели. Итеративное рецензирование называется циклом автор/читатель.

Цикл автор/читатель начинается в тот момент, когда автор принимает решение распространить информацию о какой-либо части своей работы с целью получения отзыва о ней. Материал для распространения оформляется в виде "папок" - небольших пакетов с результатами работы, которые критически обсуждаются другими специалистами в течение определенного времени. Сделанные письменные замечания также помещаются в папку в виде нумерованных комментариев. Папки с замечаниями являются, таким образом, обратной связью, которую авторы получают на свою работу. Читатели - это те, кто читает и критикует создаваемую модель (см. блок 4 на рис. 1), а затем помещает замечания в папки. Их работа возможна благодаря тому, что графический язык SADT-диаграмм позволяет создавать диаграммы и модели, которые можно легко и быстро читать. (Простота графического языка потому не случайна. Она позволяет получить представление о системе, на основе которого можно дать обоснованное заключение о достоверности модели.)

Обычно отдельная папка рецензируется одновременно несколькими читателями, и все их замечания поступают к определенному сроку к автору. Затем автор отвечает на каждое замечание и обобщает критику, содержащуюся в замечаниях. С помощью таких обсуждений можно достаточно быстро обмениваться идеями. Таким образом, методология SADT поддерживает как параллельный, так и асинхронный просмотр модели, что является наиболее эффективным способом распределения работы в коллективе. Это показывает, что моделирование в SADT является инженерной дисциплиной, потому что итеративная коллективная деятельность - признак инженерной деятельности. Это связано с тем, что модель в SADT очень редко создается одним автором. На практике над различными частями модели могут совместно работать множество авторов, потому что каждый функциональный блок модели представляет отдельный субъект, который может быть независимо проанализирован и декомпозирован. Таким образом, модель сама координирует работу коллектива авторов, в то время как процесс моделирования SADT координирует совместное рецензирование возникающих идей. Полное описание инженерного процесса приведено в части III.

4. Координация процесса рецензирования

Организация своевременной обратной связи имеет важнейшее значение для эффективного моделирования, потому что устаревшая информация потенциально способна свести на нет все усилия по разработке системы. Вот почему SADT выделяет специальную роль наблюдателя за процессом рецензирования. На рис. 1 показано, что эту роль выполняет библиотекарь, который является главным координатором процесса моделирования в SADT, обеспечивая своевременное и согласованное распространение рабочих материалов. Библиотекарь распространяет полученные от авторов папки, контролирует их движение, рассылает напоминания о своевременном возвращении авторам папок с замечаниями и о сроках ответов авторов на предложения читателей. Кроме того, библиотекарь печатает законченные модели после того, как они одобрены и приняты к использованию.

5. Модели используются после их одобрения

Вспомним, что SADT-модели создаются с конкретной целью, и эта цель записана на диаграмме A-0 модели. В каком-то смысле эта цель определяет, как будет использоваться модель. Таким образом, как только завершено создание модели с требуемым уровнем детализации и модель проверена, она может применяться для достижения поставленной цели. Например, модель экспериментального механического цеха создана для описания деятельности различных работников механического цеха, хотя результирующая модель всегда предназначалась как основа учебного руководства для нового персонала. Если эта модель точно описывает работу персонала в цехе, но не может служить для подготовки учебного руководства - она бесполезна. Точная модель не всегда полезна.

В процессе SADT-моделирования рекомендуется выделить специальную группу людей, ответственных за то, что создаваемая в процессе анализа модель будет точна и используема в дальнейшем. Эта группа, называемая Комитетом технического контроля (см. блок 5 на рис. 1), отвечает за контроль качества моделей, создаваемых авторами SADT-проекта. Комитет следит за выполняемой работой и ее соответствием конечным целям всего проекта. Члены Комитета обсуждают модель и оценивают, насколько она может быть использована и будет использована соответствующим образом в ходе выполнения проекта для достижения его глобальных целей.

Таким образом Комитет технического контроля находится в наиболее выгодном положении при определении текущего направления развития проекта и выработке предложений по его корректировке. Комитет реализует это с помощью рецензий. Модели, которые достигли желаемого уровня детализации и точности с точки зрения технических требований, направляются членам Комитета технического контроля для обсуждения и утверждения. Комитет оценивает, насколько применима данная модель. Если модель признана Комитетом применимой, она публикуется. В противном случае авторам направляются замечания для необходимой доработки.

Содержание отчета и его форма

Отчет по лабораторной работе оформляется в виде текстового документа и должен включать:

5. Название лабораторной работы.
6. Цель лабораторной работы.
7. Формулировку индивидуального задания и результат его выполнения.
8. Краткие выводы по результатам выполнения лабораторной работы.

Контрольные вопросы

1. Что называют моделью предметной области?
2. Что такое язык моделирования?
3. Что такое нотация?
4. Укажите главный критерий адекватности структурной модели
5. Что такое "Объект"?
6. Что такое "Функция (операция)"?
7. Что такое "Событие"?
8. Укажите идеи, лежащие в понимании термина " системный структурный анализ"

9. Что называют IDEF0-технологией структурного анализа и проектирования?
10. Что называют IDEF3-технологией структурного анализа и проектирования?
11. Что называют DFD технологией структурного анализа и проектирования?
12. Что называют функциональной диаграммой?
13. Что называют моделью данных?
14. Что такое методология SADT?
15. Что называют функциональной моделью SADT?
16. Что такое действие (фукнция) в модели IDEF0?
17. Что такое "Функциональный блок (Activity Box)" в функциональной методике IDEF0?
18. Что такое "Интерфейсная дуга (Arrow)" в функциональной методике IDEF0?
19. Что такое стрелка типа I (Input) в модели IDEF0?
20. Что такое стрелка типа C (Control) в модели IDEF0?
21. Что такое стрелка типа O (Output) в модели IDEF0?
22. Что такое стрелка типа M (Mechanism) в модели IDEF0?
23. Что такое "Декомпозиция (Decomposition)" в функциональной методике IDEF0?
24. Что такое "Глоссарий (Glossary)" в функциональной методике IDEF0?
25. Что называют Стрелками входа в модели IDEF0?
26. Что называют Стрелками управления в модели IDEF0?
27. Что называют Стрелками выхода в модели IDEF0?
28. Что называют Стрелками механизма исполнения в модели IDEF0?
29. Укажите рисунок с изображением комбинации стрелок выход - вход модели IDEF0 (и так для всех типов стрелок)
30. Что называют туннелем в модели IDEF0?
31. Укажите рисунок с изображением комбинации стрелок выход - управление модели IDEF0. (И так для разных сочетаний)

32. Что называют методологией описания процессов IDEF3?
33. Укажите цель составления диаграмм IDEF3.
34. Что называют сценарием процесса в модели IDEF3?
35. Укажите тип связи, изображенный на рисунке (для всех трех типов связи)
36. Что называют соединением в модели IDEF3?
37. Укажите изображение асинхронного "И"-соединения в модели IDEF3.
38. Укажите изображение асинхронного "ИЛИ"-соединения в модели IDEF3.
39. Укажите изображение асинхронного соединения "Эксклюзивное ИЛИ" в модели IDEF3.
40. Укажите изображение синхронного "И"-соединения в модели IDEF3.
41. Укажите изображение синхронного "ИЛИ"-соединения в модели IDEF3.
42. Укажите изображение синхронного соединения "Эксклюзивное ИЛИ" в модели IDEF3.

Защита лабораторной работы

Оформленный, в соответствии с требованиями ГОСТов, отчет представляется студентом преподавателю для проверки и последующей защиты. Защита отчета по лабораторной работе производится студентом только индивидуально.

В ходе защите лабораторной работы студент отвечает на вопросы преподавателя (поясняет методику выполнения индивидуального задания, отвечает на контрольные вопросы и т.д.).

Отчет, оформленный с отступлениями от требований ГОСТов,

небрежно и неаккуратно к защите не принимается.

Лабораторная работа № 8

Построение диаграмм SADT (IDEF0) по варианту предметной области

Цель и содержание: Лабораторные работы №7-10 ориентированы на изучение CASE-средства RAMUS. С помощью данного средства проектирования необходимо разработать модель "Как есть (As Is)" для выбранной предметной области.

Процесс создания диаграмм начинается с этапа изучения предметной области, краткое описание которой приведено у вас в варианте. Все вопросы, не освещенные в задании, вы либо придумываете, либо уточняете у преподавателя.

Формируемые компетенции: ПК-2, ПК-6. ПК-7.

Теоретическое обоснование

Методология структурного анализа и проектирования ПО определяет шаги работы, которые должны быть выполнены, их последовательность, правила распределения и назначения операций и методов.

В настоящее время успешно используются такие методологии, как SADT (Structure Analysis and Design Technique), структурный системный анализ Гейна-Сарсона, структурный анализ и проектирование Йодана/Де Марко, развитие систем Джексона и другие.

Перечисленные структурные методологии жестко регламентируют фазы анализа требований и проектирования спецификаций и отражают подход к разработке ПО с позиций рецептов "кулинарной книги".

В настоящее время все большую популярность приобретают инженерные методы реорганизации предприятий на основе современных информационных технологий. Понятия «бизнес-процессы», «процессно-стоимостной подход», «структурный системный анализ», «функциональное моделирование», «информационное моделирование », «реинжиниринг» и многие другие, с ними связанные, уверенно входят в лексикон аналитиков всех уровней. Соответственно расширяется спектр компьютеризованных инструментальных методов анализа экономических процессов и бизнес-процессов в частности.

Перечень специальностей, где активно используются новейшие технологии, первоначально ориентированный в основном на специальности «Информационные системы в экономике», «Менеджмент», постоянно расширяется. В этом списке в данный момент уже нашли себе место такие специальности, как «Бухгалтерский учет и аудит», «Финансы и кредит», «Мировая экономика», «Антикризисное управление », «Маркетинг» и т.д.

Аппаратура и программное обеспечение

Аппаратура. Для выполнения лабораторной работы необходимы:

- персональный суперкомпьютер i7-5930K/ 8D8192D4-2133/ K2200/ 2K40M;

- рабочие станции: монитор 29 + клавиатура + мышь. ПК MDT750/ i75930K/ 4D8192D42133";

- аппаратный сетевой обучающий комплекс мод.VDCN-1502.

Программное обеспечение. Для выполнения лабораторной работы необходима операционная система WINDOWS 7/8/10 и CASE-средство ERwin7.3.

Указания по технике безопасности

При выполнении лабораторной работы **запрещается**:

7. Самостоятельно производить ремонт персонального компьютера, а так же, установку и удаление имеющегося программного обеспечения;
8. Нарушать общепринятые правила техники безопасности при работе с электрооборудованием, в частности, касаться электрических розеток металлическими предметами и т.д.;
9. Принимать пищу, напитки и сорить на рабочем месте пользователя персонального компьютера.

В случае неисправности персонального компьютера необходимо **немедленно** сообщить об этом обслуживающему персоналу лаборатории (системному администратору, оператору).

Методика и порядок выполнения работы

- ознакомиться с вариантами индивидуальных задания и закрепить за собой один из предложенных или предложить свой вариант;
- повторить основные принципы методологии IDEF0;
- повторить основные принципы работы с системой RAMUS;
- сформировать контекстную диаграмму системы, указав в качестве названия название проектируемой системы и указав глобальные потоки для системы;
- провести декомпозицию системы на 3-4 основные функции
- связать глобальные потоки с выделенными функциями;
- провести связи между функциями;
- посчитать количественные характеристики для построенной диаграммы;
- выполнить декомпозицию следующего уровня для двух блоков, повторив пункты 5-8 для каждого блока.

Содержание отчета и его форма

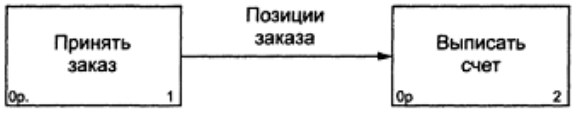
Отчет по лабораторной работе оформляется в виде текстового документа и должен включать:

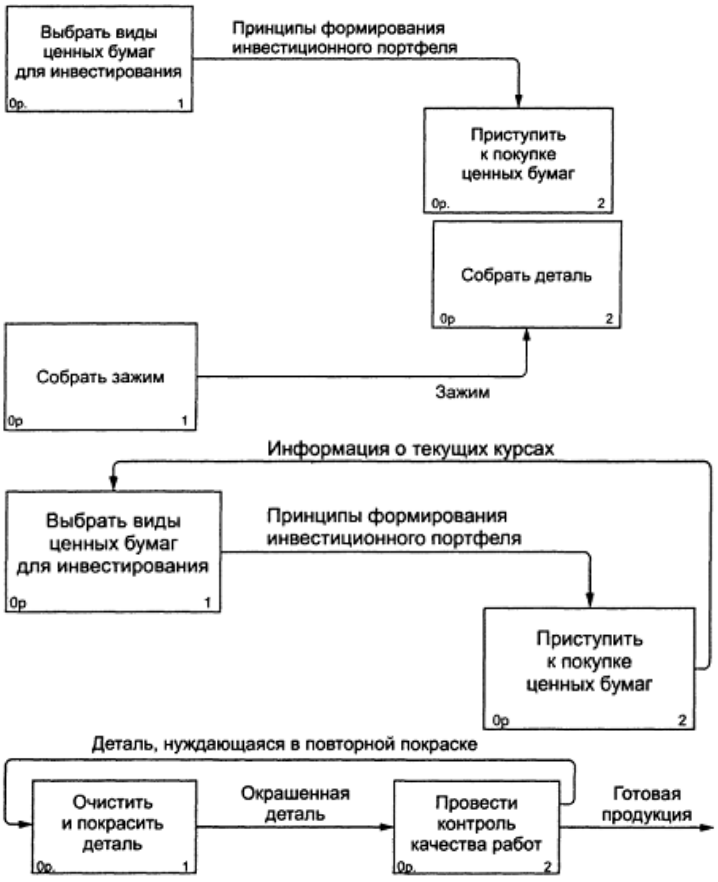
9. Название лабораторной работы.
10. Цель лабораторной работы.
11. Формулировку индивидуального задания и результат его выполнения.
12. Краткие выводы по результатам выполнения лабораторной работы.

Контрольные вопросы

43. Что называют моделью предметной области?
44. Что такое язык моделирования?

45. Что такое нотация?
46. Укажите главный критерий адекватности структурной модели
47. Что такое "Объект"?
48. Что такое "Функция (операция)"?
49. Что такое "Событие"?
50. Укажите идеи, лежащие в понимании термина " системный структурный анализ"
51. Что называют IDEF0-технологией структурного анализа и проектирования?
52. Что называют IDEF3-технологией структурного анализа и проектирования?
53. Что называют DFD технологией структурного анализа и проектирования?
54. Что называют функциональной диаграммой?
55. Что называют моделью данных?
56. Что такое методология SADT?
57. Что называют функциональной моделью SADT?
58. Что такое действие (фукнция) в модели IDEF0?
59. Что такое "Функциональный блок (Activity Box)" в функциональной методике IDEF0?
60. Что такое "Интерфейсная дуга (Arrow)" в функциональной методике IDEF0?
61. Что такое стрелка типа I (Input) в модели IDEF0?
62. Что такое стрелка типа C (Control) в модели IDEF0?
63. Что такое стрелка типа O (Output) в модели IDEF0?
64. Что такое стрелка типа M (Mechanism) в модели IDEF0?
65. Что такое "Декомпозиция (Decomposition)" в функциональной методике IDEF0?
66. Что такое "Глоссарий (Glossary)" в функциональной методике IDEF0?
67. Что называют Стрелками входа в модели IDEF0?
68. Что называют Стрелками управления в модели IDEF0?
69. Что называют Стрелками выхода в модели IDEF0?
70. Что называют Стрелками механизма исполнения в модели IDEF0?
71. Укажите рисунок с изображением комбинации стрелок выход - вход модели IDEF0 (и так для всех типов стрелок)
72. Что называют туннелем в модели IDEF0?
73. Укажите рисунок с изображением комбинации стрелок выход - управление модели IDEF0. (И так для разных сочетаний)





74. Что называют методологией описания процессов IDEF3?
75. Укажите цель составления диаграмм IDEF3.
76. Что называют сценарием процесса в модели IDEF3?
77. Укажите тип связи, изображенный на рисунке (для всех трех типов связи)
78. Что называют соединением в модели IDEF3?
79. Укажите изображение асинхронного "И"-соединения в модели IDEF3.
80. Укажите изображение асинхронного "ИЛИ"-соединения в модели IDEF3.
81. Укажите изображение асинхронного соединения "Эксклюзивное ИЛИ" в модели IDEF3.
82. Укажите изображение синхронного "И"-соединения в модели IDEF3.
83. Укажите изображение синхронного "ИЛИ"-соединения в модели IDEF3.
84. Укажите изображение синхронного соединения "Эксклюзивное ИЛИ" в модели IDEF3.

Защита лабораторной работы

Оформленный, в соответствии с требованиями ГОСТов, отчет представляется студентом преподавателю для проверки и последующей защиты. Защита отчета по лабораторной работе производится студентом

только индивидуально.

В ходе защите лабораторной работы студент отвечает на вопросы преподавателя (поясняет методику выполнения индивидуального задания, отвечает на контрольные вопросы и т.д.).

Отчет, оформленный с отступлениями от требований ГОСТов, небрежно и неаккуратно к защите не принимается.

Лабораторная работа № 9

Построение диаграмм DFD. Типовой вариант

Цель и содержание: Лабораторные работы №7-10 ориентированы на изучение CASE-средства RAMUS. С помощью данного средства проектирования необходимо разработать модель "Как есть (As Is)" для выбранной предметной области.

Процесс создания диаграмм начинается с этапа изучения предметной области, краткое описание которой приведено у вас в варианте. Все вопросы, не освещенные в задании, вы либо придумываете, либо уточняете у преподавателя.

Формируемые компетенции: ПК-2, ПК-6. ПК-7.

Теоретическое обоснование

Диаграммы потоков данных (DFD) используются для описания документооборота и обработки информации. Подобно IDEF0, DFD представляет систему как сеть связанных между собой функций. Их можно использовать как дополнение к модели IDEF0 для более наглядного отображения текущих операций документооборота в системах обработки информации.

Источники информации - **внешние сущности** - порождают информационные **потоки данных**, переносящие информацию к **процессам**. Те в свою очередь преобразуют информацию и порождают новые потоки, которые переносят информацию к другим процессам, **хранилищам данных** или **внешним сущностям** - потребителям информации.

Внешняя сущность - представляет собой материальный объект или физическое лицо (заказчик, склад), представляющее собой источник или приемник информации, который находится за пределами границ анализируемой системы. Обозначается в виде прямоугольника с тенью и обычно располагаются по краям диаграммы. Одна внешняя сущность может быть использована многократно на одной диаграмме, чтобы не рисовать слишком запутанных стрелок.

Процесс - представляет собой функции системы, преобразующие входные потоки в выходные в соответствии с определенным алгоритмом. Изображаются прямоугольниками со скругленными углами. Их смысл совпадает со смыслом функций IDEF0. Они также имеют входы и выходы, но не поддерживают управления и механизмы. Физически процесс - это например, отдел, выполняющее обработку входных документов и выпуск отчетов, или программа, аппаратно реализованное логическое устройство.

Поток данных - определяет информацию, передаваемую через некоторое соединение от источника к приемнику. Реальный поток

данных может быть информацией передаваемой по кабелю, пересылаемыми по почте письмами, магнитными дискетами. Поскольку в DFD каждая сторона блока не имеет четкого назначения, как в IDEF0, стрелки могут подходить и выходить из любой грани блока. В DFD также применяется двунаправленные стрелки для описания диалогов типа "команда-ответ".

Хранилище данных - позволяет описать данные, которые необходимо сохранить в памяти прежде, чем использовать в функциях. То есть в отличие от стрелок, описывающих объекты в движении, хранилища изображают объекты в покое. Хранилище физически может быть реализован в виде ящика в картотеки, таблицы в оперативной памяти, файла на магнитном диске.

