

Федеральное агентство по образованию
Пермский государственный технический университет
Кафедра Информационных технологий
и автоматизированных систем

Файзрахманов Р. А., Селезнев К. А.

**СТРУКТУРНО ФУНКЦИОНАЛЬНЫЙ ПОДХОД К
ПРОЕКТИРОВАНИЮ ИНФОРМАЦИОННЫХ
ТЕХНОЛОГИЙ И АВТОМАТИЗИРОВАННЫХ СИСТЕМ С
ИСПОЛЬЗОВАНИЕМ CASE-СРЕДСТВ**

Пермь, 2005

УДК

Рецензенты:

д.э.н., профессор *Н.И. Артемов*
(директор Государственного научно-исследовательского института
управляющих машин и систем ГосНИИУМС)
д.э.н., профессор *А.Н. Румянцев*
(Пермский Государственный Университет, кафедра экономической кибер-
нетики)

Файзрахманов Р. А., Селезнев К. А.

Учебное пособие к практическим занятиям «Структурно функциональный
подход к проектированию информационных технологий и автоматизиро-
ванных систем с использованием CASE-средств» / Перм. гос. техн. ун-т. –
Пермь, 2005. – 245 с.

Предназначено для студентов, обучающихся по направлению 230100 «Ин-
форматика и вычислительная техника».

УДК

© Пермский государственный
технический университет, 2005

СОДЕРЖАНИЕ

| | |
|--|-----|
| Содержание | 3 |
| Введение | 4 |
| 1. Инструментальные средства компании Computer Associations | 5 |
| 2. Инструментальная среда BPwin | 9 |
| Лабораторная работа №1. Создание контекстной диаграммы | 9 |
| Лабораторная работа №2. Диаграммы декомпозиции | 21 |
| Лабораторная работа №3. Тоннелирование стрелок | 29 |
| Лабораторная работа №4. Вспомогательные диаграммы | 35 |
| Лабораторная работа №5. Коллективная работа над проектом | 41 |
| Лабораторная работа №6. Методология IDEF3 | 48 |
| Лабораторная работа №7. Стоимостной анализ | 58 |
| Лабораторная работа №8. Реинжиниринг процессов | 70 |
| Лабораторная работа №9. Методология DFD | 79 |
| 3. Инструментальная среда ERwin | 86 |
| Лабораторная работа №1. Построение модели | 86 |
| Лабораторная работа №2. Хранимые изображения | 101 |
| Лабораторная работа №3. Отношения | 108 |
| Лабораторная работа №4. Индексация базы данных | 125 |
| Лабораторная работа №5. Прямое и обратное проектирование | 139 |
| Лабораторная работа №6. Проектные слои | 151 |
| Лабораторная работа №7. Отчеты и сообщения | 166 |
| Лабораторная работа №8. Работа с доменами | 175 |
| Лабораторная работа №9. Триггеры | 185 |
| Лабораторная работа №10. Хранимые процедуры | 206 |
| Лабораторная работа №11. Связывание моделей | 214 |
| 4. Порядок отчетности по лабораторным работам | 223 |
| 5. Задание на индивидуальную работу | 224 |
| Приложение А. Модель предприятия: "Отдел продаж сотовых телефонов" | 225 |
| Список литературы | 245 |

ВВЕДЕНИЕ

Технология создания информационных систем (далее – ИС) предъявляет особые требования к методикам реализации и программным инструментальным средствам, а именно:

1. Реализация проектов по созданию ИС состоит из стадии анализа (прежде чем создавать ИС, необходимо понять и описать бизнес-логику предметной области), проектирования (необходимо определить модули и архитектуру будущей системы), непосредственного кодирования, тестирования и сопровождения. Известно, что исправление ошибок, допущенных на предыдущей стадии, обходится примерно в 10 раз дороже, чем на текущей, откуда следует, что наиболее критическими являются первые стадии проекта. Поэтому крайне важно иметь эффективные средства автоматизации ранних этапов реализации проекта.

2. Проект по созданию сложной ИС невозможно реализовать в одиночку. Коллективная работа существенно отличается от индивидуальной, поэтому при реализации крупных проектов необходимо иметь средства координации и управления коллективом разработчиков.

3. Жизненный цикл создания сложной ИС сопоставим с ожидаемым временем ее эксплуатации. Другими словами, в современных условиях компании перестраивают свои бизнес-процессы примерно раз в два года, столько же требуется (если работать в традиционной технологии) для создания ИС. Может оказаться, что к моменту сдачи ИС она уже никому не нужна, поскольку компания, ее заказавшая, вынуждена перейти на новую технологию работы. Следовательно, для создания ИС жизненно необходим инструмент, значительно (в несколько раз) уменьшающий время разработки ИС.

4. Вследствие значительного жизненного цикла может оказаться, что в процессе создания системы внешние условия изменились. Обычно внесение изменений в проект на поздних этапах создания ИС – весьма трудоемкий и дорогостоящий процесс. Поэтому для успешной реализации крупного проекта необходимо, чтобы инструментальные средства, на которых он реализуется, были достаточно гибкими к изменяющимся требованиям.

На современном рынке средств разработки ИС достаточно много систем, в той или иной степени удовлетворяющих перечисленным требованиям. В настоящем пособии рассматриваются современные технологии и инструментальные средства компании *Computer Associations: BPWin4.0* и *ERWin4.0* для проектирования ИС.

1. ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА КОМПАНИИ COMPUTER ASSOCIATIONS

Схема взаимодействия существующих инструментальных средств показана на рис. 1.1.

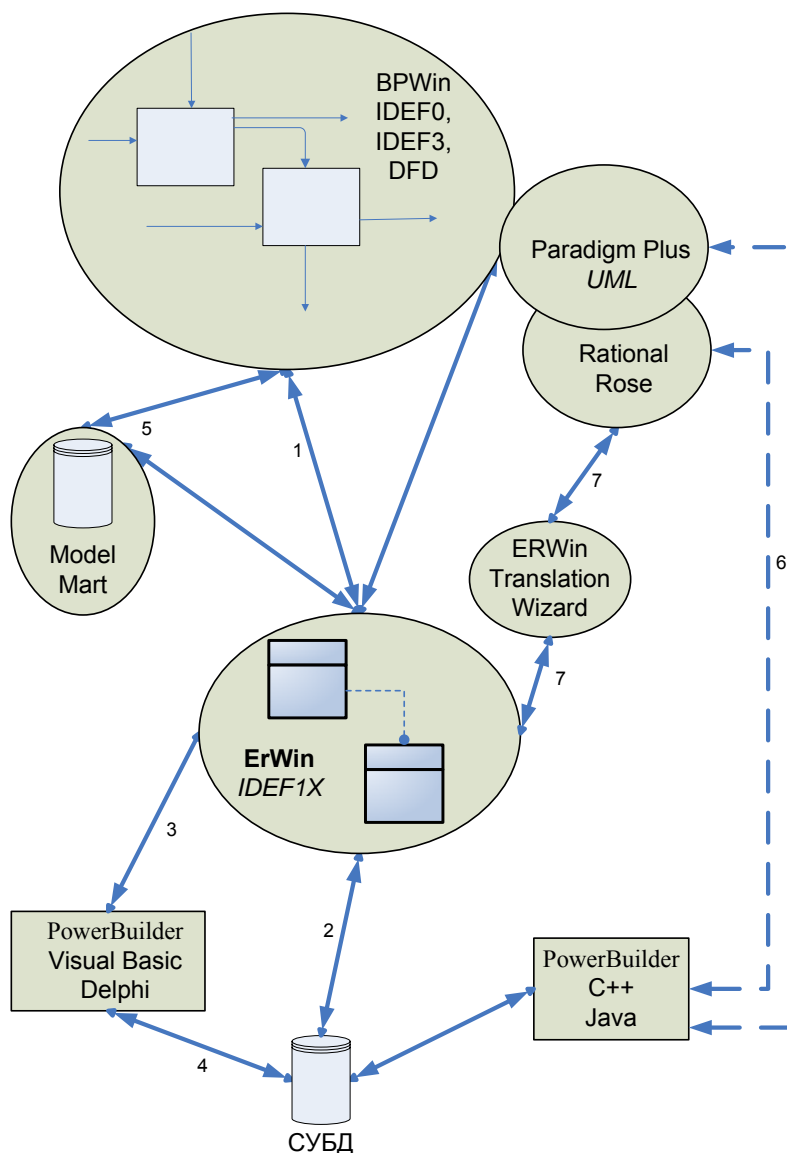


Рис. 1.1. Общая схема взаимодействия инструментальных средств

Одна из технологий разработки основана на решениях фирмы *Computer Associations* (<http://ca.com>). Рассматриваемые CASE-средства **ERwin** и **BPwin** были разработаны фирмой *Logic Works*, которая сейчас входит в компанию *Computer Associations*.

Для проведения анализа и реорганизации бизнес-процессов *Computer Associations* предлагает CASE-средство верхнего уровня **BPwin 4.0** поддерживающее методологии *IDEFO* (функциональная модель), *IDEF3* (*WorkFlow Diagram*) и *DFD* (*Dataflow Diagramm*). Функциональная модель

предназначена для описания существующих бизнес-процессов на предприятии (так называемая модель *AS-IS*) и идеального положения вещей – того, к чему нужно стремиться (модель *TO-BE*). Методология *IDEFO* предписывает построение иерархической системы диаграмм – единичных описаний фрагментов системы. Сначала проводится описание системы в целом и ее взаимодействия с окружающим миром (контекстная диаграмма), после чего проводится функциональная декомпозиция – система разбивается на подсистемы и каждая подсистема описывается отдельно (диаграммы декомпозиции). Затем каждая подсистема разбивается на более мелкие и так далее до достижения нужной степени подробности. После каждого сеанса декомпозиции проводится сеанс экспертизы: каждая диаграмма проверяется экспертами предметной области, представителями заказчика, людьми, непосредственно участвующими в бизнес-процессе. Такая технология создания модели позволяет построить модель, адекватную предметной области на всех уровнях абстрагирования. Если в процессе моделирования нужно осветить специфические стороны технологии предприятия, ***BPwin*** позволяет переключиться на любой ветви модели на нотацию *IDEF3* или *DFD* и создать смешанную модель. Нотация *DFD* включает такие понятия, как внешняя ссылка и хранилище данных, что делает ее более удобной (по сравнению с *IDEFO*) для моделирования документооборота. Методология *IDEF3* включает элемент "перекресток", что позволяет описать логику взаимодействия компонентов системы.

После построения функциональной модели можно построить модель данных. Для построения модели данных *Computer Associations* предлагает мощный и удобный инструмент – ***ERwin***. Хотя процесс преобразования модели ***BPwin*** в модель данных, плохо формализуется и поэтому полностью не автоматизирован, *Computer Associations* предлагает удобный инструмент для облегчения построения модели данных на основе функциональной модели – механизм двунаправленной связи ***BPwin*** – ***ERwin*** (стрелка 1 рис. 1.1). ***ERwin*** имеет два уровня представления модели – логический и физический. На логическом уровне данные не связаны с конкретной СУБД, поэтому могут быть наглядно представлены даже для неспециалистов. Физический уровень данных – это по существу отображение системного каталога, который зависит от конкретной реализации СУБД. ***ERwin*** позволяет проводить процессы прямого и обратного проектирования БД (стрелка 2 рис. 1.1). Это означает, что по модели данных можно сгенерировать схему БД или автоматически создать модель данных на основе информации системного каталога. Кроме того, ***ERwin*** позволяет выравнивать модель и содержимое системного каталога после редактирования того либо другого. ***ERwin*** интегрируется с популярными средствами разработки клиентской части - *PowerBuilder*, *Visual Basic*, *Delphi* (стрелка 3 рис. 1.1), что позволяет автоматически генерировать код приложения, который полностью готов к компиляции и выполнению (стрелка 4 рис. 1.1). Для разных сред разработки реализована различная техника кодогенера-

ции. Код для *PowerBuilder* генерируется непосредственно в среде **ERwin**, код для *Visual Basic* – с помощью *add-in* компонентов и библиотек, подключаемых в проект *Visual Basic*. **ERwin** не поддерживает непосредственно кодогенерацию для *Delphi*. Код клиентского приложения для *Delphi* на основе модели данных **ERwin** можно сгенерировать с помощью *MetaBASE* – продукта фирмы *gs-soft* (<http://www.gs-soft.com>).

Создание современных ИС, основанных на широком использовании распределенных вычислений, объединении традиционных и новейших информационных технологий, требует тесного взаимодействия всех участников проекта: менеджеров, бизнес-аналитиков и системных аналитиков, администраторов БД, разработчиков. Для этого использующиеся на разных этапах и разными специалистами средства моделирования и разработки должны быть объединены общей системой организации совместной работы. Фирма *Computer Associations* предлагает систему *Model Mart* – хранилище моделей, к которому открыт доступ для участников проекта создания ИС (стрелка 5 рис. 1.1). *Model Mart* удовлетворяет всем требованиям, предъявляемым к средствам разработки крупных ИС, а именно:

1. Совместное моделирование. Каждый участник проекта имеет инструмент поиска и доступа к интересующей его модели в любое время. При совместной работе используются три режима: незащищенный, защищенный и режим просмотра. В режиме просмотра запрещается любое изменение моделей. В защищенном режиме модель, с которой работает один пользователь, не может быть изменена другими пользователями. В незащищенном режиме пользователи могут работать с общими моделями в реальном масштабе времени. Возникающие при этом конфликты разрешаются при помощи специального модуля – *Intelligent Conflict Resolution (ICR)*. В дополнение к стандартным средствам организации совместной работы *Model Mart* позволяет сохранять множество версий, снабженных аннотациями, с последующим сравнением предыдущих и новых версий. При необходимости возможен возврат к предыдущим версиям.

2. Создание библиотек решений. *Model Mart* позволяет формировать библиотеки стандартных решений, включающие наиболее удачные фрагменты реализованных проектов, накапливать и использовать типовые модели, объединяя их при необходимости "сборки" больших систем. На основе существующих БД с помощью **ERwin** возможно восстановление моделей (обратное проектирование), которые в процессе анализа пригодности их для новой системы могут объединяться с типовыми моделями из библиотек моделей.

3. Управление доступом. Для каждого участника проекта определяются права доступа, в соответствии с которыми они получают возможность работать только с определенными моделями. Права доступа могут быть определены как для групп, так и для отдельных участников проекта. Роль специалистов, участвующих в различных проектах, может меняться, поэтому в *Model Mart* можно определять права доступа и управ-

лять правами доступа участников проекта к библиотекам, моделям и даже к специфическим областям модели.

4. Архитектура. Архитектура *Model Mart* реализована на архитектуре клиент – сервер. В качестве платформы реализации хранилища выбраны РСУБД *Sybase*, *Microsoft SQL Server*, *Informix* и *Oracle*. Клиентскими приложениями являются **ERwin** и **BPwin**. В *Model Mart* реализован доступ к хранилищу моделей через *API*, что позволяет постоянно наращивать возможности интегрированной среды путем включения новых инструментов моделирования и анализа.

Как было указано выше, при разработке крупных проектов критичным становится время реализации проекта. Одним из решений проблемы может стать автоматическая генерация кода приложения (клиентской части) *CASE*-средствами на основе модели предметной области. Хотя **ERwin** решает эту задачу, код генерируется на основе модели *IDEFIX*, т. е. фактически на основе реляционной модели данных, которая непосредственно не содержит информации о бизнес-процессах. Как следствие этого сгенерированный код не может полностью обеспечить функциональность приложения со сложной бизнес-логикой. Объектно-ориентированное проектирование – альтернативная технология кодогенерации, которая лишена этого недостатка.

2. ИНСТРУМЕНТАЛЬНАЯ СРЕДА BPWIN

Лабораторная работа №1. Создание контекстной диаграммы

Цель работы: Создать контекстную диаграмму в среде *BPwin*.

Теоретические сведения

BPwin имеет достаточно простой и понятный интерфейс пользователя, дающий возможность аналитику создавать сложные модели при минимальных усилиях.

При запуске *BPwin* по умолчанию появляется основная панель инструментов, палитра инструментов (вид которой зависит от выбранной нотации) и, в левой части, навигатор модели – *Model Explorer* (рис. 2.1).

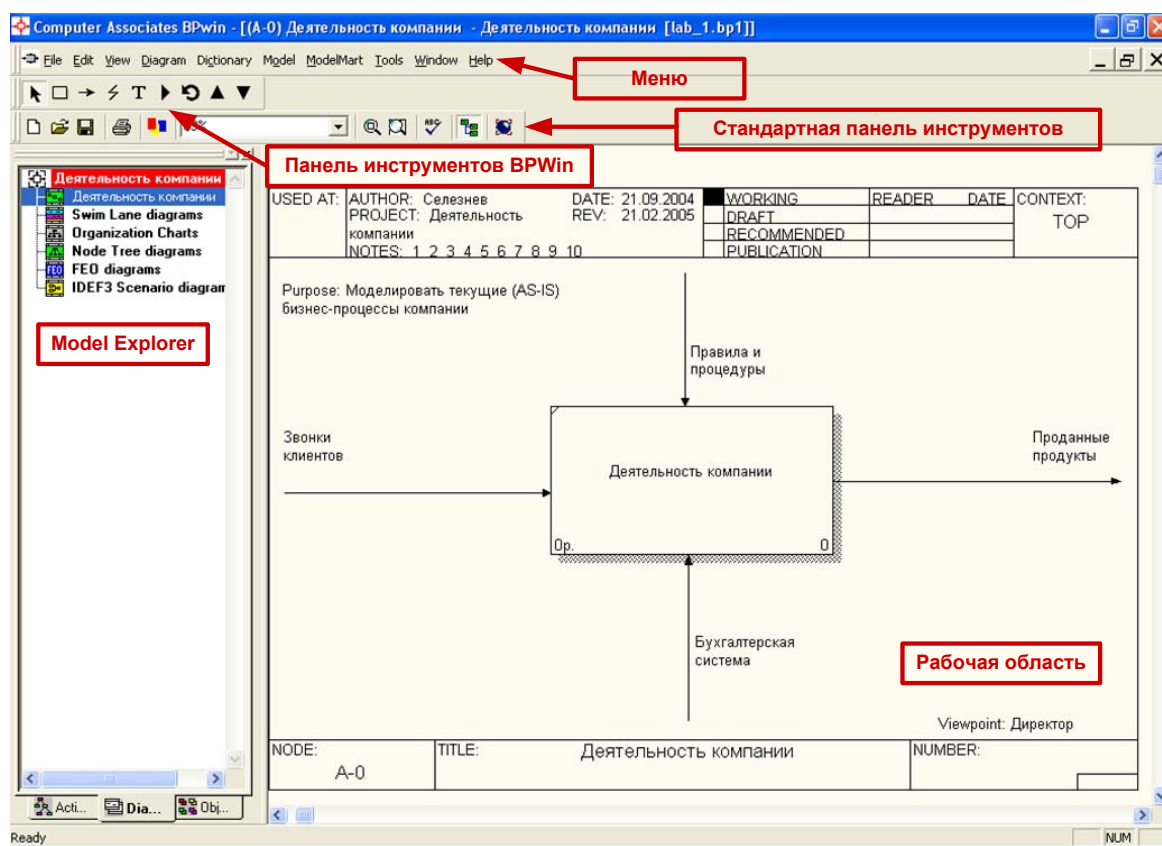





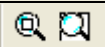





Рис. 2.1. Интегрированная среда разработки модели *BPwin* 4.0

Функциональность панели инструментов доступна из основного меню *BPwin* (табл. 2.1).

Описание элементов управления основной панели инструментов **Bpwin 4.0**

| Элемент управления | Описание | Пункт меню |
|---|---|----------------------------|
|  | Создать новую модель | <i>File/New</i> |
|  | Открыть модель | <i>File/Open</i> |
|  | Сохранить модель | <i>File/Save</i> |
|  | Напечатать модель | <i>File/Print</i> |
|  | Выбор масштаба | <i>View/Zoom</i> |
|  | Масштабирование | <i>View/Zoom</i> |
|  | Проверка правописания | <i>Tools/Spelling</i> |
|  | Включение и выключение навигатора модели <i>Model Explorer</i> | <i>View/Model Explorer</i> |
|  | Включение и выключение дополнительной панели инструментов работы с <i>ModelMart</i> | <i>ModelMart</i> |

При создании новой модели возникает диалог, в котором следует указать, будет ли создана модель заново, или она будет открыта из файла либо из репозитория *ModelMart*, внести имя модели и выбрать методологию, в которой будет построена модель (рис. 2.2). Система *ModelMart* – хранилище моделей, к которому открыт доступ для участников проекта создания информационной системы.

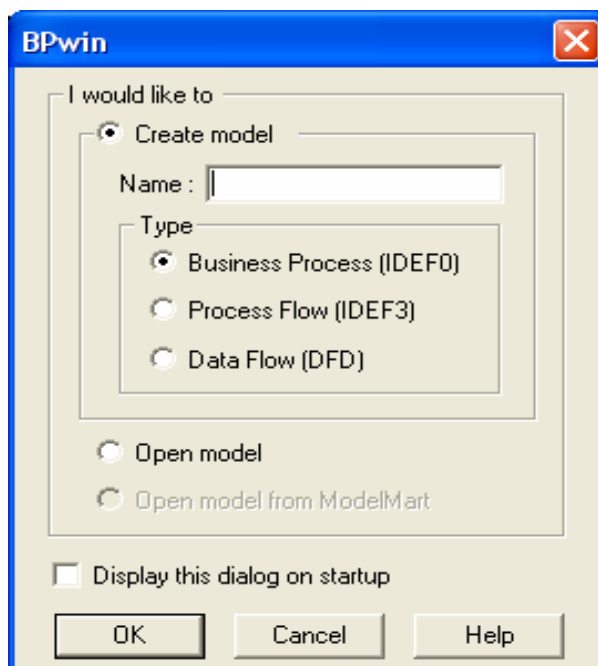


Рис. 2.2. Диалог создания модели

BPwin поддерживает три методологии – *IDEF0*, *IDEF3* и *DFD*, каждая из которых решает свои специфические задачи. В **BPwin** возможно построение смешанных моделей, т.е. модель, может содержать одновременно как диаграммы *IDEF0*, так и *IDEF3* и *DFD*. Состав палитры инструментов изменяется автоматически, когда происходит переключение с одной нотации на другую.

Установка цвета и шрифта объектов

Пункты контекстного меню *Font Editor* и *Color Editor* вызывают соответствующие диалоги для установки шрифта (в том числе его размера и стиля) и цвета объекта. **BPwin** позволяет установить шрифт по умолчанию для объектов определенного типа на диаграммах и в отчетах. Для этого следует выбрать меню *Model/Default Fonts*, после чего появляется каскадное меню, каждый пункт которого служит для установки шрифтов для определенного типа объектов:

Context Activity – работа на контекстной диаграмме;

Context Arrow – стрелки на контекстной диаграмме;

Decomposition Activity – работы на диаграмме декомпозиции;

Decomposition Arrow – стрелки на диаграмме декомпозиции;

NodeTree Text – текст на диаграмме дерева узлов;

Frame User Text – текст, вносимый пользователем в каркасе диаграмм;

Frame System Text – системный текст в каркасе диаграмм;

Text Blocks – текстовые блоки;

Parent Diagram Text – текст родительской диаграммы;

Parent Diagram Title Text – текст заголовка родительской диаграммы;

Report Text – текст отчетов.

Методология IDEF0

Принципы построения модели IDEF0

Наиболее удобным языком моделирования бизнес-процессов является *IDEF0*. Под моделью в *IDEF0* понимают описание системы (текстовое и графическое), которое должно дать ответ на некоторые заранее определенные вопросы. В *IDEF0* система представляется как совокупность взаимодействующих работ или функций. Такая чисто функциональная ориентация является принципиальной – функции системы анализируются независимо от объектов, которыми они оперируют, благодаря чему можно более четко смоделировать логику и взаимодействие процессов организации.

Моделируемая система рассматривается как произвольное подмножество неограниченного множества. Система имеет границу. Взаимодействие системы с окружающим миром описывается как вход (нечто, что перерабатывается системой), выход (результат деятельности системы), управление (стратегии и процедуры, под управлением которых производится работа) и механизм (ресурсы, необходимые для проведения работы). Находясь под управлением, система преобразует входы в выходы, используя механизмы.

Процесс моделирования какой-либо системы в *IDEF0* начинается с определения контекста, т.е. наиболее абстрактного уровня описания системы в целом. В контекст входит определение субъекта моделирования, цели и точки зрения на модель. Под субъектом понимается сама система. Описание области как системы в целом, так и ее компонентов является основой построения модели. Она должна быть в основном сформулирована изначально, поскольку именно область определяет направление моделирования и когда должна быть закончена модель. При формулировании области необходимо учитывать два компонента – широту и глубину. Широта подразумевает определение границ модели – мы определяем, что будет рассматриваться внутри системы, а что снаружи. Глубина определяет, на каком уровне детализации модель является завершенной. При определении глубины системы необходимо помнить об ограничениях времени – трудоемкость построения модели растет в геометрической прогрессии от глубины декомпозиции. После определения границ модели предполагается, что новые объекты не должны вноситься в моделируемую систему; поскольку все объекты модели взаимосвязаны, внесение нового объекта может быть не просто арифметической добавкой, но в состоянии изменить существующие взаимосвязи.

Цель моделирования (*Purpose*). Модель не может быть построена без четко сформулированной цели. Цель должна отвечать на следующие вопросы:

- Почему этот процесс должен быть замоделирован?
- Что должна показывать модель?
- Что может получить читатель?

Формулировка цели позволяет команде аналитиков сфокусировать усилия в нужном направлении.

Точка зрения (*Viewpoint*). Хотя при построении модели учитываются мнения различных людей, модель должна строиться с единой точки зрения. Точку зрения можно представить как взгляд человека, который видит систему в нужном для моделирования аспекте. Точка зрения должна соответствовать цели моделирования. Для этой цели обычно используют диаграммы *FEO* (*For Exposition Only*), которые будут описаны в дальнейшем.

IDEF0-модель предполагает наличие четко сформулированной цели, единственного субъекта моделирования и одной точки зрения. Для внесения области, цели и точки зрения в модели *IDEF0* в ***BPwin*** следует выбрать пункт меню *Model/Model Properties*, вызывающий диалог *Model Properties* (рис. 2.3).

В закладке *Status* того же диалога можно описать статус модели (черновой вариант, рабочий, окончательный и т.д.), время создания и последнего редактирования (отслеживается в дальнейшем автоматически по системной дате). В закладке *Source* описываются источники информации для построения модели (например, "Опрос экспертов предметной области и анализ документации"). Закладка *General* служит для внесения имени про-

екта и модели, имени и инициалов автора и временных рамок модели – *AS-IS* и *TO-BE*.

Рис. 2.3. Диалог *Model Properties*

Модели *AS-IS* и *TO-BE*. Обычно сначала строится модель существующей организации работы – *AS-IS* (как есть). На основе модели *AS-IS* достигается консенсус между различными единицами бизнеса по тому, "кто что сделал" и что каждая единица бизнеса добавляет в процесс. Модель *AS-IS* позволяет выяснить, "что мы делаем сегодня" перед тем, как перепрыгнуть на то, "что мы будем делать завтра". Анализ функциональной модели позволяет понять, где находятся наиболее слабые места, в чем будут состоять преимущества новых бизнес-процессов и насколько глубоким изменениям подвергнется существующая структура организации бизнеса. Найденные в модели *AS-IS* недостатки можно исправить при создании модели *TO-BE* (как будет) – модели новой организации бизнес-процессов. Модель *TO-BE* нужна для анализа альтернативных/лучших путей выполнения работы и документирования того, как компания будет делать бизнес в будущем.

Распространенная ошибка при создании модели *AS-IS* – это создание идеализированной модели. Примером может служить создание модели на основе знаний руководителя, а не конкретного исполнителя работ. Руководитель знаком с тем, как предполагается выполнение работы по руково-

дствам и должностным инструкциям и часто не знает, как на самом деле подчиненные выполняют рутинные работы. В результате получается приукрашенная, искаженная модель, которая несет ложную информацию и которую невозможно в дальнейшем использовать для анализа. Такая модель называется *SHOULD_BE* (как должно бы быть).

Технология проектирования ИС подразумевает сначала создание модели *AS-IS*, ее анализ и улучшение бизнес-процессов, т.е. создание модели *TO-BE*, и только на основе модели *TO-BE* строится модель данных, прототип и затем окончательный вариант ИС.

Диаграммы *IDEF0*

Основу методологии *IDEF0* составляет графический язык описания бизнес-процессов. Модель в нотации *IDEF0* представляет собой совокупность иерархически упорядоченных и взаимосвязанных диаграмм. Каждая диаграмма является единицей описания системы и располагается на отдельном листе.

Модель может содержать четыре типа диаграмм:

- контекстную диаграмму (в каждой модели может быть только одна контекстная диаграмма);
- диаграммы декомпозиции;
- диаграммы дерева узлов;
- диаграммы только для экспозиции (*FEO*).

Контекстная диаграмма является вершиной древовидной структуры диаграмм и представляет собой самое общее описание системы и ее взаимодействия с внешней средой. После описания системы в целом проводится разбиение ее на крупные фрагменты. Этот процесс называется функциональной декомпозицией, а диаграммы, которые описывают каждый фрагмент и взаимодействие фрагментов, называются диаграммами декомпозиции. После декомпозиции контекстной диаграммы проводится декомпозиция каждого большого фрагмента системы на более мелкие и так далее, до достижения нужного уровня подробности описания. Синтаксис описания системы в целом и каждого ее фрагмента одинаков во всей модели.

Диаграмма дерева узлов показывает иерархическую зависимость работ, но не взаимосвязи между работами. Диаграмм деревьев узлов может быть в модели сколь угодно много, поскольку дерево может быть построено на произвольную глубину и не обязательно с корня.

Диаграммы для экспозиции (*FEO*) строятся для иллюстрации отдельных фрагментов модели, для иллюстрации альтернативной точки зрения, либо для специальных целей.

Работы (*Activity*)

Работы обозначают поименованные процессы, функции или задачи, которые происходят в течение определенного времени и имеют распознаваемые результаты. Работы изображаются в виде прямоугольников. Все

работы должны быть названы и определены. Имя работы должно быть выражено отглагольным существительным, обозначающим действие (например, "Изготовление детали", "Прием заказа" и т.д.).

Стрелки (*Arrow*)

Взаимодействие работ с внешним миром и между собой описывается в виде стрелок. Стрелки представляют собой некую информацию и именуются существительными (например, "Заготовка", "Изделие", "Заказ").

В *IDEF0* различают пять типов стрелок:

Вход (*Input*) – материал или информация, которые используются или преобразуются работой для получения результата (выхода). Допускается, что работа может не иметь ни одной стрелки входа. Каждый тип стрелок подходит к определенной стороне прямоугольника, изображающего работу, или выходит из нее. Стрелка входа рисуется как входящая в левую грань работы.

Управление (*Control*) – правила, стратегии, процедуры или стандарты, которыми руководствуется работа. Каждая работа должна иметь хотя бы одну стрелку управления. Стрелка управления рисуется как входящая в верхнюю грань работы. Управление влияет на работу, но не преобразуется работой. Если цель работы – изменить процедуру или стратегию, то такая процедура или стратегия будет для работы входом. В случае возникновения неопределенности в статусе стрелки (управление или вход) рекомендуется рисовать стрелку управления.


Выход (*Output*) – материал или информация, которые производятся работой. Каждая работа должна иметь хотя бы одну стрелку выхода. Работа без результата не имеет смысла и не должна моделироваться. Стрелка выхода рисуется как исходящая из правой грани работы.


Механизм (*Mechanism*) – ресурсы, которые выполняют работу, например персонал предприятия, станки, устройства и т.д. Стрелка механизма рисуется как входящая в нижнюю грань работы. По усмотрению аналитика стрелки механизма могут не изображаться в модели.

Вызов (*Call*) – специальная стрелка, указывающая на другую модель работы. Стрелка вызова рисуется как исходящая из нижней грани работы. Стрелка вызова используется для указания того, что некоторая работа выполняется за пределами моделируемой системы. В *BPwin* стрелки вызова используются в механизме слияния и разделения моделей.

Граничные стрелки. Стрелки на контекстной диаграмме служат для описания взаимодействия системы с окружающим миром. Они могут начинаться у границы диаграммы и заканчиваться у работы, или наоборот. Такие стрелки называются граничными.

Для внесения граничной стрелки входа следует:

- щелкнуть по кнопке с символом стрелки  в палитре инструментов и переместить курсор в левую часть экрана, пока не появится черная полоска;

- щелкнуть один раз по полоске в левой стороне экрана (откуда выходит стрелка) и еще раз в левой части работы со стороны входа (где заканчивается стрелка);
- вернуться в палитру инструментов и выбрать опцию редактирования стрелки ;
- щелкнуть правой кнопкой мыши на линии стрелки, в контекстном меню выбрать *Name* и ввести имя стрелки в закладке *Name* диалога *Arrow Properties*.

Стрелки управления, выхода, механизма и выхода изображаются аналогично. Для рисования стрелки выхода, например, следует щелкнуть по кнопке с символом стрелки в палитре инструментов, щелкнуть в правой части работы со стороны выхода (где начинается стрелка), перенести курсор к правой стороне экрана, пока не появится начальная штриховая полоска, и щелкнуть один раз по штриховой полоске.

Имена вновь внесенных стрелок автоматически заносятся в словарь *Arrow Dictionary* (рис. 2.4).







| Dictionary Edit View Help | |
|---|---|
|       | |
| Name | Definition |
| Бухгалтерская система | Оформление счетов, оплата счетов, работа с заказами |
| Звонки клиентов | Запрос информации, заказы, тех. поддержка и т.д. |
| Правила и процедуры | Правила продаж, инструкции по сборке, процедуры тестирования. |
| Проданные продукты | Настольные и портативные компьютеры |
| | |

Рис. 2.4. Словарь *Arrow Dictionary*

Изменить данные о стрелках можно с помощью меню *Arrow Properties* (рис. 2.5).

Порядок выполнения работы

В качестве примера рассматривается деятельность вымышленной компании. Компания занимается в основном сборкой и продажей настольных компьютеров и ноутбуков. Компания не производит компоненты самостоятельно, а только собирает и тестирует компьютеры.

Основные процедуры в компании таковы:

- продавцы принимают заказы клиентов;
- операторы группируют заказы по типам компьютеров;

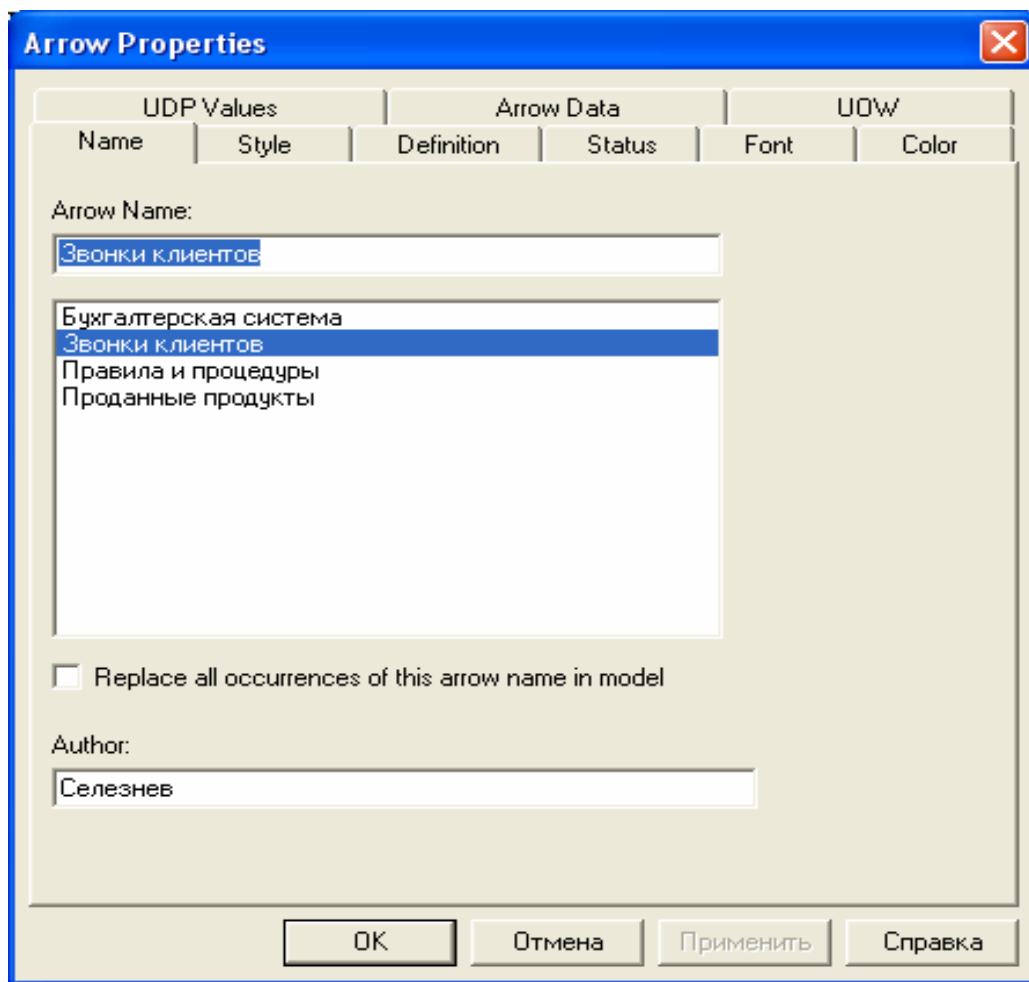




Рис. 2.5. Диалог *Arrow Properties*

- операторы собирают и тестируют компьютеры;
- операторы упаковывают компьютеры согласно заказам;
- кладовщик отгружает клиентам заказы.

Компания использует купленную бухгалтерскую информационную систему, которая позволяет оформить заказ, счет и отследить платежи по счетам.

1. Запустите **BPwin**. (Кнопка «*Start/Bpwin*»). Если появляется диалог «*ModelMart Connection Manager*», нажмите на кнопку *Cancel*.
2. Щелкните по кнопке . Появляется диалог *I would like to*. Внесите имя модели "Деятельность компании" и выберите Type – *IDEF0*. Нажмите *OK*.
3. Появляется меню *Properties for New Models*. Во вкладке *General* вводится фамилия и инициалы автора, остальные вкладки используются для определения настроек проекта.
4. Автоматически создается контекстная диаграмма.
5. Обратите внимание на кнопку  на панели инструментов. Эта кнопка включает и выключает инструмент просмотра и навигации – *Model*

Explorer (появляется слева). *Model Explorer* имеет три вкладки: *Activities*, *Diagrams* и *Objects*. Во вкладке *Activities* щелчок правой кнопкой по объекту позволяет редактировать его свойства. Если вам непонятно, как выполнить то или иное действие, вы можете вызвать помощь – клавиша *F1* или меню *Help*.

Измените шрифт для правильного отображения русских букв. Зайдите в меню *Model/Default Fonts* и выберите *Parent Diagram Text*. Поставьте галочку *change all occurrences* и нажмите *OK*. Если ничего не изменилось, повторите эту операцию с другим подменю, например: *Parent Diagram Title Text*.

6. Для изменений свойств модели используется меню *Model Properties*. По умолчанию тип модели: *Time Frame: AS-IS*. Во вкладке *Purpose* внесите цель – "*Purpose: Моделировать текущие (AS-IS) бизнес-процессы компании*" и точку зрения – "*Viewpoint: Директор*".

7. Во вкладке *Definition* внесите определение "Это учебная модель, описывающая деятельность компании" и цель *Scope*: "Общее управление бизнесом компании: исследование рынка, закупка компонентов, сборка, тестирование и продажа продуктов".

8. Перейдите на контекстную диаграмму и правой кнопкой мыши щелкните по работе. В контекстном меню выберите *Name*. Во вкладке *Name* внесите имя "Деятельность компании".


9. Во вкладке *Definition* внесите определение "Текущие бизнес-процессы компании".

10. Создайте стрелки на контекстной диаграмме с помощью меню *Model/Arrow Editor* согласно табл. 2.2.

Таблица 2.2

Стрелки контекстной диаграммы

| <i>Arrow Name</i> | <i>Arrow Definition</i> |
|-----------------------|--|
| Бухгалтерская система | Оформление счетов, оплата счетов, работа с заказами |
| Звонки клиентов | Запросы информации, заказы, техподдержка и т.д. |
| Правила и процедуры | Правила продаж, инструкции по сборке, процедуры тестирования, критерии производительности и т.д. |
| Проданные продукты | Настольные и портативные компьютеры |

11. С помощью кнопки  внесите текст в поле диаграммы - точку зрения и цель (рис. 2.6) как показано на рис. 2.7.

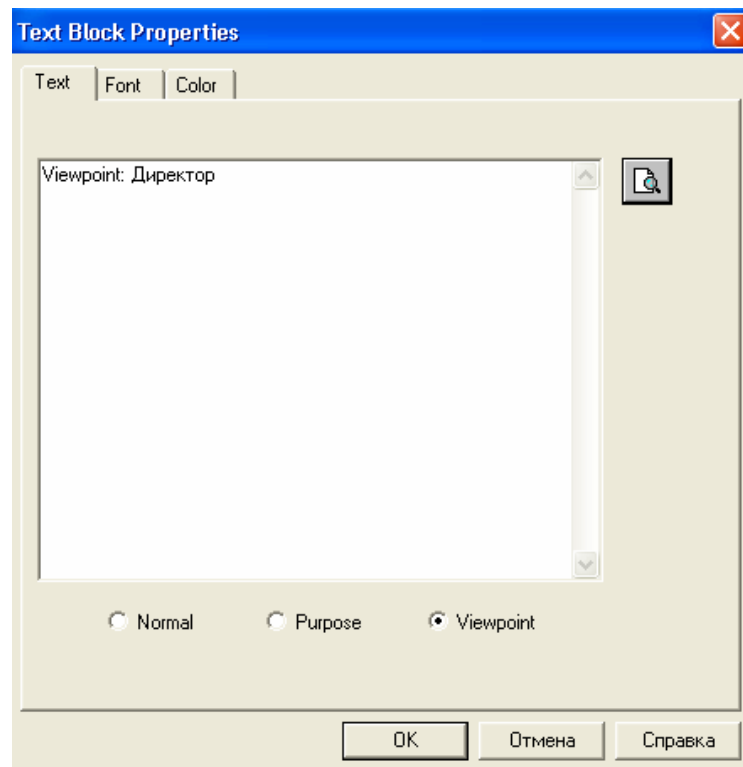


Рис. 2.6. Внесение текста в поле диаграммы с помощью редактора *Text Block Editor*

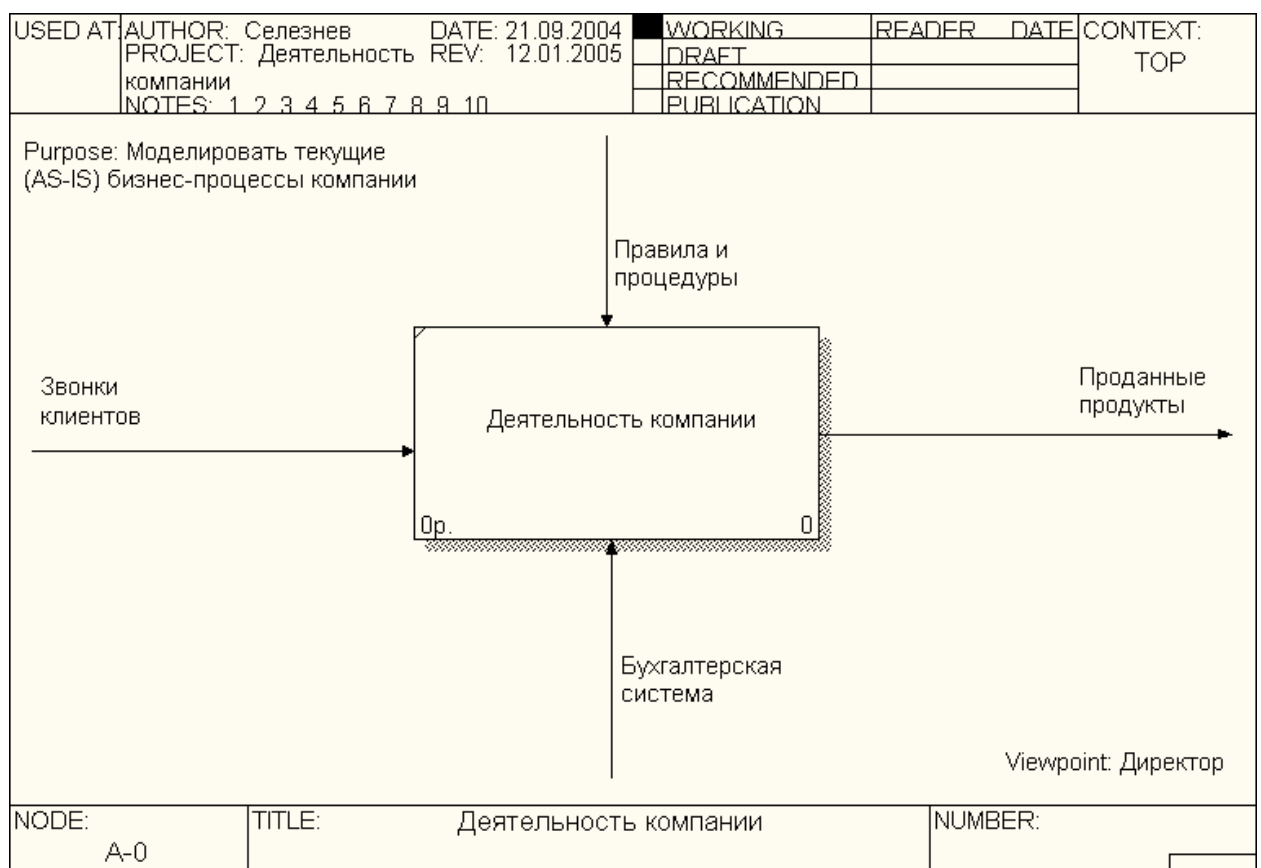
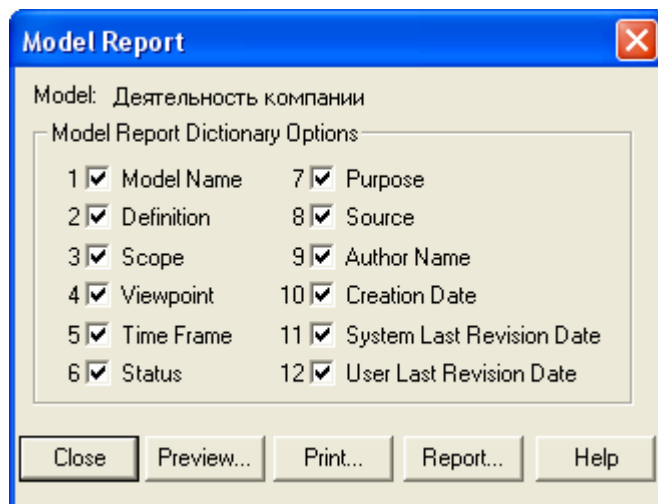
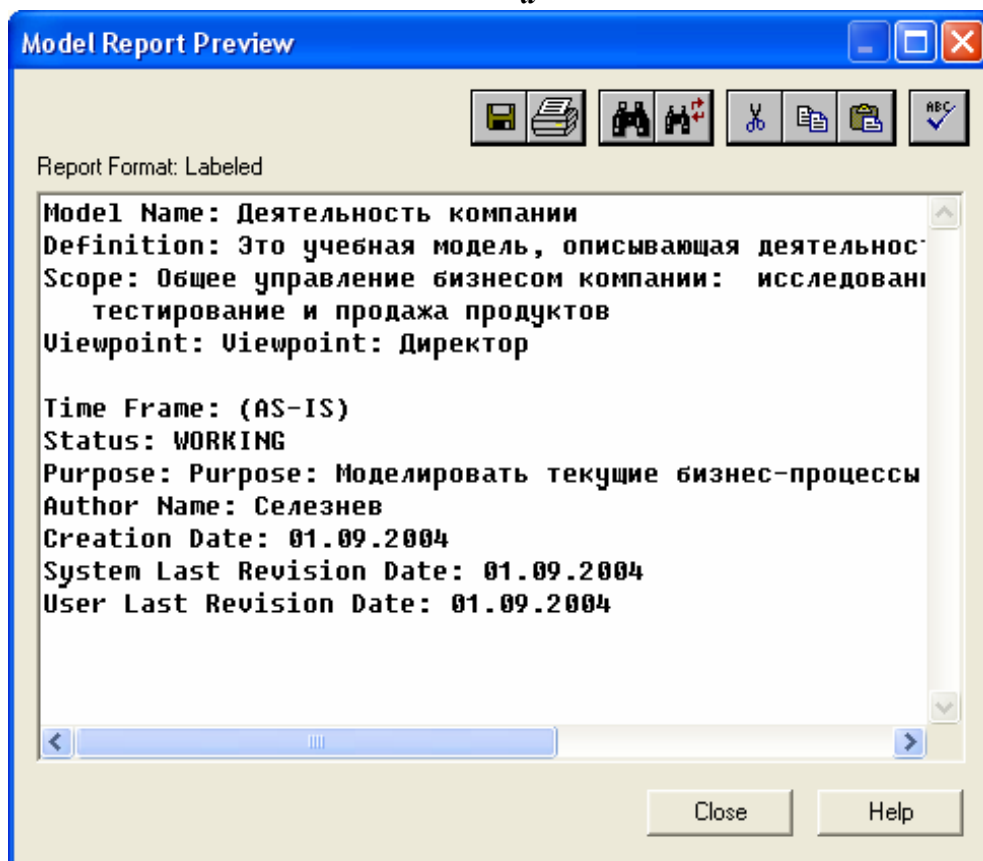


Рис. 2.7. Контекстная диаграмма

12. Создайте отчет по модели. Меню *Tools/Reports/Model Report* (рис. 2.8).



a



б

Рис. 2.8. Отчет *Model Report*

Контрольные вопросы

1. Основная цель использования *BPWin*.
2. Принципы построения диаграмм *IDEF0*.
3. В чем отличие моделей *AS-IS* и *TO-BE*?
4. Что такое *Purpose* и *Viewpoint*?
5. Как создать отчет по модели?

Лабораторная работа №2. Диаграммы декомпозиции

Цель работы: Создание диаграммы декомпозиции.

Теоретические сведения

Диаграмма декомпозиции предназначена для детализации работы. В отличие от моделей, отображающих структуру организации, работа на диаграмме верхнего уровня в *IDEF0* – это не элемент управления нижестоящими работами. Работы нижнего уровня – это то же самое, что работы верхнего уровня, но в более детальном изложении. Как следствие этого границы работы верхнего уровня – это то же самое, что границы диаграммы декомпозиции.

ICOM-коды

ICOM (аббревиатура от *Input, Control, Output* и *Mechanism*) – коды, предназначенные для идентификации граничных стрелок. Код *ICOM* содержит префикс, соответствующий типу стрелки (*I, C, O* или *M*), и порядковый номер.

BPwin вносит *ICOM*-коды автоматически. Для отображения *ICOM*-кодов следует включить опцию *Show ICOM codes* на закладке *Display* диалога *Model Properties* (меню *Model/Model Properties*).

Чтобы не возникло неоднозначных трактовок, в словаре стрелок каждому понятию можно дать расширенное и, при необходимости, формальное определение.

Содержимое словаря стрелок можно распечатать в виде отчета (меню *Tools/Reports/Arrow Report*) и получить тем самым толковый словарь терминов предметной области, использующихся в модели.

Связи работ

При декомпозиции работы входящие в нее и исходящие из нее стрелки (кроме стрелки вызова) автоматически появляются на диаграмме декомпозиции (миграция стрелок), но при этом не касаются работ. Такие стрелки называются несвязанными (*unconnected border arrow*) и воспринимаются в **BPwin** как синтаксическая ошибка.

Для связывания стрелок входа, управления или механизма необходимо перейти в режим редактирования стрелок, щелкнуть по конечнику стрелки и щелкнуть по соответствующему сегменту работы. Для связывания стрелки выхода необходимо перейти в режим редактирования стрелок, щелкнуть по сегменту выхода работы и затем по стрелке.

Для связи работ между собой используются внутренние стрелки, т.е. стрелки, которые не касаются границы диаграммы, начинаются у одной и кончаются у другой работы. Для рисования внутренней стрелки необходимо в режиме рисования стрелок щелкнуть по сегменту (например, выхода) одной работы и затем по сегменту (например, входа) другой.

В *IDEF0* различают пять типов связей работ:

Связь по входу (*output-input*), когда стрелка выхода вышестоящей работы (далее – просто выход) направляется на вход нижестоящей.

Связь по управлению (*output-control*), когда выход вышестоящей работы направляется на управление нижестоящей. Связь по управлению показывает доминирование вышестоящей работы. Данные или объекты выхода вышестоящей работы не меняются в нижестоящей.

Обратная связь по входу (*output-input feedback*), когда выход нижестоящей работы направляется на вход вышестоящей. Такая связь, как правило, используется для описания циклов.

Обратная связь по управлению (*output-control feedback*), когда выход нижестоящей работы направляется на управление вышестоящей. Обратная связь по управлению часто свидетельствует об эффективности бизнес-процесса.

Связь выход-механизм (*output-mechanism*), когда выход одной работы направляется на механизм другой. Эта взаимосвязь используется реже остальных и показывает, что одна работа подготавливает ресурсы, необходимые для проведения другой работы.

Явные стрелки. Явная стрелка имеет источником одну-единственную работу и назначением тоже одну-единственную работу.

Разветвляющиеся и сливающиеся стрелки. Одни и те же данные или объекты, порожденные одной работой, могут использоваться сразу в нескольких других работах. С другой стороны, стрелки, порожденные в разных работах, могут представлять собой одинаковые или однородные данные или объекты, которые в дальнейшем используются или перерабатываются в одном месте. Для моделирования таких ситуаций в *IDEF0* используются разветвляющиеся и сливающиеся стрелки. Для разветвления стрелки нужно в режиме редактирования стрелки щелкнуть по фрагменту стрелки и по соответствующему сегменту работы. Для слияния двух стрелок выхода нужно в режиме редактирования стрелки сначала щелкнуть по сегменту выхода работы, а затем по соответствующему фрагменту стрелки.

Смысл разветвляющихся и сливающихся стрелок передается именованием каждой ветви стрелок. Существуют определенные правила именования таких стрелок. Рассмотрим их на примере разветвляющихся стрелок. Если стрелка именована до разветвления, а после разветвления ни одна из ветвей не именована, то подразумевается, что каждая ветвь моделирует те же данные или объекты, что и ветвь до разветвления. Если при этом какая-либо ветвь после разветвления осталась неименованной, то подразумевается, что она моделирует те же данные или объекты, что и ветвь до разветвления. Недопустима ситуация, когда стрелка до разветвления не именована, а после разветвления не именована какая-либо из ветвей. **BPwin** определяет такую стрелку как синтаксическую ошибку.

Правила именования сливающихся стрелок полностью аналогичны – ошибкой будет считаться стрелка, которая после слияния не именована, а

до слияния не именована какая-либо из ее ветвей. Для именования отдельной ветви разветвляющихся и сливающихся стрелок следует выделить на диаграмме только одну ветвь, после этого вызвать редактор имени и присвоить имя стрелке. Это имя будет соответствовать только выделенной ветви.

Тоннелирование стрелок

Вновь внесенные граничные стрелки на диаграмме декомпозиции нижнего уровня изображаются в квадратных скобках и автоматически не появляются на диаграмме верхнего уровня.

Если щелкнуть по кнопке *Resolve Border Arrow*, стрелка мигрирует на диаграмму верхнего уровня, если по кнопке *Change To Tunnel* – стрелка будет затоннелирована и не попадет на другую диаграмму. Тоннельная стрелка изображается с круглыми скобками на конце.

Тоннелирование может быть применено для изображения малозначимых стрелок. Если на какой-либо диаграмме нижнего уровня необходимо изобразить малозначимые данные или объекты, которые не обрабатываются или не используются работами на текущем уровне, то их необходимо направить на вышестоящий уровень (на родительскую диаграмму). Если эти данные не используются на родительской диаграмме, их нужно направить еще выше, и т. д. В результате малозначимая стрелка будет изображена на всех уровнях и затруднит чтение всех диаграмм, на которых она присутствует. Выходом является тоннелирование стрелки на самом нижнем уровне. Такое тоннелирование называется "не-в-родительской-диаграмме".

Другим примером тоннелирование может быть ситуация, когда стрелка механизма мигрирует с верхнего уровня на нижний, причем на нижнем уровне этот механизм используется одинаково во всех работах без исключения. (Предполагается, что не нужно детализировать стрелку механизма, т. е. стрелка механизма на дочерней работе именована до разветвления, а после разветвления ветви не имеют собственного имени). В этом случае стрелка механизма на нижнем уровне может быть удалена, после чего на родительской диаграмме она может быть затоннелирована, а в комментарии к стрелке или в словаре можно указать, что механизм будет использоваться во всех работах дочерней диаграммы декомпозиции. Такое тоннелирование называется "не-в-дочерней-работе".

Работы (Activity)

Работы обозначают поименованные процессы, функции или задачи, которые происходят в течение определенного времени и имеют распознаваемые результаты. Работы изображаются в виде прямоугольников. Все работы должны быть названы и определены. Имя работы должно быть выражено отглагольным существительным, обозначающим действие (например, "Изготовление детали", "Прием заказа" и т.д.). Работа "Изготовление детали" может иметь, например, следующее определение: "Работа относится к полному циклу изготовления изделия от контроля качества сырья до отгрузки готового упакованного изделия". При создании новой модели

(меню *File/New*) автоматически создается контекстная диаграмма с единственной работой, изображающей систему в целом.

Для внесения имени работы следует щелкнуть по работе правой кнопкой мыши, выбрать в меню *Name – Editor* и в появившемся диалоге внести имя работы. Для описания других свойств работы служит диалог "*Activity Properties*".

Диаграммы декомпозиции содержат родственные работы, т.е. дочерние работы, имеющие общую родительскую работу. Декомпозировать работу на одну работу не имеет смысла: диаграммы с количеством работ более восьми получаются перенасыщенными и плохо читаются. Для обеспечения наглядности и лучшего понимания моделируемых процессов рекомендуется использовать от трех до шести блоков на одной диаграмме.


Работы на диаграммах декомпозиции обычно располагаются по диагонали от левого верхнего угла к правому нижнему. Такой порядок называется порядком доминирования. Согласно этому принципу расположения в левом верхнем углу располагается самая важная работа или работа, выполняемая по времени первой. Далее вправо вниз располагаются менее важные или выполняемые позже работы. Такое расположение облегчает чтение диаграмм, кроме того, на нем основывается понятие взаимосвязей работ.

Каждая из работ на диаграмме декомпозиции может быть в свою очередь декомпозирована. На диаграмме декомпозиции работы нумеруются автоматически слева направо. Номер работы показывается в правом нижнем углу. В левом верхнем углу изображается небольшая диагональная черта, которая показывает, что данная работа не была декомпозирована.

Стрелки

Альтернативный метод внесения имен и свойств стрелок – использование словаря стрелок (вызов словаря – меню *Dictionary/Arrow*). Если внести имя и свойства стрелки в словарь, ее можно будет внести в диаграмму позже. Стрелку нельзя удалить из словаря, если она используется на какой-либо диаграмме. Если удалить стрелку из диаграммы, из словаря она не удаляется. Имя и описание такой стрелки может быть использовано в дальнейшем. Для добавления стрелки необходимо перейти в конец списка и щелкнуть правой кнопкой по последней строке. Возникает новая строка, в которой нужно внести имя и свойства стрелки.

Порядок выполнения работы

1. Выберите кнопку перехода на нижний уровень в палитре инструментов  и в диалоге *Activity Box Count* установите число работ на диаграмме нижнего уровня – 3 – и нажмите *OK* (рис. 2.9).

Автоматически будет создана диаграмма декомпозиции. Правой кнопкой мыши щелкните по работе, выберите *Name* и внесите имя работы. Повторите операцию для всех трех работ. Затем внесите определение, статус и источник для каждой работы согласно табл. 2.3.