Рассмотрим фрагмент проекта системы, организующей работу банкомата по обслуживанию клиента по его кредитной карте.

1. Запустите CASE-средство BPwin
2. Создайте новую модель: выберите **File | New**, в поле **Name** введите **Работа банкомата**, в области **Type** выберите **Data Flow (DFD)**, нажмите **OK,** в поле **Autor** введите ваше имя, например, **Иванов И.И.,** нажмите **OK.**
3. Установите русский шрифт для объекта: выберите **Model | Default Fonts**, выберите **Context Activity**, выберите **Courier New** или **TimesET**, нажмите **OK**
4. Установите русский шрифт для **всех** типов объектов имеющихся в меню **Model | Default Fonts.**
5. Введите имя процесса и описание: выберите процесс, нажмите **МП,** выберите **Name**, введите **Обслужить**. выберите вкладку **Defenition,** в поле **Defenition** введите определение **Процесс обслуживания клиента по его кредитной карте с помощью банкомата**, выберите вкладку **Color,** выберите **розовый,** нажмите **OK**
6. Добавьте внешнюю сущность: нажмите кнопку **External Reference Tool** , выберите мышью место размещения - в верхнем левом углу, введите имя сущности **Клиент ,** нажмите **OK,** нажмите кнопку **Poiter Tool.**
7. Введите описание: выберите внешнюю сущность **Клиент**, нажмите **МП,** выберите **Name**, выберите вкладку **Definition,** в поле **Defenition** введите определение **Клиент банка с кредитной картой**, выберите вкладку **Color,** выберите **синий,** нажмите **OK.**
8. Добавьте внешнюю сущность **Компьютер банка** в нижнем правом углу и ее описание **Хранит информацию о счетах всех клиентов**.

9. Введите поток данных: нажмите кнопку **Precedence Arrow Tool** , выберите правую грань внешней сущности **Клиент,** выберите левую грань процесса **Обслужить**, нажмите кнопку **Poiter Tool.**
10. Введите имя и тип потока данных: выберите созданный поток данных, нажмите **МП**, выберите **Name**, в поле **Arrow Name** введите **Кредитная карта**, выберите вкладку **Style,** в области **Shape** выберите **двунаправленную** стрелку, нажмите **OK,**
11. Введите описание потока данных: выберите поток данных **Кредитная карта**, нажмите **МП**, выберите вкладку **Definition,** в поле **Definition** введите **Для банковского обслуживания клиент должен предоставить кредитную карту для автоматического считывания с нее информации (пароль, лимит денег, детали клиента)**, нажмите **OK**
12. Введите поток данных **Ключевые данные** и его описание **Для банковского обслуживания клиент должен ввести ключевые данные - пароль, запрос на обслуживание**

    . "Банковское обслуживание" должно: (1) выдать **сообщение**, приглашающее клиента ввести **ключевые данные**, (2) выдать клиенту **деньги**, (3) выдать клиенту **выписку** по проведенному обслуживанию, включающую **выписку о деньгах, выписку по балансу, выписку по операции,** проведенной банком.
13. Введите потоки данных **Сообщение**, **Деньги**, **Выписка** - см. рисунок 1.
14. Введите описание потоков **Сообщение**, **Деньги**, **Выписка**. Процесс "Обслужить" и внешняя сущность "Компьютер банка" обмениваются следующей информацией: (1) **данные по счету** клиента в банке, (2) **протокол обслуживания**, включающий информацию об **обработанной документации**, изымаемой **денежной сумме** и **данные по истории запроса**.
15. Введите потоки данных **Данные по счету**, **Протокол обслуживания** - см. рисунок 1.

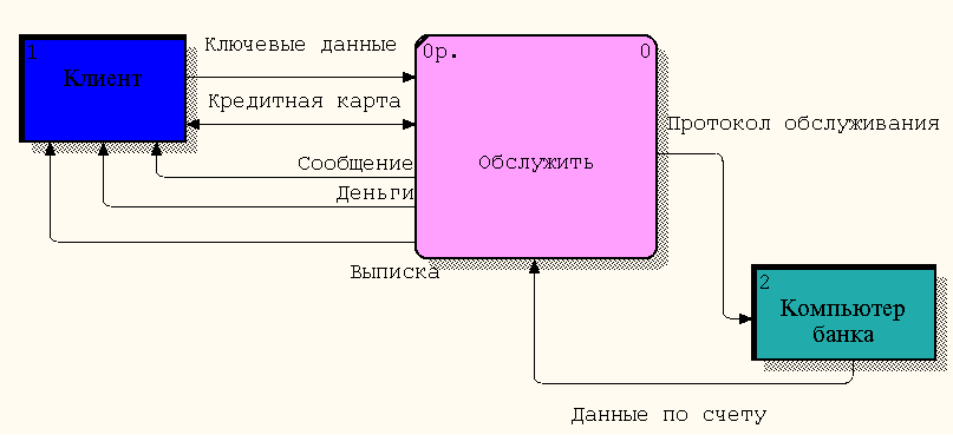


Рисунок 1.

16. Введите описание потоков **Данные по счету**, **Протокол обслуживания**
17. Создайте диаграмму декомпозиции: выберите процесс **Обслужить**, выберите на палитре инструментов кнопку декомпозиции, в окне **Activity Box Count** выберите **DFD**, в списке **Number of Activities**... выберите **4**, выберите **OK.**
18. Добавьте хранилище данных: нажмите кнопку **Date store tool**, введите имя хранилища **Данные кредитной карты -** см. рисунок 2.

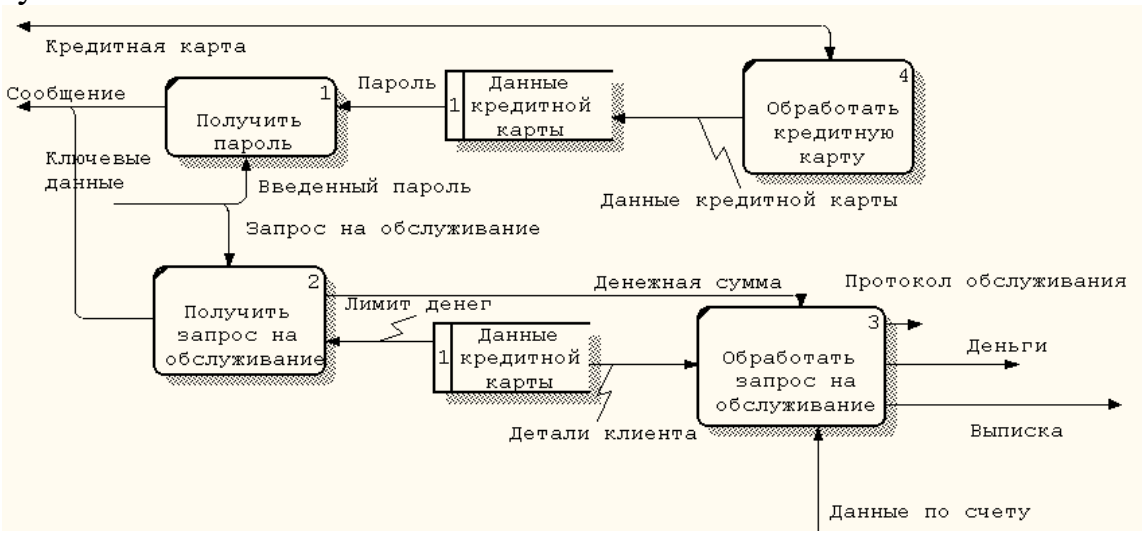


Рисунок 2.

19. Добавьте еще раз хранилище данных **Данные кредитной карты** с целью избежать пересечений линий потоков данных. Процесс **Получить пароль** осуществляет прием и проверку пароля клиента и имеет потоки: (1) внешний выходной поток **Собщение** для информирования клиента о своей готовности принять пароль, (2) входной поток **Введенный пароль** как элемент внешнего потока **Ключевые данные,** (3) входной поток **Пароль** из хранилища **Данные кредитной карты** для проверки вводимого клиентом пароля.
20. Введите имя процесса **Получить пароль** - см. рисунок 2
21. Введите потоки данных **Сообщение**, **Введенный пароль**, **Пароль** - см. рисунок 2.
22. Введите описание процесса **Получить пароль** и потоков данных **Сообщение**, **Введенный пароль**, **Пароль.** Процесс **Получить запрос на обслуживание** осуществляет прием и проверку запроса клиента на проведение необходимой ему банковской операции и имеет на входе-выходе следующие процессы: (1) внешний выходной поток **Сообщение** для информирования клиента о своей готовности принять запрос на обслуживание, (2) входной поток **Запрос на обслуживание** как элемент внешнего потока **Ключевые данные,** (3) входной

поток **Лимит денег** из хранилища **Данные кредитной карты** для контроля наличия денег на счете клиента.

23. Введите имя процесса **Получить запрос на обслуживание** - см. рисунок 2.
24. Введите потоки данных **Сообщение, Запрос на обслуживание, Лимит денег** - см. рисунок 2.
25. Введите описание процесса **Получить запрос на обслуживание** и потоков данных **Сообщение, Запрос на обслуживание, Лимит денег**. Процесс **Обработать запрос на обслуживание** имеет (1) внешний входной поток **Данные по счету** (из сущности **Компьютер банка),** (2) входной поток **Детали клиента,** (3) входной поток **Денежная сумма** (из процесса **Получить запрос на обслуживание**), (4) внешние выходные потоки **Выписка, Деньги, Протокол обслуживания.**
26. Введите имя процесса **Обработать запрос на обслуживание** - см. рисунок 2
27. Введите потоки данных **Данные по счету, Детали клиента, Выписка, Деньги, Протокол обслуживания, Денежная сумма** - см. рисунок 2.
28. Введите описание процесса **Обработать запрос на обслуживание** и потоков данных **Данные по счету, Детали клиента, Выписка, Деньги, Протокол обслуживания, Денежная сумма**. Процесс **Обработать кредитную карту** осуществляет считывание информации с кредитной карты и имеет на входе внешний поток **Кредитная карта,** на выходе - поток **Данные кредитной карты.**
29. Введите имя процесса **Обработать кредитную карту** - см. рисунок 2
30. Введите потоки данных **Кредитная карта, Данные кредитной карты** - см. рисунок 2.
31. Введите описание процесса **Обработать кредитную карту** и потоков данных **Кредитная карта, Данные кредитной карты**.
32. Создайте диаграмму декомпозиции процесса **Обработать запрос на обслуживание**: Процесс **Обработать документацию банка** выполняет обработку внутренней банковской документации по клиенту и имеет (1) входной поток **Детали клиента**, (2) выходной поток **Обработанная документация** (часть внешнего потока Протокол обслуживания).
33. Введите имя процесса **Обработать документацию банка**.
34. Введите потоки данных **Детали клиента, Обработанная документация** - см. рисунок 3.
35. Введите описание процесса **Обработать документацию банка** и потоков данных **Детали клиента, Обработанная документация**. Процесс **Распечатать баланс клиента** выдает справку по истории счета клиента и по балансу клиента и имеет

(1) входные потоки **Детали клиента** и **Данные по балансу** (часть внешнего потока Данные по счету), (2) выходные потоки **Выписка по балансу** (часть внешнего потока Выписка) и **Данные по истории запроса** (часть внешнего потока Протокол обслуживания).

36. Введите имя процесса **Распечатать баланс клиента**.
37. Введите потоки данных **Детали клиента, Данные по балансу**, **Выписка по балансу, Данные по истории запроса -** см. рисунок 3.
38. Введите описание процесса **Распечатать баланс клиента** и потоков данных **Детали клиента, Данные по балансу**, **Выписка по балансу, Данные по истории запроса**. Процесс **Приготовить деньги клиенту** обеспечивает выдачу наличных денег и информирование компьютера банка об изъятии из банка денег. Процесс имеет (1) входные потоки **Денежная сумма** и **Детали клиента**, (2) выходные потоки **Деньги** и Денежная сумма (часть потока Протокол обслуживания).
39. Введите имя процесса **Приготовить деньги клиента**.
40. Введите потоки данных **Денежная сумма**, **Детали клиента, Деньги, Денежная сумма -** см. рисунок 3.
41.

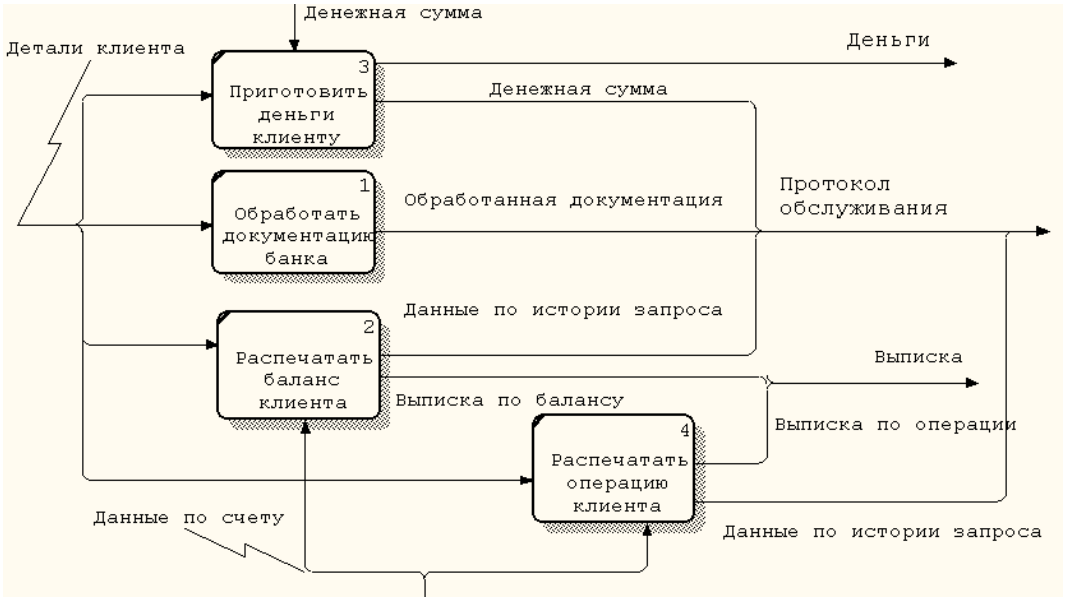


Рисунок 3.

42. Введите описание процесса **Приготовить деньги клиента** и потоков данных **Денежная сумма**, **Детали клиента, Деньги, Денежная сумма**. Процесс **Распечатать операцию клиента** выдает справку по истории счета клиента и уведомление по проведенной операции и имеет потоки (1) входные - **Данные по счету** и **Детали клиента**, (2) выходные - **Выписка по операции** (часть потока Выписка) и **Данные по истории запроса** (часть потока Протокол обслуживания).

43. Введите имя процесса **Распечатать операцию клиента**.
44. Введите **Данные по счету, Детали клиента**, **Выписка по операции, Данные по истории запроса** - см. рисунок 3.
45. Введите описание процесса **Распечатать операцию клиента** и потоков данных **Данные по счету, Детали клиента**, **Выписка по операции, Данные по истории запроса**

Аппаратура и программное обеспечение

Аппаратура. Для выполнения лабораторной работы необходимы:

- персональный суперкомпьютер i7-5930K/ 8D8192D4-2133/ K2200/ 2K40M;

- рабочие станции: монитор 29 + клавиатура + мышь. ПК MDT750/ i75930K/ 4D8192D42133";

- аппаратный сетевой обучающий комплекс мод.VDCN-1502.

Программное обеспечение. Для выполнения лабораторной работы необходима операционная система WINDOWS 7/8/10 и CASE-средство Ramus Educational.

Указания по технике безопасности

При выполнении лабораторной работы **запрещается**:

10. Самостоятельно производить ремонт персонального компьютера, а так же, установку и удаление имеющегося программного обеспечения;

11. Нарушать общепринятые правила техники безопасности при работе с электрооборудованием, в частности, касаться электрических розеток металлическими предметами и т.д.;

12. Принимать пищу, напитки и сорить на рабочем месте пользователя персонального компьютера.

В случае неисправности персонального компьютера необходимо **немедленно** сообщить об этом обслуживающему персоналу лаборатории (системному администратору, оператору).

Методика и порядок выполнения работы

- ознакомиться с вариантами индивидуальных задания и закрепить за собой один из предложенных или предложить свой вариант;
- повторить основные принципы методологии DFD;
- повторить основные принципы работы с системой RAMUS;
- сформировать диаграмму потоков данных системы, указав в качестве названия название проектируемой системы и указав глобальные потоки для системы;
- провести декомпозицию системы на 3-4 основные функции
- связать глобальные потоки с выделенными функциями;
- провести связи между функциями;

- посчитать количественные характеристики для построенной диаграммы.

Содержание отчета и его форма

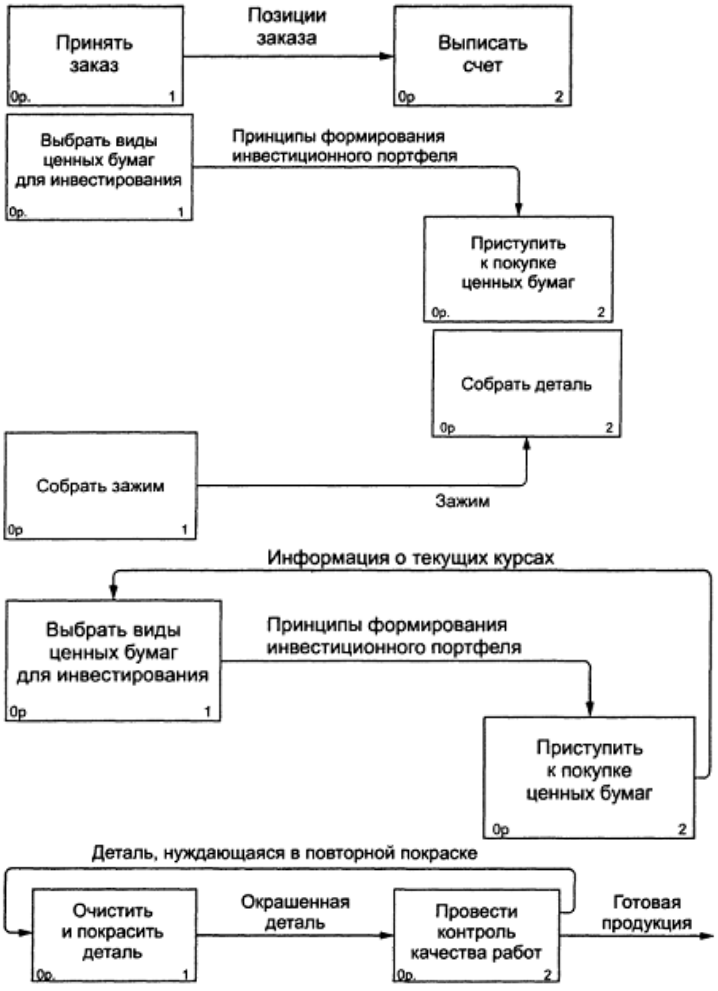
Отчет по лабораторной работе оформляется в виде текстового документа и должен включать:

1. Название лабораторной работы.
2. Цель лабораторной работы.
3. Формулировку индивидуального задания и результат его выполнения.
4. Краткие выводы по результатам выполнения лабораторной работы.