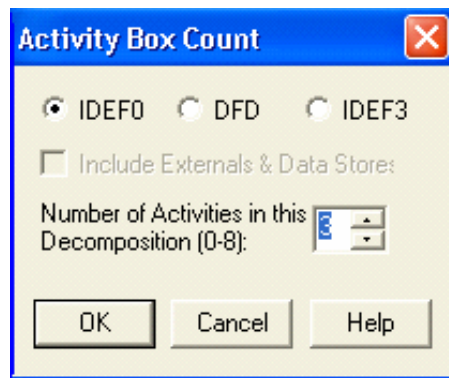


Рис. 2.9. Создание диаграммы декомпозиции

Таблица 2.3



Работы диаграммы декомпозиции A0


| <i>Activity Name</i> | <i>Definition</i> |
|-----------------------------------|--|
| Продажи и маркетинг | Телемаркетинг и презентации, выставки |
| Сборка и тестирование компьютеров | Сборка и тестирование настольных и портативных компьютеров |
| Отгрузка и получение | Отгрузка заказов клиентам и получение компонентов от поставщиков |

2. Для изменения свойств работ после их внесения в диаграмму можно воспользоваться словарем работ. Вызов словаря – меню *Dictionary /Activity* (рис. 2.10).

| Name | Definition |
|-----------------------|--|
| Деятельность компании | Текущие бизнес-процессы компании |
| Отгрузка и получение | Отгрузка заказов клиентам и получение компонентов |
| Продажи и маркетинг | Телемаркетинг и презентации, выставки |
| Сборка и | Сборка и тестирование настольных и портативных ком |

Рис. 2.10. Словарь *Activity Dictionary*

Если описать имя и свойства работы в словаре, ее можно будет внести в диаграмму позже с помощью кнопки  в палитре инструментов. Невозможно удалить работу из словаря, если она используется на какой-либо диаграмме. Если работа удаляется из диаграммы, из словаря она не удаляется. Имя и описание такой работы может быть использовано в дальнейшем. Для добавления работы в словарь необходимо перейти в конец списка и щелкнуть правой кнопкой по последней строке. Возникает новая строка, в которой нужно внести имя и свойства работы. Также можно воспользоваться клавишей табуляции для введения информации. Для удаления всех имен работ, не используемых в модели, щелкните по кнопке  (*Purge*).

3. Измените стрелку "Проданные продукты" и "Звонки клиентов". Перейдите в режим рисования стрелок. Свяжите граничные стрелки (кнопка  на палитре инструментов) так, как показано на рис. 2.11.

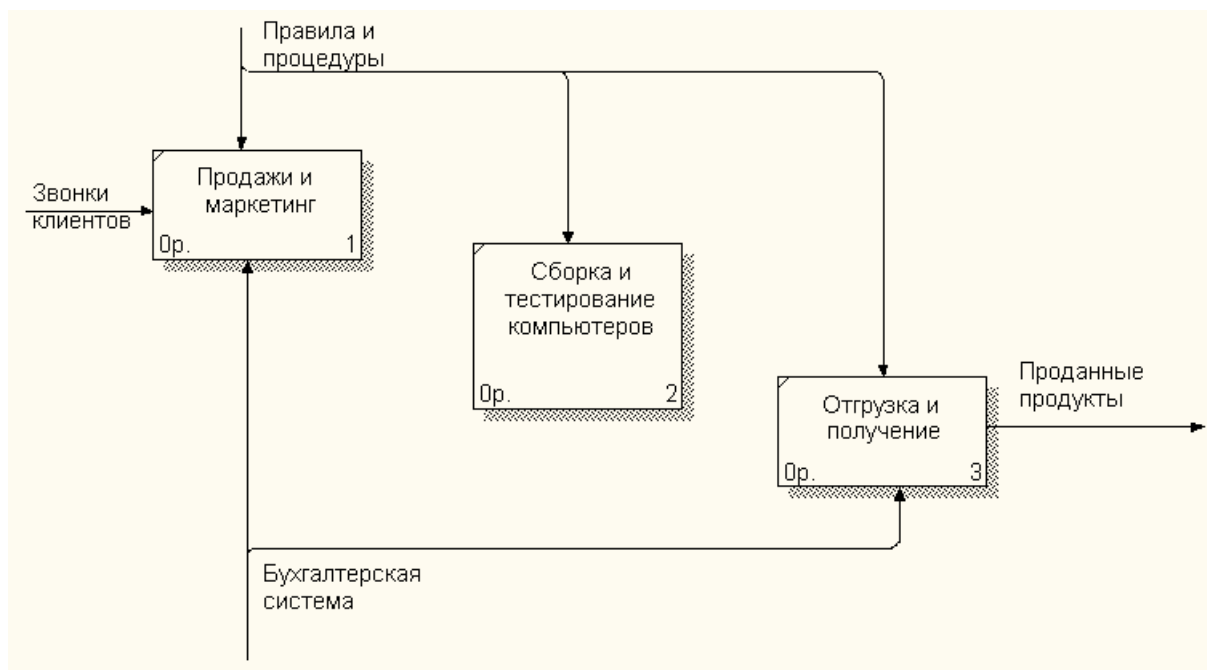


Рис. 2.11. Связанные граничные стрелки на диаграмме A0

4. Переименуйте работу "Сборка и тестирование компьютеров" на "Правила сборки и тестирования" (рис. 2.12). Внесите определение для ветви "Бухгалтерская система": "Инструкции по сборке, Процедуры тестирования, критерии производительности и т.д." Правой кнопкой мыши щелкните по ветви "Бухгалтерская система", переименуйте ее в "Система оформления заказов".

5. Переименуйте работу "Правила сборки и тестирования" в "Сборка и тестирование компьютеров". Создайте новые внутренние стрелки "Заказы клиентов", "Собранные компьютеры" и "Неисправные компоненты" так, как показано на рис. 2.13. Дайте название "Правила сборки и тестирования" стрелке управления, входящую в работу "Сборка и тестирование компьютеров".

6. Создайте стрелку обратной связи (по управлению) "Результаты сборки и тестирования", идущую от работы, "Сборка и тестирование компьютеров" к работе "Продажи и маркетинг". Измените стиль стрелки (увеличьте толщину линии) и установите опцию *Extra Arrowhead* (из контекстного меню). Методом *drag&drop* перенесите имена стрелок так, чтобы их было удобнее читать. Если необходимо, установите *Squiggle* (из контекстного меню). Результат изменений показан на рис. 2.14.

7. Свяжите работу "Продажи и маркетинг" с граничной стрелкой выхода "Маркетинговые материалы". На уровне работы "Сборка и тестирование компьютеров" для стрелки "Маркетинговые материалы" выберите опцию *Trim* из контекстного меню в целях уменьшения длины стрелки. Результат выполнения лабораторной работы №2 показан на рис. 2.15.

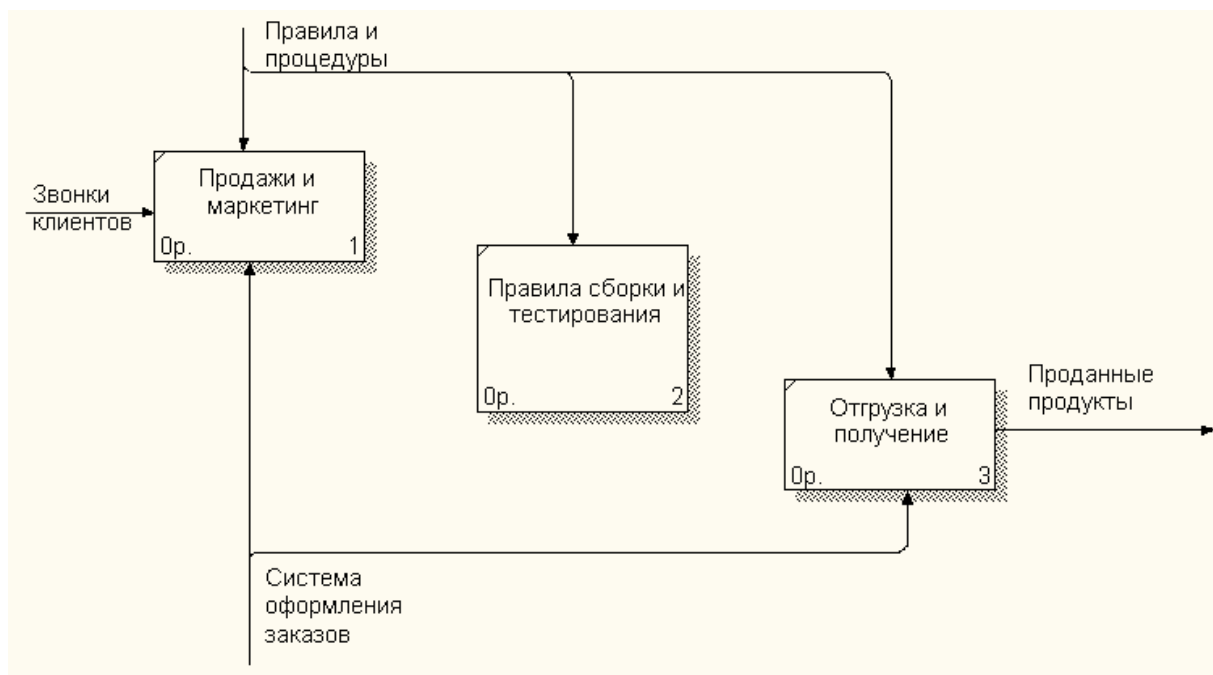


Рис. 2.12. Стрелка "Правила сборки и тестирования"

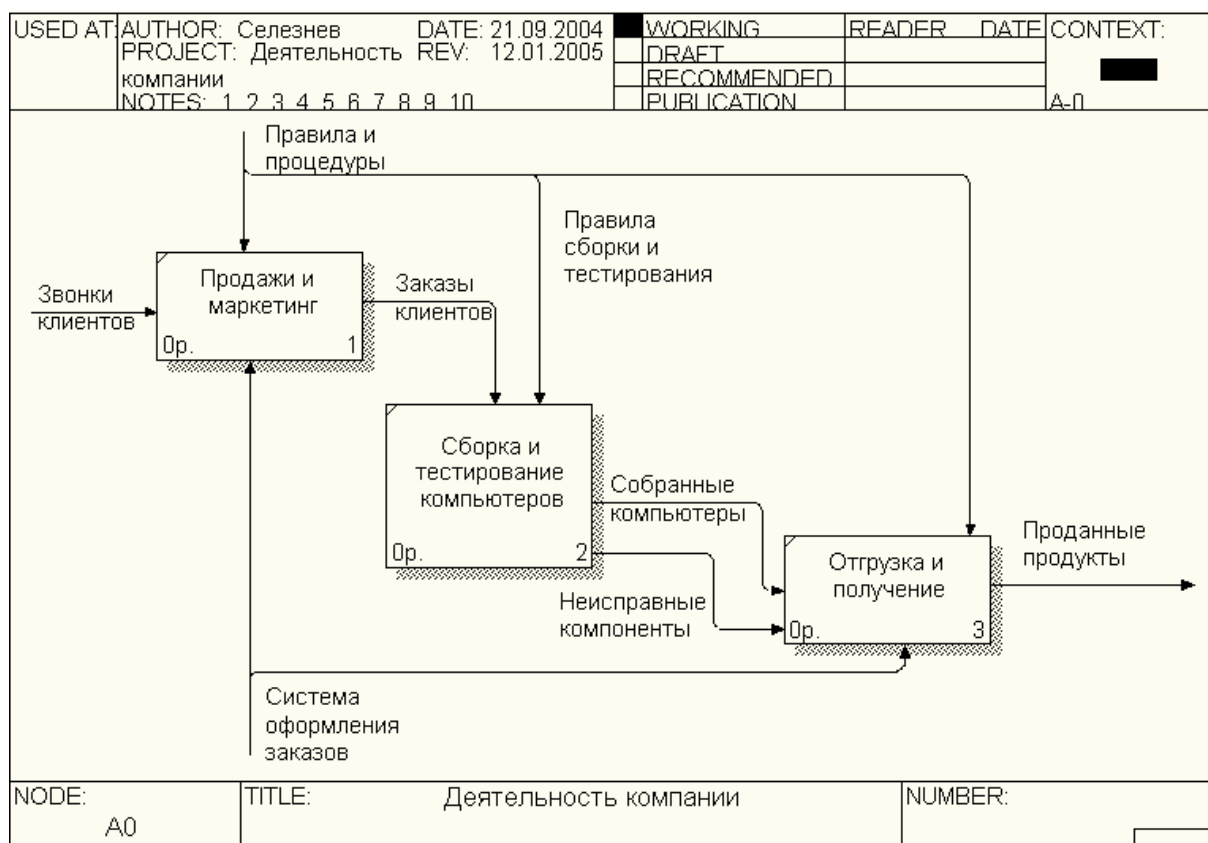


Рис. 2.13. Внутренние стрелки диаграммы AO

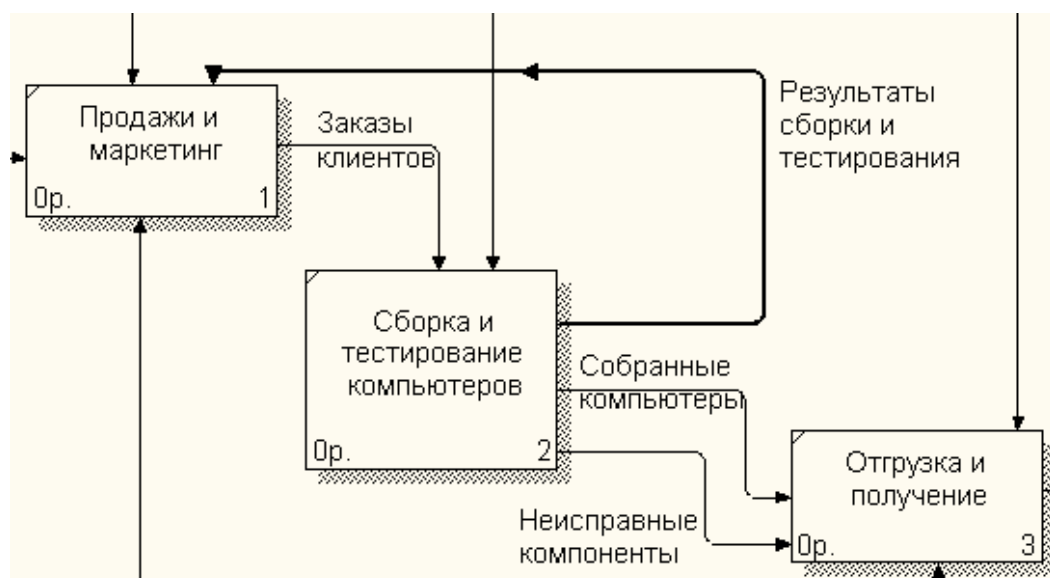


Рис. 2.14. Результат выполнения задания 6

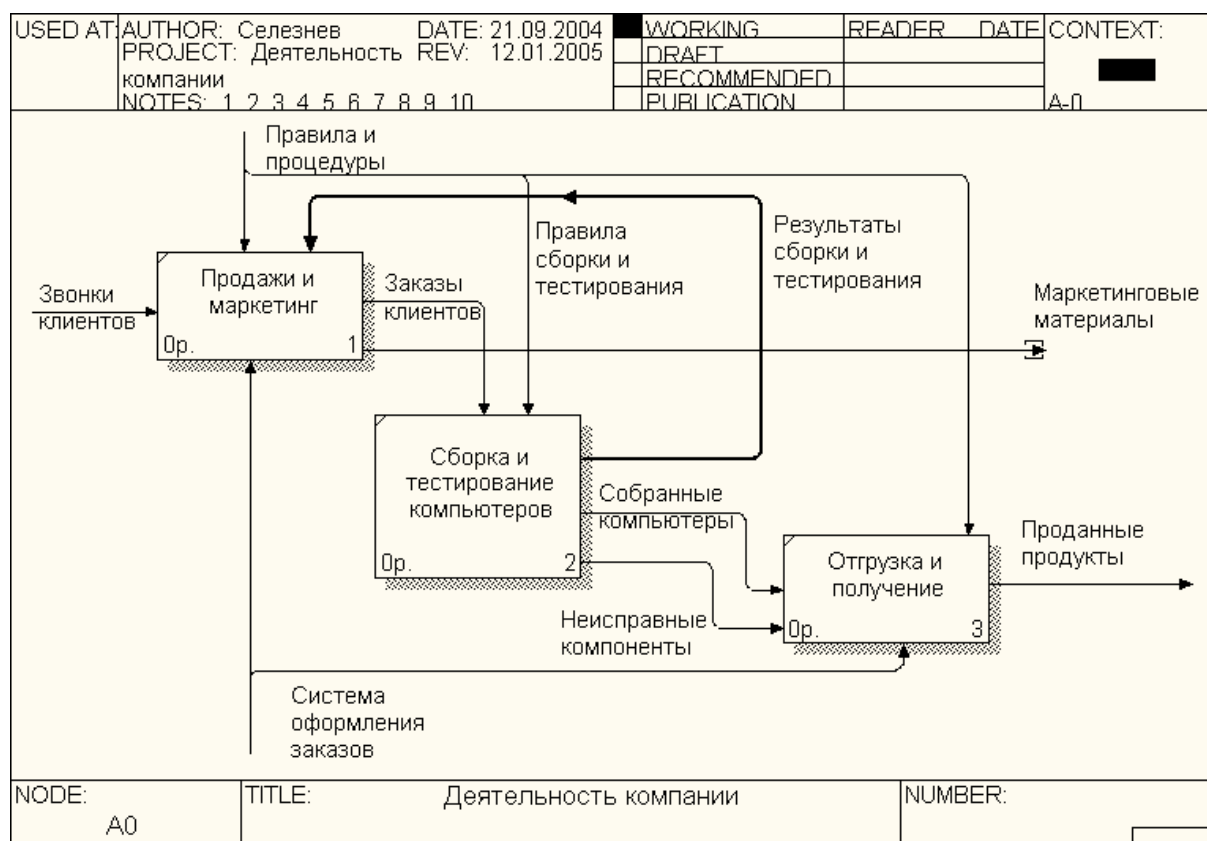


Рис. 2.15. Результат выполнения лабораторной работы №2 – диаграмма A0

Контрольные вопросы

1. Что такое *ICOM*-коды?
2. Какие бывают типы стрелок?
3. Что такое словарь работ, стрелок?
4. Какие бывают типы связей работ?
5. Каким образом происходит слияние и расщепление стрелок?

Лабораторная работа №3. Тоннелирование стрелок

Цель работы: Создание диаграммы декомпозиции A2

Теоретические сведения

В реальных диаграммах к каждой работе может подходить и от каждой может отходить около десятка стрелок. Если диаграмма содержит 6-8 работ, то она может содержать 30-40 стрелок, причем они могут сливаться, разветвляться и пресекаться. Такие диаграммы могут стать очень плохо читаемыми. В *IDEF0* существуют соглашения по рисованию диаграмм, которые призваны облегчить чтение и экспертизу модели. Некоторые из этих правил **BPwin** поддерживает автоматически, выполнение других следует обеспечить вручную.

- Прямоугольники работ должны располагаться по диагонали с левого верхнего в правый нижний угол (порядок доминирования). При создании новой диаграммы декомпозиции **BPwin** автоматически располагает работы именно в таком порядке. В дальнейшем можно добавлять новые работы или изменить расположение существующих, но нарушать диагональное расположение работ по возможности не следует. Порядок доминирования подчеркивает взаимосвязь работ, позволяет минимизировать изгибы и пересечения стрелок.
- Следует максимально увеличивать расстояние между входящими или выходящими стрелками на одной грани работы. Если включить опцию *Line Drawing: Automatically space arrows* на закладке *Layout* диалога *Model Properties* (меню *Edit/Model Properties*), **BPwin** будет располагать стрелки нужным образом автоматически.
- Следует максимально увеличить расстояние между работами, поворотами и пересечениями стрелок.
- Если две стрелки проходят параллельно (начинаются из одной и той же грани одной работы и заканчиваются на одной и той же грани другой работы), то по возможности следует их объединить и назвать единым термином.
- Обратные связи по входу рисуются "нижней" петлей, обратная связь по управлению – "верхней" (см. рис. 1.15, 1.17). **BPwin** автоматически рисует обратные связи нужным образом. Конечно, можно нарисовать их по другому, но это будет неправильно.
- Циклические обратные связи следует рисовать только в случае крайней необходимости, когда подчеркивают значение повторно используемого объекта. Принято изображать такие связи на диаграмме декомпозиции. **BPwin** не позволяет создать циклическую обратную связь за один прием. Если все же необходимо изобразить такую связь, следует сначала создать обычную связь по выходу, затем разветвить стрелку, направить новую, ветвь обратно ко входу работы-

источника и, наконец, удалить старую ветвь стрелки выхода (рис. 2.16).

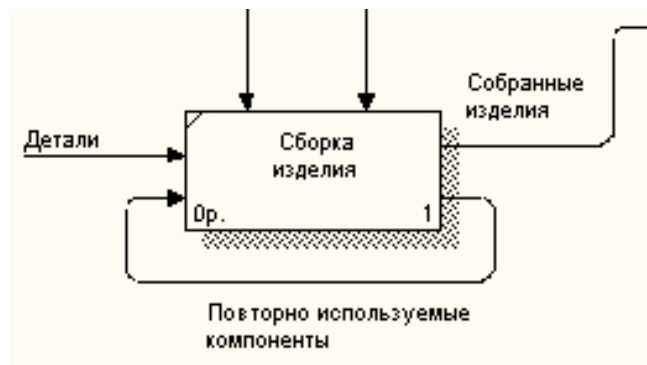


Рис. 2.16. Пример обратной циклической связи

- Следует минимизировать число пересечений, петель и поворотов стрелок. Это ручная и, в случае насыщенных диаграмм, творческая работа (рис. 2.17).

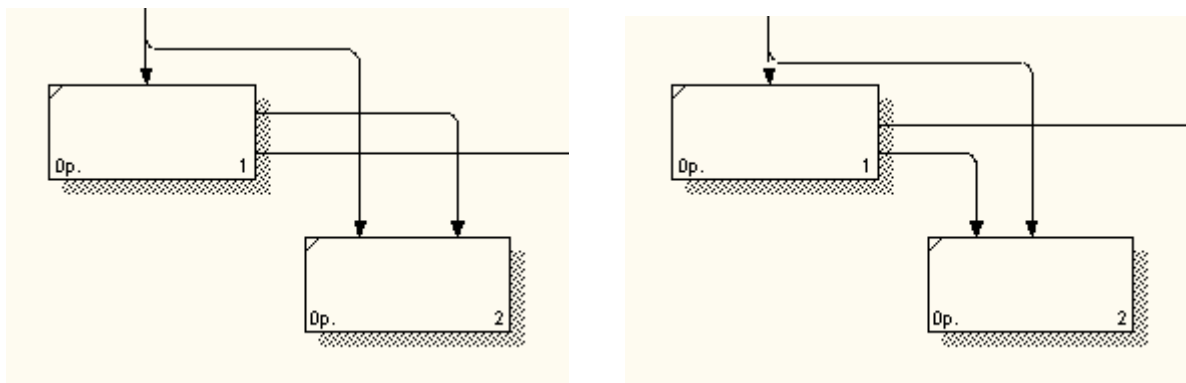


Рис. 2.17. Минимизация пересечений и поворотов стрелок

- Если нужно изобразить связь по входу, необходимо избегать "нависания" работ друг над другом. В этом случае **BPwin** изображает связи по входу в виде петли, что затрудняет чтение диаграмм (рис. 2.18).

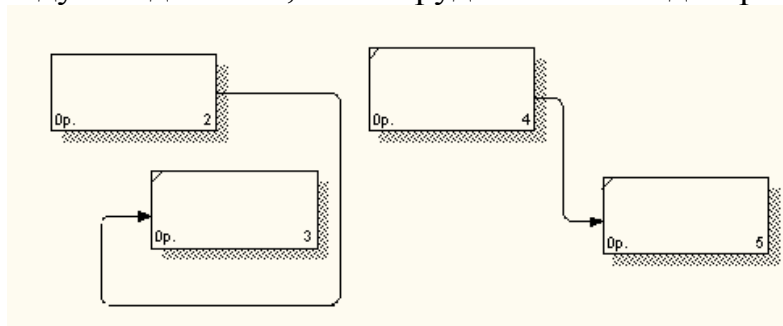


Рис. 2.18. Пример правильного (справа) и неправильного (слева) расположения работ при изображении связи по входу

Порядок выполнения работы

Переименуйте стрелку "Система оформления заказов" в стрелку "Бухгалтерская система".

1. Декомпозируем работу "Сборка и тестирование компьютеров" с 4-мя работами, используя методологию *IDEF0* по следующим критериям:

- Производственный отдел получает заказы клиентов от отдела продаж по мере их поступления.
- Диспетчер координирует работу сборщиков, сортирует заказы, группирует их и дает указание на отгрузку компьютеров, когда они готовы.
- Каждые 2 часа диспетчер группирует заказы – отдельно для настольных компьютеров и ноутбуков – и направляет на участок сборки.
- Сотрудники участка сборки собирают компьютеры согласно спецификациям заказа и инструкциям по сборке. Когда группа компьютеров, соответствующая группе заказов, собрана, она направляется на тестирование. Тестеры проверяют каждый компьютер и в случае необходимости заменяют неисправные компоненты.
- Тестеры направляют результаты тестирования диспетчеру, который на основании этой информации принимает решение о передаче компьютеров, соответствующих группе заказов, на отгрузку.
- На основе этой информации внесите новые работы и стрелки (табл. 2.4. и 2.5).

Таблица 2.4

Работы диаграммы декомпозиции *A2*

| <i>Activity Name</i> | <i>Activity Definition</i> |
|--|---|
| Отслеживание расписания и управление сборкой и тестированием | Просмотр заказов, установка расписания выполнения заказов, просмотр результатов тестирования, формирование групп заказов на сборку и отгрузку |
| Сборка настольных компьютеров | Сборка настольных компьютеров в соответствии с инструкциями и указаниями диспетчера |
| Сборка ноутбуков | Сборка ноутбуков в соответствии с инструкциями и указаниями диспетчера |
| Тестирование компьютеров | Тестирование компьютеров и компонентов. Замена неработающих компонентов |

Стрелки диаграммы декомпозиции A2

| <i>Arrow Name</i> | <i>Arrow Source</i> | <i>Arrow Source Type</i> | <i>Arrow Destination</i> | <i>Arrow Destination. Type</i> |
|-----------------------------------|--|--------------------------|--|--------------------------------|
| Диспетчер | Персонал производственного отдела | <i>Mechanism</i> | Отслеживание расписания и управление сборкой и тестированием | <i>Mechanism</i> |
| Заказы клиентов | Граница диаграммы | | Отслеживание расписания и управление сборкой и тестированием | <i>Control</i> |
| Заказы на настольные компьютеры | Отслеживание расписания и управление сборкой и тестированием | <i>Output</i> | Сборка настольных компьютеров | <i>Control</i> |
| Заказы на ноутбуки | Отслеживание расписания и управление сборкой и тестированием | <i>Output</i> | Сборка ноутбуков | <i>Control</i> |
| Компоненты | "Tunnel" | <i>Input</i> | Сборка настольных компьютеров | <i>Input</i> |
| | | | Сборка ноутбуков | <i>Input</i> |
| | | | Тестирование компьютеров | <i>Input</i> |
| Настольные компьютеры | Сборка настольных компьютеров | <i>Output</i> | Тестирование компьютеров | <i>Input</i> |
| Ноутбуки | Сборка ноутбуков | <i>Output</i> | Тестирование компьютеров | <i>Input</i> |
| Персонал производственного отдела | "Tunnel" | <i>Mechanism</i> | Сборка настольных компьютеров | <i>Mechanism</i> |
| | | <i>Mechanism</i> | Сборка ноутбуков | <i>Mechanism</i> |
| Правила сборки и тестирования | Граница диаграммы | | Сборка настольных компьютеров | <i>Control</i> |
| | | | Сборка ноутбуков | <i>Control</i> |
| | | | Тестирование компьютеров | <i>Control</i> |
| Результаты сборки и тестирования | Сборка настольных компьютеров | <i>Output</i> | Граница диаграммы | <i>Output</i> |
| | Сборка ноутбуков | <i>Output</i> | | |
| | Тестирование компьютеров | <i>Output</i> | | |

| <i>Arrow Name</i> | <i>Arrow Source</i> | <i>Arrow Source Type</i> | <i>Arrow Destination</i> | <i>Arrow Destination Type</i> |
|--|--|--------------------------|--|-------------------------------|
| Результаты тестирования | Тестирование компьютеров | <i>Output</i> | Отслеживание расписания и управление сборкой и тестированием | <i>Input</i> |
| Собранные компьютеры | Тестирование компьютеров | <i>Output</i> | Граница диаграммы | <i>Output</i> |
| Тестер | Персонал производственного отдела | <i>Mechanism</i> | Тестирование компьютеров | <i>Mechanism</i> |
| Указания передать компьютеры на отгрузку | Отслеживание расписания и управление сборкой и тестированием | <i>Output</i> | Тестирование компьютеров | <i>Control</i> |

2. Тоннелируйте и свяжите на верхнем уровне граничные стрелки "Компоненты" и "Персонал производственного отдела" (*Border Arrow Editor/Change it to resolved rounded tunnel*). Результат выполнения лабораторной работы №3 показан на рис. 2.19.

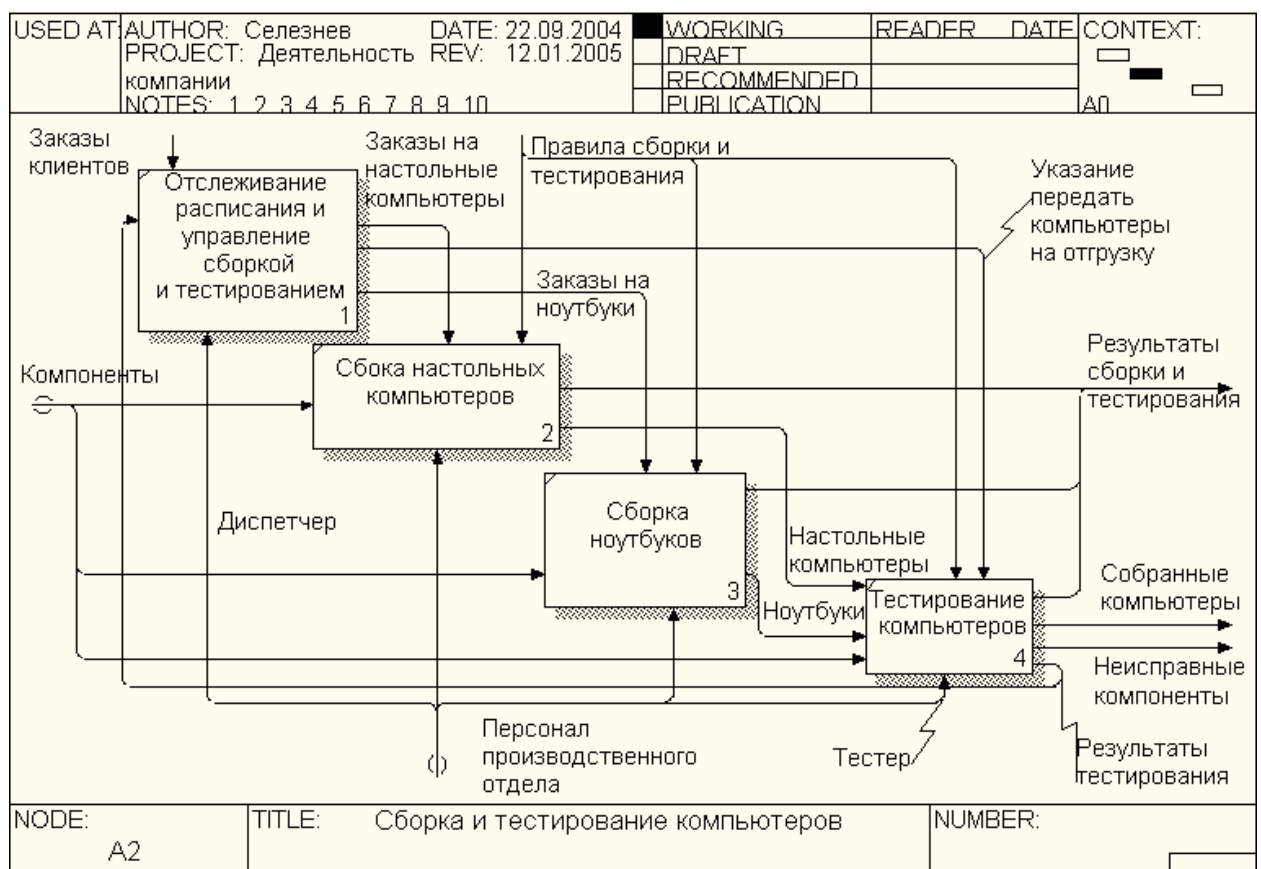


Рис. 2.19. Результат выполнения лабораторной работы №3

Контрольные вопросы

1. Что такое тоннелирование?
2. Типы тоннелирования и их отличия.
3. Правильное именование стрелок при слиянии и расщеплении?
4. Что такое *Squiggle*?
5. Как реализуется соглашение по рисованию диаграмм?

Лабораторная работа №4. Вспомогательные диаграммы

Цель работы: Построение диаграммы узлов и FEO-диаграммы.

Теоретические сведения


Каркас диаграммы

Каркас содержит заголовок (верхняя часть рамки) и подвал (нижняя часть). Заголовок каркаса используется для отслеживания диаграммы в процессе моделирования. Нижняя часть используется для идентификации и позиционирования в иерархии диаграммы.

Смысл элементов каркаса приведен в таблицах 2.6 и 2.7.

Таблица 2.6

Поля заголовка каркаса (слева направо)

| Поле | Смысл |
|----------------------------------|--|
| <i>Used At</i> | Используется для указания на родительскую работу в случае, если на текущую диаграмму ссылались посредством стрелки вызова |
| <i>Autor, Date, Rev, Project</i> | Имя создателя диаграммы, дата создания и имя проекта, в рамках которого была создана диаграмма. <i>REV</i> -дата последнего редактирования диаграммы |
| <i>Notes 123456789 10</i> | Используется при проведении сеанса экспертизы. Эксперт должен (на бумажной копии диаграммы) указать число замечаний, вычеркивая цифру из списка каждый раз при внесении нового замечания |
| <i>Working</i> | Новая диаграмма, кардинально обновленная диаграмма или новый автор диаграммы |
| <i>Draft</i> | Диаграмма прошла первичную экспертизу и готова к дальнейшему обсуждению |
| <i>Recommended</i> | Диаграмма и все ее сопровождающие документы прошли экспертизу. Новых изменений не ожидается |
| <i>Publication</i> | Диаграмма готова к окончательной печати и публикации |
| <i>Reader</i> | Имя читателя (эксперта) |
| <i>Context</i> | Схема расположения работ в диаграмме верхнего уровня. Работа, являющаяся родительской, показана темным прямоугольником, остальные – светлым. На контекстной диаграмме (<i>A-0</i>) показана надпись <i>TOP</i> . В левом нижнем углу показывается номер по узлу родительской диаграммы:  |

Поля подвала каркаса

| Поле | Смысл |
|---------------|--|
| <i>Node</i> | Номер узла диаграммы (номер родительской работы) |
| <i>Title</i> | Имя диаграммы. По умолчанию - имя родительской работы |
| <i>Number</i> | <i>C-Number</i> , уникальный номер версии диаграммы |
| <i>Page</i> | Номер страницы, может использоваться как номер страницы при формировании палки |

Значения полей каркаса задаются в диалоге *Diagram Properties* (меню *Diagram/Diagram Properties*).

Диаграммы дерева узлов

Диаграмма дерева узлов показывает иерархию работ в модели и позволяет рассмотреть всю модель целиком, но не показывает взаимосвязи между работами (стрелки). Процесс создания модели работ является итерационным, работы могут менять свое расположение в дереве узлов многократно. Чтобы не запутаться и проверить способ декомпозиции, следует после каждого изменения создавать диаграмму дерева узлов. По умолчанию нижний уровень декомпозиции показывается в виде списка, остальные работы – в виде прямоугольников. Для отображения всего дерева в виде прямоугольников следует выключить опцию *Bullet Last Level*. При создании дерева узлов следует указать имя диаграммы, поскольку, если в нескольких диаграммах в качестве корня на дереве узлов использовать одну и ту же работу. Все эти диаграммы получают одинаковый номер (номер узла + постфикс *N*, например *AON*) и в списке открытых диаграмм (пункт меню *Window*) их можно будет различить только по имени.

Диаграммы декомпозиции FEO

Диаграммы "только для экспозиции" (*FEO*) часто используются в модели для иллюстрации других точек зрения, для отображения отдельных деталей, которые не поддерживаются явно синтаксисом *IDEF0*. Диаграммы *FEO* позволяют нарушить любое синтаксическое правило поскольку, по сути, являются просто картинками – копиями стандартных диаграмм и не включаются в анализ синтаксиса. Но если *FEO* используется для иллюстрации альтернативных точек зрения (альтернативный контекст), рекомендуется придерживаться синтаксиса *IDEF0*. Для создания диаграммы *FEO* следует выбрать пункт меню *Diagram/Add FEO diagram*. В возникающем диалоге *Add New FEO Diagram* следует указать имя диаграммы *FEO* и тип родительской диаграммы. Новая диаграмма получает номер, который генерируется автоматически (номер родительской диаграммы по узлу + постфикс *F*, например *AIF*).

Порядок выполнения работы

1. Выберите меню *Diagram/Add Note Tree*. В первом диалоге *Node Tree Wizard* внесите имя диаграммы, укажите корневую диаграмму дерева (по

умолчанию – родительская работа текущей диаграммы) и количество уровней (рис. 2.20).

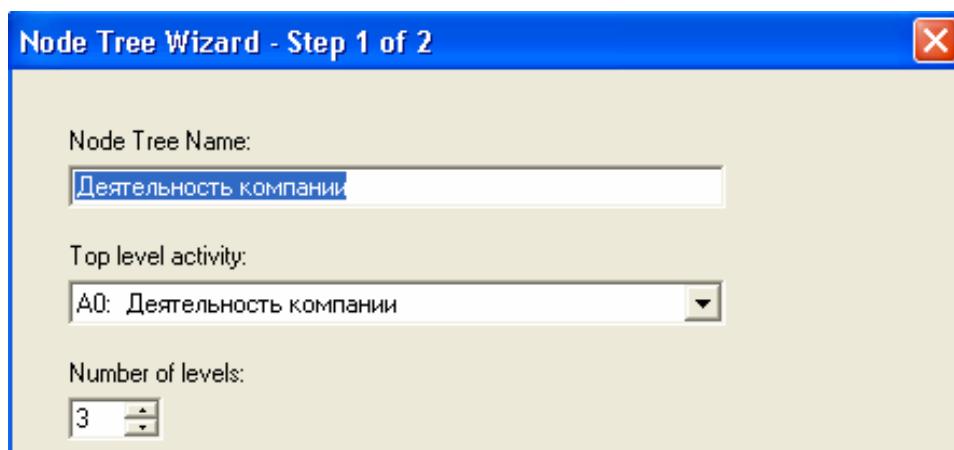


Рис. 2.20. Первый диалог помощника *Node Tree Wizard*

2. Во втором диалоге выберите опции, заданные по умолчанию (рис. 2.21).

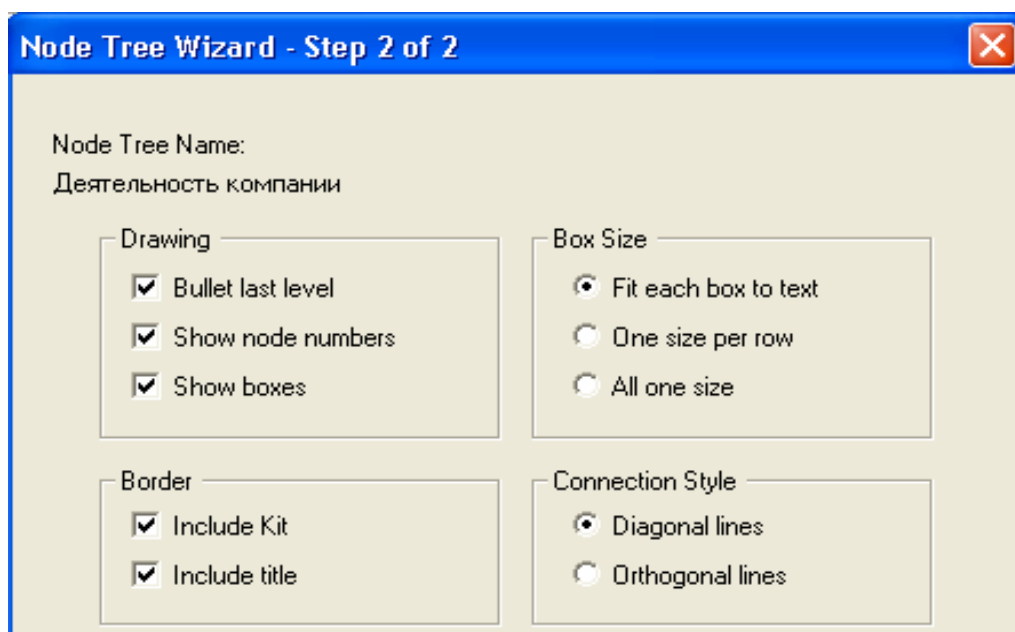


Рис. 2.21. Второй диалог помощника *Node Tree Wizard*

3. Щелкните "Готово", создается диаграмма дерева узлов. Результат приведен на рис. 2.22.

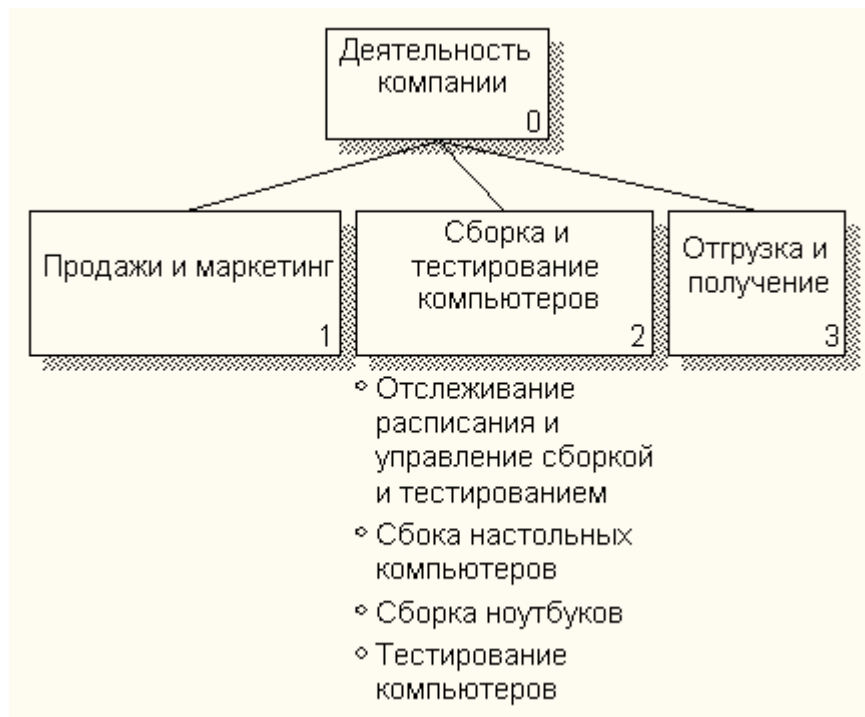


Рис. 2.22. Диаграмма дерева узлов

4. Создайте новую диаграмму дерева узлов. Диаграмму дерева узлов можно модифицировать. Нижний уровень может быть отображен не в виде списка, а в виде прямоугольников, так же как и верхние уровни. Для модификации диаграммы правой кнопкой мыши щелкните по свободному месту, не занятому объектами, выберите меню *Node Tree Diagram Properties* и во вкладке *Style* диалога *Node Tree Properties* отключите опцию *Bullet Last Level*. Результат приведен на рис. 2.23.

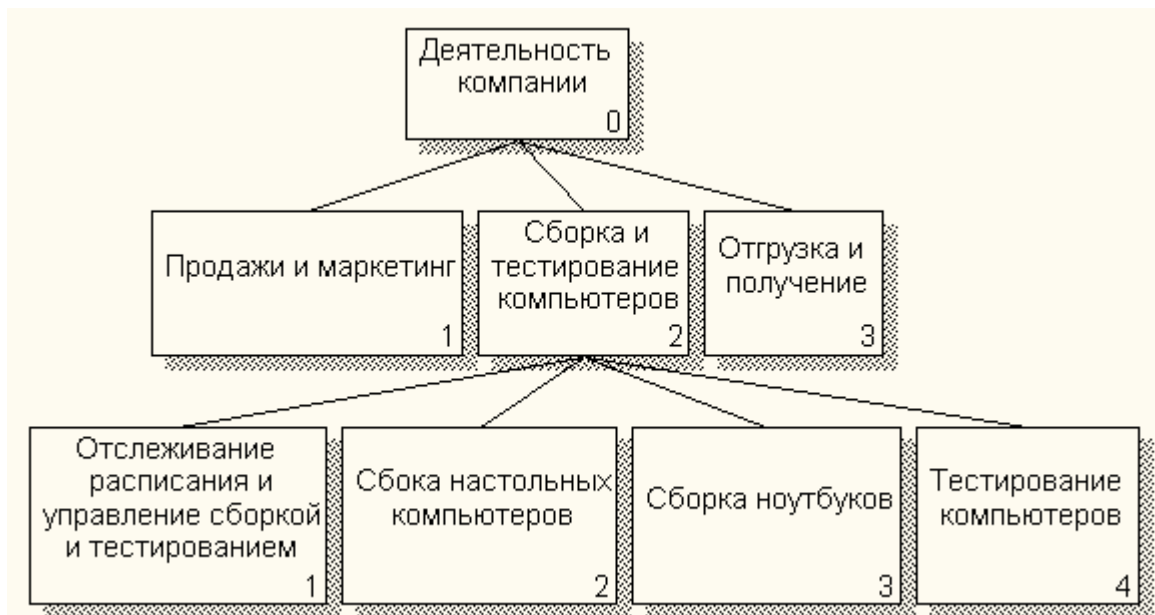


Рис. 2.23. Модифицированная диаграмма дерева узлов

5. Детально рассмотрим взаимодействие работы “Сборка и тестирование компьютеров” с другими работами. Создадим *FEO*-диаграмму, на которой будут только стрелки работы “Сборка и тестирование компьютеров”. Выберите пункт меню *Diagram/Add FEO Diagram*.

6. В диалоге *Add New FEO Diagram* выберите тип *Decomposition Diagram* и внесите имя “Моя диаграмма” диаграммы *FEO*. В качестве имени диаграммы – источника выберите “Сборка и тестирование компьютеров”.

7. Для определения диаграммы перейдите в *Diagram/ Diagram Properties* и во вкладке *Diagram Text* внесите определение: “Моя диаграмма основана на диаграмме Сборка и тестирование компьютеров. Это первая учебная диаграмма”.

8. Удалите некоторые стрелки на диаграмме *FEO* (в диалогах на подтверждение удаления отвечать ДА), как показано на рис. 2.24. эта диаграмма будет являться результатом выполнения данной лабораторной работы.

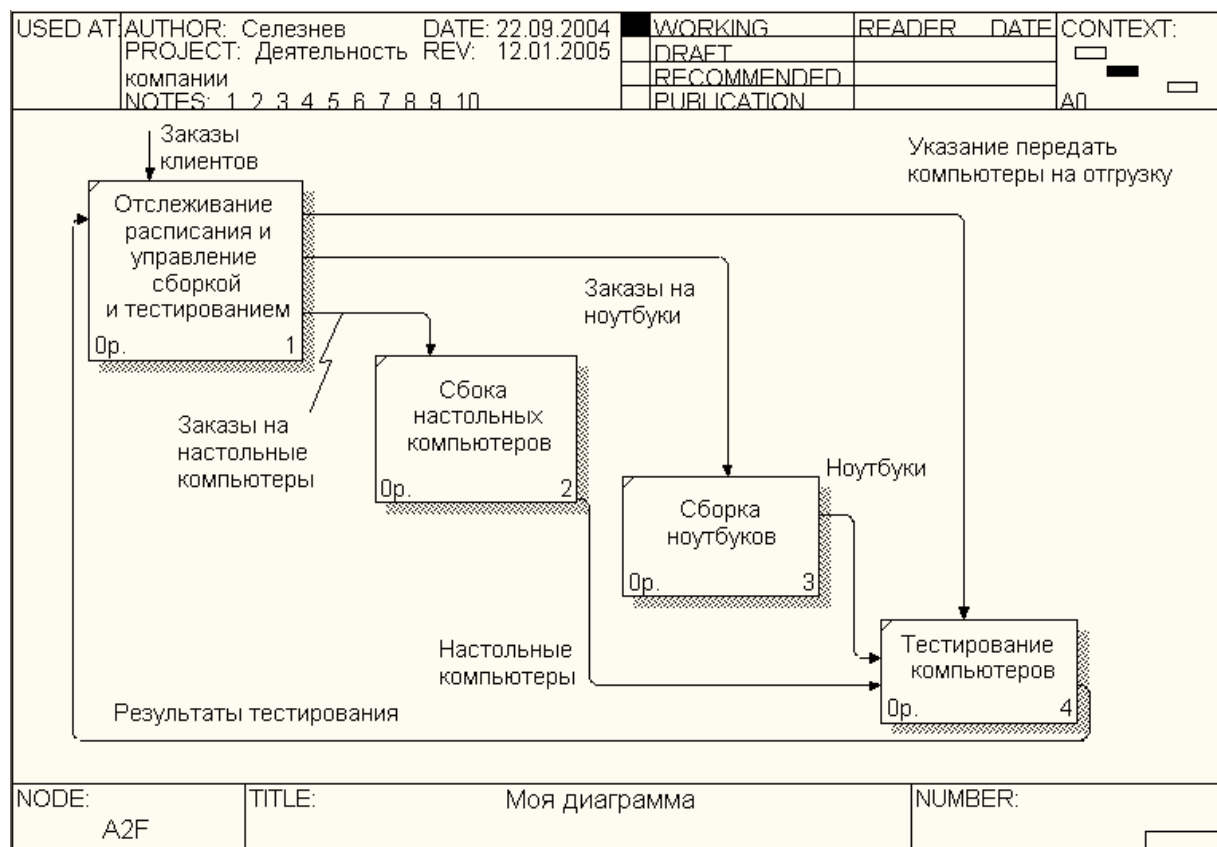


Рис. 2.24. Учебная диаграмма

9. Для перехода между стандартной диаграммой, деревом узлов и *FEO* используйте кнопку на палитре инструментов.

Контрольные вопросы

1. Что такое диаграмма дерева узлов?
2. Что такое *FEO*-диаграмма?
3. Что означает опция *Bullet Last Level*?
4. При дальнейшей работе как будет изменяться диаграмма дерева узлов?
5. Каким образом можно посмотреть созданные диаграммы узлов и *FEO*?

Лабораторная работа №5. Коллективная работа над проектом

Цель работы: Слияние и расщепление моделей. Копирование работ.

Теоретические сведения

Слияние и расщепление моделей

Возможность слияния и расщепления моделей обеспечивает коллективную работу над проектом. Отдельная ветвь модели может быть отщеплена для использования в качестве независимой модели, для доработки или архивирования.

BPwin использует для слияния и разветвления моделей стрелки вызова. Для слияния необходимо выполнить следующие условия:

- Обе сливаемые модели должны быть открыты в **BPwin**;
- Имя модели-источника, которое присоединяют к модели-цели, должно совпадать с именем стрелки вызова работы в модели-цели;
- Стрелка вызова должна исходить из недекомпозируемой работы (работа должна иметь диагональную черту в левом верхнем углу);
- Имена контекстной работы подсоединяемой модели-источника и работы на модели-цели, к которой мы подсоединяем модель-источник, должны совпадать;
- Модель-источник должна иметь, по крайней мере, одну диаграмму декомпозиции.

Для слияния моделей нужно щелкнуть правой кнопкой мыши по работе со стрелкой вызова в модели-цели и во всплывающем меню выбрать пункт *Merge Model*.

При слиянии моделей объединяются словари стрелок и работ. В случае одинаковых определений возможна перезапись определений или принятие определений из модели-источника. То же относится к именам стрелок, хранилищам данных и внешним ссылкам. (Хранилища данных и внешние ссылки – объекты диаграмм потоков данных, *DFD*, будут рассмотрены ниже).

После подтверждения слияния (кнопка *OK*) модель-источник подсоединяется к модели-цели, стрелка вызова исчезает, а работа, от которой отходила стрелка вызова, становится недекомпозируемой – к ней подсоединяется диаграмма декомпозиции первого уровня модели-источника. Стрелки, касающиеся работы на диаграмме модели-цели, автоматически не мигрируют в декомпозицию, а отображаются как неразрешенные. Их следует тоннелировать вручную.

В процессе слияния модель-источник остается неизменной и к модели-цели подключается фактически ее копия. Если в дальнейшем модель-источник будет редактироваться, эти изменения автоматически не попадут в соответствующую ветвь модели-цели.

Разделение моделей производится аналогично. Для отщепления ветви от модели следует щелкнуть правой кнопкой мыши по декомпозированной

работе (работа не должна иметь диагональной черты в левом верхнем углу) и выбрать во всплывающем меню пункт *Split Model*. В появившемся диалоге *Split Options* следует указать имя создаваемой модели. После подтверждения расщепления в старой модели работа станет недекомпозированной (признак – диагональная черта в левом верхнем углу), будет создана стрелка вызова, причем ее имя будет совпадать с именем новой модели, и, наконец, будет создана новая модель, причем имя контекстной работы будет совпадать с именем работы, от которой была "оторвана" декомпозиция.

Порядок выполнения работы

1. Перейдите на диаграмму *АО* и щелкните правой кнопкой мыши по работе "Отгрузка и получение". В контекстном меню выберите *Split Model*. В появившемся диалоге *Split Option* установите опцию *Enable Merge/Overwrite Option*, внесите имя новой модели – "Отгрузка и получение" и щелкните по *OK*. Обратите внимание, что у работы "Отгрузка и получение" появилась стрелка вызова. **BPwin** создал также новую модель "Отгрузка и получение" (рис. 2.25).

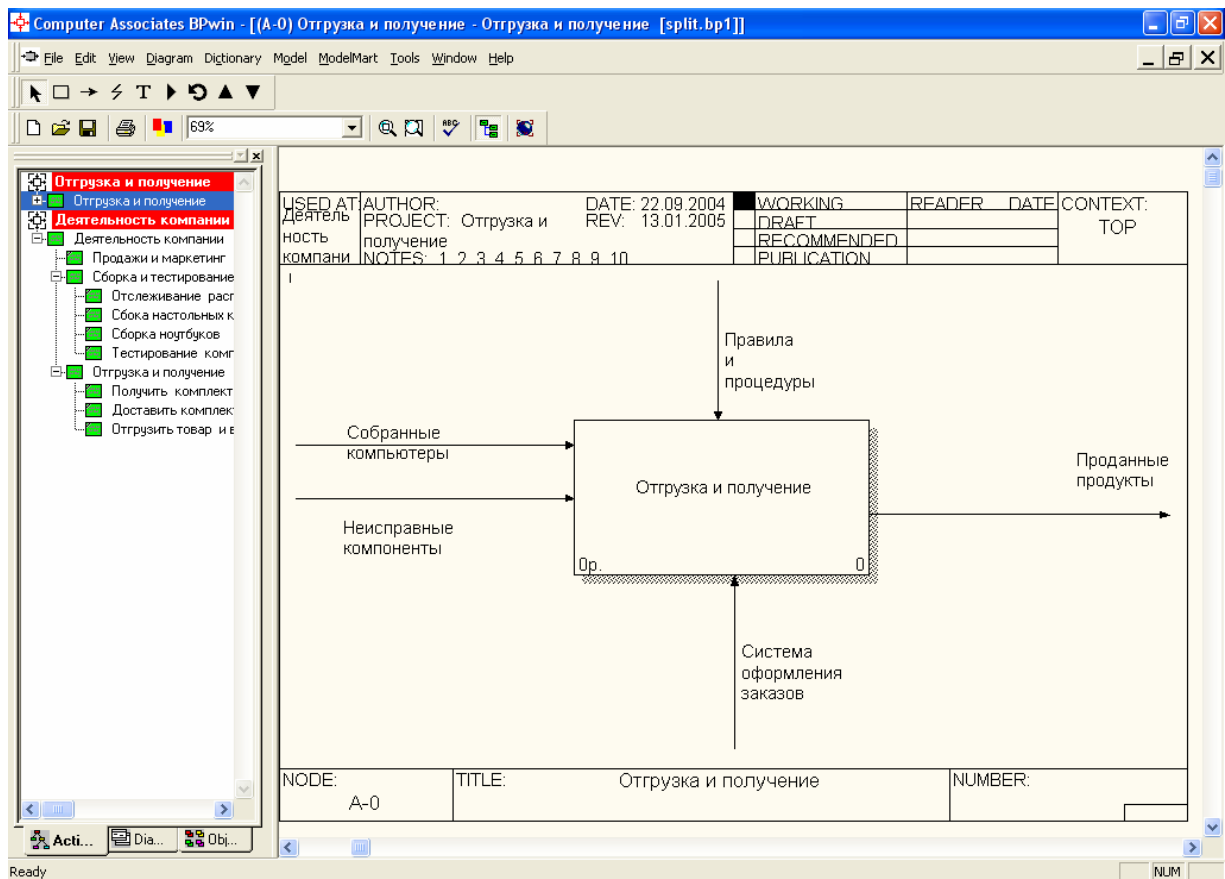


Рис. 2.25. Создание новой модели "Отгрузка и получение"

2. Внесите свойства новой модели:
 - *Time Frame*: AS-IS;
 - *Purpose*: Документировать работу "Отгрузка и получение";
 - *Viewpoint*: Начальник отдела;
 - *Definition*: Модель создается для иллюстрации возможности **BPwin** по расщеплению и слиянию моделей;
 - *Scope*: Работы по получению комплектующих и отправке готовой продукции.
3. Сохраните новую модель, например "*split.bp1*".
4. Декомпозируйте контекстную работу на 3 работы по методологии *IDEF0* (табл. 2.8).

Таблица 2.8

Декомпозиция работы "Отгрузка и получение"

| <i>Activity Name</i> | <i>Activity Definition</i> |
|---------------------------|--|
| Получить комплектующие | Физически получить комплектующие и сделать соответствующие записи в информационной системе |
| Доставить комплектующие | Доставить комплектующие сборщикам и тестерам |
| Отгрузить товар и возврат | Отгрузить товар клиентам и неисправные компоненты (возврат) поставщикам |

5. Свяжите граничные стрелки, как показано на рис. 2.26.

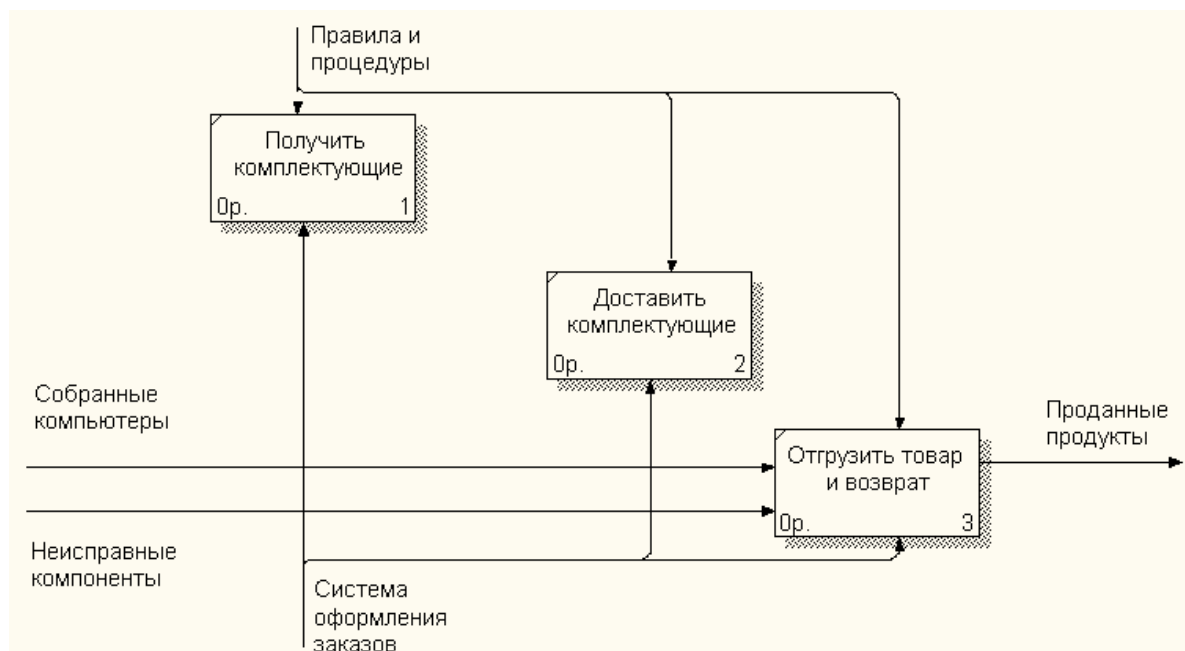


Рис. 2.26. Внутренние стрелки на диаграмме "Отгрузка и получение"

6. Внесите следующие внутренние и граничные стрелки (табл. 2.9).