Контрольные вопросы

1. Что называют моделью предметной области?
2. Что такое язык моделирования?
3. Что такое нотация?
4. Укажите главный критерий адекватности структурной модели
5. Что такое "Объект"?
6. Что такое "Функция (операция)"?
7. Что такое "Событие"?
8. Укажите идеи, лежащие в понимании термина " системный структурный анализ"
9. Что называют IDEF0-технологией структурного анализа и проектирования?
10. Что называют IDEF3-технологией структурного анализа и проектирования?
11. Что называют DFD технологией структурного анализа и проектирования?
12. Что называют функциональной диаграммой?
13. Что называют моделью данных?
14. Что такое методология SADT?
15. Что называют функциональной моделью SADT?
16. Что такое действие (фукнция) в модели IDEF0?
17. Что такое "Функциональный блок (Activity Box)" в функциональной методике IDEF0?
18. Что такое "Интерфейсная дуга (Arrow)" в функциональной методике IDEF0?
19. Что такое стрелка типа I (Input) в модели IDEF0?
20. Что такое стрелка типа C (Control) в модели IDEF0?
21. Что такое стрелка типа O (Output) в модели IDEF0?
22. Что такое стрелка типа M (Mechanism) в модели IDEF0?
23. Что такое "Декомпозиция (Decomposition)" в функциональной методике IDEF0?

24. Что такое "Глоссарий (Glossary)" в функциональной методике IDEF0?
25. Что называют Стрелками входа в модели IDEF0?
26. Что называют Стрелками управления в модели IDEF0?
27. Что называют Стрелками выхода в модели IDEF0?
28. Что называют Стрелками механизма исполнения в модели IDEF0?
29. Укажите рисунок с изображением комбинации стрелок выход - вход модели IDEF0 (и так для всех типов стрелок)
30. Что называют туннелем в модели IDEF0?
31. Укажите рисунок с изображением комбинации стрелок выход - управление модели IDEF0. (И так для разных сочетаний)



32. Что называют методологией описания процессов IDEF3?
33. Укажите цель составления диаграмм IDEF3.
34. Что называют сценарием процесса в модели IDEF3?
35. Укажите тип связи, изображенный на рисунке (для всех трех типов связи)
36. Что называют соединением в модели IDEF3?
37. Укажите изображение асинхронного "И"-соединения в модели IDEF3.
38. Укажите изображение асинхронного "ИЛИ"-соединения в модели IDEF3.

39. Укажите изображение асинхронного соединения "Эксклюзивное ИЛИ" в модели IDEF3.
40. Укажите изображение синхронного "И"-соединения в модели IDEF3.
41. Укажите изображение синхронного "ИЛИ"-соединения в модели IDEF3.
42. Укажите изображение синхронного соединения "Эксклюзивное ИЛИ" в модели IDEF3.

Защита лабораторной работы

Оформленный, в соответствии с требованиями ГОСТов, отчет представляется студентом преподавателю для проверки и последующей защиты. Защита отчета по лабораторной работе производится студентом только индивидуально.

В ходе защите лабораторной работы студент отвечает на вопросы преподавателя (поясняет методику выполнения индивидуального задания, отвечает на контрольные вопросы и т.д.).

Отчет, оформленный с отступлениями от требований ГОСТов, небрежно и неаккуратно к защите не принимается.

Лабораторная работа № 10

Построение диаграмм DFD по варианту предметной области

Цель и содержание: Лабораторные работы №7-10 ориентированы на изучение CASE-средства RAMUS. С помощью данного средства проектирования необходимо разработать модель "Как есть (As Is)" для выбранной предметной области.

Процесс создания диаграмм начинается с этапа изучения предметной области, краткое описание которой приведено у вас в варианте. Все вопросы, не освещенные в задании, вы либо придумываете, либо уточняете у преподавателя.

Формируемые компетенции: ПК-2, ПК-6. ПК-7.

Теоретическое обоснование

Диаграммы потоков данных (DFD) используются для описания документооборота и обработки информации. Подобно IDEF0, DFD представляет систему как сеть связанных между собой функций. Их можно использовать как дополнение к модели IDEF0 для более наглядного отображения текущих операций документооборота в системах обработки информации.

Источники информации - **внешние сущности** - порождают информационные **потоки данных**, переносящие информацию к **процессам**. Те в свою очередь преобразуют информацию и порождают новые потоки, которые переносят информацию к другим процессам, **хранилищам данных** или **внешним сущностям** - потребителям информации.

Внешняя сущность - представляет собой материальный объект или физическое лицо (заказчик, склад), представляющее собой источник или приемник информации, который находится за пределами границ анализируемой системы. Обозначается в виде прямоугольника с тенью и обычно располагаются по краям диаграммы. Одна внешняя сущность может быть использована многократно на одной диаграмме, чтобы не рисовать слишком запутанных стрелок.

Процесс - представляет собой функции системы, преобразующие входные потоки в выходные в соответствии с определенным алгоритмом. Изображаются прямоугольниками со скругленными углами. Их смысл совпадает со смыслом функций IDEF0. Они также имеют входы и выходы, но не поддерживают управления и механизмы. Физически процесс - это например, отдел, выполняющее обработку входных документов и выпуск отчетов, или программа, аппаратно реализованное логическое устройство.

Поток данных - определяет информацию, передаваемую через некоторое соединение от источника к приемнику. Реальный поток

данных может быть информацией передаваемой по кабелю, пересылаемыми по почте письмами, магнитными дискетами. Поскольку в DFD каждая сторона блока не имеет четкого назначения, как в IDEF0, стрелки могут подходить и выходить из любой грани блока. В DFD также применяется двунаправленные стрелки для описания диалогов типа "команда-ответ".

Хранилище данных - позволяет описать данные, которые необходимо сохранить в памяти прежде, чем использовать в функциях. То есть в отличие от стрелок, описывающих объекты в движении, хранилища изображают объекты в покое. Хранилище физически может быть реализован в виде ящика в картотеки, таблицы в оперативной памяти, файла на магнитном диске.

Аппаратура и программное обеспечение

Аппаратура. Для выполнения лабораторной работы необходимы:

- персональный суперкомпьютер i7-5930K/ 8D8192D4-2133/ K2200/ 2K40M;

- рабочие станции: монитор 29 + клавиатура + мышь. ПК MDT750/ i75930K/ 4D8192D42133";

- аппаратный сетевой обучающий комплекс мод.VDCN-1502.

Программное обеспечение. Для выполнения лабораторной работы необходима операционная система WINDOWS 7/8/10 и CASE-средство Ramus Educational.

Указания по технике безопасности

При выполнении лабораторной работы **запрещается**:

13. Самостоятельно производить ремонт персонального компьютера, а так же, установку и удаление имеющегося программного обеспечения;

14. Нарушать общепринятые правила техники безопасности при работе с электрооборудованием, в частности, касаться электрических розеток металлическими предметами и т.д.;

15. Принимать пищу, напитки и сорить на рабочем месте пользователя персонального компьютера.

В случае неисправности персонального компьютера необходимо **немедленно** сообщить об этом обслуживающему персоналу лаборатории (системному администратору, оператору).

Методика и порядок выполнения работы

- ознакомиться с вариантами индивидуальных задания и закрепить за собой один из предложенных или предложить свой вариант;

- повторить основные принципы методологии DFD;
- повторить основные принципы работы с системой RAMUS;
- сформировать диаграмму потоков данных системы, указав в качестве названия название проектируемой системы и указав глобальные потоки для системы;
- провести декомпозицию системы на 3-4 основные функции
- связать глобальные потоки с выделенными функциями;
- провести связи между функциями;
- посчитать количественные характеристики для построенной диаграммы.

Содержание отчета и его форма

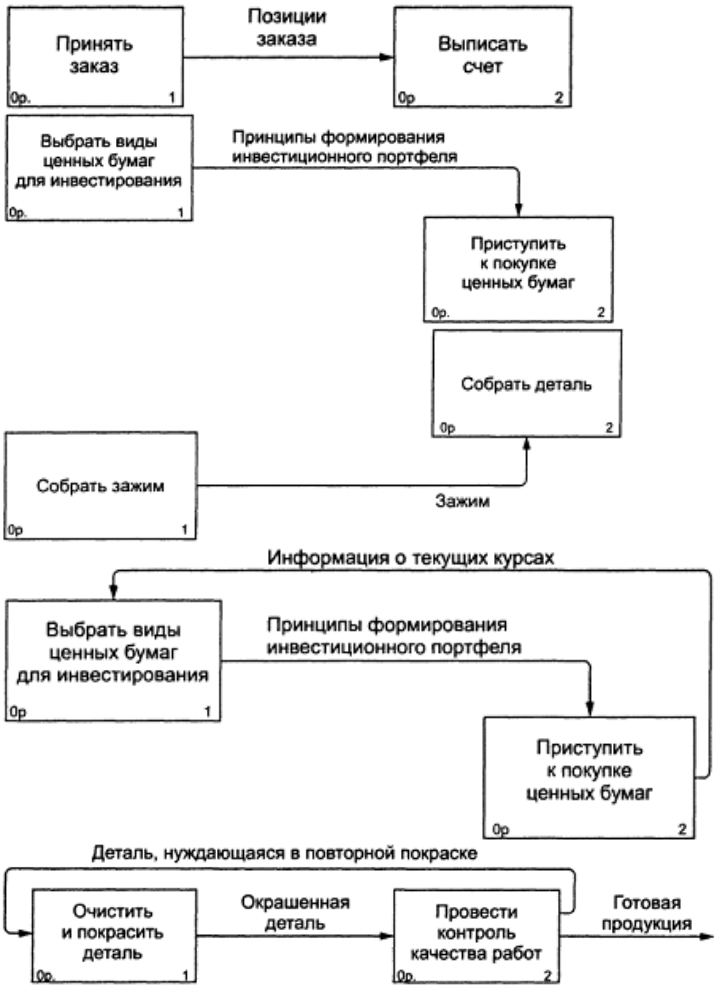
Отчет по лабораторной работе оформляется в виде текстового документа и должен включать:

5. Название лабораторной работы.
6. Цель лабораторной работы.
7. Формулировку индивидуального задания и результат его выполнения.
8. Краткие выводы по результатам выполнения лабораторной работы.

Контрольные вопросы

43. Что называют моделью предметной области?
44. Что такое язык моделирования?
45. Что такое нотация?
46. Укажите главный критерий адекватности структурной модели
47. Что такое "Объект"?
48. Что такое "Функция (операция)"?
49. Что такое "Событие"?
50. Укажите идеи, лежащие в понимании термина " системный структурный анализ"
51. Что называют IDEF0-технологией структурного анализа и проектирования?
52. Что называют IDEF3-технологией структурного анализа и проектирования?
53. Что называют DFD технологией структурного анализа и проектирования?
54. Что называют функциональной диаграммой?
55. Что называют моделью данных?
56. Что такое методология SADT?
57. Что называют функциональной моделью SADT?
58. Что такое действие (фукнция) в модели IDEF0?
59. Что такое "Функциональный блок (Activity Box)" в функциональной методике IDEF0?

60. Что такое "Интерфейсная дуга (Arrow)" в функциональной методике IDEF0?
61. Что такое стрелка типа I (Input) в модели IDEF0?
62. Что такое стрелка типа C (Control) в модели IDEF0?
63. Что такое стрелка типа O (Output) в модели IDEF0?
64. Что такое стрелка типа M (Mechanism) в модели IDEF0?
65. Что такое "Декомпозиция (Decomposition)" в функциональной методике IDEF0?
66. Что такое "Глоссарий (Glossary)" в функциональной методике IDEF0?
67. Что называют Стрелками входа в модели IDEF0?
68. Что называют Стрелками управления в модели IDEF0?
69. Что называют Стрелками выхода в модели IDEF0?
70. Что называют Стрелками механизма исполнения в модели IDEF0?
71. Укажите рисунок с изображением комбинации стрелок выход - вход модели IDEF0 (и так для всех типов стрелок)
72. Что называют туннелем в модели IDEF0?
73. Укажите рисунок с изображением комбинации стрелок выход - управление модели IDEF0. (И так для разных сочетаний)



74. Что называют методологией описания процессов IDEF3?
75. Укажите цель составления диаграмм IDEF3.

76. Что называют сценарием процесса в модели IDEF3?
77. Укажите тип связи, изображенный на рисунке (для всех трех типов связи)
78. Что называют соединением в модели IDEF3?
79. Укажите изображение асинхронного "И"-соединения в модели IDEF3.
80. Укажите изображение асинхронного "ИЛИ"-соединения в модели IDEF3.
81. Укажите изображение асинхронного соединения "Эксклюзивное ИЛИ" в модели IDEF3.
82. Укажите изображение синхронного "И"-соединения в модели IDEF3.
83. Укажите изображение синхронного "ИЛИ"-соединения в модели IDEF3.
84. Укажите изображение синхронного соединения "Эксклюзивное ИЛИ" в модели IDEF3.

Защита лабораторной работы

Оформленный, в соответствии с требованиями ГОСТов, отчет представляется студентом преподавателю для проверки и последующей защиты. Защита отчета по лабораторной работе производится студентом только индивидуально.

В ходе защите лабораторной работы студент отвечает на вопросы преподавателя (поясняет методику выполнения индивидуального задания, отвечает на контрольные вопросы и т.д.).

Отчет, оформленный с отступлениями от требований ГОСТов, небрежно и неаккуратно к защите не принимается.

Лабораторные работы № 11 – 13. Построение диаграмм вариантов использования. Типовой вариант. Построение диаграмм вариантов использования по варианту предметной области. Построение диаграмм вариантов использования. Описание

Цель и содержание: Лабораторные работы №11-13 ориентированы на изучение CASE-средства **ArgoUML**.

- изучение основных этапов проведения проектирования в редакторах UML,
- изучение основных элементов нотации, применяемых в CASE-пакетах объектно-ориентированного проектирования,
- изучение диаграмм вариантов использования,
- изучение их применения в процессе постановки задачи

Формируемые компетенции: ПК-2, ПК-6. ПК-7.

Теоретическое обоснование

Системы не существуют в изоляции. Как правило, они взаимодействуют с актерами - людьми или программами - которые используют систему в своих целях, причем каждый актер ожидает, что она будет вести себя определенным, вполне предсказуемым образом. **Прецедент** (Use case) специфицирует поведение системы или ее части и представляет собой описание множества последовательностей действий (включая варианты), выполняемых системой для того, чтобы актер мог получить определенный результат.

С помощью прецедентов можно описать поведение разрабатываемой системы, не определяя ее реализацию. Таким образом, они позволяют достичь взаимопонимания между разработчиками, экспертами и конечными пользователями продукта. Кроме того, прецеденты помогают проверить архитектуру системы в процессе ее разработки. Реализуются они кооперациями.

Хорошо структурированные прецеденты описывают только существенное поведение системы или подсистемы и не являются ни слишком общими, ни слишком специфическими.

Хорошо спроектированный жилой дом - это не просто ряд стен, подпирающих крышу, которая защищает жильцов от непогоды. Работая вместе с архитектором над проектом дома, вы наверняка будете учитывать предполагаемое использование помещения. Если вы любите приглашать гостей, следует продумать план гостиной, чтобы людям было удобно общаться. Проектируя кухню, нужно определить местонахождение шкафчиков и бытовой техники. Неплохо было бы также принять во внимание маршрут транспортировки продуктов из машины на кухню, от которого во многом зависит взаимное расположение комнат. Если у вас большая семья, надо позаботиться о количестве и размещении ванных комнат, иначе каждое утро там будут выстраиваться очереди.

Размышления о том, как вы и ваша семья будете распоряжаться домом, - это пример анализа на основе прецедентов. Вы рассматриваете разные способы использования дома, которые в конечном счете обусловливают архитектуру. Для многих семей прецеденты использования схожи - во всех домах едят, спят, растят детей, отдыхают и хранят воспоминания. Но в каждом случае выдвигаются индивидуальные требования (или разновидности базовых требований) к жилищу. Потребности большой семьи, например, будут отличаться от запросов молодого человека, только что закончившего колледж. Эти различия окажут решающее влияние на то, как будет выглядеть готовый дом.

Важнейшая особенность разработки прецедентов состоит в том, что вы не специфицируете, как они будут реализованы. Например, поведение банкомата можно

описать с помощью вариантов взаимодействия с ним пользователей, но вам не обязательно знать, как он устроен. Прецеденты специфицируют желаемое поведение, но ничего не говорят о том, как его достичь. И, что очень важно, это позволяет вам как эксперту или конечному пользователю общаться с разработчиками, конструирующими систему в соответствии с вашими требованиями, не углубляясь в детали реализации. Подробности будут рассмотрены позже, а на данном этапе вы можете сконцентрироваться на наиболее существенных проблемах.

В UML поведение моделируется с помощью прецедентов, специфируемых в отрыве от реализации. **Прецедент** - это описание множества последовательностей действий (включая их варианты), которые выполняются системой для того, чтобы актер получил результат, имеющий для него определенное значение. Это определение включает в себя несколько важных пунктов.

Прецедент описывает множество последовательностей, каждая из которых представляет взаимодействие сущностей, находящихся вне системы (ее актеров), с системой как таковой и ее ключевыми абстракциями. Такие взаимодействия являются в действительности функциями уровня системы, которыми вы пользуетесь для визуализации, специфицирования, конструирования и документирования ее желаемого поведения на этапах сбора и анализа требований. Прецедент представляет функциональные требования к системе в целом. Например, основной прецедент в работе банка - это обработка займов.

Прецеденты предполагают взаимодействие актеров и системы. Актер представляет собой логически связанное множество ролей, которые играют пользователи прецедентов во время взаимодействия с ними. Актерами могут быть как люди, так и автоматизированные системы. Например, при моделировании работы банка процесс обработки займов включает в себя, помимо всего прочего, взаимодействие между клиентом и сотрудником кредитного отдела.

Развитие прецедентов может осуществляться по-разному. В любой хорошо продуманной системе существуют прецеденты, которые либо являются специализированными версиями других, более общих, либо входят в состав прочих прецедентов, либо расширяют их поведение. Общее поведение множества прецедентов, допускающее повторное применение, можно выделить, организуя их в соответствии с тремя описанными видами отношений. В частности, при моделировании работы банка базовый прецедент, описывающий обработку займов, подразделяется на несколько вариаций, от оформления крупной закладной до выдачи маленькой деловой ссуды. Однако все эти прецеденты имеют нечто общее в особенностях поведения, например оценку платежеспособности клиента.

Всякий прецедент должен выполнять некоторый объем работы. С точки зрения данного актера, прецедент делает нечто представляющее для него определенную ценность, например вычисляет результат, создает новый объект или изменяет состояние другого объекта. В примере с работой банка процесс обработки заявки на займ приводит к подписанию расходного ордера и материализуется в виде некоторой суммы денег, вручаемой клиенту.

Прецеденты могут быть применены ко всей системе или к ее части, в том числе к подсистемам или даже к отдельным классам и интерфейсам. В любом случае прецеденты не только представляют желаемое поведение этих элементов, но могут быть использованы как основа для их тестирования на различных этапах разработки. Прецеденты в применении к подсистемам - это прекрасный источник регрессионных тестов, а в применении к системе в целом - источник комплексных и системных тестов. Графическое изображение прецедента и актера показано на рис. 24. Эта нотация позволяет визуализировать прецедент в контексте других прецедентов и отдельно от его реализации.

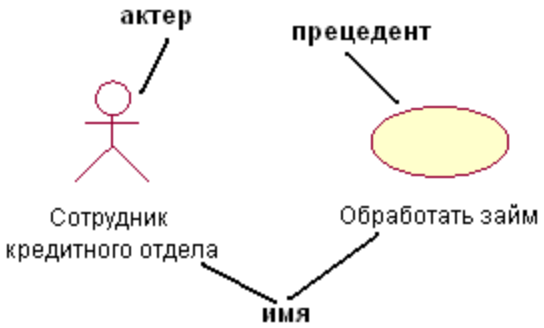


Рисунок 24. Актеры и прецеденты

Любой прецедент должен иметь **имя**, отличающее его от других прецедентов. Оно должно быть уникально внутри объемлющего пакета. Имя прецедента представляет собой текстовую строку. Взятое само по себе, оно называется простым именем. К составному имени спереди добавлено имя пакета, в котором он находится. Обычно при изображении прецедента указывают только его имя, как показано на рис. 25. Имя прецедента может состоять из любого числа букв, цифр и некоторых знаков препинания (за исключением таких, как двоеточия, которые применяются для отделения имени прецедента от имени объемлющего пакета). Имя может занимать несколько строк. На практике для именования прецедентов используют короткие глагольные фразы в активной форме, обозначающие некоторое поведение и взятые из словаря моделируемой системы.