Внутренние и граничные стрелки на декомпозиции работы
"Отгрузка и получение"

| <i>Arrow Name</i> | <i>Arrow Definition</i> |
|--------------------------|--|
| Возврат поставщику | Неисправные компоненты |
| Компоненты | Выберите название из списка (словаря) |
| Компоненты от поставщика | Компоненты от поставщика |
| Проверенные компоненты | Проверенные и подготовленные для передачи сборщикам и тестировщикам компоненты |

Результат добавления стрелок показан на рис. 2.27

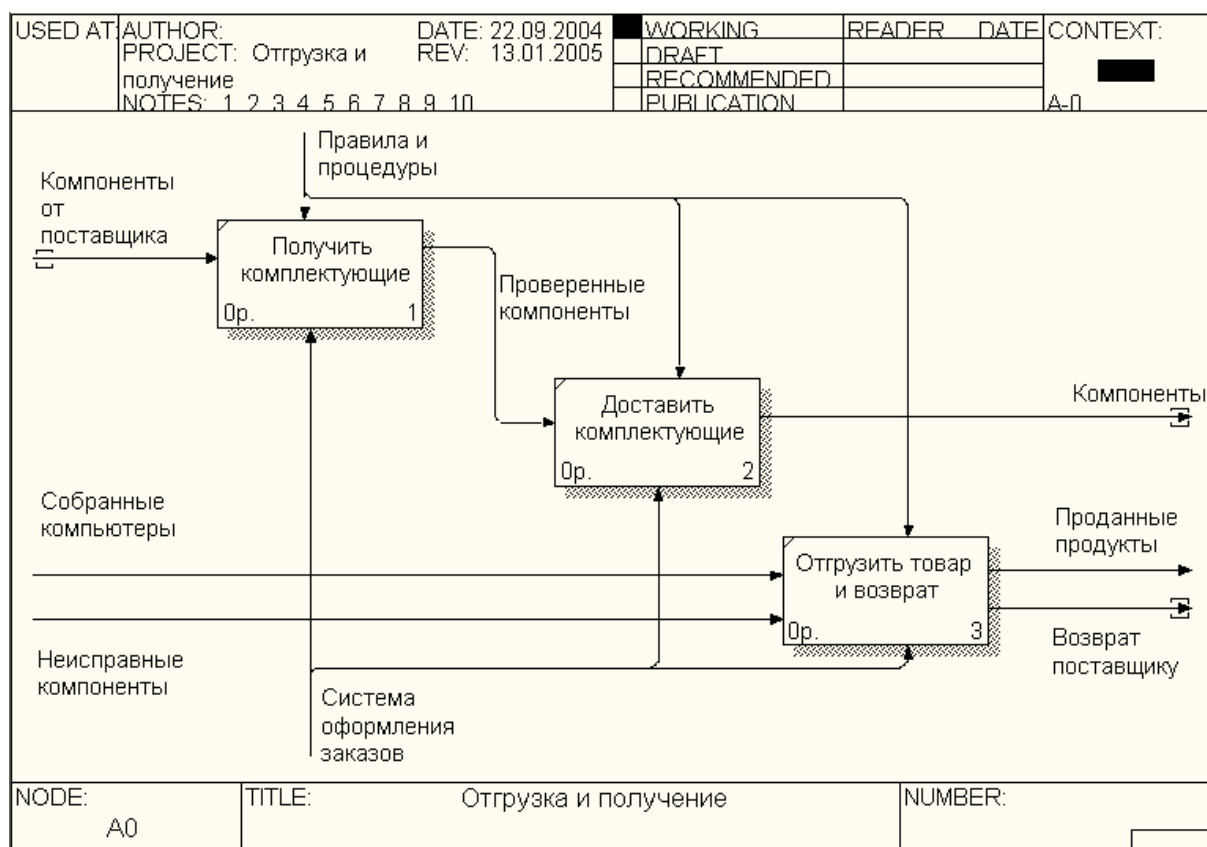


Рис. 2.27. Внутренние и граничные стрелки на диаграмме
"Отгрузка и получение"

7. Тоннелируйте граничные стрелки (*Resolve Border Arrow*). Диаграмма декомпозиции показана на рис. 2.28. Родительская диаграмма "Отгрузка и получение" изображена на рис. 2.29.

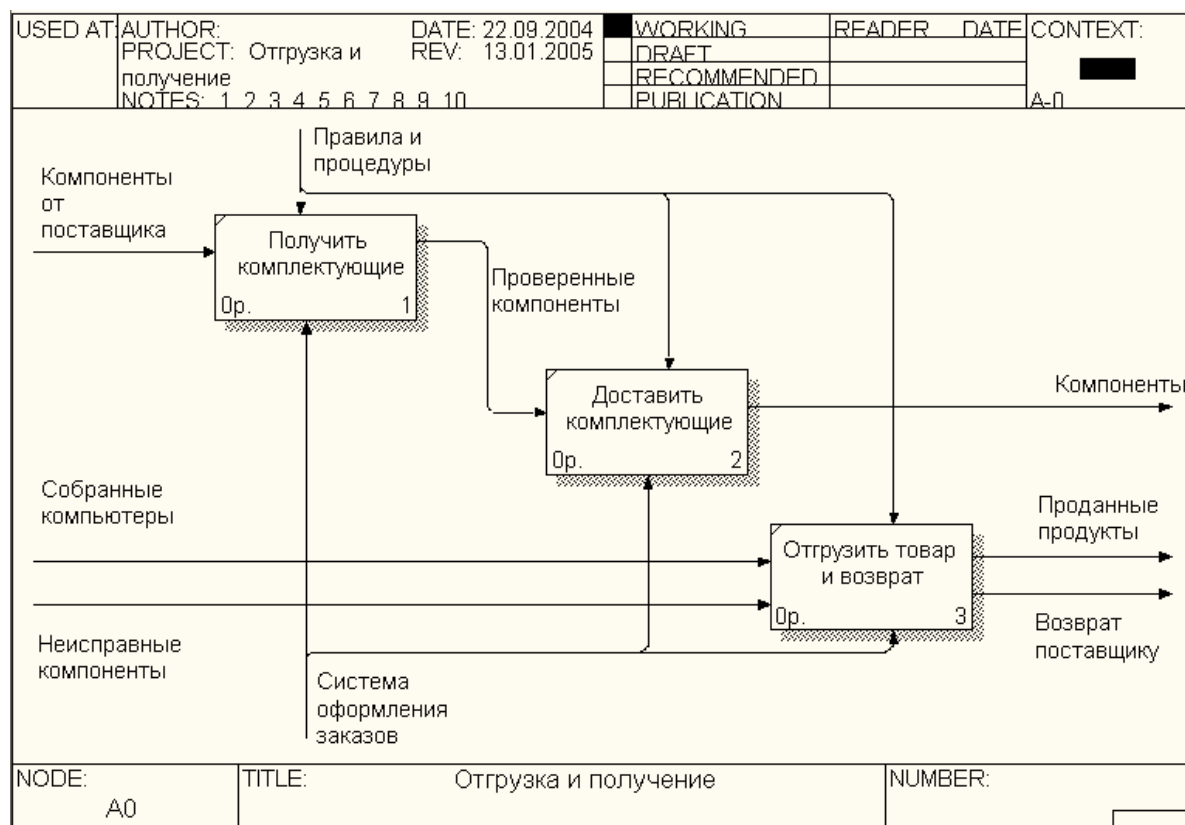


Рис. 2.28. Диаграмма декомпозиции работы "Отгрузка и получение"

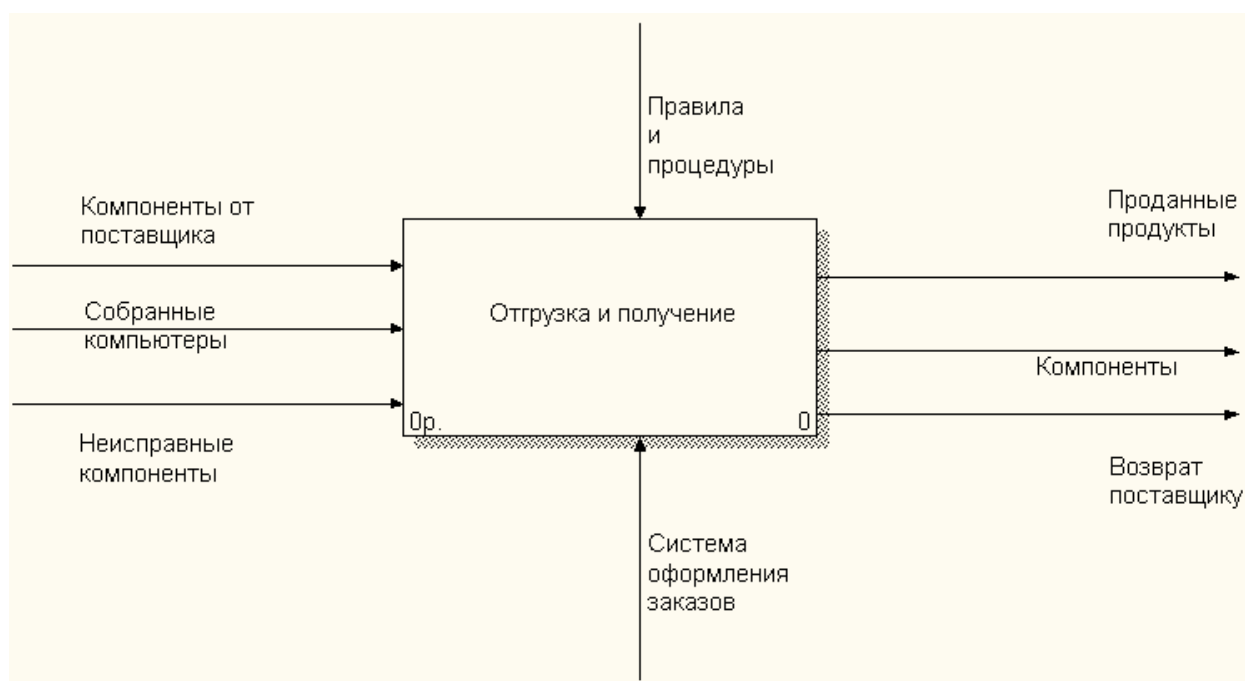


Рис. 2.29. Родительская диаграмма "Отгрузка и получение"

8. Перейдите в модель "Деятельность компании". На диаграмме А0 щелкните правой кнопкой мыши по работе "Отгрузка и получение". В контекстном меню выберите *Merge Model*. В появившемся диалоге *Merge Model* установите опцию *Cut/Paste entire dictionaries* и щелкните по *OK*.

Обратите внимание, что у работы "Отгрузка и получение" исчезла стрелка вызова и появилась новая декомпозиция.

9. На диаграмме A0 тоннелируйте (*Resolve Border Arrow*) и свяжите стрелки как показано на рис. 2.30.

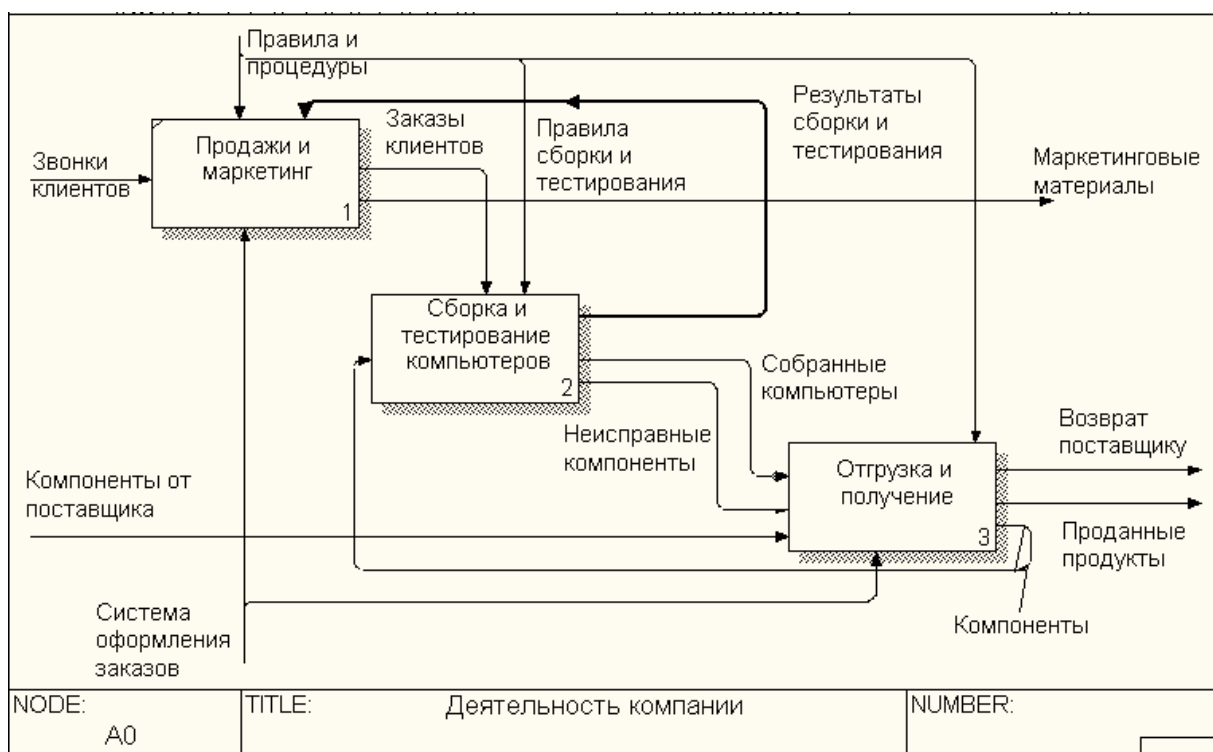


Рис. 2.30. Диаграмма "Деятельность компании" после слияния моделей

10. В связи с появлением стрелки входа "Компоненты" у работы "Сборка и тестирование компьютеров", удалите несвязанную стрелку "Компоненты" и ее ветви на диаграмме A2 "Сборка и тестирования компьютеров" и соответственно продолжите стрелку I1 "Компоненты". Проверьте, является ли стрелка "Компоненты" стрелкой выхода (O) на диаграмме A3 "Отгрузка и получение".

11. Копирование работ в другую модель. Создайте новую модель "ТЕСТ". Декомпозируйте контекстную работу в новой модели, но не вносите имена работ. Переключите *Model Explorer* во вкладку *Activity*. С помощью *drag&drop* перенесите какую-нибудь работу из модели "Деятельность компании" на диаграмму декомпозиции модели "ТЕСТ" (выделите работу для копирования, нажмите *ctrl* и, удерживая левую клавишу мыши на пиктограмме работы, перенесите работу в нужное место на диаграмме). В появившемся диалоге "*Continue with Merge?*" установите опцию *Cut/Paste entire dictionaries* и щелкните по *OK*. В результате работа из модели "Деятельность компании" копируется на новую диаграмму модели "ТЕСТ".

12. Перемещение работ в той же самой модели. Щелкните по работе в модели "ТЕСТ" и переместите работу на нужное место на другой диаграм-

ме. В появившемся диалоге *Continue with Merge?* щелкните *OK*. В результате работа переносится из одной диаграммы на другую.

Контрольные вопросы

1. Как произвести слияние и расщепление моделей?
2. Каким образом можно скопировать работу?
3. Какое надо задать имя новой модели при расщеплении?
4. Что означает опция *Cut/Paste entire dictionaries?*
5. Какие условия необходимо выполнить для слияния моделей?

Лабораторная работа №6. Методология IDEF3

Цель работы: Создание диаграммы IDEF3

Теоретические сведения

Метод описания процессов IDEF3

Методология построения моделей IDEF3, называемая также *Workflow diagramming* – методологией моделирования необходима для описания логики взаимодействия информационных потоков. Данная методология использует графическое описание информационных потоков, взаимоотношений между процессами обработки информации и объектов, являющихся частью этих процессов. Диаграммы *Workflow* могут быть применены в моделировании бизнес-процессов для анализа завершенности процедур обработки информации. С их помощью можно описывать сценарии действий сотрудников организации, например последовательность обработки заказа или события, которые необходимо обработать за конечное время. Каждый сценарий сопровождается описанием процесса и может быть использован для документирования каждой функции.

IDEF3 – это метод, имеющий основной целью дать возможность аналитикам описать ситуацию, когда процессы выполняются в определенной последовательности, а также описать объекты, участвующие совместно в одном процессе. Техника описания набора данных IDEF3 является частью структурного анализа. В отличие от некоторых методик описаний процессов IDEF3 не ограничивает аналитика чрезмерно жесткими рамками синтаксиса, что может привести к созданию неполных или противоречивых моделей. IDEF3 может быть также использован как метод создания процессов. Каждая работа в IDEF3 описывает какой-либо сценарий бизнес-процесса и может являться составляющей другой работы. Поскольку сценарий описывает цель и рамки модели, важно, чтобы работы именовались отглагольным существительным, обозначающим процесс действия, или фразой, содержащей такое существительное.

Точка зрения на модель, цель модели – те вопросы, на которые призвана ответить модель, – должны быть задокументированы.

Диаграммы. Диаграмма является основной единицей описания в IDEF3. Важно правильно построить диаграммы, поскольку они предназначены для чтения другими людьми (а не только автором).

Единицы работы – *Unit of Work (UOW)*. UOW, также называемые работами (*activity*), являются центральными компонентами модели. В IDEF3 работы изображаются прямоугольниками с прямыми углами и имеют имя, выраженное отглагольным существительным, обозначающим процесс действия, одиночным или в составе фразы, и номер (идентификатор); другое имя существительное в составе той же фразы обычно отображает основной выход (результат) работы (например, "Изготовление изделия"). Часто имя существительное в имени работы меняется в процессе моделирования, по-

скольку модель может уточняться и редактироваться. Идентификатор работы присваивается при создании и не меняется никогда. Даже если работа будет удалена, ее идентификатор не будет вновь использоваться для других работ. Обычно номер работы состоит из номера родительской работы и порядкового номера на текущей диаграмме.

Связи. Связи показывают взаимоотношения работ. Все связи в *IDEF3* однонаправлены и могут быть направлены куда угодно, но обычно диаграммы *IDEF3* стараются построить так, чтобы связи были направлены слева направо. В *IDEF3* различают три типа стрелок, изображающих связи, стиль которых устанавливается через меню *Arrow Properties*:

Старшая (*Precedence*) – сплошная линия, связывающая единицы работ (*UOW*). Рисуеться слева направо или сверху вниз. Показывает, что работа-источник должна закончиться прежде, чем работа-цель начнется.

Отношения (*Relational Link*) – пунктирная линия, использующаяся для изображения связей между единицами работ (*UOW*) а также между единицами работ и объектами ссылок.

Потоки объектов (*Object Flow*) – стрелка с двумя наконечниками, применяется для описания того факта, что объект используется в двух или более единицах работы, например, когда объект порождается в одной работе и используется в другой.




Старшая связь и поток объектов. Старшая связь показывает, что работа-источник заканчивается ранее, чем начинается работа-цель.

Отношение показывает, что стрелка является альтернативой старшей стрелке или потоку объектов в смысле задания последовательности выполнения работ – работа-источник не обязательно должна закончиться, прежде чем работа-цель начнется. Работа-цель может закончиться прежде, чем закончится работа-источник.

Перекрестки (*Junction*). Окончание одной работы может служить сигналом к началу нескольких работ, или же одна работа для своего запуска может ожидать окончания нескольких работ. Перекрестки используются для отображения логики взаимодействия стрелок при слиянии и разветвлении или для отображения множества событий, которые могут или должны быть завершены перед началом следующей работы. Различают перекрестки для слияния (*Fan-in Junction*) и разветвления (*Fan-out Junction*) стрелок. Перекресток не может использоваться одновременно для слияния и для разветвления. Смысл каждого типа перекрестков приведен в табл. 2.10.

Все перекрестки на диаграмме нумеруются, каждый номер имеет префикс *J*. Можно редактировать свойства перекрестка при помощи диалога *Definition Editor*. В отличие от *IDEF0* и *DFD* в *IDEF3* стрелки могут сливаться и разветвляться только через перекрестки.

Типы перекрестков

| Соединение | Имя | Значение <i>Fan-in</i> | Значение <i>Fan-out</i> |
|---|---------------------------|--|--|
|  | Асинхронное <i>AND</i> | Все последующие процессы должны быть полными | Все предшествующие процессы должны быть полными |
|  | Синхронное <i>AND</i> | Все следующие процессы обрабатываются одновременно комплексно | Все предшествующие процессы начинают обрабатываться одновременно |
|  | Асинхронное <i>OR</i> | Один или более предшествующих процессов должны быть завершены | Один или более последующих процессов должны быть начаты |
|  | Синхронное <i>OR</i> | Один или более предшествующим процессам завершаются одновременно | Один или более последующих процессов начинаются одновременно |
|  | <i>XOR</i> | Ровно один предшествует завершающемуся процессу | Ровно один следует за началом процесса |

Объект ссылки. Объект ссылки в *IDEF3* выражает некую идею, концепцию или данные, которые нельзя связать со стрелкой, перекрестком или работой. Объект ссылки изображается в виде прямоугольника, похожего на прямоугольник работы. Имя объекта ссылки задается в диалоге *Referent* (пункт всплывающего меню *Name Editor*), в качестве имени можно использовать имя какой-либо стрелки с других диаграмм или имя сущности из модели данных. Объекты ссылки должны быть связаны с единицами работ или перекрестками пунктирными линиями. Официальная спецификация *IDEF3* различает три стиля объектов ссылок – безусловные (*unconditional*), синхронные (*synchronous*) и асинхронные (*asynchronous*). **BPwin** поддерживает только безусловные объекты ссылок. Синхронные и асинхронные объекты ссылок, используемые в диаграммах переходов состояний объектов, не поддерживаются.

При внесении объектов ссылок помимо имени следует указывать тип объекта ссылки. Типы объектов ссылок приведены в табл. 2.11.

Типы объектов ссылок

| Тип объекта ссылки | Цель описания |
|-------------------------------|--|
| <i>OBJECT</i> | Описывает участие важного объекта в работе |
| <i>GOTO</i> | Инструмент циклического перехода (в повторяющейся последовательности работ), возможно на текущей диаграмме, но не обязательно. Если все работы цикла присутствуют на текущей диаграмме, цикл может также изображаться стрелкой, возвращающейся на стартовую работу: <i>GOTO</i> может ссылаться на перекресток |
| <i>UOB (Unit of behavior)</i> | Применяется, когда необходимо подчеркнуть множественное использование какой-либо работы, но без цикла. Например, работа "Контроль качества" может быть использована в процессе "Изготовления изделия" несколько раз, после каждой единичной операции. Обычно этот тип ссылки не используется для моделирования автоматически запускающихся работ |
| <i>NOTE</i> | Используется для документирования важной информации, относящейся к каким-либо графическим объектам на диаграмме. <i>NOTE</i> является альтернативой внесению текстового объекта в диаграмму |
| <i>ELAB (Elaboration)</i> | Используется для усовершенствования графиков или их более детального описания. Обычно употребляется для детального описания разветвления и слияния стрелок на перекрестках |

Декомпозиция работ. В *IDEF3* декомпозиция используется для детализации работ. Методология *IDEF3* позволяет декомпонировать работу многократно, т.е. работа может иметь множество дочерних работ. Это позволяет в одной модели описать альтернативные потоки. Возможность множественной декомпозиции предъявляет дополнительные требования к нумерации работ. Так, номер работы состоит из номера родительской работы, версии декомпозиции и собственного номера работы на текущей диаграмме.

Рассмотрим процесс декомпозиции диаграмм *IDEF3*, включающий взаимодействие автора (аналитика) и одного или нескольких экспертов предметной области:

Описание сценария, области и точки зрения. Перед проведением сеанса экспертизы у экспертов предметной области должны быть задокументированы сценарии и рамки модели для того, чтобы эксперт мог понять цели декомпозиции. Если точка зрения моделирования отличается от точки зрения эксперта, она должна быть особенно тщательно задокументирована.

Возможно, что эксперт самостоятельно не сможет передать необходимую информацию. В этом случае аналитик должен приготовить список вопросов для проведения интервью.

Определение работ и объектов. Обычно эксперт предметной области передает аналитику текстовое описание сценария. В дополнение к этому может существовать документация, описывающая интересующие процессы. Из всей этой информации аналитик должен составить список кандидатов на работы (отглагольные существительные, обозначающие процесс, одиночные или в составе фразы) и кандидатов на объекты (существительные, обозначающие результат выполнения работы), которые необходимы для перечисленных в списке работ.

В некоторых случаях целесообразно создать графическую модель для представления ее эксперту предметной области. Графическая модель может быть также создана после сеанса сбора информации для того, чтобы детали форматирования диаграммы не смущали участников.

Поскольку разные фрагменты модели *IDEF3* могут быть созданы разными группами аналитиков в разное время, *IDEF3* поддерживает простую схему нумерации работ в рамках всей модели. Разные аналитики оперируют разными диапазонами номеров, работая при этом независимо. Пример выделения диапазона приведен в табл. 2.12.

Таблица 2.12

Диапазоны номеров работ

| Аналитик | Диапазон номеров <i>IDEF3</i> |
|----------|-------------------------------|
| Иванов | 1-999 |
| Петров | 1000-1999 |
| Сидоров | 2000-2999 |

Последовательность и согласование. Если диаграмма создается после проведения интервью, аналитик должен принять некоторые решения, относящиеся к иерархии диаграмм, например, сколько деталей включать в одну диаграмму. Если последовательность и согласование диаграмм неочевидны, может быть проведена еще одна экспертиза для детализации и уточнения информации. Важно различать подразумевающее согласование (согласование, которое подразумевается в отсутствие связей) и ясное согласование (согласование, ясно изложенное в мнении эксперта).

Работы, перекрестки и документирование объектов. *IDEF3* позволяет внести информацию в модель различными способами. Например, логика взаимодействия может быть отображена графически в виде комбинации перекрестков. Та же информация может быть отображена в виде объекта ссылки типа *ELAB* (*Elaboration*). Это позволяет аналитику вносить информацию в удобном в данный момент времени виде. Важно учитывать, что модели могут быть реорганизованы, например, для их представления в более презентабельном виде. Выбор формата для презентации часто имеет важное значение для организации модели, поскольку комбинация перекре-

стков занимает значительное место на диаграмме и использование иерархии перекрестков затрудняет расположение работ на диаграмме.

В результате дополнения диаграмм *IDEF0* диаграммами *IDEF3* может быть создана смешанная модель, которая наилучшим образом описывает все стороны деятельности предприятия. Иерархию работ в смешанной модели можно увидеть в окне *Model Explorer*. Работы в нотации *IDEF0* изображаются зеленым цветом, *IDEF3* – желтым.

Имитационное моделирование

Имитационное моделирование – это метод, позволяющий строить модели, учитывающие время выполнения функций. Полученную модель можно "проиграть" во – времени и получить статистику происходящих процессов так, как это было бы в реальности. В имитационной модели изменения процессов и данных ассоциируются с событиями. "Проигрывание" модели заключается в последовательном переходе от одного события к другому. Обычно имитационные модели строятся для поиска оптимального решения в условиях ограничения по ресурсам, когда другие математические модели оказываются слишком сложными.

Связь между имитационными моделями и моделями процессов заключается в возможности преобразования модели процессов в неполную имитационную модель. Имитационная модель дает больше информации для анализа системы, в свою очередь результаты такого анализа могут стать причиной модификации модели процессов.

Имитационная модель включает следующие основные элементы:

- Источники и цели (*Bourses u Destinations*). Источники – это элементы, от которых в модель поступает информация или объекты. По смыслу они близки к "объект ссылки" на диаграммах *IDEF3*. Скорость поступления данных или объектов от источника обычно задается статистической функцией. Цель – это устройство для приема информации или объектов.
- Очереди (*Queues*) – это место, где объекты ожидают обработки. Времена обработки объектов (производительность) в разных работах могут быть разными. В результате перед некоторыми работами могут накапливаться объекты, ожидающие своей очереди. Часто целью имитационного моделирования является минимизация количества объектов в очередях. Тип очереди в имитационной модели может быть конкретизирован. Очередь может быть похожа на стек – пришедшие последними в очередь объекты первыми отправляются на дальнейшую обработку (*LIFO: last-in-first-out*). Альтернативой стеку, может быть, последовательная обработка, когда первыми на дальнейшую обработку отправляются объекты, пришедшие первыми (*FIFO: first -in-first-out*). Могут быть заданы и более сложные алгоритмы обработки очереди.

- Оборудование (*Facilities*). Оборудование – это аналог работ в модели процессов. В имитационной модели может быть задана производительность оборудования.

BPwin не имеет собственных инструментов, позволяющих создавать имитационные модели, однако можно экспортировать модель *IDEF3* в специализированное средство создания таких моделей – *BPSimulator 3.0*.

Для экспорта модели в *BPSimulator* необходимо настроить *ODBC*-источник и подготовить модель к экспорту. Для подготовки модели необходимо настроить свойства, определяемые пользователем *UDP*, специально включенные в **BPwin** для целей экспорта. Задание соответствующих *UDP* позволяет автоматически установить значения и свойства объектов имитационной модели в *BPSimulator*.

Для экспорта модели *IDEF3* в *BPSimulator* следует выбрать меню *File/Export* в *BPSimulator*. Экспорт осуществляется через файл *MS Excel* (.xls). Для импорта данных в *BPSimulator* необходимо открыть новую модель и импортировать соответствующий файл.

Порядок выполнения работы

1. Перейдите на диаграмму *A2* и декомпозируйте работу "Сборка настольных компьютеров". В диалоге *Activity Box Count* (рис. 2.31) установите число работ 7 и нотацию *IDEF3*.

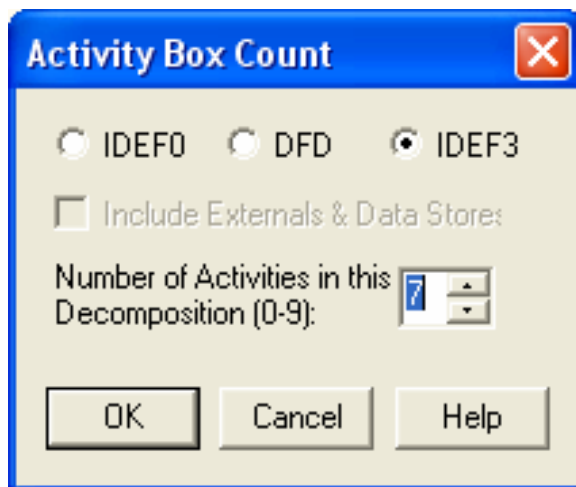


Рис. 2.31. Выбор нотации *IDEF3* в диалоге *Activity Box Count*

Возникает диаграмма *IDEF3*, содержащая работы (*UOW*). Правой кнопкой мыши щелкните по работе, выберите в контекстном меню *Name* и внесите имя работы "Подготовка компонентов". Затем во вкладке *Definition* внесите определение "Подготавливаются все компоненты компьютера согласно спецификации заказа".


2. Во вкладке *UOW* внесите свойства работы (табл. 2.13).

Свойства *UOW*


| | |
|-------------------|--|
| <i>Objects</i> | Компоненты: винчестеры, корпуса, материнские платы, видеокарты, звуковые карты, дисководы <i>CD-ROM</i> и флоппи, модемы, программное обеспечение. |
| <i>Facts</i> | Доступные операционные системы: <i>Windows 98</i> , <i>Windows NT</i> , <i>Windows 2000</i> |
| <i>Constrains</i> | Установка модема требует установки дополнительного программного обеспечения |

3. Внесите в диаграмму еще 6 работ:

- Установка материнской платы и винчестера;
- Установка модема;
- Установка *CD-ROM*;
- Установка флоппи-дисковода;
- Инсталляция операционной системы;
- Инсталляция дополнительного программного обеспечения.

4. С помощью кнопки  палитры инструментов создайте объект ссылки. Внесите имя объекта внешней ссылки "Компоненты" (выбрать из существующих стрелок). Свяжите стрелкой объект ссылки и работу "Подготовка компонентов" (стиль стрелки: ссылка – *Referent*).

5. Свяжите стрелкой работы "Подготовка компонентов" (выход) и "Установка материнской платы и винчестера" (вход). Измените стиль стрелки на *Object Flow*. В *IDEF3* имя стрелки может отсутствовать, хотя *VRwin* показывает отсутствие имени как ошибку.

6. С помощью кнопки  на палитре инструментов внесите два перекрестка типа "асинхронное или" и свяжите работы с перекрестками, как показано на рисунке 2.32.

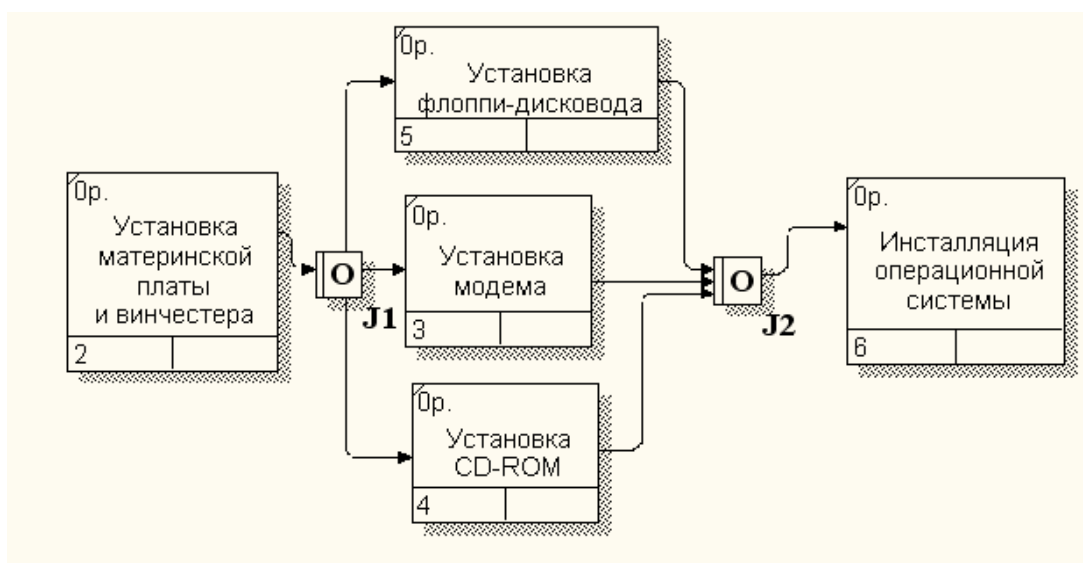


Рис. 2.32. Создание перекрестков на диаграмме

Правой кнопкой щелкните по перекрестку для разветвления (*fan-out*), выберите *Name* и внесите имя "Компоненты, требуемые в спецификации заказа".

7. Создайте два перекрестка типа исключающего "ИЛИ" и свяжите работы; добавьте объект внешней ссылки и дайте ему имя "Программное обеспечение". Для стрелки, соединяющей "Компоненты" и работу "Подготовка компонентов" измените тип стрелы на *Referent*. Для стрелки, соединяющей работу "Инсталляция дополнительного программного обеспечения" и "Программное обеспечение" измените тип стрелы на *Referent*. Для стрелки, соединяющей работу "Инсталляция операционной системы" и "Программное обеспечение" измените тип стрелы на *Referent*. Результат данной лабораторной работы должен быть как на рис. 2.33.

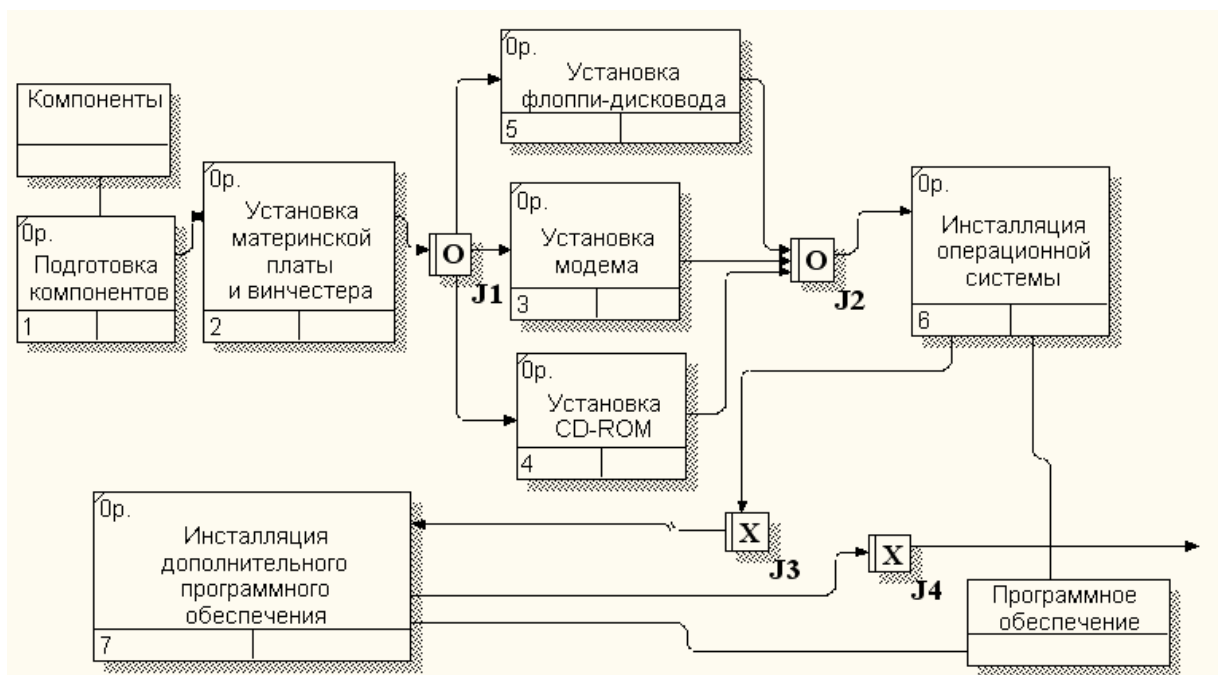


Рис. 2.33. Результат выполнения упражнения 7

8. Выберите пункт меню *Diagram/Add IDEF3 Scenario*. Создайте диаграмму сценария (декомпозиция диаграммы "Сборка настольных компьютеров") на основе диаграммы IDEF3 "Сборка настольных компьютеров" (A22). Присвойте ей имя "Мой сценарий". Отметьте галочкой *Copy contents of source diagram*.

9. Удалите элементы, не входящие в сценарий, как показано на рис. 2.34.

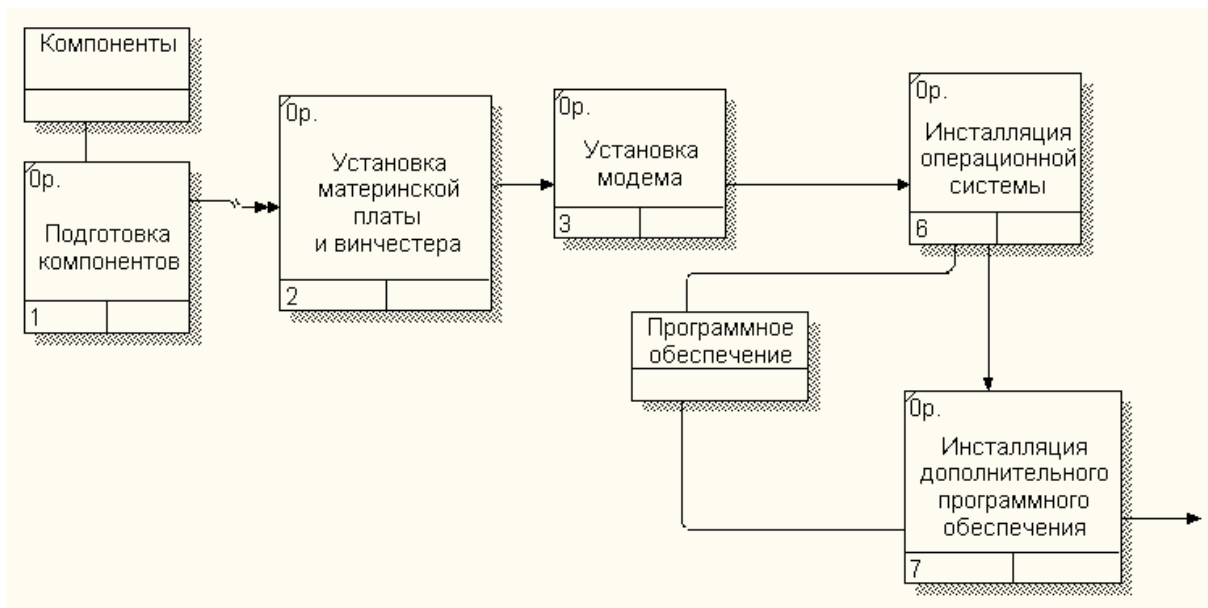


Рис. 2.34. Результат выполнения лабораторной работы №6

Контрольные вопросы

1. Чем отличаются диаграммы *IDEF3* от диаграмм *IDEF0*?
2. Какие бывают виды перекрестков?
3. Что такое объект ссылки?
4. Для чего необходимо строить *IDEF3*-сценарий?
5. Что такое имитационное моделирование?

Лабораторная работа №7. Стоимостной анализ

Цель работы: Стоимостной анализ (ABC) и свойства, определяемые пользователем (UDP).

Теоретические сведения

Создание отчетов в *BPwin*

BPwin имеет мощный инструмент генерации. Отчеты по модели вызываются из пункта меню *Report*. Всего имеется семь типов отчетов:

1. *Model Report*. Отчет включает информацию о контексте модели - имя модели, точку зрения, область, цель, имя автора, дату создания и др.

2. *Diagram Report*. Отчет по конкретной диаграмме. Включает список объектов (работ, стрелок, хранилищ данных, внешних ссылок и т.д.).

3. *Diagram Object Report*. Наиболее полный отчет по модели. Может включать полный список объектов модели (работ, стрелок с указанием их типа и др.) и свойства, определяемые пользователем.

4. *Activity Cost Report*. Отчет о результатах стоимостного анализа.

5. *Arrow Report*. Отчет по стрелкам. Может содержать информацию из словаря стрелок, информацию о работе-источнике, работе-назначении стрелки и информацию о разветвлении и слиянии стрелок.

6. *Data Usage Report*. Отчет о результатах связывания модели процессов и модели данных. (будет рассмотрен ниже.)

7. *Model Consistency Report*. Отчет, содержащий список синтаксических ошибок модели.

Синтаксические ошибки *IDEF0* разделяются на три типа:

- Во-первых, это ошибки, которые *BPwin* выявить не в состоянии. Например, синтаксис *IDEF0* требует, чтобы имя работы было выражено отглагольным существительным или глагольной формой, выражающей действие ("Изготовление изделия", "Обслуживание клиента", "Выписка счета" и т.д.), а имя стрелки также должно быть выражено существительным. *BPwin* не позволяет анализировать синтаксис естественного языка (английского и русского) и смысл имен объектов и поэтому игнорирует ошибки этого типа. Выявление таких ошибок - ручная работа, которая ложится на плечи аналитиков и должна контролироваться руководителем проекта;
- Ошибки второго типа *BPwin* просто не допускает. Например, каждая грань работы предназначена для определенного типа стрелок. *BPwin* просто не позволит создать на диаграмме *IDEF0* внутреннюю стрелку, выходящую из левой грани работы и входящую в правую грань;
- Третий тип ошибок *BPwin* позволяет допустить, но детектирует их. Полный их список можно получить в отчете *Model Consistency Report*. Это единственный неопциональный отчет в *BPwin*. Список ошибок может содержать, например, неименованные работы и стрелки (*unnamed arrow*, *unnamed activity*), несвязанные стрелки (*un-*

connected border arrow), неразрешенные стрелки (*unresolved (square tunneled) arrow connections*), работы, не имеющие, по крайней мере, одной стрелки выхода и одной стрелки управления (*Activity "Сборка блоков" has no Control, Activity "Сборка блоков" has no Output*), и т.д.

При выборе пункта меню, который соответствует какому-либо отчету, появляется диалог настройки отчета. Для каждого из семи типов отчетов он выглядит по-своему.

Раскрывающийся список *Standard Reports* позволяет выбрать один из стандартных отчетов. Стандартный отчет – это запоминаемая комбинация переключателей, флажков и других элементов управления диалога. Для создания собственного стандартного отчета необходимо задать опции отчета, ввести имя отчета в поле списка выбора и щелкнуть по кнопке *New*. **BPwin** сохраняет информацию о стандартном отчете в файле *BPWINRPT.INI*. Все определения этого файла доступны из любой модели. Единственное ограничение – свойства, определяемые пользователем (*User-Defined Properties*). Они сохраняются в виде указателя и поэтому доступны только из "родной" модели. Стандартный отчет можно изменить (кнопка *Update*) или удалить (кнопка *Delete*).

В правом верхнем углу диалога находится группа управляющих элементов для выбора формата отчета. Доступны следующие форматы:

- *Labeled* – отчеты включают метку поля, затем, в следующей строке, печатается содержимое поля;
- *Fixed Column* – каждое поле печатается в собственной колонке;
- *Tab-Comma Delimited* – каждое поле печатается в собственной колонке. Колонки разделяются знаком табуляции или запятыми;
- *DDE Table* – данные передаются по *DDE* приложению, например *MS Word* или *Excel*;
- *RPTwin* – отчет создается в формате *Platinum RPTwin* – специализированного генератора отчетов, который входит в поставку **BPwin**.

Опция *Ordering* (на отчете по стрелкам отсутствует) сортирует данные по какому-либо значению.

Опция *Multi-Valued Format* регулирует вывод полей в отчете при группировке данных:

- *Repeating Group* – детальные данные объединяются в одно поле, между значениями вставляется +;
- *Filled* – дублирование данных для каждого заголовка группы;
- *Header* (опция по умолчанию) – печатается заголовок группы, затем – детальная информация.

Стоимостный анализ (ЛВС) и свойства, определяемые пользователем (UDP)

BPwin предоставляет аналитику два инструмента для оценки модели – стоимостный анализ, основанный на работах (*Activity Based Costing, ABC*), и свойства, определяемые пользователем (*User Defined Properties, UDP*).

ABC является широко распространенной методикой, используемой международными корпорациями и государственными организациями (в том числе Департаментом обороны США) для идентификации истинных движителей затрат в организации.

Стоимостный анализ представляет собой соглашение об учете, используемое для сбора затрат, связанных с работами, с целью определить общую стоимость процесса. Стоимостный анализ основан на модели работ, потому что количественная оценка невозможна без детального понимания функциональности предприятия. Обычно *ABC* применяется для того, чтобы понять происхождение выходных затрат и облегчить выбор нужной модели работ при реорганизации деятельности предприятия (*Business Process Reengineering, BPR*). С помощью стоимостного анализа можно решить такие задачи, как определение действительной стоимости производства продукта, определение действительной стоимости поддержки клиента, идентификация работ, которые стоят больше всего (те, которые должны быть улучшены в первую очередь), обеспечение менеджеров финансовой мерой предлагаемых изменений т.д.

ABC может проводиться только тогда, когда модель работы последовательная (следует синтаксическим правилам *IDEF0*), корректная (отражает бизнес), полная (охватывает всю рассматриваемую область) и стабильная (проходит цикл экспертизы без изменений), другими словами, создание модели работы закончено.

ABC включает следующие основные понятия:

- объект затрат – причина, по которой работа выполняется, обычно, основной выход работы, стоимость работ есть суммарная стоимость объектов затрат;
- движитель затрат – характеристики входов и управлений, которые влияют на то, как выполняется и как долго длится работа;
- центры затрат, которые можно трактовать как статьи расхода.

При проведении стоимостного анализа в ***BPwin*** сначала задаются единицы измерения времени и денег. Если в списке выбора отсутствует необходимая валюта (например, рубль), ее можно добавить. Символ валюты по умолчанию берется из настроек *Windows*. Диапазон измерения времени в списке *Unit of measurement* достаточен для большинства случаев - от секунд до лет. Затем описываются центры затрат (*cost centers*). Для внесения центров затрат необходимо вызвать диалог *Cost Center Editor* (меню *Edit/ABC Cost Centers*). Каждому центру затрат следует дать подробное описание в окне *Definition*. Список центров затрат упорядочен. Порядок в списке можно менять при помощи стрелок, расположенных справа от списка. Задание определенной последовательности центров затрат в списке, во-первых, облегчает последующую работу при присвоении стоимости работам, а во-вторых, имеет значение при использовании единых стандартных отчетов в разных моделях. Информация о центрах затрат и *UDP* сохраняется в виде указателей, т.е. хранятся номера центров затрат. Поэтому,

если нужно использовать один и тот же стандартный отчет в разных моделях, списки центров затрат должны быть в них одинаковы.

Общие затраты по работе рассчитываются как сумма по всем центрам затрат. При вычислении затрат вышестоящей (родительской) работы сначала вычисляется произведение затрат дочерней работы на частоту работы (число раз, которое работа выполняется в рамках проведения родительской работы), затем результаты складываются. Если во всех работах модели включен режим *Compute from Decompositions*, подобные вычисления автоматически проводятся по всей иерархии работ снизу вверх. Этот принцип подсчета справедлив, если работы выполняются последовательно. Встроенные возможности **BPwin** позволяют разрабатывать упрощенные модели стоимости, которые оказываются полезными при предварительной оценке затрат. Если схема выполнения более сложная (например, работы производятся альтернативно), можно отказаться от подсчета и задать итоговые суммы для каждой работы вручную (*Override Decompositions*). В этом случае результаты расчетов с нижних уровней декомпозиции будут игнорироваться, при расчетах на верхних уровнях будет учитываться сумма, заданная вручную. На любом уровне результаты расчетов сохраняются независимо от выбранного режима, поэтому при выключении опции *Override Decompositions* расчет снизу вверх производится обычным образом.

Для проведения более тонкого анализа можно воспользоваться специализированным средством стоимостного анализа *EasyABC* (*ABC Technology, Inc.*). **BPwin** имеет двунаправленный интерфейс с *EasyABC*. Для экспорта данных в *EasyABC* следует выбрать пункт меню *File/Export/Node Tree*, задать в диалоге *Export Node Tree* необходимые настройки и экспортировать дерево узлов в текстовый файл (.txt). Файл экспорта можно импортировать в *EasyABC*. После проведения необходимых расчетов результирующие данные можно импортировать из *EasyABC* в **BPwin**. Для импорта нужно выбрать меню *File/Import/Costs* и в диалоге *Import Activity Costs* выбрать необходимые установки.

Результаты стоимостного анализа могут существенно повлиять на очередность выполнения работ.

Результаты стоимостного анализа наглядно представляются на специальном отчете **BPwin** – *Activity Cost Report* (меню *Report/Activity Cost Report*). Отчет позволяет документировать имя, номер, определение и стоимость работ, как суммарную, так и отдельно по центрам затрат. Результаты отображаются и непосредственно на диаграммах. В левом нижнем углу прямоугольника работы может показываться либо стоимость (по умолчанию), либо продолжительность, либо частота проведения работы. Настройка отображения осуществляется в диалоге *Model Properties* (меню *Edit/Model Properties*), закладка *Display, ABC Data, ABC Units*.

ABC позволяет оценить стоимостные и временные характеристики системы. Если стоимостных показателей недостаточно, имеется возможность внесения собственных метрик – свойств, определенных пользовате-

лем (*User Defined Properties, UDP*). *UDP* позволяют провести дополнительный анализ, хотя и без суммирующих подсчетов.

Для описания *UDP* служит диалог *User-Defined Property Name Editor* (меню *Edit/UDP Definition*). В верхнем окне диалога вносится имя *UDP*, в списке выбора *Datatype* описывается тип свойства. Имеется возможность задания 18 различных типов *UDP*, в том числе управляющих команд и массивов, объединенных по категориям. Для внесения категории следует задать имя категории в окне *New Category/Member* и щелкнуть по кнопке *Add Category*. Для присвоения свойства категории необходимо выбрать *UDP* из списка, затем категорию из списка категорий и щелкнуть по кнопке *Update*. Одна категория может объединять несколько свойств, в то же время одно свойство может входить в несколько категорий. Свойство типа *List* может содержать массив предварительно определенных значений. Для определения области значений *UDP* типа *List* следует задать значение свойства в окне *New Category/Member* и щелкнуть по кнопке *Add Member*. Значения из списка можно редактировать и удалять.

Каждой работе можно поставить в соответствие набор *UDP*. Для этого следует щелкнуть правой кнопкой мыши по работе и выбрать пункт меню *UDP Editor*. В закладке *UDP Values* диалога *IDEF0 Activity Properties* можно задать значения *UDP*. Свойства типа *List* отображаются списком выбора, который заполнен предварительно определенными значениями. Свойства типа *Command* могут иметь в качестве значения командную строку, которая выполняется при нажатии на кнопку. Например, свойство "Спецификации" категории "Дополнительная документация" может иметь значение "*C:\MSOffice97\Office\WINWORD.EXE sped.doc*".

Кнопка *Categories* служит для задания фильтра по категориям *UDP*. По умолчанию в списке показываются свойства всех категорий,

В левом нижнем углу диалога настройки отчета показывается список *UDP*. С помощью кнопки *Activity Categories* можно установить фильтр по категориям.

Порядок выполнения работы

1. В диалоге *Model Properties* (вызывается из меню *Model/Model Properties*) во вкладке *ABC Units* (рис. 2.35) установите единицы измерения и денег и времени – рубли и дни (как правило, все установки есть по умолчанию).

2. Перейдите в *Dictionary/Cost Center* и в диалоге *Cost Center Dictionary* внесите название и определение центров затрат (табл. 2.14).

Для подтверждения ввода данных используйте клавишу табуляции.

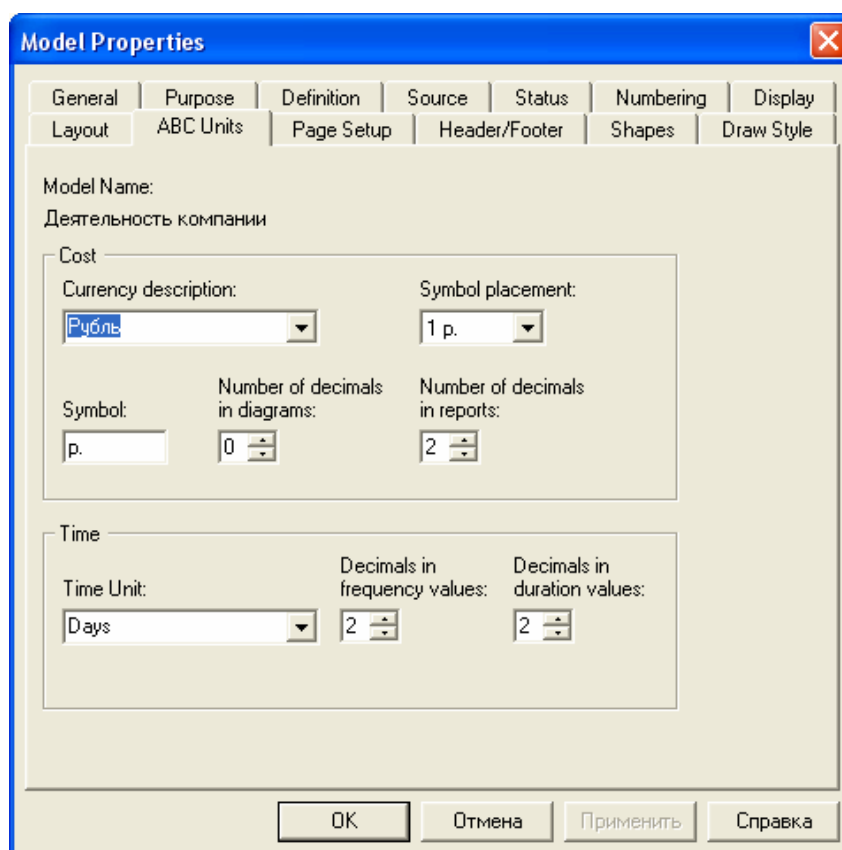


Рис. 2.35. Вкладка *ABC Units* диалога *Model Properties*

Таблица 2.14

Цены затрат *ABC*

| Центр затрат | Определение |
|--------------|--|
| Управление | Затраты на управление, связанные с составлением графика работ, формированием партий компьютеров, контролем над сборкой и тестированием |
| Рабочая сила | Затраты на оплату рабочих, занятых сборкой и тестированием компьютеров |
| Компоненты | Затраты на закупку компонентов |

3. Для отображения стоимости каждой работы в нижнем левом углу прямоугольника перейдите в меню *Model/Model Properties* и во вкладке *Display* диалога *Model Properties* включите опцию *ABC Data* (как правило, эта опция установлена по умолчанию).

4. Установите стоимости работ (табл. 2.15). Для назначения стоимости работе следует щелкнуть по ней правой кнопкой мыши и выбрать в контекстном меню *Costs*. Если данные не вводятся, выберите опцию *Override decompositions*.

Таблица 2.15

Стоимости работ на диаграмме A2

| <i>Activity Name</i> | <i>Cost Center</i> | <i>Cost Center Cost, руб.</i> | <i>Frequency</i> | <i>Duration, день</i> |
|--|--------------------|-------------------------------|------------------|-----------------------|
| Отслеживание расписания и управление сборкой и тестированием | Управление | 500,00 | 1,00 | 1,00 |
| Сборка настольных компьютеров | Рабочая сила | 100,00 | 12,00 | 1,00 |
| | Компоненты | 16000,00 | | |
| Сборка ноутбуков | Рабочая сила | 140,00 | 20,00 | 1,00 |
| | Компоненты | 28000,00 | | |
| Тестирование компьютеров | Рабочая сила | 60,00 | 32,00 | 1,00 |

5. Сгенерируйте отчет *Activity Cost Report (Tools/Reports)*: установите флаги у *Activity Name*, *Activity Costs*, *Cost center name*, *Cost center cost*, а для остальных установок – оставьте те, что были заданы по умолчанию. В качестве формата отчета (*Report Format*) следует выбрать "DDE Table".

Таблица 2.16

Отчет *Activity Cost Report*

| <i>Activity Name</i> | <i>Activity Cost (Рубль)</i> | <i>Cost Center</i> | <i>Cost Center Cost (Рубль)</i> |
|--|------------------------------|--------------------|---------------------------------|
| Деятельность компании | 758 420,00 | Компоненты | 752 000,00 |
| | | Рабочая сила | 5 920,00 |
| | | Управление | 500,00 |
| Продажи и маркетинг | 0,00 | | |
| Сборка и тестирование компьютеров | 758 420,00 | Компоненты | 752 000,00 |
| | | Рабочая сила | 5 920,00 |
| | | Управление | 500,00 |
| Отслеживание расписания и управление сборкой и тестированием | 500,00 | Управление | 500,00 |
| Сборка настольных компьютеров | 16 100,00 | Компоненты | 16 000,00 |
| | | Рабочая сила | 100,00 |
| Подготовка компонентов | 0,00 | | |
| Установка материнской платы и винчестера | 0,00 | | |
| Инсталляция дополнительного программного обеспечения | 0,00 | | |
| Установка флоппи-дисков | 0,00 | | |
| Установка модема | 0,00 | | |

Окончание табл. 2.16

| <i>Activity Name</i> | <i>Activity Cost (Рубль)</i> | <i>Cost Center</i> | <i>Cost Center Cost (Рубль)</i> |
|----------------------------------|----------------------------------|--------------------|-------------------------------------|
| Установка <i>CD-ROM</i> | 0,00 | | |
| Инсталляция операционной системы | 0,00 | | |
| Сборка ноутбуков | 28 140,00 | Компоненты | 28 000,00 |
| | | Рабочая сила | 140,00 |
| Тестирование компьютеров | 60,00 | Рабочая сила | 60,00 |
| Отгрузка и получение | 0,00 | | |
| Получить комплектующие | 0,00 | | |
| Доставить комплектующие | 0,00 | | |
| Отгрузить товар и возврат | 0,00 | | |

6. Сохраните полученный файл *MSWord* в рабочую папку с названием *Activity Cost Report*

7. Перейдите в меню *Dictionary/UDP Keywords* и в диалоге *UDP Keywords List* внесите ключевые слова *UDP* (рис. 2.36):

- Документация;
- Информационная систем;
- Расход ресурсов.