Рисунок 25. Простые и составные имена

Прецеденты и актеры

Актер представляет собой связное множество ролей, которые пользователи прецедентов исполняют во время взаимодействия с ними. Обычно актер представляет роль, которую в данной системе играет человек, аппаратное устройство или даже другая система. Например, если вы работаете в банке, то можете играть роль *СотрудникКредитногоОтдела*. Если в этом банке у вас имеется счет, вы играете роль *Клиента*. Таким образом, экземпляр актера представляет собой конкретную личность, взаимодействующую с системой определенным образом. Хотя вы и используете актеров в своих моделях, они не являются частью системы, так как существуют вне ее.

Как показано на рис. 26, актеров изображают в виде человеческих фигурок. Можно определить общие типы актеров (например, *Клиент*) и затем специализировать их (например, создав разновидность *КоммерческийКлиент*) с помощью отношений обобщения.

Рисунок 26. Актеры

Можно использовать механизмы расширения UML для приписывания актеру стереотипа, чтобы создать другую пиктограмму, адекватную поставленным целям.

Актеров можно связывать с прецедентами только отношениями ассоциации. Ассоциация между актером и прецедентом показывает, что они общаются друг с другом, возможно, посылая или принимая сообщения.

Прецеденты и поток событий

Прецедент описывает, **что** делает система (подсистема, класс или интерфейс), но не определяет, **каким образом** она это делает. В процессе моделирования всегда важно разделять внешнее и внутреннее представления.

Можно специфицировать поведение прецедента путем описания потока событий в текстовой форме - в виде, понятном для постороннего читателя. В описание необходимо включить указание на то, как и когда прецедент начинается и заканчивается, когда он взаимодействует с актерами и какими объектами они обмениваются. Важно обозначить также основной и альтернативный потоки поведения системы.

Например, в контексте банкомата можно было бы следующим образом описать прецедент *ValidateUser* (*ПроверитьПользователя*):

- *Основной поток событий*. Прецедент начинается, когда система запрашивает у клиента его *персональный идентификационный номер* (PIN). *Клиент* (*Customer*) может ввести его с клавиатуры. Завершается ввод нажатием клавиши Enter. После этого система проверяет введенный PIN и, если он правильный, подтверждает ввод. На этом прецедент заканчивается.
- *Исключительный поток событий*. *Клиент* может прекратить транзакцию в любой момент, нажав клавишу Cancel. Это действие начинает прецедент заново. Никаких изменений на счету клиента не производится.
- *Исключительный поток событий*. *Клиент* может в любой момент до нажатия клавиши Enter стереть свой PIN и ввести новый.
- *Исключительный поток событий*. Если *Клиент* ввел неправильный PIN, прецедент запускается сначала. Если это происходит три раза подряд, система отменяет всю транзакцию и не позволяет данному *Клиенту* снова начать работу с банкоматом в течение 60 секунд.

Поток событий в прецеденте можно описать различными способами, в том числе в виде неформализованного структурированного текста (как в примере выше), формализованного структурированного текста (с пред- и постусловиями) или с помощью псевдокода.

Прецеденты и сценарии

Как правило, в начале работы потоки событий прецедента описывают в текстовой форме. По мере уточнения требований к системе будет удобнее перейти к графическому изображению потоков на диаграммах взаимодействия. Обычно для описания главного потока прецедента используют **диаграмму последовательностей**, а для дополнительных - ее варианты.

Желательно отделять главный поток от альтернативных, поскольку прецедент описывает не одну, а множество последовательностей, и выразить все детали интересующего вас прецедента с помощью одной последовательности невозможно.

Например, в системе управления человеческими ресурсами присутствует прецедент *Нанять работника*. У этой общей бизнес-функции существует множество вариаций. Вы можете переманить сотрудника из другой компании (так бывает чаще всего), перевести человека из одного подразделения в другое (эта практика распространена в транснациональных компаниях) или нанять иностранца (особый случай, регулируемый специальными правилами). Каждый вариант описывается своей последовательностью.

Таким образом, один прецедент *Нанять работника* описывает несколько последовательностей, или сценариев, каждый из которых представляет одну из возможных вариаций данного потока событий. **Сценарий** (Scenario) - это некоторая последовательность действий, иллюстрирующая поведение системы. Сценарии находятся в таком же отношении к прецедентам, как экземпляры к классам, то есть сценарий - это экземпляр прецедента.

Относительно сложная система содержит несколько десятков прецедентов, каждый из которых может разворачиваться в несколько десятков сценариев. Для любого прецедента можно выделить основные сценарии, описывающие важнейшие последовательности, и вспомогательные, описывающие альтернативные последовательности.

Прецеденты и кооперации

Как было указано, прецедент описывает желательное поведение системы (подсистемы, класса или интерфейса), но не специфицирует его реализацию. Это важная особенность, поскольку анализ системы (по результатам которого специфицируется ее поведение) по возможности не должен учитывать проблемы реализации (иными словами, как это поведение должно быть материализовано). В конце концов, однако, прецеденты придется реализовать. Для этого необходимо будет создать сообщество классов и других элементов, в результате совместной работы которых будет достигнуто желаемое поведение. Такое сообщество, включая его динамическую и статическую структуру, называется в UML **кооперацией**.

Как видно из рис. 27, реализацию прецедента с помощью кооперации можно специфицировать явно. Но чаще всего один прецедент реализуется в точности одной кооперацией, так что явно описывать отношение не нужно. Хотя вы можете не визуализировать это отношение явно, применяемые для работы с моделями инструментальные средства, скорее всего, будут так или иначе его поддерживать.

Рисунок 27. Прецеденты и кооперации

Нахождение минимального набора хорошо структурированных кооперации, реализующих определенный во всех прецедентах системы поток событий, - основная задача системной архитектуры.

Организация прецедентов

Для организации прецедентов их группируют в пакеты, так же как и классы.

Кроме того, прецеденты можно организовать, определив между ними отношения обобщения, включения и расширения. Эти отношения применяют, чтобы выделить некоторое общее поведение (извлекая его из других прецедентов, которые его включают) или, наоборот, вариации (поместив такое поведение в другие прецеденты, которые его расширяют).

Отношение обобщения между прецедентами аналогично отношениям обобщения между классами. Это означает, что прецедент-потомок наследует поведение и семантику

своего родителя, может замещать его или дополнять его поведение, а кроме того, может быть подставлен всюду, где появляется его родитель (как родитель, так и потомок могут иметь конкретные экземпляры). Например, в банковской системе возможно наличие прецедента *Проверить Клиента*, который отвечает за проверку личности клиента. Он может иметь двух специализированных потомков (*Проверить пароль* и *Сканирование сетчатки*). Оба потомка ведут себя так же, как прецедент *Проверить клиента*, и могут использоваться везде, где используется их родитель, но при этом каждый из них добавляет и свое собственное поведение (первый проверяет текстовый пароль, а второй - рисунок сетчатки глаза). Как показано на рис. 28, обобщения между прецедентами изображаются точно так же, как и обобщения между классами в виде линии с незакрашенной стрелкой.

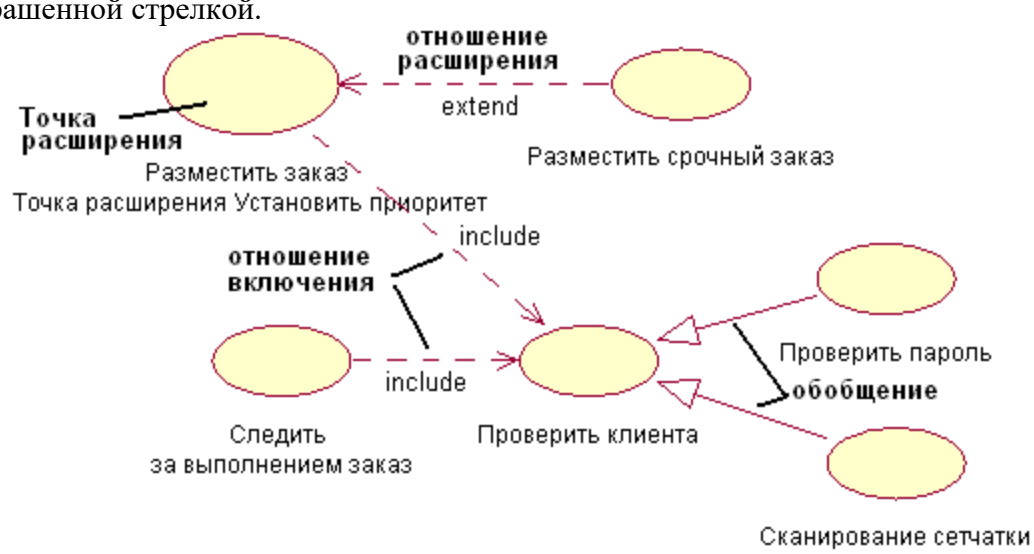


Рисунок 28. Обобщения, включения и расширения

Отношение включения между прецедентами означает, что в некоторой точке базового прецедента инкорпорировано поведение другого прецедента. Включаемый прецедент никогда не существует автономно, а инстанцируется только как часть объемлющего прецедента. Можно считать, что базовый прецедент заимствует поведение включаемых.

Благодаря наличию отношений включения удается избежать многократного описания одного и того же потока событий, поскольку общее поведение можно описать в виде самостоятельного прецедента, включаемого в базовые. Отношение включения является примером делегирования, при котором ряд обязанностей системы описывается в одном месте (во включаемом прецеденте), а остальные прецеденты, когда необходимо, включают эти обязанности в свой набор.

Отношения включения изображаются в виде зависимостей со стереотипом **include**. Чтобы специфицировать место в потоке событий, где базовый прецедент включает поведение другого, вы просто пишете слово **include**, за которым следует имя включаемого прецедента. Проиллюстрируем это на примере, описывающем поток для прецедента *Следить за выполнением заказа*.

Основной поток событий. Получить и проверить номер заказа, **include** (*Проверить клиента*). Запросить статус каждой части заказа и доложить клиенту.

Отношение расширения подразумевает, что базовый прецедент неявно содержит поведение другого прецедента в точке, которая косвенно задается расширяющим прецедентом. Базовый прецедент может быть автономным, но при определенных обстоятельствах его поведение расширяется за счет другого. Базовый прецедент допустимо расширить только в некоторых точках, называемых точками расширения. Можно считать, что расширяющий прецедент передает свое поведение базовому.

Отношение расширения применяют для моделирования таких частей прецедента, которые пользователь воспринимает как необязательное поведение системы. Тем самым можно разделить обязательное и необязательное поведение. Отношения расширения используются также для моделирования отдельных субпотоков, выполняемых лишь при определенных обстоятельствах. Наконец, их применяют для моделирования нескольких потоков, которые могут включаться в некоторой точке сценария в результате явного взаимодействия с актером.

Отношение расширения изображают в виде зависимости со стереотипом **extend**. Точки расширения базового сценария перечисляются в дополнительном разделе. Они являются просто метками, которые могут появляться в потоке базового прецедента. Например, поток для прецедента *Разместить заказ* можно было бы описать нижеследующим образом.

Основной поток событий. **include** (*Проверить Клиента*). Собрать все пункты сделанного клиентом заказа. (*Установить приоритет*). Отправить заказ на обработку.

В данном примере фраза *Установить приоритет* - это точка расширения. Прецедент может содержать несколько точек расширения (причем каждую по несколько раз), идентифицируемых по именам. При обычных обстоятельствах базовый прецедент в этом примере выполняется без учета приоритетности заказа. Если же поступает приоритетный заказ, то поток будет выполняться, как обычно, до точки расширения (*Установить приоритет*), а в ней будет выполнен расширяющий прецедент (*Разместить срочный заказ*), после чего возобновится работа главного потока. Если определено несколько точек расширения, то расширяющие прецеденты будут последовательно выполняться в собственных потоках.

Организация прецедентов путем выделения общего поведения (отношение включения) и различиых вариаций (отношение расширения) является важной составной частью процесса разработки простого, сбалансированного и понятного набора прецедентов системы.

Другие возможности

Прецеденты являются классификаторами и могут иметь **атрибуты** и **операции**, которые изображаются так же, как для классов. **Атрибуты** можно считать объектами внутри прецедента, которые требуются для описания его внешнего поведения, а **операции** - действиями системы, необходимыми для описания потока событий. Эти объекты и операции разрешается включать в диаграммы взаимодействия, чтобы специфицировать поведение прецедента.

Как и ко всем остальным классификаторам, к прецедентам можно присоединять **автоматы**. Позволительно расценивать их как еще один способ описания поведения прецедента.

Типичные приемы моделирования. Поведение элемента

Чаще всего с помощью прецедентов моделируют поведение элемента: системы в целом, подсистемы или класса. При этом важно сконцентрироваться исключительно на том, что должен делать элемент, а не на том, как он это будет делать.

Подобное применение прецедентов к элементам представляет важность по трем причинам. Во-первых, моделируя поведение элемента с помощью прецедентов, эксперты в предметной области могут описать взгляд на систему извне с такой степенью детализации, что разработчики сумеют сконструировать ее внутреннее представление. Прецеденты дают возможность экспертам, конечным пользователям и разработчикам общаться на одном языке. Во-вторых, прецеденты позволяют разработчикам понять назначение элемента. Система, подсистема или класс могут быть сложными образованиями с большим числом операций и других составных частей. Описав прецеденты элемента, вы поможете их потенциальным пользователям разобраться в том, как с ним обращаться. В противном случае им пришлось бы на собственном опыте постигать, как следует использовать тот или иной элемент. В- третьих, прецеденты

являются основой для тестирования каждого элемента на всем протяжении его разработки. Постоянно сравнивая функционирование каждого элемента с прецедентами, вы будете контролировать корректность его реализации. При этом вы не только получаете источник регрессионных тестов, но будете вынуждены при появлении нового прецедента данного элемента пересмотреть реализацию, чтобы убедиться в том, что элемент в достаточной степени изменяем. Если это не так, следует пересмотреть архитектуру.

Моделирование поведения элемента осуществляется следующим образом:

1. Идентифицируйте актеров, взаимодействующих с данным элементом. К числу актеров-кандидатов относятся группы, которые требуют определенного поведения для выполнения своих задач либо необходимы, прямо или косвенно, для выполнения функций элемента.
2. Организуйте актеров, выделив общие и специализированные роли.
3. Для каждого актера рассмотрите основные пути его взаимодействия с элементом. Рассмотрите также взаимодействия, изменяющие состояние элемента или его окружения либо предполагающие реакцию на некоторое событие.
4. Рассмотрите альтернативные (исключительные) способы взаимодействия актеров с элементом.
5. Организуйте выявленное поведение в виде прецедентов, применяя отношения включения и расширения для выделения общего и исключительного поведения.

Например, система розничной торговли должна взаимодействовать с клиентами, которые размещают заказы и хотят отслеживать их продвижение. Система будет отгружать выполненные заказы и выставлять счета клиентам. Как видно из рис. 29, моделировать поведение такой системы можно, объявляя его в виде прецедентов (*Разместить заказ*, *Следить за выполнением заказа*, *Отгрузить заказ* и *Выставить счет*). Можно выделить общее поведение (*Проверить клиента*) и вариации (*Отгрузить частично выполненный заказ*). Для каждого из этих прецедентов следует включить спецификацию поведения с помощью текста, автомата или взаимодействий.

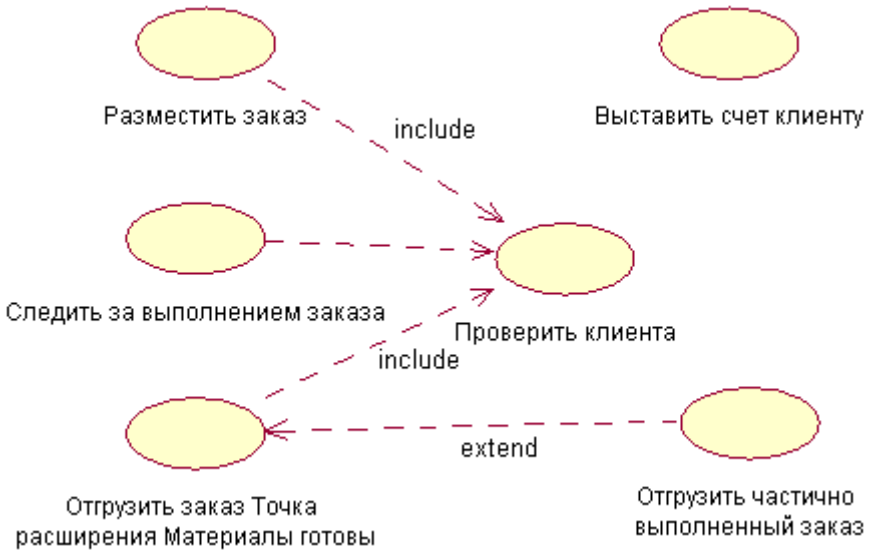


Рисунок 29. Моделирование поведения элемента

По мере развития модели вы обнаружите тенденцию к объединению прецедентов в концептуально и семантически близкие группы. В UML для моделировании таких групп применяются пакеты.

Советы

Моделируя прецеденты в UML, помните, что каждый из них должен представлять некоторое четко идентифицируемое поведение системы или ее части. Хорошо структурированный прецедент обладает следующими свойствами:

- именует простое, идентифицируемое и в некоторой степени атомарное поведение системы или ее части;
- выделяет общее поведение, извлекая его из всех прецедентов, которые его включают;
- выделяет вариации, помещая некоторое поведение в другие прецеденты, которые его расширяют;
- описывает поток событий в степени, достаточной для понимания посторонним читателем;
- описывается с помощью минимального набора сценариев, специфицирующих его нормальную и дополнительную семантику.

Изображая прецеденты в UML, пользуйтесь следующими правилами:

- показывайте только такие прецеденты, которые важны для понимания поведения системы или ее части в данном контексте;
- показывайте только те актеры, которые связаны с этими прецедентами.

Аппаратура и программное обеспечение

Аппаратура. Для выполнения лабораторной работы необходимы:

- персональный суперкомпьютер i7-5930K/ 8D8192D4-2133/ K2200/ 2K40M;

- рабочие станции: монитор 29 + клавиатура + мышь. ПК MDT750/ i75930K/ 4D8192D42133";

- аппаратный сетевой обучающий комплекс мод.VDCN-1502.

Программное обеспечение. Для выполнения лабораторной работы необходима операционная система WINDOWS 7/8/10 и CASE-средство ArgoUML.

Указания по технике безопасности

При выполнении лабораторной работы **запрещается**:

16. Самостоятельно производить ремонт персонального компьютера, а так же, установку и удаление имеющегося программного обеспечения;

17. Нарушать общепринятые правила техники безопасности при работе с электрооборудованием, в частности, касаться электрических розеток металлическими предметами и т.д.;

18. Принимать пищу, напитки и сорить на рабочем месте пользователя персонального компьютера.

В случае неисправности персонального компьютера необходимо **немедленно** сообщить об этом обслуживающему персоналу лаборатории (системному администратору, оператору).

Методика и порядок выполнения работы

- ознакомиться с вариантами индивидуальных задания и закрепить за собой один из предложенных или предложить свой вариант;
- повторить основные принципы методологии моделирования Use Case диаграмм;
- повторить основные принципы работы с системой ArgoUML;
- сформировать диаграмму Use Case системы, указав в качестве названия название проектируемой системы и указав прецеденты и актеров для системы;
- привести подробное описание спроектированной модели Use Case диаграмм;
- посчитать количественные характеристики для построенной диаграммы.

Содержание отчета и его форма

Отчет по лабораторной работе оформляется в виде текстового документа и должен включать:

1. Название лабораторной работы.
2. Цель лабораторной работы.
3. Формулировку индивидуального задания и результат его выполнения.
4. Краткие выводы по результатам выполнения лабораторной работы.