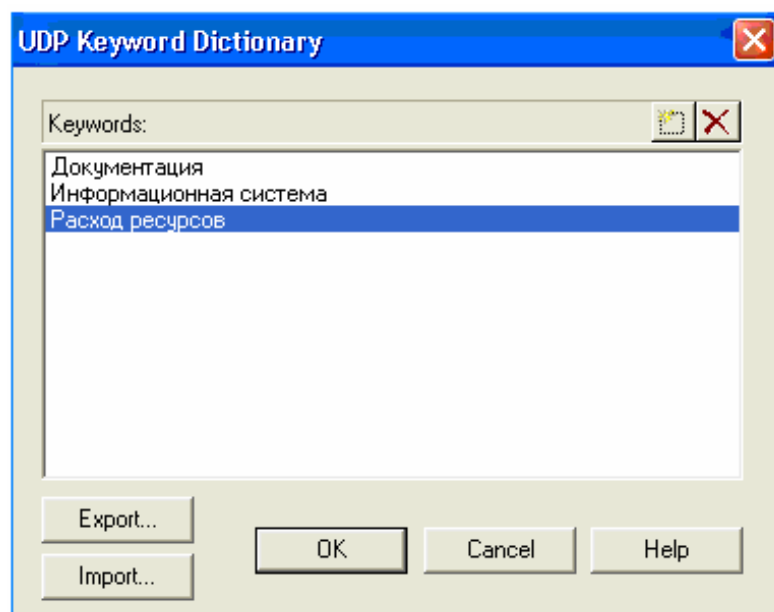


Рис. 2.36. Словарь ключевых слов *UDP*

8. Откройте *UDP Dictionary (Dictionary/UDP)*. Внесите значения в соответствии с табл. 2.17. Для подключения к *UDP* ключевого слова перейдите к полю *Keyword* и щелкните по полю выбора. Для *UDP* типа *List* необходимо в поле *Value* задать список значений. Для подтверждения ввода данных используйте клавишу табуляции.

Таблица 2.17

Наименование и свойства *UDP*

| Наименование <i>UDP</i> | Тип | Значение | Ключевое слово |
|------------------------------|---|--|------------------------|
| Приложения | <i>Text List (Multiple Selection)</i> | Модуль оформления заказов Модуль создания и контроля расписания выполнения работ Модуль учета комплектующих и оборудования Модуль процедур сборки и поиска неисправностей | Информационная система |
| Дополнительная документация | <i>Command List</i> | <i>Winword.EXE sample1.doc</i> <i>Winword.EXE sample2.doc</i> <i>POWERPNT.EXE sample3.doc</i> | Документация |
| История изменения | <i>Paragraph Text</i> | | Документация |
| Загрязнение окружающей среды | <i>Text List (Single Selection)</i> | Очень высокое Высокое Среднее Низкое | |
| Расход электроэнергии | <i>Real Number</i> | | Расход ресурсов |

После внесенных изменений словарь будет выглядеть следующим образом

| Name | UDP Datatype | Value | Keyword |
|------------------------------|---------------------------------|---------------------------|------------------------|
| Дополнительная документация | Command List | POWERPNT.EXE sample3.doc | Документация |
| Загрязнение окружающей среды | Text List (Single selection) | Высокое | |
| История изменения | Paragraph Text | | Документация |
| Приложения | Text List (Multiple selections) | Модуль оформления заказов | Информационная система |
| Расход электроэнергии | Real Number | | Расход ресурсов |
| | Text | | |

Рис. 2.37. Словарь *UDP*

9. Для назначения *UDP* работе следует щелкнуть по ней правой кнопкой мыши и выбрать в контекстном меню *UDP*. Появляется вкладка *UDP Values* диалога *Activity Properties* (рис. 2.38).

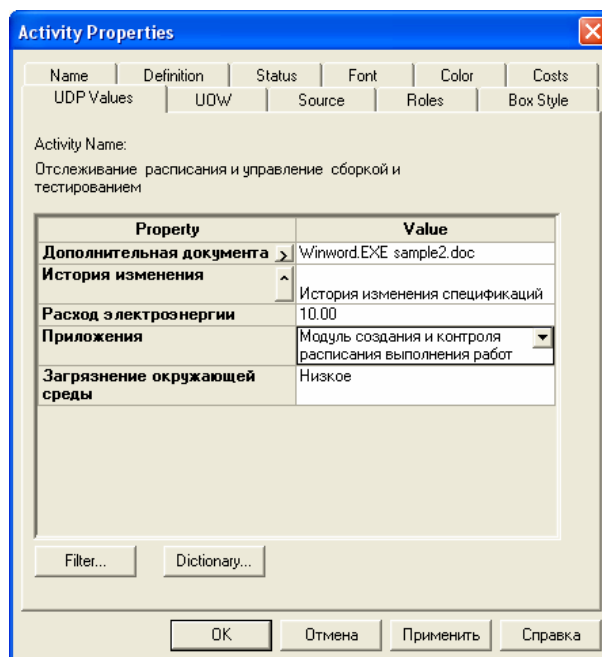




Рис. 2.38. Вкладка *UDP Values* диалога *Activity Properties*

Внесите значения *UDP* для работ (табл. 2.18).

Таблица 2.18

Значения *UDP* для работ

| <i>Activity Name</i> | Дополнительная документация | Приложения | История изменения | Расход электроэнергии | Загрязнение окружающей среды |
|--|--------------------------------|---|--------------------------------|-----------------------|------------------------------|
| Отслеживание расписания и управление сборкой и тестированием | <i>Winword.EXE sample2.doc</i> | Модуль создания и контроля расписания выполнения работ | История изменения спецификаций | 10,00 | Низкое |
| Сборка настольных компьютеров | | Модуль учета комплектующих и оборудования. Модуль процедур сборки и поиска неисправностей | | 20,00 | Среднее |
| Сборка ноутбуков | | Модуль учета комплектующих и оборудования. Модуль процедур сборки и поиска неисправностей | | 25,00 | Среднее |
| Тестирование компьютеров | | Модуль учета комплектующих и оборудования. Модуль процедур сборки и поиска неисправностей | | 40,00 | Среднее |

10. После внесения *UDP* типа *Command* или *Command List* щелчок по кнопке  приведет к запуску приложения (если это приложение уже создано). Создайте файл "*sample2.doc*" в каталоге, из которого вы работаете. Щелкните по кнопке . В результате произойдет запуск приложения. Закройте его.

11. В диалоге *Activity Properties* щелкните по кнопке *Filter*. В появившемся диалоге *Diagram object UDP filter* (рис. 2.39) отключите ключевые слова "Информационная система". В результате в диалоге *Activity Properties* не будут отображаться *UDP* с ключевыми словами "Информационная система".

12. Создайте отчет по *UDP*. Меню *Tools/Report/Diagram object Report*. Выберите опции отчета:

- *Start from Activity: A2*;
- *Number of Levels: 2*;
- *User Defined Properties: Расход электроэнергии*;
- *Report Format: DDE Table (MS Word)*.

13. Щелкните по кнопке *Report*. Сохраните созданный документ *UDP report* в рабочую папку. Содержание отчета отображает табл. 2.19.

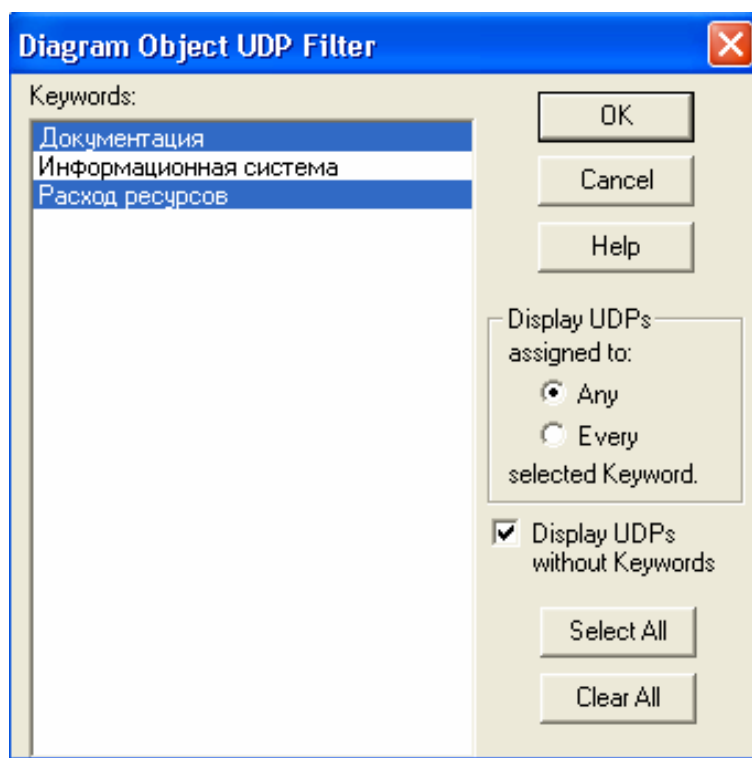


Рис. 2.39. Диалог *Diagram object UDP filter*

UDP Report

| <i>Activity Name</i> | Расход электроэнергии |
|--|------------------------------|
| Сборка и тестирование компьютеров | |
| Отслеживание расписания и управление сборкой и тестированием | 10,00 |
| Сбока настольных компьютеров | 20,00 |
| Сборка ноутбуков | 25,00 |
| Тестирование компьютеров | 40,00 |

Контрольные вопросы

1. Какие бывают типы отчетов?
2. Какие бывают синтаксические ошибки?
3. Что такое стоимостной анализ?
4. Что означает *UDP*?
5. Каким образом можно сгенерировать отчет?

Лабораторная работа №8. Реинжиниринг процессов

Цель работы: Создание модели *TO-BE* (реинжиниринг бизнес-процессов).

Теоретические сведения

Проведение экспертизы. Цикл автор-читатель. Цикл автор-читатель предназначен для обеспечения обратной связи при построении модели. Он включает определенные формализованные процедуры, предписывающие правила координации деятельности участников создания модели. В работе над моделью принимают участие специалисты разных специальностей – аналитики (авторы), эксперты предметной области (читатели), библиотекари и комитет технического контроля. Обычно библиотекарь выделяется для больших проектов. Цикл автор-читатель содержит следующие этапы:

На очередном этапе декомпозиции аналитик создает диаграмму на основе общих знаний, анализа документации и опроса экспертов. Общие знания не позволяют создать диаграмму достаточно корректно, поэтому она нуждается в уточнении и дополнении.

Все коммуникации при создании модели контролируются библиотекарем. Он ответственен за прохождение папок и архивирование диаграмм модели. После создания диаграмма посылается библиотекарю для помещения в архив.

Автором формируется папка и передается для распространения библиотекарю (одна копия направляется автору). В папку должна входить текущая диаграмма. Кроме того, в папку могут включаться сопутствующие отчеты, в том числе словарь стрелок и работ, диаграмма верхнего уровня, дерево узлов и любая необходимая дополнительная документация. На папке регистрируются входящие данные – дата, автор, данные читателя и т. д., после чего папка направляется эксперту предметной области (читателю).

Читатель рецензирует папку и записывает свои комментарии. Замечания вносятся в диаграмму по определенным правилам. Если читатель решил внести замечание, он должен указать номер замечания, затем внести текст замечания и в каркасе диаграммы в разделе *Notes* зачеркнуть цифру, соответствующую номеру замечания (рис. 2.40).

После рецензирования папки возвращаются библиотекарю. Библиотекарь должен обеспечивать проведение рецензирования в срок. Затем папки регистрируются и направляются автору.

Автор вносит ответ на замечания и, если он согласен с замечаниями, вносит изменения в модель. На практике зачастую сеанс экспертизы проводится в форме устного собеседования между автором и экспертом. В этом случае особенно важно вносить замечания эксперта и комментарии автора в диаграмму для документирования всех идей, возникших в результате моделирования.

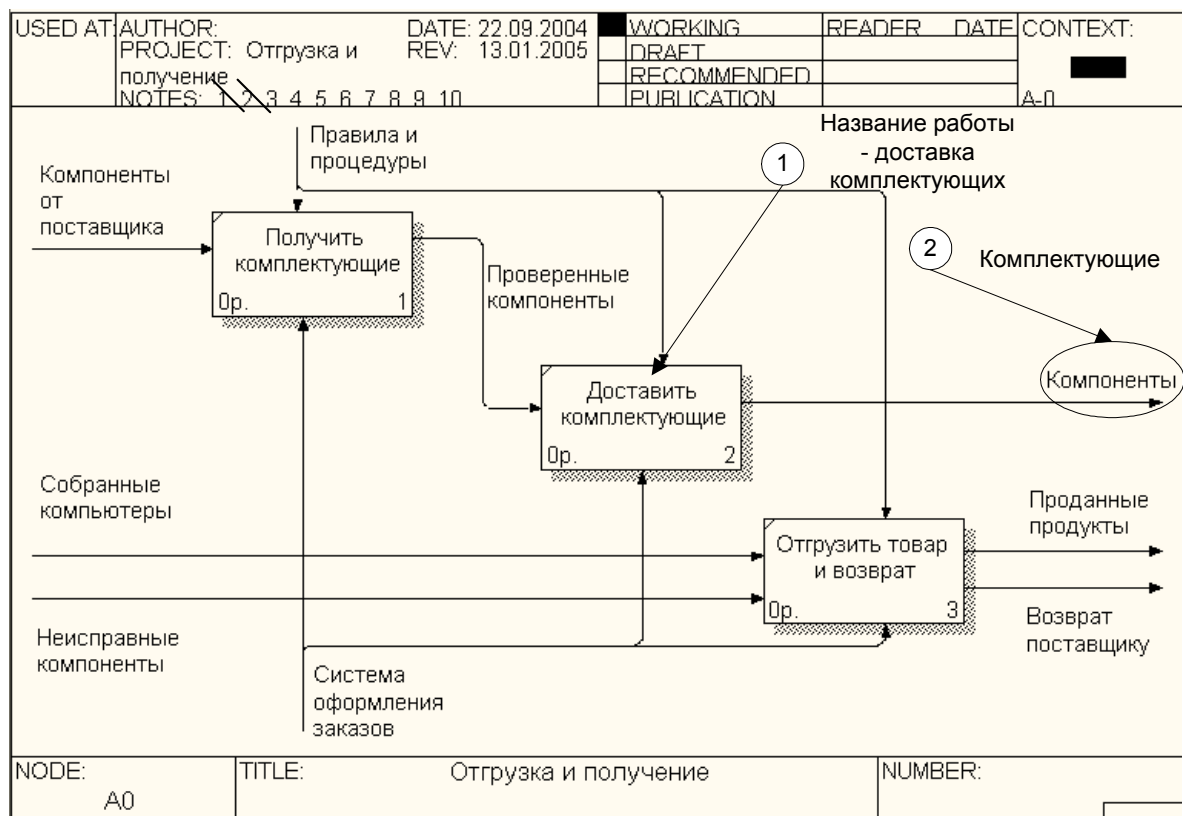


Рис. 2.40. Внесение замечаний в диаграмму

Если это необходимо, проводится дополнительная экспертиза у того же или у другого эксперта.

После прохождения нескольких циклов число замечаний обычно уменьшается, и диаграмма становится стабильной. В процессе изменения диаграмма может менять свой статус, который должен быть отражен в каркасе диаграммы. Когда автор считает, что диаграмма уже достаточно проработана и достигла уровня "*Recommended*", он пересылает ее на утверждение в комитет технического контроля, где она проходит окончательную экспертизу. После внесения замечаний и окончательных изменений диаграмма (или набор диаграмм) окончательно утверждается, получает статус "*Publication*" и может быть распечатана и распространена среди участников проекта.

Порядок выполнения работы

Модель *TO-BE* создается на основе анализа модели *AS-IS*. Анализ может проводиться как по формальным признакам (отсутствие выходов или управления, отсутствие обратных связей и т.д.), так и по неформальным – на основе знаний предметной области.

Допустим, в результате анализа принимается решение реорганизовать функции производства и тестирования компьютеров и оставить функциональности "Продажи и маркетинг" и "Отгрузка и получение" пока без изменений.

Принято решение сформировать отдел дизайна, который должен формировать конфигурацию компьютеров, разрабатывать корпоративные стандарты, подбирать приемлемых поставщиков, разрабатывать инструкции по сборке, процедуры тестирования и устранения неполадок для всего производственного отдела.

Работа "Сборка и тестирование компьютеров" должна быть реорганизована и названа "Производство продукта". Будут созданы работы "Разработать конфигурацию", "Планировать производство" и "Собрать продукт".

Рассмотрим новые роли персонала. Дизайнер должен разрабатывать систему, стандарты на продукцию, документировать и передавать спецификации в отдел маркетинга и продаж. Он должен определять, какие компоненты (аппаратные и программные) должны закупаться для сборки компьютеров, обеспечивать документацией и управлять процедурами сборки, тестирования и устранения неполадок.

Функции диспетчера в работе "Сборка и тестирование компьютеров" должны быть заменены на функции планировщика.

Планировщик должен обрабатывать заказы клиентов и генерировать заказы на сборку, получить коммерческий прогноз из отдела маркетинга и формировать требования на закупку компонентов и собирать информацию от поставщиков.

Диспетчер должен составлять расписание производства на основании заказов на сборку, полученных в результате работы "Планировать производство", получать копии заказов клиентов и отвечать за упаковку и комплектацию заказанных компьютеров, передаваемых в работу "Отгрузка и получение".

1. Расщепление и модификация модели

1. Измените свойства модели "Деятельность компании":

- *Model Name*: Предлагаемая модель компании;
- *Time Frame*: TO-BE;
- *Purpose*: Документировать предлагаемые изменения бизнес-процессов компании.

2. Переименуйте работу "Сборка и тестирование компьютеров" в "Производство продукта". Расщепите эту работу в модель с тем же названием.

3. Модифицируйте отщепленную модель. Переместите работу "Тестирование компьютеров" с диаграммы АО "Производство продукта" на диаграмму А2.1 "Сборка настольных компьютеров".

4. Переименуйте работу "Сборка настольных компьютеров" на диаграмме АО в "Сборка продукта".

5. Удалите работу "Сборка ноутбуков".

6. Переименуйте стрелку "Заказы на настольные компьютеры" в "Заказы на изготовление".

7. Переименуйте "Отслеживание расписания и управление сборкой и тестированием" в "Планирование производства".

8. Создайте работу "Разработать конфигурацию".
9. Создайте ветвь стрелки "Персонал производственного отдела", назовите ее "Дизайнер" и направьте как механизм к работе "Разработать конфигурацию".
10. Создайте стрелку "Стандарты на продукцию" и направьте ее от выхода "Разработать конфигурацию" к границе диаграммы. Тоннелируйте эту стрелку (*Resolve Border Arrow*). Создайте ветвь этой стрелки, идущую к управлению работы "Планирование производства" и назовите ее "Список необходимых компонентов".
11. Удалите стрелку "Правила сборки и тестирования" с родительской диаграммы и диаграммы декомпозиции. Создайте ветвь стрелки "Список необходимых компонентов", идущую к управлению работы "Сборка продукта" и назовите ее "Правила сборки и тестирования".
12. Переименуйте стрелку "Диспетчер" в "Планировщика производства".
13. Добавьте стрелки "Прогноз продаж" и "Информация от поставщика" как граничные управляющие к работе "Планирование производства".
14. Добавьте стрелку "Заказ поставщику" как граничную стрелку выхода от работы "Планирование производства".
15. Тоннелируйте необходимые стрелки (*Resolve Border Arrow*) и свяжите их с работами на родительской диаграмме A0.
16. Тоннелируйте стрелки "Собранные компьютеры" и "Неисправные компоненты" (*Resolve Border Arrow*) и свяжите их с выходом работы "Сборка продукта".
17. Результат выполнения первой части лабораторной работы №8 приведен на рис. 2.41 и рис. 2.42.

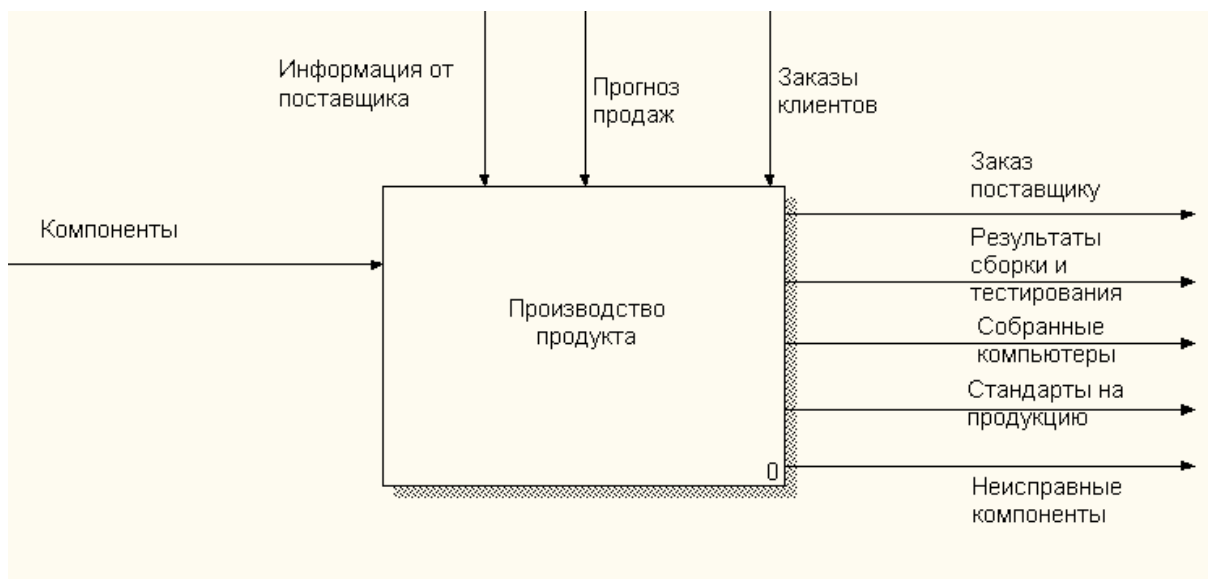


Рис. 2.41. Родительская диаграмма A0 "Производство продукта"

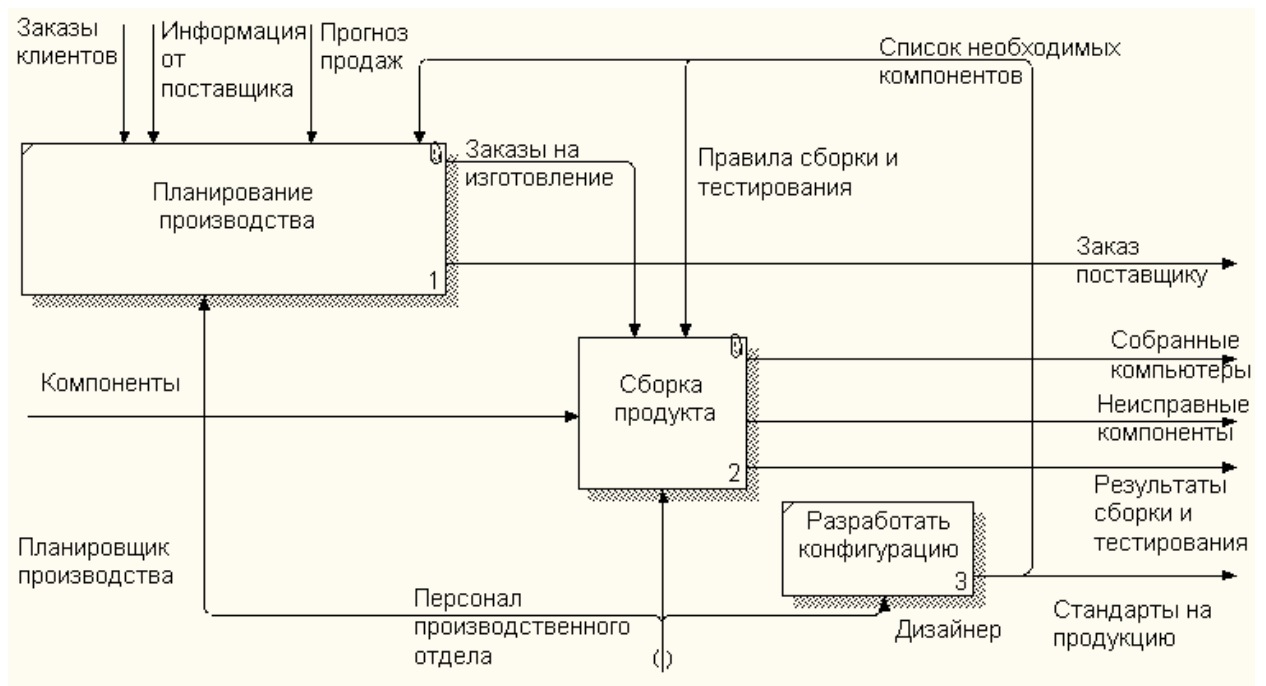


Рис. 2.42. Диаграмма декомпозиции A0 "Производство продукта"

II. Слияние модели

1. Перейдите к работе "Производство продукта" в модели "Деятельность компании". Щелкните правой кнопкой мыши по работе. В контекстном меню выберите *Merge Model*. В появившемся диалоге *Merge Model* установите опцию *Cut/Paste entire dictionaries*, опцию *Overwrite existing fields* и щелкните по *OK*. Модели должны слиться.

2. На диаграмме A0 тоннелируйте стрелки (*Resolve Border Arrow*) "Информация от поставщика" и "Заказ поставщику".

3. Измените стрелку "Прогноз продаж". Направьте ее с выхода "Продажи и маркетинг" на управление "Производство продукта" (измените стрелку "Прогноз продаж" на диаграмме A2 "Производство продукта" в связи с появившейся стрелкой контроля).

4. Измените стрелку "Стандарты на продукцию". Направьте ее с выхода "Производство продукта" на управление "Продажи и маркетинг" (измените стрелку "Стандарты на продукцию" выхода работы "Разработать конфигурацию" на диаграмме A2 "Производство продукта" в связи с появившейся стрелкой выхода).

5. Удалите ветвь стрелки управления "Правила и процедуры" работы "Производство продукта".

6. Закройте модель "Производство продукта". Результат выполнения второй части лабораторной работы №8 приведен на рис. 2.43 и рис. 2.44.

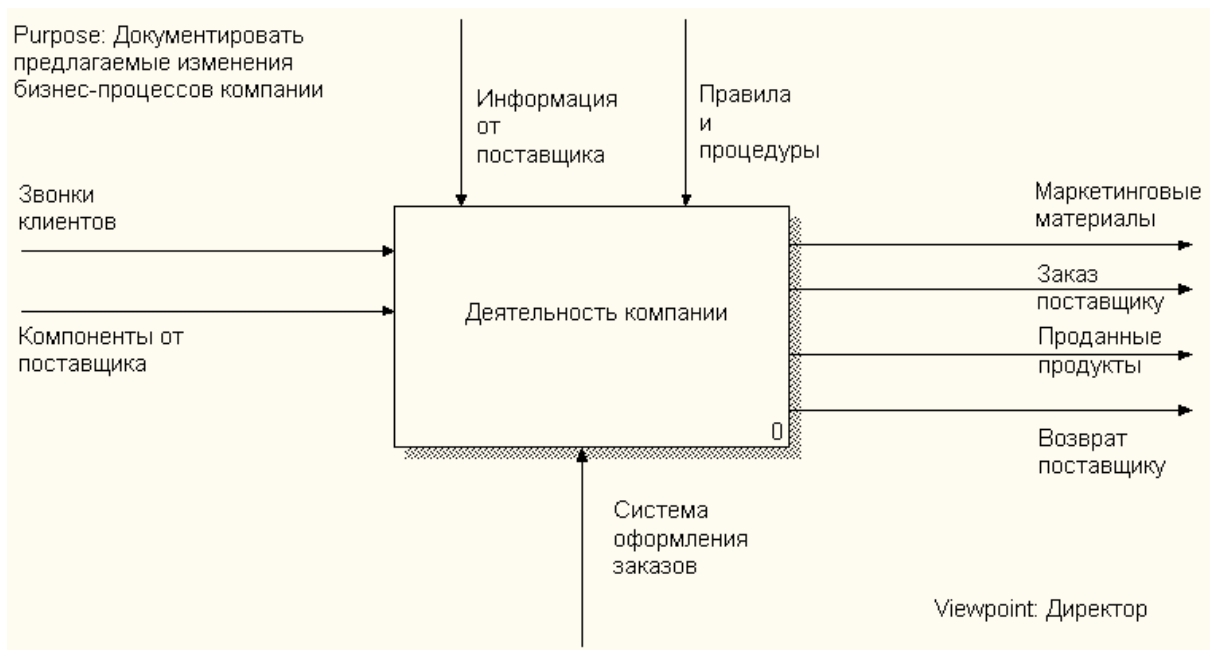


Рис. 2.43. Родительская диаграмма АО "Деятельность компании"

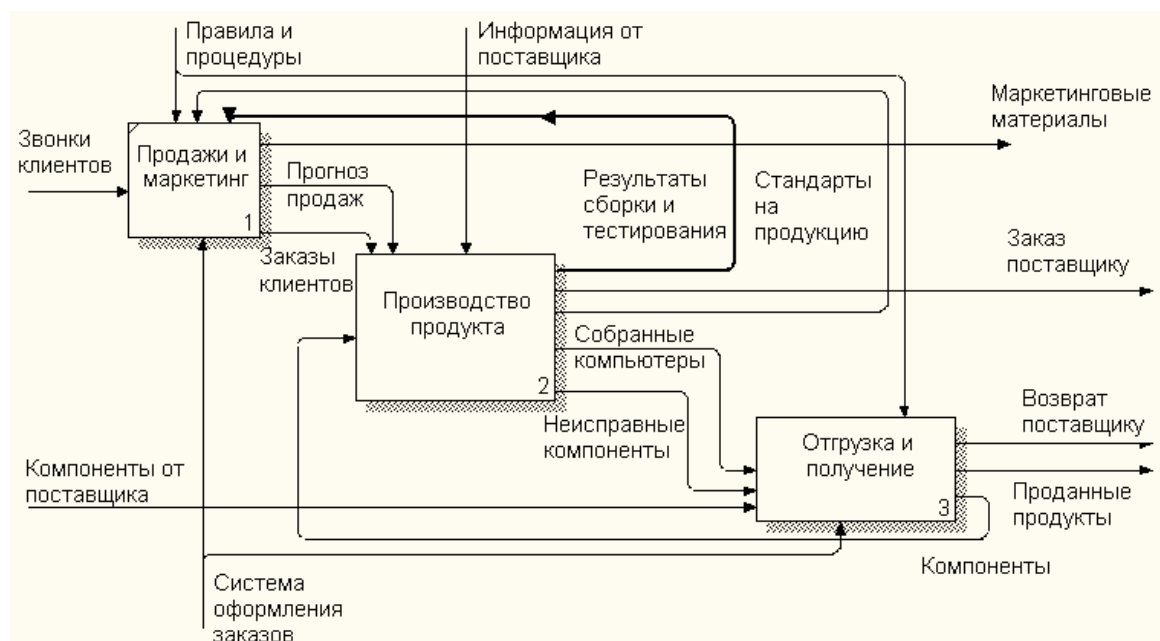


Рис. 2.44. Диаграмма АО "Деятельность компании"

III. Использование *Model Explorer* для реорганизации дерева декомпозиции

1. Существуют причины, по которым работа "Разработать конфигурацию" должна быть на верхнем уровне, на диаграмме АО. Дизайнер разрабатывает стандарты на продукцию, включая правила сборки и тестирования, и список необходимых для закупки компонентов. Тем самым дизайнер управляет производством продукта в целом, кроме того, управляет ра-

ботой "Продажи и маркетинг". Логично перенести эту работу на уровень выше. Используя возможности *Model Explorer*, перенесите работу "Разработать конфигурацию" с диаграммы A2 "Производство продукта" на диаграмму A0.