Контрольные вопросы

1. Какие три типа моделей используются при проектировании?
2. Каково назначение концептуальной модели?
3. Назовите основной вид диаграмм в концептуальной модели.
4. Каково назначение логической модели?
5. Назовите основной вид диаграмм в логической модели.
6. Назовите два взгляда на моделируемую систему в логической модели.
7. Какова роль диаграмм взаимодействия объектов в логической модели?
8. Какова роль диаграмм последовательности взаимодействий в логической модели?
9. Каково назначение физической модели?
10. Назовите основной вид диаграмм в физической модели.
11. В чем смысл процедуры итерационного моделирования?

12. В чем смысл варианта использования?
13. Каково назначение диаграмм вариантов использования?
14. Назовите основные свойства вариантов использования.
15. Назовите основные компоненты диаграмм вариантов использования.
16. Что такое «действующее лицо»?
17. Какую роль могут играть действующие лица по отношению к варианту использования?
18. Каким образом анализ внешних событий позволяет определить варианты использования системы?

Защита лабораторной работы

Оформленный, в соответствии с требованиями ГОСТов, отчет представляется студентом преподавателю для проверки и последующей защиты. Защита отчета по лабораторной работе производится студентом только индивидуально.

В ходе защите лабораторной работы студент отвечает на вопросы преподавателя (поясняет методику выполнения индивидуального задания, отвечает на контрольные вопросы и т.д.).

Отчет, оформленный с отступлениями от требований ГОСТов, небрежно и неаккуратно к защите не принимается.

Лабораторные работы № 15 – 16. Построение диаграмм взаимодействия. Типовой вариант. Построение диаграмм взаимодействия по варианту предметной области

Цель и содержание: Лабораторные работы №14-15 ориентированы на изучение CASE-средства Umbrello.

Формируемые компетенции: ПК-2, ПК-6. ПК-7.

Диаграммы взаимодействия являются моделями, описывающими поведение взаимодействующих групп объектов.

Как правило, диаграмма взаимодействия охватывает поведение только одного варианта использования. На такой диаграмме отображается ряд объектов и те сообщения, которыми они обмениваются между собой в рамках данного варианта использования.

Данный подход будет проиллюстрирован на примере простого варианта использования, который описывает следующее поведение:

- «Менеджер» запрашивает текущий «Отчет» «Исполнителя»;
- если «Отчет» устарел, «Менеджер» посылает запрос «Исполнителю» на обновление «Отчета»;
- «Исполнитель» создает новый «Отчет»;
- «Менеджер» делает повторный запрос «Отчета».

Существует два вида диаграмм взаимодействия: диаграммы последовательности (sequence diagrams) и кооперативные диаграммы (collaboration diagrams).

Теоретическое обоснование

Обычно веб-сайты представляют собой сложные и динамические структуры. На их разработку выделяют короткое время, чтобы сайт как можно быстрее заработал и давал эффект. Часто разработчики сразу же садятся за написание кода, даже не обдумав, что они собираются создавать, и как они это собираются сделать. Код для сервера часто пишется с листа, таблицы в базах данных создаются по мере надобности, и в итоге иногда архитектура системы начинает развиваться в совершенно непредсказуемом направлении. Однако с помощью простого моделирования и строгого подхода к программированию можно сделать процесс разработки в гораздо более гладким и гарантировать, что созданную web-систему будет просто поддерживать в дальнейшем.

UML (Unified Modeling Language) - унифицированный язык моделирования - это язык, с помощью которого описываются системы. Следовательно этот язык может замечательно помочь вам описать и отобразить вашу будущую систему. Данная статья продемонстрирует некоторые способы, с помощью которых, используя язык UML, вы можете моделировать свои web-сайты. Язык UML может быть очень сложным в усвоении, но некоторыми частями UML пользоваться очень легко, а это поможет вам создавать лучшие системы в меньшие сроки. В качестве примера мы возьмем приложение, которое было описано в статье "Создание WML-приложения".

Поскольку в данной статье мы не будем вдаваться в специфику языка UML, я предлагаю вам ознакомиться с отдельной статьей под названием "Что нужно знать о UML", в которой как в справочнике описаны некоторые термины и знаки языка. В выноске к статье перечислены ссылки на ресурсы, из которых можно почерпнуть более глубокие знания как о UML, так и о достойных примерах его использования в дизайне.

Стадия планирования

Вне зависимости от того, строите ли вы систему с нуля, переносите ее на другую платформу или улучшаете ее функциональность, с самого начала требуется составить план, чтобы гарантировать принятие правильного решения. Когда вы работаете в группе, состоящей из нескольких человек, ясное понимание плана движения вперед и четкое распределение работы становятся просто неоценимыми. Постарайтесь основательно разобраться в следующих аспектах системы на стадии планирования:

- Кто будет пользователями системы, и каковы будут роли этих пользователей
- Требования приложения
- Взаиморасположение страниц и порядок перемещения между ними
- Инструменты и технологии, которые будут использоваться на сайте

Знайте своих пользователей

Важно знать, какие типы пользователей будут работать с вашей системой. И не только потому, что для анализа системы вам придется беседовать с представителями этих пользователей (анкеты, письма, личные беседы и т.д.), но и потому, что часто вам придется моделировать систему так, чтобы в ней работали пользователи с различными ролями и привилегиями. Классифицируя пользователей и изучая их требования, вы узнаете такие сведения, которые скажут вам, как надо строить базу данных, какие ограничения обеспечить, как сгруппировать страницы, как организовать обучение и помощь, какая специфическая информация должна быть на сайте, а кроме того - ваши знания о пользователях могут быть интересны и вашим рекламодателям.

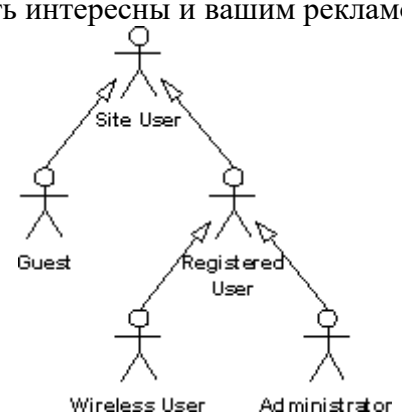


Рис. 1: Иерархия исполнитель-роль

На представленном выше рисунке изображены несколько групп пользователей (называемые на языке UML "исполнителями" (actors), которые будут работать с web-сайтом. В данном случае самый общий тип пользователя ("Пользователь сайта" - Site user) расположен вверху диаграммы. Сплошная стрелка обозначает отношение обобщения, то есть у нас имеются две более специфические категории пользователей: посетители-гости и зарегистрированные пользователи. Характеристики, которые будут общими у обеих групп пользователей, будут приписаны исполнителю "пользователь сайта", а привилегии, специфичные для посетителей-гостей и зарегистрированных пользователей, будут приписаны более мелкой соответствующей роли. В зависимости от используемой программы, вы можете прямо в диаграмме создавать описания и прикреплять их к определенному исполнителю, т.е. вам нет необходимости создавать отдельно диаграмму и отдельно документы, ее описывающие. В данном приложении помимо прочего пользователи делятся на посетителей, подключившихся к сайту по беспроводной связи, и администраторов. Обе эти группы - дальнейшая подразделение группы зарегистрированных пользователей, в системе у них будут различные функции.

Определите требования

Уточните, что именно вы собираетесь построить, прежде чем начнете работу. Хоть вас и будет мучить соблазн выяснять это по мере написания кода, гораздо более эффективно - провести "мозговой штурм", нарисовать пару схем и записать все на бумаге. Часто нерентабельно писать подробную спецификацию требований к сайту, но вы обязаны хотя бы приблизительно набросать пару диаграмм и написать пару строк о том, какие функции будет выполнять сайт. Вот именно здесь и приходят на помощь блочные диаграммы. Рассматривайте блоки-действия как пакет функций - как нечто, что обычно соответствует странице сайта, приложению или действию, осуществляемому на сайте (например: проверка пароля посетителя, изменение настроек пользователя или удаление устаревших пользователей). Следующий рисунок представляет собой краткую диаграмму блоков-действий, с помощью которой планируется web-сайт. Обратите внимание, что на данной диаграмме не показаны все действия, предусмотренные в нашем приложении - обычно для описания всей функциональности будущего сайта создается несколько диаграмм.

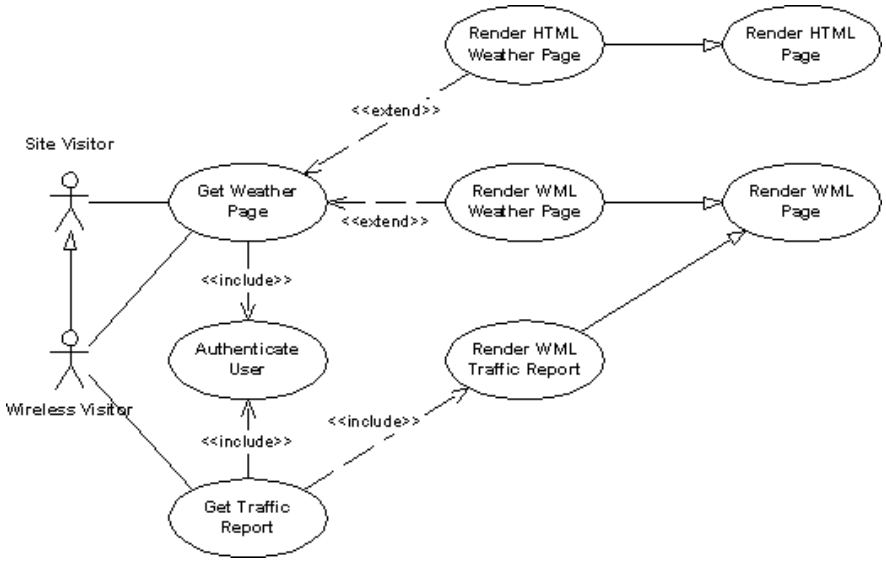


Рис.2: Диаграмма блоков-действий

Даже с помощью такой простой диаграммы как эта, можно запросто описать огромный объем информации. Например, отношение включения ("include"), показывает, что два определенных действия включают в себя одну и туже функцию аутентификации пользователя. Отношение расширения ("extend") указывает, что страница с информацией о погоде может выдаваться как в HTML так и в WML-формате. Отношение обобщения ("generalization") показывает, что для выдачи конкретных страниц будет использовать более общая процедура, под названием "выдача HTML-страницы" или "выдача WML-страницы" с целью обеспечить единство внешнего вида и поведения всех страниц на web-сайте.

Также диаграмма показывает, что посетители подключенные к сайту по беспроводной связи, будут иметь доступ к тем его разделам, которые не будут видны для других посетителей. В данной диаграмме только эта группа пользователей может посмотреть отчеты о дорожной обстановке. Мы посчитали, что отчеты о дорожной обстановке понадобятся лишь людям, находящимся в пути, и потому нам нет нужды прилагать усилия по генерации страниц в каких-то иных форматах, кроме формата WML. Следовательно, действие "Выдать отчет о дорожной обстановке" не нужно расширять (extend) вариантами WML или HTML страниц, оно может напрямую включить (include) действие "Создать WML-страницу".

Как правило все эти диаграммы и блоки-действия сопровождаются какими-то краткими описаниями. Есть несколько статей, где обсуждается то, как создаются диаграммы с блоками-действиями и как пишутся к ним объяснения (например, статья

Алистер Кокбёрн - "Шаблон блоков-действий" - Alistair Cockburn's Use Case Template). Если вкратце, то вы обычно описываете, что должно произойти внутри блока-действия, кто этим действие будет пользоваться, как оно будет вызываться, как останавливаться, и какие варианты результатов действия могут получиться.

Спланируйте страницы

Во время работы над блоками-действиями у вас сформируются некоторые представления по поводу того, каких и сколько страниц понадобится создавать для сайта. Возможно еще до начала работу у вас будут хорошие идеи о некоторых из страниц, однако с помощью блочных диаграмм вы сможете взглянуть на проблему с другой точки зрения. Нужно ли посетителю сайта столько много страниц? Не слишком ли много функций возложено на эту страницу? Трудно ли перемещаться по сайту? Скажем, трудно ли попасть с первой страницы на одну из часто выполняемых функций? Найдите ответы на все эти вопросы в блочных диаграммах, т.е. до того, как вы начнете делать наброски своих страниц и создавать прототип сайта.

После того, как вырисуется в общих чертах диаграмма блоков-действий, можно приступать к грубой прикидке структуры сайта. Для этого опять-таки можно воспользоваться языком UML. Кто-то скажет, что страницы и файлы нужно моделировать с помощью инструмента, где используются компонентные диаграммы. Лично я нахожу, что с инструментами, где используются классовые диаграммы, работать проще (см. следующий рисунок).

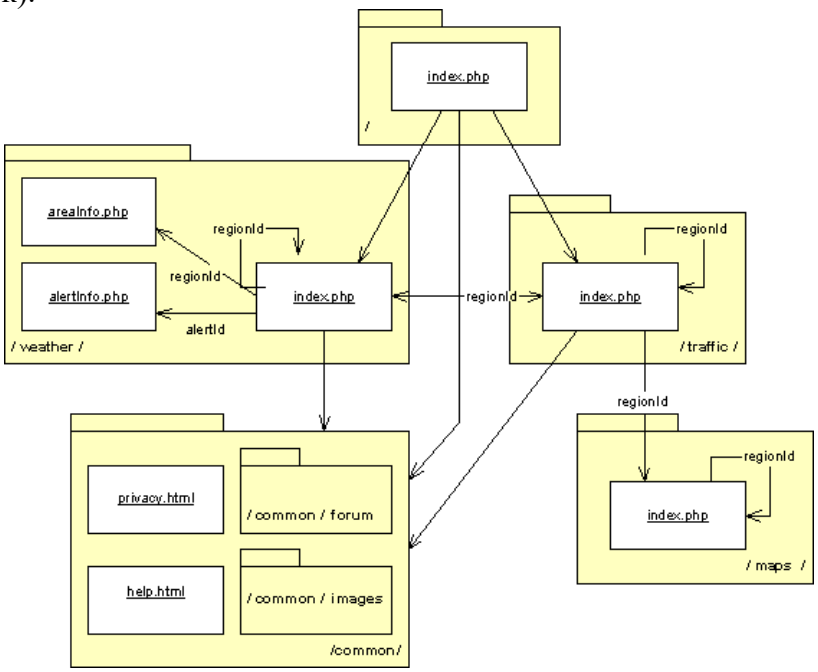


Рис. 3: Диаграмма страниц

На данной диаграмме все функции сайта были распределены на несколько областей:

- / - корень сайта
- /common/ - общие функции, изображения, скрипты, таблицы стилей и так далее
- /maps/ - изображения карт и направлений
- /traffic/ - отчеты о дорожной обстановке
- /weather/ - отчеты о погоде

Данный рисунок также показывает, какие параметры передаются между страницами. Переменная regionId - самая главная, так как она обозначает район, в котором заинтересован пользователь (будь то код округа, города или провинции). С помощью переменной regionId вы передаете информацию о регионе между страницами, таким

образом пользователь сможет перейти от отчета о погоде к отчету о дорожной обстановке в том же самом регионе. Что касается каталога common, то здесь вы видите, что область состоит из целого пакета (package), а не из отдельных файлов. Это просто прием укрощения хаоса в диаграмме, так как пакеты будут взаимно использовать большинство, если не все, файлы, расположенные в каталоге /common/.

Данная диаграмма очень полезна для планирования сайта, так как она помогает избежать путаницы. Кроме того она служит вам шаблоном, которым вы будете пользоваться при создании других web-сайтов с такой же структурой.

Выберите инструменты

Для маленьких сайтов выбрать инструменты и технологии достаточно просто. В особенности в тех случаях, когда важную роль играет стоимость проекта, возможны лишь несколько комбинаций - Apache, MySQL или PostgreSQL, PHP или Perl или JSP/сервлеты. Самым популярным решением обычно оказывается комбинация Apache + PHP + MySQL, а наличие дешевых хостингов с этой комбинацией только подталкивает к ее использованию. Более мощным web-сайтам требуется оценивать и проверять инструменты более тщательно, прежде чем вкладывать в них свои время и силы. Более подробное обсуждение по оценке и выбору сервера приложений можно найти в статье "Выберите правильный сервер приложений J2EE". На следующем рисунке показана примерная компонентная диаграмма, с помощью которой описывается архитектура сайта. Данная простая диаграмма пожалуй подойдет для описания многих web-сайтов, работающих сейчас в Сети. Следовательно ее не придется каждый раз воссоздавать заново, так как в ней нет какой-либо интересной, специфической или уникальной информации.

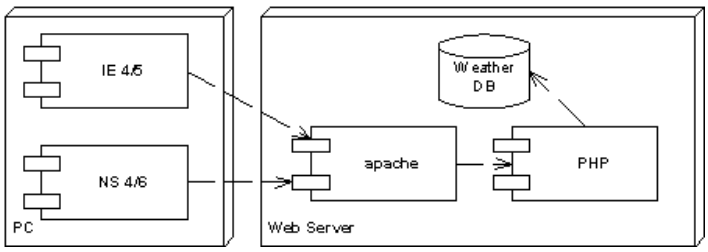


Рис. 4: Диаграмма архитектуры

Мы не будем описывать весь цикл разработки программного продукта в данной статье, но важно отметить, что именно теперь модно начинать создавать макет сайта и прототипы страниц. Запишите на будущее пару идей по структуре сайта и организации его страниц, так как в конечном счете вам захочется создать какой-то общий код, генерирующий меню, боковые панели и общую компоновку страниц, чтобы использовать его в других сайтах. Кроме того, если вы переносите проект на новые инструменты и технологии, прототипы помогут выяснить вам, насколько дизайн осуществим при данных условиях, и насколько ваша команда подготовлена к использованию новых инструментов.

Этап дизайна

Этап дизайна должен накладываться на этап анализа - уже в то время, когда вы узнаете все больше и больше подробностей о будущей системы, вы начинаете делать наброски того, как она будет построена. Невозможно сначала на 100% проанализировать систему, а затем прямо перейти к ее дизайну. Часто требования к системе развиваются, а иногда и ваша идея построения системы может послужить толчком к развитию требований к ней (и наоборот). Все разработчики занимаются дизайном - но некоторые занимаются им прямо во время написания кода. Вообще-то это работает, но при таком методе труднее становится управлять сложным кодом и совместно работать на ним в команде. Несколько минут потраченные на моделирование системы с помощью диаграмм в конце окупят с лихвой.

Дизайн на будущее

Большинство разработчиков проводит большую часть времени за отлаживанием и переделкой кода чем, за его написанием. В особенности это справедливо, когда вам

приходится работать над несколькими web-сайтами сразу. Хорошие дизайнерские находки можно переносить с одного сайта на другой в части структуры, организации и построения кода. Однако если вы наворотите код без предварительного планирования, преимущества его использования в перспективе сойдут на нет. Очень эффективным способом визуального изображения сайта является построение нескольких диаграмм классов. Приведенные рисунки показывают несколько ключевых отношений, используемых в диаграммах классов.

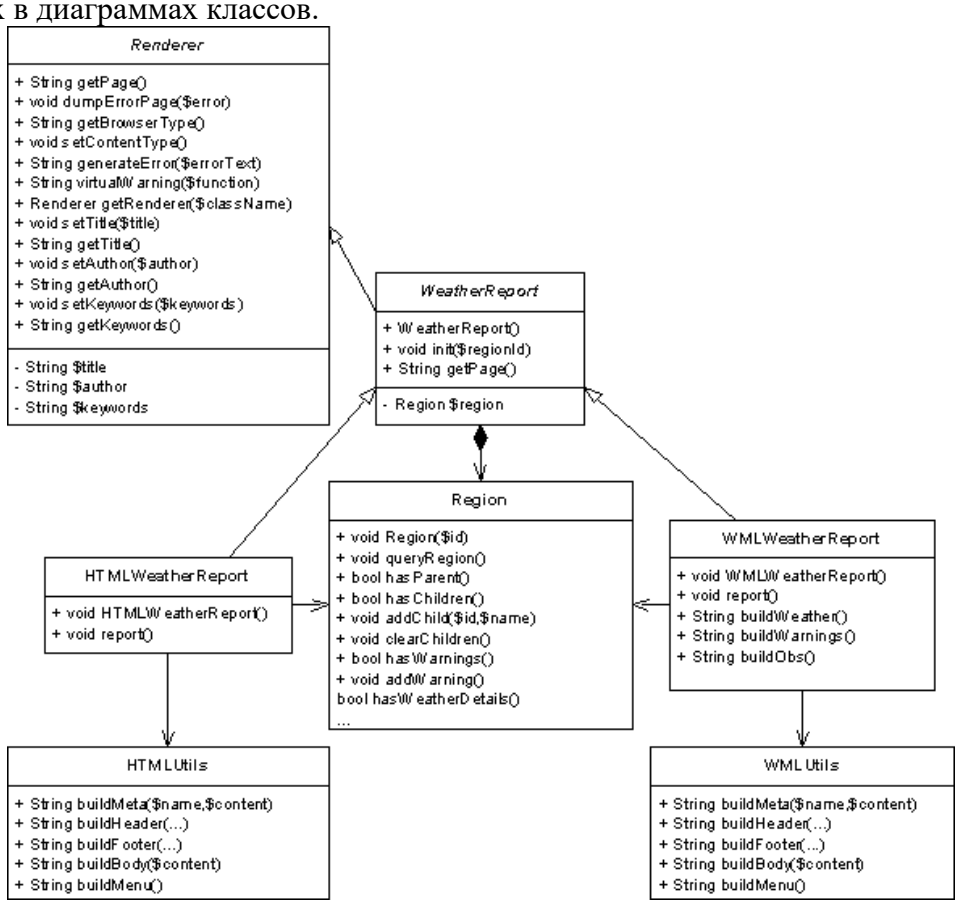


Рис. 5: Диаграмма классов

Диаграммы, подобные приведенной выше, можно построить очень быстро. Эта диаграмма, например, изображает мой сайт, о котором я говорил в предыдущей статье - Введение в WML. На ее создание ушло примерно 15 минут. Однако она сослужила хорошую службу при создании и поддержке демо-версии сайта, и позволила избежать переделки кода, которая была бы неизбежной в другом случае. Мне бы хотелось обратить ваше внимание в диаграмме на несколько ключевых моментов (пожалуй, вам предварительно стоит ознакомиться описанием приложения в предыдущей статье о WML).

- Класс Renderer является абстрактным классом - его имя выделено курсивом. Это значит, что он не будет использоваться напрямую, лишь его потомки-подклассы будут инициализированы (например Region()). Все страницы, выводящие информацию на экран, должны быть подклассами этого класса, чтобы обеспечить задачу вывода информации для любого типа броузера.
- Класс WeatherReport отвечает за создание и владение всех объектов Region. На схеме это отношение изображено с помощью черного ромба, обозначающего сильное отношение

группировки (aggregation). Это значит, что один объект владеет другими объектами и создает их.

- Знак "+" перед названиями методов обозначает, что эти методы имеют тип public, и они могут быть вызваны из других объектов или функций. Знак "-" обозначает, что переменная или метод имеют тип private, и они видимы только функциям, принадлежащим данному объекту. В PHP все методы и переменны имеют тип public, но мы должны всегда рассматривать эти переменные как private-переменные, и не должны к ним обращаться напрямую.
- Класс HTMLWeatherReport зависит от класса HTMLUtils. Отношение зависимости (dependency) обозначает, что один класс инициализирует и/или вызывает функции другого класса.
- В каждом классе диаграммы классов должны быть указаны все методы (и переменные. Если таковые имеются), их видимость (тип public, private или protected), тип значения, возвращаемого каждой функцией, аргументы-параметры этих функций, и также типы данных переменных. Функции указываются первыми, а за ними в отдельном блоке перечисляются переменные.

Если система, которую вы строите, не является объектно-ориентированной, диаграммы классов все равно могут помочь вам создать модель приложения. Классы вместо объектов просто будут обозначать различные включаемые файлы и функции. В этом случае на вашей диаграмме будут отсутствовать отношения наследования, слияния (composition), группировки (aggregation) и прочие отношения из ООП. НО тем не менее ваша диаграмма должна показывать с помощью стрелок-зависимостей, какие файлы какими файлами вызываются.

Моделирование работы системы

Иногда необходимо показать, как куски все составные части системы будут работать вместе в процессе. Диаграмма классов показывает все отношения между классами, но не показывает порядок в котором делаются вызовы и то, как результат выполнения одной функции определяет какая функция будет вызываться следующей. Для того, чтобы показать более динамические аспекты системы, язык UML предлагает ряд диаграмм различного типа. Диаграммы сценариев (Scenario diagrams) особенно полезны, если вы разрабатываете модель веб-сайта. Диаграммы сценариев делятся на два вида: диаграммы взаимодействия (collaboration diagrams) и диаграммы последовательностей (sequence diagrams). Как правило не приходится моделировать все действия, происходящие в системе. Обычно, диаграммы сценариев служат для отображения самых сложных частей системы, либо для общего схематического изображения работы кода. Например, с помощью этой диаграммы можно показать, как код проверки пользователя вставляется в определенную страницу, либо показать, как набор утилит служит в различных страницах для создания единого интерфейса сайта. Два типа диаграмм сценариев приведены ниже на рисунках.

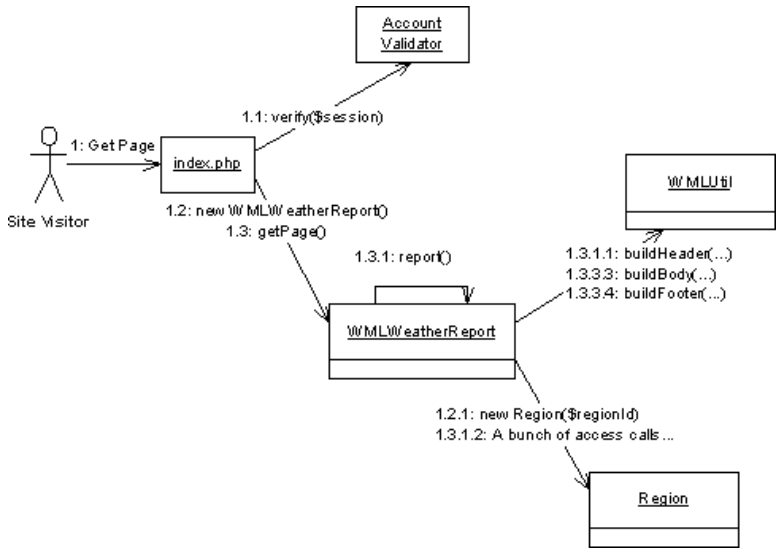


Рис. 6. Диаграмма взаимодействия.

Данная диаграмма взаимодействия показывает дизайн того, как на веб-сайте будет создаваться отчет о погоде. Реальный код, стоящий за диаграммой, можно найти в предыдущей статье - введении в WML (исключая механизм проверки пользователя). Некоторые незначительные методы на диаграмме не присутствуют потому, что я был заинтересован в моделировании ключевых шагов процесса. Вы сами можете проследить выполнение методов от "1" до "1.3.3.4". в некоторых командах принято нумеровать диаграммы как 1, 2, 3, 4, но вообще-то лучше показывать глубину вызовов используя следующую нотацию: 1, 1.1, 1.2, 2, 2.1 и так далее. эта схема нумерации показывает систему передачи вызовов более наглядно. Например, диаграмма показывает, что метод report() вызывает несколько функций в объектах WMLUtil и Region. Функция buildHeader(...) в WMLUtil вызывается до того, как мы начали создавать отчет с помощью других функций. Наконец, мы вызываем метод buildFooter(...) в модуле WMLUtil до того, как возвращаемся к методу report() и из него - к методу getPage(). В диаграммы взаимодействия можно добавлять более подробную информацию, например, тип возвращаемых данных, ограничения на данные и прочие условия.

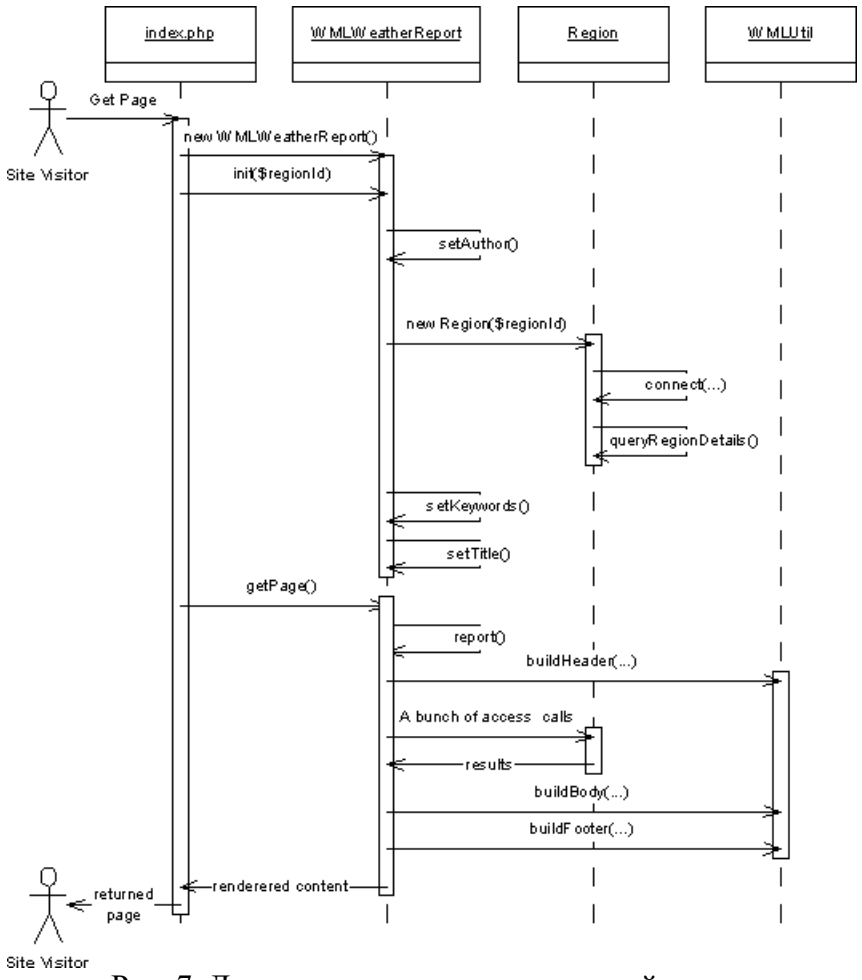


Рис. 7. Диаграмма последовательностей

Диаграмма последовательностей очень похожа на диаграмму взаимодействий по тому, какую информацию они показывают. Вообще-то, многие программы UML способны создавать диаграммы последовательностей из диаграмм взаимодействий и обратно. Главное отличие диаграммы последовательностей в том, что на этой диаграмме легче проследить последовательность действий. Кроме того, на этой диаграмме можно указать подробную информацию о времени существования того или иного объекта или о его поведении (например, время ожидания, взаимодействие параллельных нитей процесса, момент создания и уничтожения объектов).

Когда возникает вопрос, какую диаграмму строить, я использую несколько критериев, которые помогают мне выбрать наиболее подходящий вариант:

- Используйте диаграмму последовательностей, если вы хотите показать детали кода, относящиеся к временным и межпроцессорным взаимодействиям
- Используйте диаграмму взаимодействий, если вы пытаетесь показать цепь взаимодействий между объектами
- Используйте диаграмму последовательностей, если вы хотите показать одновременные взаимодействия между несколькими объектами
- Используйте диаграмму взаимодействий, если вы хотите показать большое количество взаимодействий или передачи сообщений между малым количеством объектов.

План развертывания приложения

Как было сказано в первой части данной статьи, большинство веб-сайтов не отличаются сложной архитектурой. Однако диаграммы развертывания системы все равно могут представлять для вас интерес по двум причинам: они показывают архитектуру и организацию файлов. Что касается организации файлов, то я предпочитаю работать с инструментами моделирования классов так, как описано в первой части в разделе " Спланируйте страницы ". Здесь же, я упомяну для справки диаграмму компонентов, но в зависимости от сложности сайта вы можете сами решать, нужна она или нет. Следующий ниже пример намеренно использует больше машин (узлов), чем наш простой реальный пример, который вполне уместился бы на одной машине, ну разве что в будущем его понадобилось бы промасштабировать для обслуживания более массовой аудитории.

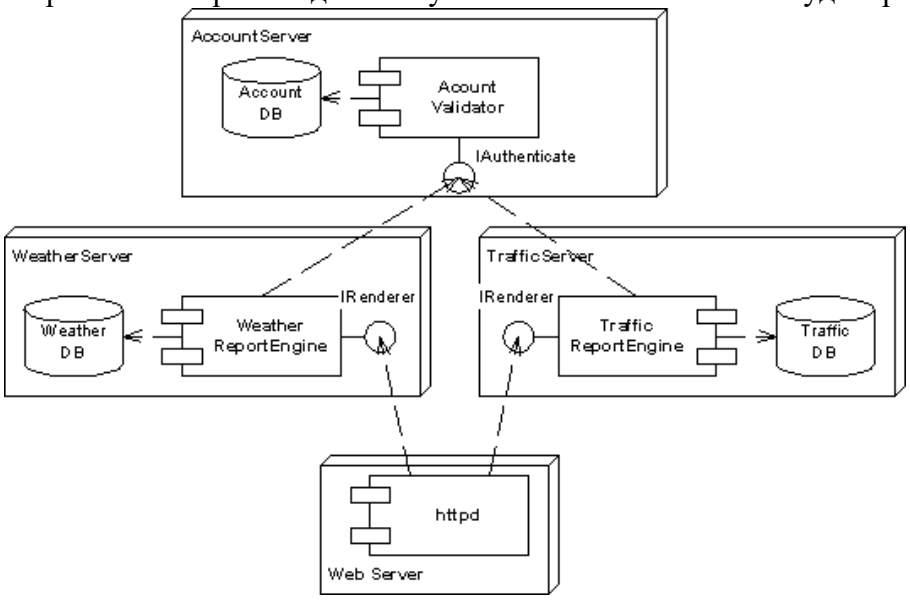


Рис.8: Диаграмма компонентов

Принципы хорошего дизайна

UML - это всего лишь инструмент. Если им правильно пользоваться, он облегчит вам работу по созданию хороших сайтов. Но ключом к созданию хорошего сайта являются принципы хорошего дизайна, и никакая нотация не обучит вас им. На эту тем есть множество книг, но я попытаюсь осветить лишь несколько самых главных вопросов.

Пишите цельный и несплетенный код

Эту фразу постоянно повторяют при описании принципов объектно-ориентированного программирования. Цельный класс объединяет в себе все поведения и всю информацию, которые имеют общее назначение или задачу. Это значит, что вам не следует будет наполнять код пользовательского интерфейса математическими алгоритмами, или например вам следует поместить все операции и информацию о пользователе в один класс UserAccount. Цельный дизайн важен по ряду причин. Он помогает сократить число зависимостей классов друг от друга, позволяет создать более интуитивный и понятный дизайн, упрощает процесс разъяснения дизайна для других разработчиков, и сокращает количество классов, с которыми разработчику приходится одновременно работать. Например, если вам придется делать изменения в механизм проверки пользователя на вашем сайте, удобнее было бы изменить только один файл, чем несколько.

Несплетенный (decoupled) дизайн означает уменьшение числа взаимодействий между классами или файлами. Такой дизайн не только проще понять и объяснить своим коллегам по команде, но его и проще поддерживать. Посмотрите на следующий пример:

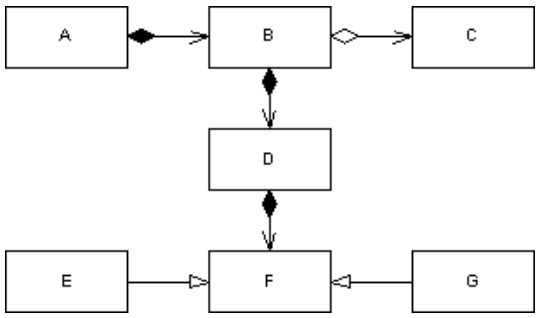


Рис. 9: Пример дизайна №1

Не зная ничего о назначении каждого из изображенных классов, не разумно давать оценку их целостности. Тем не менее, диаграмма показывает, что отношения между классами изящно сведены к минимуму. Из-за этого легче понять поведение кода. Что более важно, изменения в любом из классов минимально влияют на другие классы (например, изменения в классе D непосредственно затронут лишь класс B). Более того, что получения доступа к классу D разработчику не нужно что-то знать о классах E, F или G. А теперь посмотрите на другой пример и сравните.

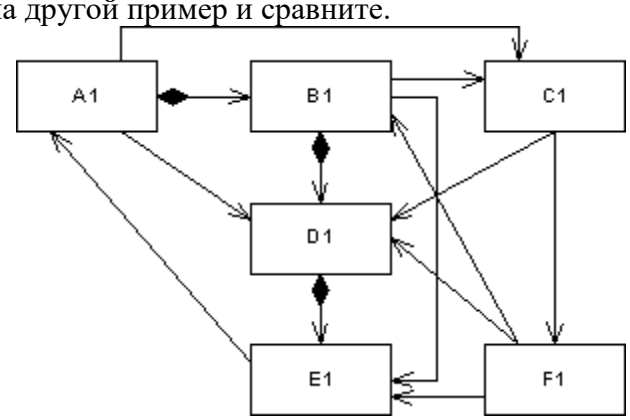


Рис.10: Пример дизайна №2

В этом пример дизайна объекты явно слишком сильно сплетены. Изменения в классе D1 потребуют длительных тестов других классов, чтобы проверить, насколько изменения сказались на них. Более того, если вы хотите выполнить какую-то функцию, вам придется гадать, где она спрятана - в самом классе D1 или в его соседях C1, E1 и/или F1.

Чтобы избежать подобного слишком сложного дизайна требуются навыки, но вот советы, которые могут вам в этом помочь:

- Ставьте себе задачу создавать цельные классы. Не распыляйте функциональность по нескольким файлам или классам.
- Называйте свои объекты ясно и понятно. Если кто-то другой не знает, что ваш класс или файл делает, или что означает ваша переменная, это значит, что вам дизайн не будет интуитивно понятным в не зависимости от того, насколько хороша его структура. Сокращения только затруднят понимание вашего дизайна.
- Не бойтесь жонглировать (refactor) кодом. Очень часто перемещение функциональности в другой файл или класс может существенно упростить код.
- Старайтесь делать классы компактными. Водопад кода лишь свидетельствует, что класс теряет целостность. Огромные

классы, модули или файлы как правило свидетельствуют об отсутствии у них четкого назначения.

- Проведите обзор вашего дизайна с кем-нибудь. Может статься, что вам подкинут новую идею, или укажут на моменты, которые были ясны для вас, но не ясны для других.
- Не позволяйте фактору эффективности кода слишком сильно давить на ваш дизайн на ранних стадиях. Чистый и хорошо проработанный код гораздо проще оптимизировать и отточить, чем запутанный код, оптимизирующий проблемы, которые уже пересмотрены и сняты.

Найдите инструменты, которые поддерживают процесс дизайна

Если вы пользуетесь UML для создания и поддержки ПО вашего веб-сайта, вот несколько инструментов, которые вам могут пригодиться. Подробное их обсуждение выходит за рамки данной статьи, но вы можете отправить мне письмо, если у вас возникнут какие-либо вопросы.

- Ручка и бумага - эти инструменты не так уж и сложны. Пару быстрых набросков в понятной форме будут гораздо более полезными, чем диаграммы, смысл которых непонятен. С помощью этих инструментов конечно не так просто описывать интерфейсы классов и создавать диаграммы сценариев, но общие планы системы вы в любом случае сможете набросать. Microsoft Visio - Visio Professional 2000 теперь поддерживает UML. Вполне неплохой инструмент, по сравнению с другими. Если вы пользуетесь версией ниже 2000, можно скачать набор UML объектов, созданных работниками компании Navision. Rational Rose - лично мне нравится этот инструмент, но для мелких разработчиков веб-сайтов он слишком дорого. Инструменты Rational Rose позволяют с легкостью управлять разрастающимся дизайном проекта, создавать отчеты по моделям, совместно (поочередно или параллельно) работать над моделью с другими пользователями.

Вот другие варианты, которыми я пользовался очень мало или вообще не пользовался, но которые могут быть отличным вариантом для вас:

MagicDraw - дешевая Java-система моделирования на UML Together - очень хорошо привязывается к C/C++ и Java, и поддерживает UML. Начать рисовать диаграммы классов можно и в бесплатном варианте программы Together WhiteBoard. Objecteering UML - распространяет UML-програмум для личного пользования бесплатно System Architect - популярная дорогая система UML-моделирования со службой поддержки на местах

Аппаратура и программное обеспечение

Аппаратура. Для выполнения лабораторной работы необходимы:

- персональный суперкомпьютер i7-5930K/ 8D8192D4-2133/ K2200/

2K40M;

- рабочие станции: монитор 29 + клавиатура + мышь. ПК MDT750/ i75930K/ 4D8192D42133";

- аппаратный сетевой обучающий комплекс мод.VDCN-1502.

Программное обеспечение. Для выполнения лабораторной работы необходима операционная система WINDOWS 7/8/10 и CASE-средство ArgoUML.

Указания по технике безопасности

При выполнении лабораторной работы **запрещается**:

1. Самостоятельно производить ремонт персонального компьютера, а так же, установку и удаление имеющегося программного обеспечения;
2. Нарушать общепринятые правила техники безопасности при работе с электрооборудованием, в частности, касаться электрических розеток металлическими предметами и т.д.;
3. Принимать пищу, напитки и сорить на рабочем месте пользователя персонального компьютера.

В случае неисправности персонального компьютера необходимо **немедленно** сообщить об этом обслуживающему персоналу лаборатории (системному администратору, оператору).

Методика и порядок выполнения работы

- Повторить теоретический материал.
- Ответить на контрольные вопросы. Ответы сохранить в текстовом файле.
- Открыть в Umbrello созданный ранее проект с названием pr04_UID, где UID - ваш номер типа 40pov40.
- Выбрать в моделируемой системе вариант использования, для которого будут строиться диаграммы взаимодействия.
- Построить для выбранного варианта использования диаграмму последовательности.
- Построить для того же варианта использования кооперативную диаграмму.
- Найти численную оценку для каждой из диаграмм.
- Добавить на диаграммы метки с комментариями, содержащими найденные численные оценки диаграмм.
- Сформулировать достоинства и недостатки каждого вида диаграмм при моделировании данного варианта использования и добавить метку с комментарием с ответом на данный вопрос.
- Прислать полученный файл с проектом и текстовый файл с ответами на контрольные вопросы преподавателю.

Содержание отчета и его форма

Отчет по лабораторной работе оформляется в виде текстового документа и должен включать:

1. Название лабораторной работы.
2. Цель лабораторной работы.
3. Формулировку индивидуального задания и результат его выполнения.
4. Краткие выводы по результатам выполнения лабораторной работы.

Контрольные вопросы

1. Каково назначение диаграмм взаимодействия?
2. Как относятся между собой диаграммы вариантов использования и диаграммы взаимодействия?
3. Назовите два вида диаграмм взаимодействия.
4. Что такое «жизненная линия» на диаграмме последовательности?
5. Как на диаграмме последовательности представляются сообщения?
6. Что такое самоделегирование?
7. Что показывает активизация объекта?
8. В чем отличие кооперативных диаграмм от диаграмм взаимодействия?
9. Каковы преимущества и недостатки каждого вида взаимодействия?
10. Как отображается условное поведение на диаграммах взаимодействия?

Защита лабораторной работы

Оформленный, в соответствии с требованиями ГОСТов, отчет представляется студентом преподавателю для проверки и последующей защиты. Защита отчета по лабораторной работе производится студентом только индивидуально.

В ходе защите лабораторной работы студент отвечает на вопросы преподавателя (поясняет методику выполнения индивидуального задания, отвечает на контрольные вопросы и т.д.).

Отчет, оформленный с отступлениями от требований ГОСТов, небрежно и неаккуратно к защите не принимается.

Лабораторная работа № 16-18. Проектирование корпоративной сети предприятия. Построение схемы сетевых устройств

Цель и содержание: Лабораторные работы №16-18 ориентированы на изучение многофункциональной программы моделирования сетей Cisco Packet Tracer.

Формируемые компетенции: ПК-2, ПК-6. ПК-7.

Cisco Packet Tracer 6.3 – Cisco Packet Tracer – это многофункциональная программа моделирования сетей, которая позволяет студентам экспериментировать с поведением сети и оценивать возможные сценарии. Являясь неотъемлемой частью комплексной среды обучения Сетевой академии, Packet Tracer предоставляет функции моделирования, визуализации, авторской разработки, аттестации и сотрудничества, а также облегчает преподавание и изучение сложных технологических принципов. Packet Tracer дополняет физическое оборудование класса, позволяя студентам создавать сети с практически неограниченным количеством устройств, поддерживая накопление практического опыта, тягу к открытиям и развитие навыков устранения неполадок. Учебная среда на основе имитационных моделей помогает студентам развивать навыки XXI века, такие как принятие решений, творческое и критическое мышление и решение задач. Packet Tracer дополняет программы Сетевых академий, что позволяет преподавателям легко описать и показать сложные технические принципы и проекты сетевых систем.

Аппаратура и программное обеспечение

Аппаратура. Для выполнения лабораторной работы необходимы:

- персональный суперкомпьютер i7-5930K/ 8D8192D4-2133/ K2200/ 2K40M;
- рабочие станции: монитор 29 + клавиатура + мышь. ПК MDT750/ i75930K/ 4D8192D42133";
- аппаратный сетевой обучающий комплекс мод.VDCN-1502.

Программное обеспечение. Для выполнения лабораторной работы необходима операционная система WINDOWS 7/8/10 и CASE-средство ArgoUML.

Указания по технике безопасности

При выполнении лабораторной работы **запрещается**:

1. Самостоятельно производить ремонт персонального компьютера, а так же, установку и удаление имеющегося программного обеспечения;
2. Нарушать общепринятые правила техники безопасности при работе с электрооборудованием, в частности, касаться электрических розеток металлическими предметами и т.д.;

3. Принимать пищу, напитки и сорить на рабочем месте пользователя персонального компьютера.

В случае неисправности персонального компьютера необходимо **немедленно** сообщить об этом обслуживающему персоналу лаборатории (системному администратору, оператору).

Методика и порядок выполнения работы

Packet Tracer. Интернет и электронная почта

Задачи

Часть 1. Настройка и проверка веб-сервисов

Часть 2. Настройка и проверка почтовых сервисов

Общие сведения

В этом упражнении, используя программу Packet Tracer, вы будете выполнять настройку веб-сервисов и электронной почты на учебном сервере. Затем необходимо выполнить настройку клиентов для доступа к веб-сервисам и электронной почте.

Примечание. Программа Packet Tracer только моделирует процесс настройки этих сервисов. Процессы установки и настройки программного обеспечения веб- и электронной почты описаны в отдельных инструкциях.

Часть 1. Настройка и проверка веб-сервисов

Шаг 1. Настройте веб-сервисы на CentralServer и BranchServer.

А. Щелкните **CentralServer** (Центральный сервер), откройте вкладку **Services** (Сервисы) и выберите раздел **HTTP**.

Б. Выберите вариант **On**, чтобы включить HTTP и HTTP Secure (HTTPS).

В. Необязательный шаг. Измените HTML-код.

Г. Повторите шаги с 1A по 1B для **BranchServer** (Сервер филиала).

Шаг 2. Проверьте работоспособность веб-серверов, открыв их веб-страницы.

В этой сети много конечных устройств, но для целей данного шага используйте компьютер **PC3**.

А. Щелкните **PC3**, откройте вкладку **Desktop** (Рабочий стол) и выберите раздел **Web Browser** (Веб-браузер).

Б. В поле URL введите IP-адрес **10.10.10.2** и нажмите кнопку **Go** (Перейти). Откроется веб-сайт **CentralServer**.

В. В поле URL введите IP-адрес **64.100.200.1** и нажмите кнопку **Go** (Перейти). Откроется веб-сайт **BranchServer**.

Г. В поле URL введите **centralserver.pt.pka** нажмите кнопку **Go** (Перейти). Откроется веб-сайт **CentralServer**.

Д. В поле URL введите **branchserver.pt.pka** нажмите кнопку **Go** (Перейти). Откроется веб-сайт **BranchServer**.

Е. Какой протокол преобразует имена **centralserver.pt.pka** и **branchserver.pt.pka** в IP-адреса?

Часть 2. Настройка и проверка почтовых сервисов на серверах

Шаг 1. Настройте CentralServer для отправки (SMTP) и получения (POP3) сообщений электронной почты.

А. Щелкните **CentralServer**, откройте вкладку **Services** (Сервисы) и нажмите кнопку **EMAIL**.

Б. Выберите вариант **On**, чтобы включить SMTP и POP3.

В. Назначьте имя домена **centralserver.pt.pka** и нажмите кнопку **Set** (Установить).

Г. Создайте пользователя с именем **central-user** и паролем **cisco**. Нажмите **+** для добавления пользователя.

Шаг 2. Настройте BranchServer для отправки (SMTP) и получения сообщений электронной почты (POP3).

А. Щелкните **BranchServer**, откройте вкладку **Services** (Сервисы) и выберите **EMAIL**.

Б. Выберите вариант **On**, чтобы включить SMTP и POP3.

В. Назначьте имя домена **branchserver.pt.pka** и нажмите кнопку **Set** (Установить).

Г. Создайте пользователя с именем **branch-user** и паролем **cisco**. Нажмите **+** для добавления пользователя.

Шаг 3. Настройте PC3 для использования сервиса электронной почты CentralServer.

А. Щелкните **PC3**, откройте вкладку **Desktop** (Рабочий стол) и выберите раздел **E Mail**.

Б. Введите следующие значения в соответствующие поля:

1) Your Name (Ваше имя): **Central User**
2) Email Address (Адрес эл. почты): **central-user@centralserver.pt.pka**
3) Incoming Mail Server (Сервер входящей почты): **10.10.10.2**
4) Outgoing Mail Server (Сервер исходящей почты): **10.10.10.2**
5) User Name (Имя пользователя): **central-user**
6) Password (Пароль): **cisco**

В. Нажмите**Save** (Сохранить). Отобразится окно почтового обозревателя.

Г. Нажмите кнопку **Receive** (Получить). Если все настройки клиента и сервера выполнены правильно, в окне почтового обозревателя появится сообщение о подтверждении Receive Mail Success (Сообщение успешно получено).

Шаг 4. Настройте Sales (Отдел продаж) для использования сервиса электронной почты BranchServer.

А. Щелкните **Sales** (Отдел продаж), откройте вкладку **Desktop** (Рабочий стол) и выберите раздел **E Mail**.

Б. Введите следующие значения в соответствующие поля:

1) Your Name (Ваше имя): **Branch User**
2) Email Address (Адрес эл. почты): **branch-user@branchserver.pt.pka**
3) Incoming Mail Server (Сервер входящей почты): **172.16.0.3**
4) Outgoing Mail Server (Сервер исходящей почты): **172.16.0.3**
5) User Name (Имя пользователя): **branch-user**
6) Password (Пароль): **cisco**

В. Нажмите**Save** (Сохранить). Отобразится окно почтового обозревателя.

Г. Нажмите кнопку **Receive** (Получить). Если все настройки клиента и сервера выполнены правильно, в окне почтового обозревателя появится сообщение о подтверждении Receive Mail Success (Сообщение успешно получено).

Д. Упражнение должно быть выполнено на 100 %. Не закрывайте окно настройки отдела продаж или окно почтового обозревателя.

Шаг 5. Отправьте сообщение электронной почты от клиента Sales и клиента PC3.

А. В окне **Sales Mail Browser** нажмите **Compose** (Создать сообщение).

Б. Введите следующие значения в соответствующие поля:

1) To (Получатель): **central-user@centralserver.pt.pka**
2) Subject (Тема): *укажите тему сообщения*.
3) **Email** Body (Текст письма): *введите текст письма*.

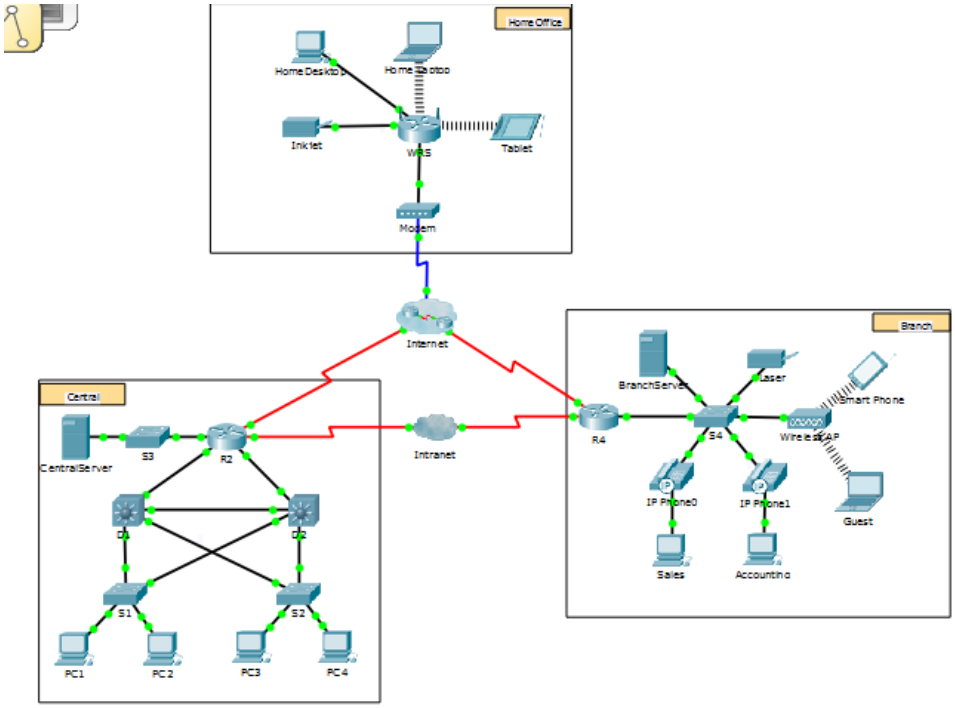
В. Нажмите **Send** (Отправить).

Г. Убедитесь, что **PC3** получил сообщение электронной почты. Щелкните **PC3**. Если окно почтового обозревателя закрыто, щелкните **E Mail**.

Д. Нажмите кнопку **Receive** (Получить). Отобразится сообщение от отдела продаж. Дважды щелкните сообщение электронной почты.

Е. Нажмите кнопку **Reply** (Ответить), введите ответ и нажмите кнопку **Send** (Отправить).

Ж. Убедитесь, что отдел продаж **Sales** получил ответ.



Содержание отчета и его форма

Отчет по лабораторной работе оформляется в виде текстового документа и должен включать:

1. Название лабораторной работы.
2. Цель лабораторной работы.
3. Формулировку индивидуального задания и результат его выполнения.
4. Краткие выводы по результатам выполнения лабораторной работы.

Защита лабораторной работы

Оформленный, в соответствии с требованиями ГОСТов, отчет представляется студентом преподавателю для проверки и последующей защиты. Защита отчета по лабораторной работе производится студентом только индивидуально.

В ходе защите лабораторной работы студент отвечает на вопросы преподавателя (поясняет методику выполнения индивидуального задания, отвечает на контрольные вопросы и т.д.).

Отчет, оформленный с отступлениями от требований ГОСТов, небрежно и неаккуратно к защите не принимается.

Список литературы
Основная литература:

1 Альбекова З.М. Корпоративные информационные системы: учебное пособие – Ставрополь: Изд-во СКФУ, 2021. – 120 с

2 Ермаков, А. Е. Основы конфигурирования корпоративных сетей Cisco : Учебное пособие / Ермаков А. Е. - Москва : Учебно-методический центр по образованию на железнодорожном транспорте, 2013. - 248 с. - Книга находится в базовой версии ЭБС IPRbooks. - ISBN 978-5-89035-677-2

3 Золотарёв, О. В. Технология внедрения корпоративных информационных систем : Методические указания к лабораторным работам / Золотарёв О. В. - Москва : Российский новый университет, 2013. - 40 с. - Книга находится в базовой версии ЭБС IPRbooks.

4 Корпоративные информационные системы управления : учебник / [Н.М. Абдикеев, Н.Б. Завьялова, А.Д. Киселев и др.] ; под ред. Н.М. Абдикеева, О.В. Китовой. - М. : ИНФРА-М, 2011. - 464 с. : ил. - (Учебники для МВА). - Библиогр. в конце глав. - ISBN 978-5-16-004373-9

5 Матяш С.А. Корпоративные информационные системы: учебное пособие [Электронный ресурс] / С.А. Матяш. – М., Берлин: Директ-Медиа, 2015. – 471. – URL: http://biblioclub.ru/index.php?page=book_red&id=435245&sr=1

Дополнительная литература:

1 Боброва, Е. И. Корпоративные библиотечно-информационные системы / Е.И. Боброва. - Кемерово : КемГУКИ, 2015. - 36 с. - ISBN 978-8154-0306-2

2 Воронцов, Ю.А. Распределённые информационные системы : учебно-методическое пособие / сост. Ю.А. Воронцов Электронный ресурс. - Распределённые информационные системы,2022-04-04 : Московский технический университет связи и информатики ; Москва, 2016. - 16 с. - Книга находится в базовой версии ЭБС IPRbooks.

Интернет-ресурсы:

1) http://www.exponenta.ru – образовательный математический сайт;

2) http://www.intuit.ru – Национальный открытый университет «ИНТУИТ»;

3) http://www.window.edu.ru – Единое окно доступа к образовательным ресурсам;

4) http:://www.netacad.com – Сетевая Академия Cisco.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к самостоятельной работе по дисциплине
«Корпоративные информационные системы»
для студентов направления
09.03.02 «Информационные системы и технологии»

Ставрополь 2025

СОДЕРЖАНИЕ

1. Самостоятельная работа как важнейшая форма учебного процесса.

Самостоятельная работа - планируемая учебная, выполняемая во внеаудиторное (аудиторное) время по заданию и при методическом руководстве преподавателя, но без его непосредственного участия (при частичном непосредственном участии преподавателя, оставляющем ведущую роль за работой студентов).

Самостоятельная работа студентов в ВУЗе является важным видом учебной и научной деятельности студента. Самостоятельная работа студентов играет значительную роль в рейтинговой технологии обучения. Государственным стандартом предусматривается, как правило, 50% часов из общей трудоемкости дисциплины на самостоятельную работу студентов (далее СРС). В связи с этим, обучение в ВУЗе включает в себя две, практически одинаковые по объему и взаимовлиянию части – процесса обучения и процесса самообучения. Поэтому СРС должна стать эффективной и целенаправленной работой студента.

Концепцией модернизации российского образования определены основные задачи профессионального образования - "подготовка квалифицированного работника соответствующего уровня и профиля, конкурентоспособного на рынке труда, компетентного, ответственного, свободно владеющего своей профессией и ориентированного в смежных областях деятельности, способного к эффективной работе по специальности на уровне мировых стандартов, готового к постоянному профессиональному росту, социальной и профессиональной мобильности".

Решение этих задач невозможно без повышения роли самостоятельной работы студентов над учебным материалом, усиления ответственности преподавателей за развитие навыков самостоятельной работы, за стимулирование профессионального роста студентов, воспитание творческой активности и инициативы.

К современному специалисту общество предъявляет достаточно

широкий перечень требований, среди которых немаловажное значение имеет наличие у выпускников определенных способностей и умения самостоятельно добывать знания из различных источников, систематизировать полученную информацию, давать оценку конкретной финансовой ситуации. Формирование такого умения происходит в течение всего периода обучения через участие студентов в практических занятиях, выполнение контрольных заданий и тестов, написание курсовых и выпускных квалификационных работ. При этом самостоятельная работа студентов играет решающую роль в ходе всего учебного процесса.

Формы самостоятельной работы студентов разнообразны. Они включают в себя:

- изучение учебной, научной и методической литературы, материалов периодических изданий с привлечением электронных средств официальной, статистической, периодической и научной информации;
- подготовку докладов и рефератов;

Самостоятельная работа приобщает студентов к научному творчеству, поиску и решению актуальных современных проблем.

2. Цели и основные задачи СРС

Ведущая цель организации и осуществления СРС должна совпадать с целью обучения студента – подготовкой бакалавра с высшим образованием. При организации СРС важным и необходимым условием становятся формирование умения самостоятельной работы для приобретения знаний, навыков и возможности организации учебной и научной деятельности.

Целью самостоятельной работы студентов является овладение фундаментальными знаниями, профессиональными умениями, опытом творческой, исследовательской деятельности. Самостоятельная работа

студентов способствует развитию самостоятельности, ответственности и организованности, творческого подхода к решению проблем учебного и профессионального уровня.

Задачами СРС являются:

- систематизация и закрепление полученных теоретических знаний и практических умений студентов;
- углубление и расширение теоретических знаний;
- формирование умений использовать справочную документацию и специальную литературу;
- развитие познавательных способностей и активности студентов: творческой инициативы, самостоятельности, ответственности и организованности;
- формирование самостоятельности мышления, способностей к саморазвитию, самосовершенствованию и самореализации;
- использование материала, собранного и полученного в ходе самостоятельных занятий на семинарах, на практических и лабораторных занятиях.

3. Виды самостоятельной работы

Основными видами самостоятельной работы студентов без участия преподавателей являются:

- формирование и усвоение учебной литературы, включая информационные образовательные ресурсы (электронные учебники, электронные библиотеки и др.);
- написание рефератов;
- подготовка к семинарам и лабораторным работам, их оформление;

- составление аннотированного списка статей из соответствующих журналов по отраслям знаний;
- выполнение домашних заданий;

(В зависимости от особенностей факультета перечисленные виды работ могут быть расширены, заменены на специфические).

Основными видами самостоятельной работы студентов с участием преподавателей являются:

- текущие консультации;
- прием и разбор домашних заданий (в часы практических занятий);
- прием и защита лабораторных работ (во время проведения л/р);
- прохождение и оформление результатов практик (руководство и оценка уровня сформированности профессиональных умений и навыков);
- выполнение выпускной квалификационной работы (руководство, консультирование и защита выпускных квалификационных работ) и др.

4. Организация СРС

Методика организации самостоятельной работы студентов зависит от структуры, характера и особенностей изучаемой дисциплины, объема часов на ее изучение, вида заданий для самостоятельной работы студентов, индивидуальных качеств студентов и условий учебной деятельности.

Процесс организации самостоятельной работы студентов включает в себя следующие этапы:

- подготовительный (определение целей, составление программы, подготовка методического обеспечения, подготовка оборудования);
- основной (реализация программы, использование приемов поиска информации, усвоения, переработки, применения, передачи знаний, фиксирование результатов, самоорганизация процесса работы);
- заключительный (оценка значимости и анализ результатов, их систематизация, оценка эффективности программы и приемов работы, выводы о направлениях оптимизации труда).

Организацию самостоятельной работы студентов обеспечивают: факультет, кафедра, учебный и методический отделы, преподаватель, библиотека, ТСО, ИВТ, издательство и др.

Деятельность студентов по формированию и развитию навыков учебной самостоятельной работы.

В процессе самостоятельной работы студент приобретает навыки самоорганизации, самоконтроля, самоуправления, саморефлексии и становится активным самостоятельным субъектом учебной деятельности.

Выполняя самостоятельную работу под контролем преподавателя студент должен:

−освоить минимум содержания, выносимый на самостоятельную работу студентов и предложенный преподавателем в соответствии с Государственными образовательными стандартами высшего профессионального образования (ГОС ВПО/ГОС СПО) по данной дисциплине.

−планировать самостоятельную работу в соответствии с графиком самостоятельной работы, предложенным преподавателем.

– самостоятельную работу студент должен осуществлять в организационных формах, предусмотренных учебным планом и рабочей программой преподавателя.

– выполнять самостоятельную работу и отчитываться по ее результатам в соответствии с графиком представления результатов, видами и сроками отчетности по самостоятельной работе студентов.

студент может:

сверх предложенного преподавателем (при обосновании и согласовании с ним) и минимума обязательного содержания, определяемого ГОС ВПО/ГОС СПО по данной дисциплине:

– самостоятельно определять уровень (глубину) проработки содержания материала;

– предлагать дополнительные темы и вопросы для самостоятельной проработки;

– в рамках общего графика выполнения самостоятельной работы предлагать обоснованный индивидуальный график выполнения и отчетности по результатам самостоятельной работы;

– предлагать свои варианты организационных форм самостоятельной работы;

– использовать для самостоятельной работы методические пособия, учебные пособия, разработки сверх предложенного преподавателем перечня;

– использовать не только контроль, но и самоконтроль результатов самостоятельной работы в соответствии с методами самоконтроля, предложенными преподавателем или выбранными самостоятельно.

Самостоятельная работа студентов должна оказывать важное влияние на формирование личности будущего специалиста, она планируется студентом самостоятельно. Каждый студент самостоятельно определяет режим своей работы и меру труда, затрачиваемого на овладение учебным содержанием по каждой дисциплине. Он выполняет внеаудиторную работу

по личному индивидуальному плану, в зависимости от его подготовки, времени и других условий.

5. Общие рекомендации по организации самостоятельной работы

Основной формой самостоятельной работы студента является изучение рекомендованной литературы, активное участие на практических и семинарских занятиях. Но для успешной учебной деятельности, ее интенсификации, необходимо учитывать следующие субъективные факторы:

1. Знание школьного программного материала, наличие прочной системы зияний, необходимой для усвоения основных вузовских курсов. Необходимо отличать пробелы в знаниях, затрудняющие усвоение нового материала, от малых способностей. Затратив силы на преодоление этих пробелов, студент обеспечит себе нормальную успеваемость и поверит в свои способности.

2. Наличие умений, навыков умственного труда:

а) при работе с дополнительной литературой;

3. Специфика познавательных психических процессов: внимание, память, речь, наблюдательность, интеллект и мышление. Слабое развитие каждого из них становится серьезным препятствием в учебе.

4. Хорошая работоспособность, которая обеспечивается нормальным физическим состоянием. Ведь серьезное учение - это большой многосторонний и разнообразный труд. Результат обучения оценивается не количеством сообщаемой информации, а качеством ее усвоения, умением ее использовать и развитием у себя способности к дальнейшему самостоятельному образованию.

5. Соответствие избранной деятельности, профессии индивидуальным способностям. Необходимо выработать у себя умение саморегулировать свое эмоциональное состояние и устранять обстоятельства, нарушающие деловой настрой, мешающие намеченной работе.

6. Овладение оптимальным стилем работы, обеспечивающим успех в деятельности. Чередование труда и пауз в работе, периоды отдыха, индивидуально обоснованная норма продолжительности сна, предпочтение вечерних или утренних занятий, стрессоустойчивость на экзаменах и особенности подготовки к ним,

7. Уровень требований к себе, определяемый сложившейся самооценкой.

Адекватная оценка знаний, достоинств, недостатков - важная составляющая самоорганизации человека, без нее невозможна успешная работа по управлению своим поведением, деятельностью.

Одна из основных особенностей обучения в высшей школе заключается в том, что постоянный внешний контроль заменяется самоконтролем, активная роль в обучении принадлежит уже не столько преподавателю, сколько студенту.

Зная основные методы научной организации умственного труда, можно при наименьших затратах времени, средств и трудовых усилий достичь наилучших результатов.

Эффективность усвоения поступающей информации зависит от работоспособности человека в тот или иной момент его деятельности.

Работоспособность - способность человека к труду с высокой степенью напряженности в течение определенного времени. Различают внутренние и внешние факторы работоспособности.

К внутренним факторам работоспособности относятся интеллектуальные особенности, воля, состояние здоровья.

К внешним:

- организация рабочего места, режим труда и отдыха;
- уровень организации труда - умение получить справку и пользоваться информацией;
- величина умственной нагрузки.

Выдающийся русский физиолог Н. Е. Введенский выделил следующие условия продуктивности умственной деятельности:

- во всякий труд нужно входить постепенно;
- мерность и ритм работы. Разным людям присущ более или менее разный темп работы;
- привычная последовательность и систематичность деятельности;
- правильное чередование труда и отдыха.

7. Методические рекомендации для студентов по отдельным формам самостоятельной работы.

С первых же сентябрьских дней на студента обрушивается громадный объем информации, которую необходимо усвоить. Нужный материал содержится не только в лекциях (запомнить его – это только малая часть задачи), но и в учебниках, книгах, статьях. Порой возникает необходимость привлекать информационные ресурсы Интернет.

Система вузовского обучения подразумевает значительно большую самостоятельность студентов в планировании и организации своей деятельности. Вчерашнему школьнику сделать это бывает весьма непросто: если в школе ежедневный контроль со стороны учителя заставлял постоянно и систематически готовиться к занятиям, то в вузе вопрос об уровне знаний вплотную встает перед студентом только в период сессии. Такая ситуация оборачивается для некоторых соблазном весь семестр посвятить свободному времяпрепровождению («когда будет нужно – выучу!»), а когда приходит пора экзаменов, материала, подлежащего усвоению, оказывается так много, что никакая память не способна с ним справиться в оставшийся промежуток времени.

При работе с книгой необходимо подобрать литературу, научиться правильно ее читать, вести записи. Для подбора литературы в библиотеке используются алфавитный и систематический каталоги.

Важно помнить, что рациональные навыки работы с книгой - это всегда

большая экономия времени и сил.

Правильный подбор учебников рекомендуется преподавателем, читающим лекционный курс. Необходимая литература может быть также указана в методических разработках по данному курсу.

Изучая материал по учебнику, следует переходить к следующему вопросу только после правильного уяснения предыдущего, описывая на бумаге все выкладки и вычисления (в том числе те, которые в учебнике опущены или на лекции даны для самостоятельного вывода).

При изучении любой дисциплины большую и важную роль играет самостоятельная индивидуальная работа.

Особое внимание следует обратить на определение основных понятий курса. Студент должен подробно разбирать примеры, которые поясняют такие определения, и уметь строить аналогичные примеры самостоятельно. Нужно добиваться точного представления о том, что изучаешь. Полезно составлять опорные конспекты. При изучении материала по учебнику полезно в тетради (на специально отведенных полях) дополнять конспект лекций. Там же следует отмечать вопросы, выделенные студентом для консультации с преподавателем.

Выводы, полученные в результате изучения, рекомендуется в конспекте выделять, чтобы они при перечитывании записей лучше запоминались.

Опыт показывает, что многим студентам помогает составление листа опорных сигналов, содержащего важнейшие и наиболее часто употребляемые формулы и понятия. Такой лист помогает запомнить формулы, основные положения лекции, а также может служить постоянным справочником для студента.

Различают два вида чтения; первичное и вторичное. *Первичное* - эти внимательное, неторопливое чтение, при котором можно остановиться на трудных местах. После него не должно остаться ни одного непонятного олова. Содержание не всегда может быть понятно после первичного чтения.

Задача *вторичного* чтения полное усвоение смысла целого (по счету это чтение может быть и не вторым, а третьим или четвертым).

Правила самостоятельной работы с литературой. Как уже отмечалось, самостоятельная работа с учебниками и книгами (а также самостоятельное теоретическое исследование проблем, обозначенных преподавателем на лекциях) – это важнейшее условие формирования у себя научного способа познания. Основные советы здесь можно свести к следующим:

• Составить перечень книг, с которыми Вам следует познакомиться; «не старайтесь запомнить все, что вам в ближайшее время не понадобится, – советует студенту и молодому ученому Г. Селье, – запомните только, где это можно отыскать» (Селье, 1987. С. 325).

• Сам такой перечень должен быть систематизированным (что необходимо для семинаров, что для экзаменов, что пригодится для написания курсовых и дипломных работ, а что Вас интересует за рамками официальной учебной деятельности, то есть что может расширить Вашу общую культуру...).

• Обязательно выписывать все выходные данные по каждой книге (при написании курсовых и дипломных работ это позволит очень сэкономить время).

• Разобраться для себя, какие книги (или какие главы книг) следует прочитать более внимательно, а какие – просто просмотреть.

• При составлении перечней литературы следует посоветоваться с преподавателями и научными руководителями (или даже с более подготовленными и эрудированными сокурсниками), которые помогут Вам лучше сориентироваться, на что стоит обратить большее внимание, а на что вообще не стоит тратить время...

•Естественно, все прочитанные книги, учебники и статьи следует конспектировать, но это не означает, что надо конспектировать «все подряд»: можно выписывать кратко основные идеи автора и иногда приводить

наиболее яркие и показательные цитаты (с указанием страниц).

• Если книга – Ваша собственная, то допускается делать на полях книги краткие пометки или же в конце книги, на пустых страницах просто сделать свой «предметный указатель», где отмечаются наиболее интересные для Вас мысли и обязательно указываются страницы в тексте автора (это очень хороший совет, позволяющий экономить время и быстро находить «избранные» места в самых разных книгах).

• Если Вы раньше мало работали с научной литературой, то следует выработать в себе способность «воспринимать» сложные тексты; для этого лучший прием – научиться «читать медленно», когда Вам понятно каждое прочитанное слово (а если слово незнакомое, то либо с помощью словаря, либо с помощью преподавателя обязательно его узнать), и это может занять немалое время (у кого-то – до нескольких недель и даже месяцев); опыт показывает, что после этого студент каким-то «чудом» начинает буквально заглатывать книги и чуть ли не видеть «сквозь обложку», стоящая это работа или нет...

• «Либо читайте, либо перелистывайте материал, но не пытайтесь читать быстро... Если текст меня интересует, то чтение, размышление и даже фантазирование по этому поводу сливаются в единый процесс, в то время как вынужденное скорочтение не только не способствует качеству чтения, но и не приносит чувства удовлетворения, которое мы получаем, размышляя о прочитанном», – советует Г. Селье (Селье, 1987. – С. 325-326).

• Есть еще один эффективный способ оптимизировать знакомство с научной литературой – следует увлечься какой-то идеей и все книги просматривать с точки зрения данной идеи. В этом случае студент (или молодой ученый) будет как бы искать аргументы «за» или «против» интересующей его идеи, и одновременно он будет как бы общаться с авторами этих книг по поводу своих идей и размышлений.

Из всех рассмотренных видов чтения основным для студентов является изучающее – именно оно позволяет в работе с учебной литературой

накапливать знания в различных областях. Вот почему именно этот вид чтения в рамках учебной деятельности должен быть освоен в первую очередь. Кроме того, при овладении данным видом чтения формируются основные приемы, повышающие эффективность работы с научным текстом.

Основные виды систематизированной записи прочитанного:

1. Аннотирование – предельно краткое связное описание просмотренной или прочитанной книги (статьи), ее содержания, источников, характера и назначения;
2. Планирование – краткая логическая организация текста, раскрывающая содержание и структуру изучаемого материала;
3. Тезирование – лаконичное воспроизведение основных утверждений автора без привлечения фактического материала;
4. Цитирование – дословное выписывание из текста выдержек, извлечений, наиболее существенно отражающих ту или иную мысль автора;
5. Конспектирование – краткое и последовательное изложение содержания прочитанного.

Конспект – сложный способ изложения содержания книги или статьи в логической последовательности. Конспект аккумулирует в себе предыдущие виды записи, позволяет всесторонне охватить содержание книги, статьи. Поэтому умение составлять план, тезисы, делать выписки и другие записи определяет и технологию составления конспекта.

Методические рекомендации по составлению конспекта:

1. Внимательно прочитайте текст. Уточните в справочной литературе непонятные слова. При записи не забудьте вынести справочные данные на поля конспекта;
2. Выделите главное, составьте план;
3. Кратко сформулируйте основные положения текста, отметьте аргументацию автора;

4. Законспектируйте материал, четко следуя пунктам плана. При конспектировании старайтесь выразить мысль своими словами. Записи следует вести четко, ясно.

5. Грамотно записывайте цитаты. Цитируя, учитывайте лаконичность, значимость мысли.

В тексте конспекта желательно приводить не только тезисные положения, но и их доказательства. При оформлении конспекта необходимо стремиться к емкости каждого предложения. Мысли автора книги следует излагать кратко, заботясь о стиле и выразительности написанного. Число дополнительных элементов конспекта должно быть логически обоснованным, записи должны распределяться в определенной последовательности, отвечающей логической структуре произведения. Для уточнения и дополнения необходимо оставлять поля.

Овладение навыками конспектирования требует от студента целеустремленности, повседневной самостоятельной работы.

Практические занятия.

Для того чтобы практические занятия приносили максимальную пользу, необходимо помнить, что упражнение и решение задач проводятся по вычитанному материалу из дополнительной литературы и связаны, как правило, с детальным разбором отдельных вопросов на лабораторных и практических занятиях.

При самостоятельном решении задач нужно обосновывать каждый этап решения, исходя из теоретических положений курса. Если студент видит несколько путей решения проблемы (задачи), то нужно сравнить их и выбрать самый рациональный. Полезно до начала вычислений составить краткий план решения проблемы (задачи). Решение проблемных задач или примеров следует излагать подробно, вычисления располагать в строгом порядке, отделяя вспомогательные вычисления от основных.

Консультации

Если в процессе самостоятельной работы над изучением теоретического материала или при решении задач у студента возникают вопросы, разрешить которые самостоятельно не удается, необходимо обратиться к преподавателю для получения у него разъяснений или указаний. В своих вопросах студент должен четко выразить, в чем он испытывает затруднения, характер этого затруднения. За консультацией следует обращаться и в случае, если возникнут сомнения в правильности ответов на вопросы самопроверки.

Подготовка к экзаменам и зачетам.

Изучение многих общепрофессиональных и специальных дисциплин завершается экзаменом. Подготовка к экзамену способствует закреплению, углублению и обобщению знаний, получаемых, в процессе обучения, а также применению их к решению практических задач. Готовясь к экзамену, студент ликвидирует имеющиеся пробелы в знаниях, углубляет, систематизирует и упорядочивает свои знания. На экзамене студент демонстрирует то, что он приобрел в процессе обучения по конкретной учебной дисциплине.

Вначале следует просмотреть весь материал по сдаваемой дисциплине, отметить для себя трудные вопросы. Обязательно в них разобраться.

Систематическая подготовка к занятиям в течение семестра позволит использовать время экзаменационной сессии для систематизации знаний.

Правила подготовки к зачетам и экзаменам:

• Лучше сразу сориентироваться во всем материале и обязательно расположить весь материал согласно экзаменационным вопросам (или вопросам, обсуждаемым на семинарах), эта работа может занять много времени, но все остальное – это уже технические детали (главное – это ориентировка в материале!).

• Сама подготовка связана не только с «запоминанием». Подготовка также предполагает и переосмысление материала, и даже рассмотрение альтернативных идей.

• Сначала студент должен продемонстрировать, что он «усвоил» все, что требуется по программе обучения (или по программе данного преподавателя), и лишь после этого он вправе высказать иные, желательно аргументированные точки зрения.