2. Разрешите и перенаправьте стрелки согласно рис. 2.45 и рис. 2.46.

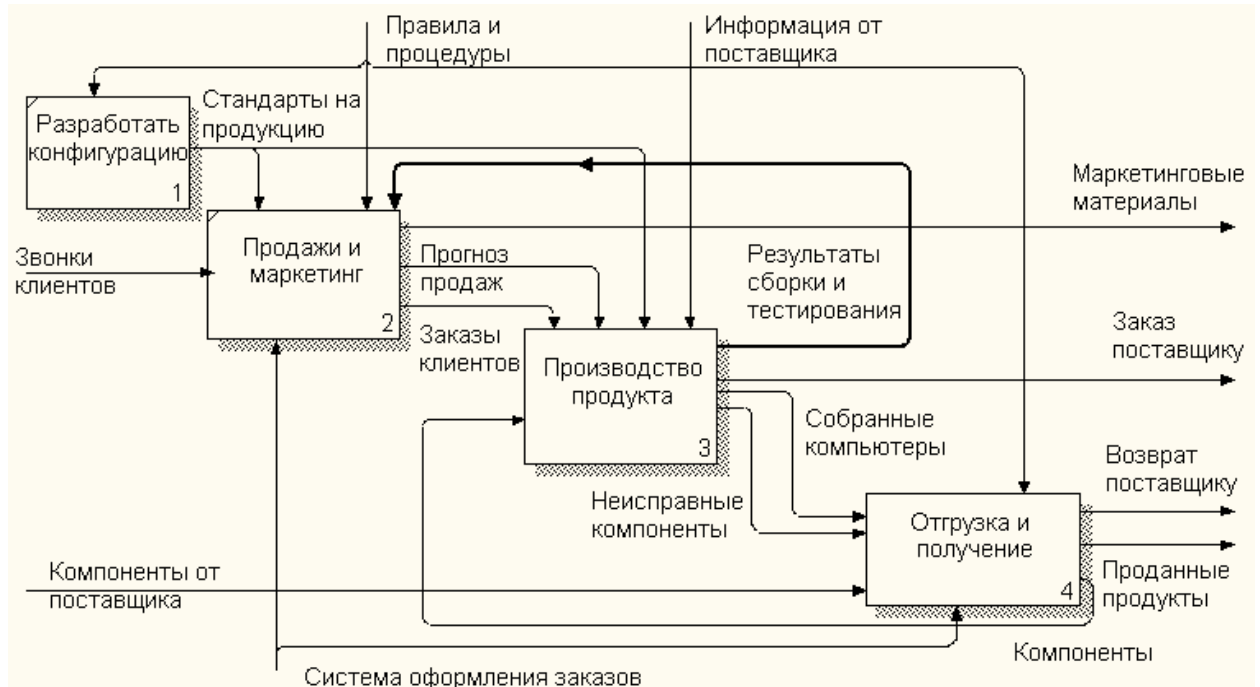


Рис. 2.45. Родительская диаграмма A0 2 "Деятельность компании"

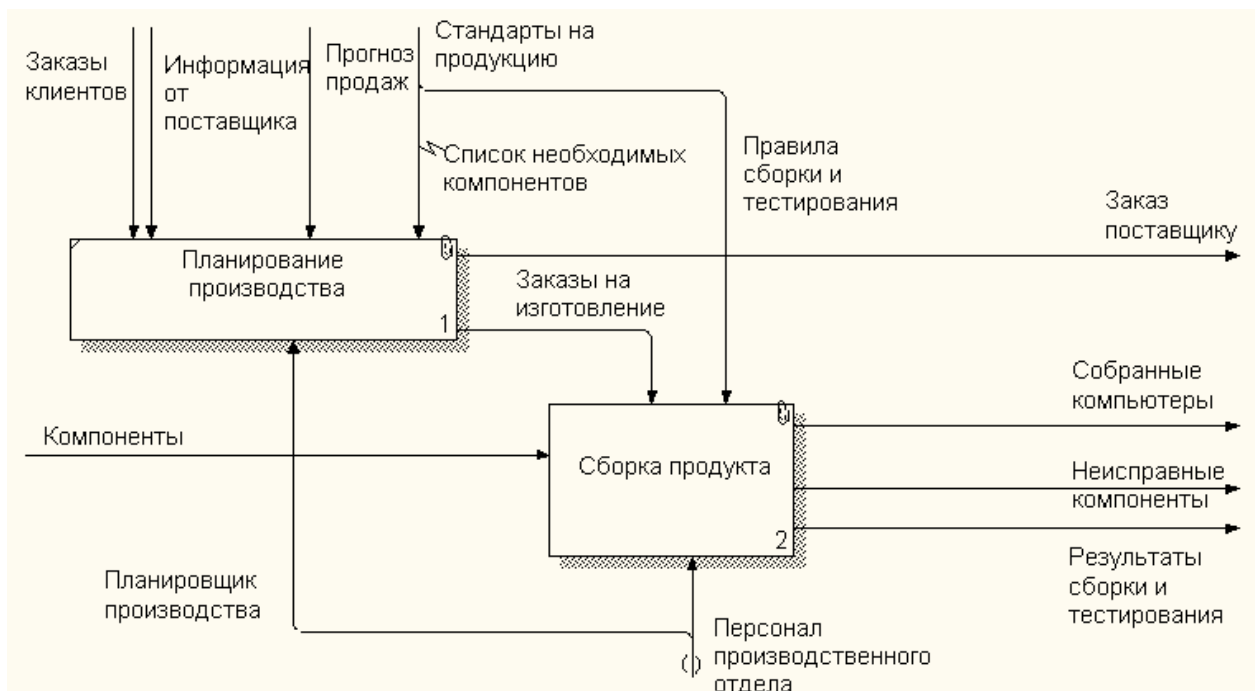


Рис. 2.46. Диаграмма A3 "Производство продукта"

IV. Модификация диаграммы IDEF3 "Сборка продукта" с целью отображения новой информации

Так же как в модели AS-IS, сборка продукта состоит из сборки компонентов и установки программного обеспечения. Однако теперь в работу "Сборка продукта" включена работа "Тестирование компьютера". Тестирование начинается после окончания процесса сборки компьютера и окончания процесса установки программного обеспечения. Если компьютер неисправен, в процессе тестирования у него заменяют компоненты, информация о неисправных компонентах может быть направлена на работу "Подготовка компонентов". Такая информация может помочь более тщательно подготавливать компоненты к сборке. Результатом процесса тестирования являются заказанные компьютеры и неисправные компоненты. Модифицируйте диаграмму IDEF3 "Сборка продукта" в соответствии с приведенной информацией, результат приведен на рис. 2.47.

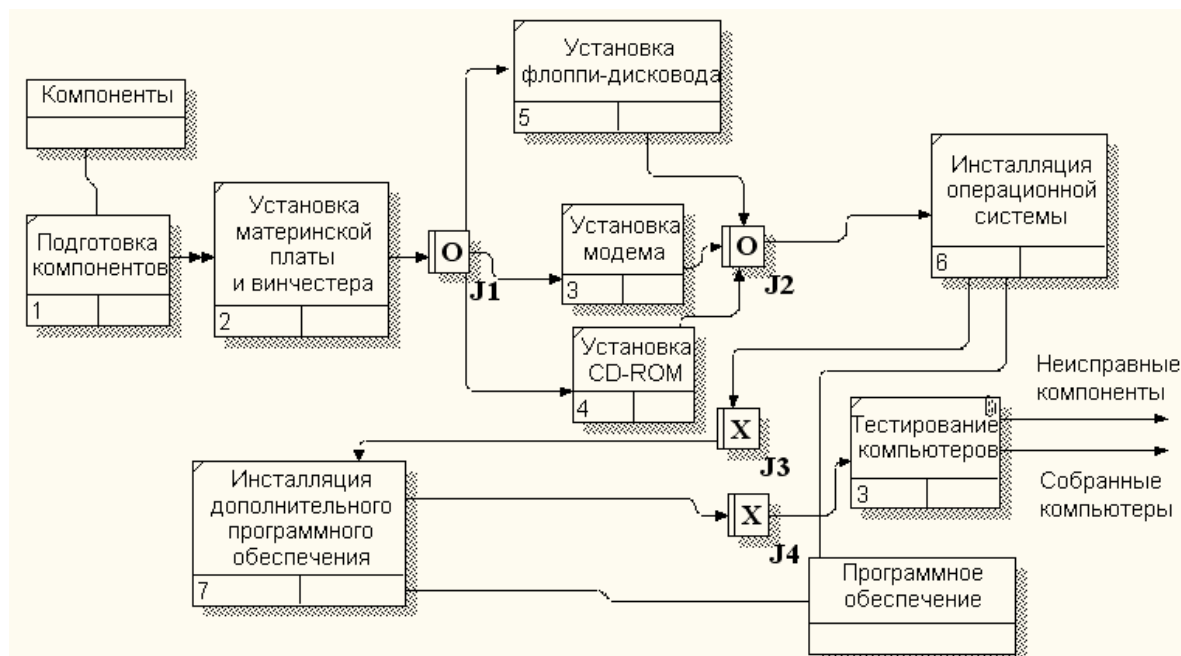


Рис. 2.47. Диаграмма A32.1 "Сборка продукта"

V. Декомпозиция работы "Продажи и маркетинг"

Работа по продажам и маркетингу заключается в ответах на телефонные звонки клиентов, предоставлении клиентам информации о ценах, оформлении заказов, внесении заказов в информационную систему и исследовании рынка. На основе этой информации декомпозируйте работу "Продажи и маркетинг" (IDEF0). Создайте следующие работы:

- Предоставление информации о ценах;
- Оформление заказов;
- Исследование рынка.

Убедитесь, что все стрелки модели связаны, если необходимо перерисуйте несвязанные стрелки. Результат декомпозиции представлен на рис. 2.48.

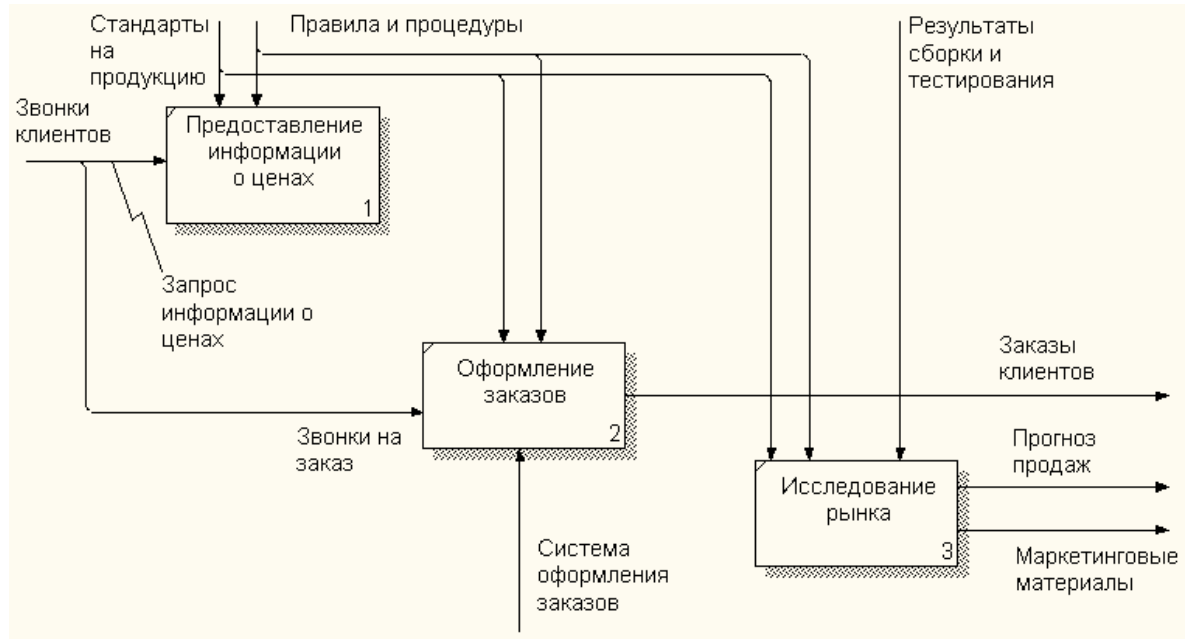


Рис. 2.48. Диаграмма A2 "Продажи и маркетинг"

Контрольные вопросы

1. Что такое реинжиниринг бизнес-процессов?
2. Опишите цикл автор – читатель.
3. Кто такой библиотекарь?
4. Каким образом заносятся замечания в проект?
5. Кто может являться экспертом?

Лабораторная работа №9. Методология DFD

Цель работы: Создание диаграммы DFD.

Теоретические сведения

Диаграммы потоков данных (Data Flow Diagramm)

Диаграммы потоков данных (DFD) используются для описания документооборота и обработки информации. В соответствии с методологией модель системы определяется как иерархия диаграмм потоков данных, описывающих асинхронный процесс преобразования информации от ее ввода в систему до выдачи пользователю. Диаграммы верхних уровней иерархии (контекстные диаграммы) определяют основные процессы или подсистемы ИС с внешними входами и выходами. Они детализируются при помощи диаграмм нижнего уровня. Такая декомпозиция продолжается, создавая многоуровневую иерархию диаграмм, до тех пор, пока не будет достигнут такой уровень декомпозиции, на котором процесс становятся элементарными и детализировать их далее невозможно. Их можно использовать как дополнение к модели IDEF0 для более наглядного отображения текущих операций документооборота в корпоративных системах обработки информации. DFD описывает:

- системы/подсистемы
- функции обработки информации (работы);
- документы (стрелки, *arrow*), объекты, сотрудников или отделы, которые участвуют в обработке информации;
- внешние ссылки (*external references*), которые обеспечивают интерфейс с внешними объектами, находящимися за границами моделируемой системы;
- таблицы для хранения документов (хранилище данных, *data store*).

В **Bpwin** для построения диаграмм потоков данных используется нотация Гейна-Сарсона.

Для того чтобы дополнить модель IDEF0 диаграммой DFD, нужно в процессе декомпозиции в диалоге *Activity Box Count* кликнуть по радиокнопке DFD. В палитре инструментов на новой диаграмме DFD появляются новые кнопки:

- добавить в диаграмму внешнюю ссылку (*External Reference*). Внешняя ссылка является источником или приемником данных извне модели;
- добавить в диаграмму хранилище данных (*Data store*). Хранилище данных позволяет описать данные, которые необходимо сохранить в памяти прежде, чем использовать в работах;
- ссылка на другую страницу. В отличие от IDEF0 инструмент *offpage reference* позволяет направить стрелку на любую диаграмму (а не только на верхний уровень).

В отличие от стрелок IDEF0, которые представляют собой жесткие взаимосвязи, стрелки DFD показывают, как объекты (включая данные)

двигаются от одной работы к другой. Это представление потоков совместно с хранилищами данных и внешними сущностями делает модели *DFD* более похожими на физические характеристики системы – движение объектов (*data flow*), хранение объектов (*data stores*), поставка и распространение объектов (*external entities*).

DFD рассматривает систему как совокупность предметов. Контекстная диаграмма часто включает работы и внешние ссылки. Работы обычно именуются по названию системы, например "Система обработки информации". Включение внешних ссылок в контекстную диаграмму не отменяет требования методологии четко определить цель, область и единую точку зрения на моделируемую систему.

Системы/подсистемы. При построении модели сложной ИС она может быть представлена в самом общем виде на так называемой контекстной диаграмме в виде одной системы как единого целого, либо может быть декомпозирована на ряд подсистем.

Работы. В *DFD* работы представляют собой функции системы, преобразующие входы в выходы. Хотя работы изображаются прямоугольниками со скругленными углами, смысл их совпадает со смыслом работ *IDEF0* и *IDEF3*. Так же как работы *IDEF3*, они имеют входы и выходы, но не поддерживают управления и механизмы, как *IDEF0*.

Внешние сущности. Внешние сущности изображают входы в систему и/или выходы из системы. Внешние сущности изображаются в виде прямоугольника с тенью и обычно располагаются по краям диаграммы. Одна внешняя сущность может быть использована многократно на одной или нескольких диаграммах. Обычно такой прием используют, чтобы не рисовать слишком длинных и запутанных стрелок.

Стрелки (Потоки данных). Стрелки описывают движение объектов из одной части системы в другую. Поскольку в *DFD* каждая сторона работы не имеет четкого назначения, стрелки могут подходить к любой грани и выходить из любой грани прямоугольника работы. В *DFD* также применяются двунаправленные стрелки для описания диалогов типа "команда-ответ" между работами, между работой и внешней сущностью и между внешними сущностями.

Хранилище данных. В отличие от стрелок, описывающих объекты в движении, хранилища данных изображают объекты в покое. В материальных системах хранилища данных изображаются там, где объекты ожидают обработки, например в очереди. В системах обработки информации хранилища данных являются механизмом, который позволяет сохранить данные для последующих процессов.

Слияние и разветвление стрелок. В *DFD* стрелки могут сливаться и разветвляться, что позволяет описать декомпозицию стрелок. Каждый новый сегмент сливающейся или разветвляющейся стрелки может иметь собственное имя.

Построение диаграмм DFD. Диаграммы DFD могут быть построены с использованием традиционного структурного анализа, подобно тому, как строятся диаграммы IDEF0. Сначала строится физическая модель, отображающая текущее состояние дел. Затем эта модель преобразуется в логическую модель, которая отображает требования к существующей системе. После этого строится модель, отображающая требования к будущей системе. И, наконец, строится физическая модель, на основе которой должна быть построена новая система.

Альтернативным подходом является подход, популярный при создании программного обеспечения, называемый событийным разделением (*event partitioning*), в котором различные диаграммы DFD выстраивают модель системы. Логическая модель строится как совокупность работ и документирования того, что они (эти работы) должны делать. Затем модель окружения (*environment model*) описывает систему как объект, взаимодействующий с событиями из внешних сущностей. Модель окружения обычно содержит описание цели системы, одну контекстную диаграмму и список событий. Контекстная диаграмма содержит один прямоугольник работы, изображающий систему в целом, и внешние сущности, с которыми система взаимодействует. Наконец, модель поведения (*behavior model*) показывает, как система обрабатывает события. Эта модель состоит из одной диаграммы, в которой каждый прямоугольник изображает каждое событие из модели окружения. Хранилища могут быть добавлены для моделирования данных, которые необходимо запоминать между событиями. Потоки добавляются для связи с другими элементами, и диаграмма проверяется с точки зрения соответствия модели окружения.

Полученные диаграммы могут быть преобразованы с целью более наглядного представления системы, в частности работы на диаграммах могут быть декомпозированы.

Нумерация объектов. В DFD номер каждой работы может включать префикс, номер родительской работы (*A*) и номер объекта. Номер объекта – это уникальный номер работы на диаграмме. Например, работа может иметь номер *A.12.4*. Уникальный номер имеют хранилища данных и внешние сущности независимо от их расположения на диаграмме. Каждое хранилище может иметь префикс *D* и уникальный номер, например *D5*. Каждая внешняя сущность – префикс *E* и уникальный номер, например *E5*.

Порядок выполнения работы

При оформлении заказа важно проверить, существует ли такой клиент в базе данных и, если не существует, внести его в базу данных и затем оформить заказ. Оформление заказа начинается со звонка клиента. В процессе оформления заказа база данных клиентов может просматриваться и редактироваться. Заказ должен включать как информацию о клиенте, так и информацию о заказанных продуктах. Оформление заказа подразумевает чтение и запись информации о прочих заказах.

В процессе декомпозиции согласно правилам *DFD* необходимо преобразовать граничные стрелки во внутренние, начинающиеся и заканчивающиеся на внешних ссылках.

1. Декомпозируйте работу "Оформление заказов" на диаграмме *A2*. В диалоге *Activity Box Count* выберите количество работ 2 и нотацию *DFD* (рис. 2.49).

2. Щелкните по *OK* и внесите в новую диаграмму *DFD A22* имена работ:

- Проверка и внесение клиента;
- Внесение заказа.

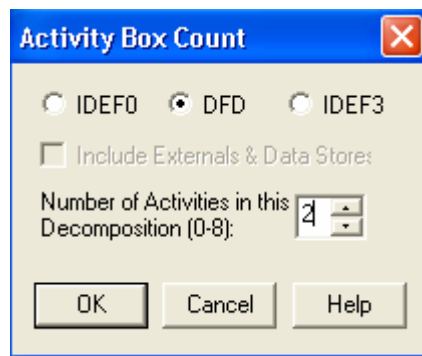




Рис. 2.49. Выбор нотации *DFD* в диалоге *Activity Box Count*

3. Используя кнопку  на палитре инструментов, внесите хранилища данных:

- Список клиентов;
- Список продуктов;
- Список заказов.

4. Удалите граничные стрелки с диаграммы *DFD A22*.

5. Используя кнопку  на палитре инструментов, внесите внешнюю ссылку:

- Звонки клиентов.

6. Создайте внутренние ссылки согласно рис. 2.50. При переименовании стрелок используйте словарь.

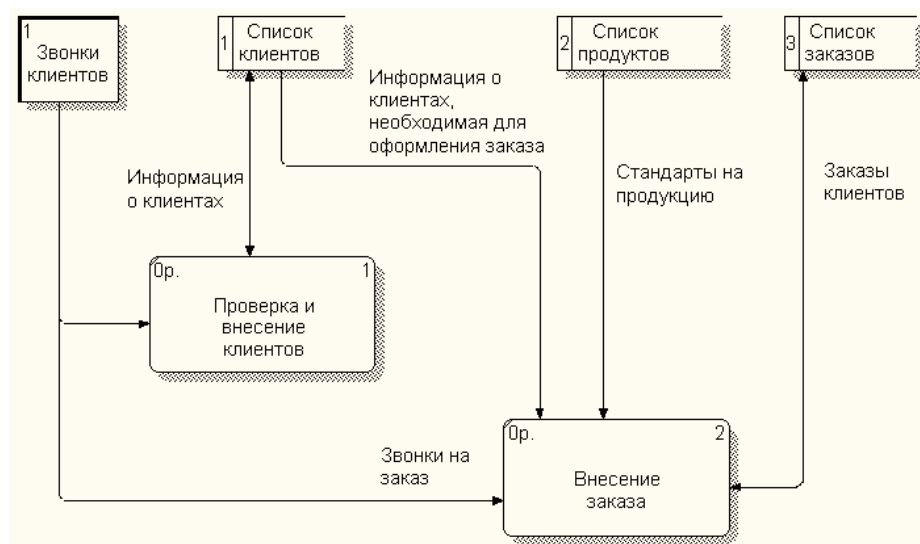


Рис. 2.50. Диаграмма A22 "Оформление заказов"

7. Обратите внимание, что стрелки "Информация о клиентах" и "Заказы клиентов" двунаправленные. Для того чтобы сделать стрелку двунаправленной, щелкните правой кнопкой по стрелке, выберите в контекстном меню пункт *Style* и во вкладке *Style* выберите опцию *Bidirectional*.

8. На родительской диаграмме A2 тоннелируйте (*Change to Tunnel*) стрелки, подходящие и исходящие из работы "Оформление заказов" (рис. 2.51).

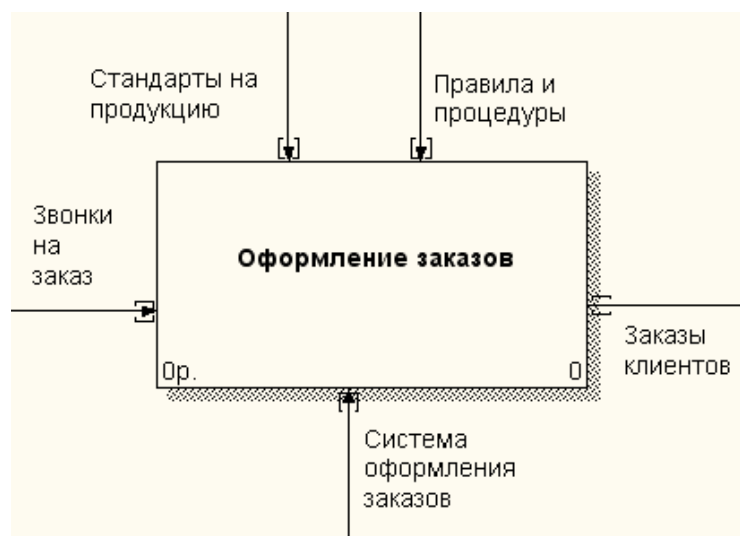



Рис. 2.51. Работа "Оформление заказов" на диаграмме A2

Некоторые стрелки с диаграмм *IDEF0* и *DFD* (не только с родительских) могут показываться на диаграмме *DFD*. Для отображения таких стрелок используется инструмент *Off-Page Reference*.

9. Декомпозируйте работу "Исследование рынка" на диаграмме A2 на диаграмму DFD. Удалите граничные стрелки. Создайте следующие работы:

- Разработка прогнозов продаж;
- Разработка маркетинговых материалов;
- Привлечение новых клиентов.

10. Используя кнопку  на палитре инструментов, внесите хранилища данных:

- Список клиентов;
- Список продуктов;
- Список заказов.

11. Добавьте две внешние ссылки:

- Маркетинговые материалы;
- Прогноз продаж.

12. Свяжите объекты диаграммы DFD стрелками, как показано на рис. 2.52.

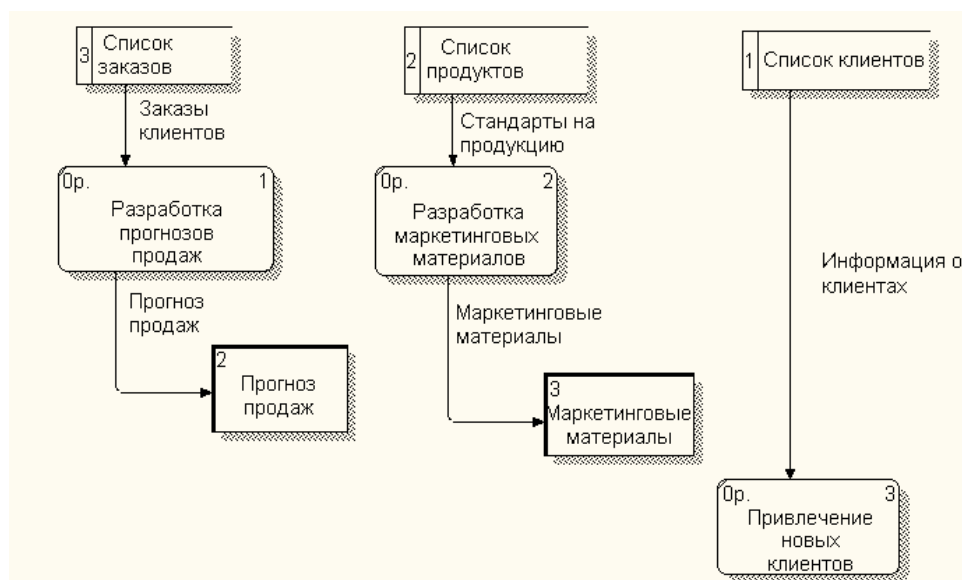


Рис. 2.52. Диаграмма A23 "Исследование рынка"

13. На родительской диаграмме A2 тоннелируйте (*Change to Tunnel*) стрелки, подходящие и исходящие из работы "Исследование рынка".

14. В случае внесения новых клиентов в работе "Проверка и внесение клиента" на диаграмме A22 "Оформление заказов" информация должна направляться к работе "Привлечение новых клиентов" диаграммы A23 "Исследование рынка". Для этого необходимо использовать инструмент *Off-Page Reference*. На диаграмме A22 "Оформление заказов" создайте новую граничную стрелку, исходящую от работы "Проверка и внесение клиента", и назовите ее "Информация о новом клиенте" (рис. 2.53).



Рис. 2.53. Стрелка "Информация о новом клиенте" на диаграмме A22

15. Правой кнопкой щелкните по кончику стрелки и выберите в меню *Off-Page Reference*. В появившемся диалоге *Off-Page Arrow Reference* (рис. 2.54) выберите в качестве диаграммы A23D "Исследование рынка". В качестве *Destination border* выберите *Input*. Выберите "Ok and Remain in current diagram".

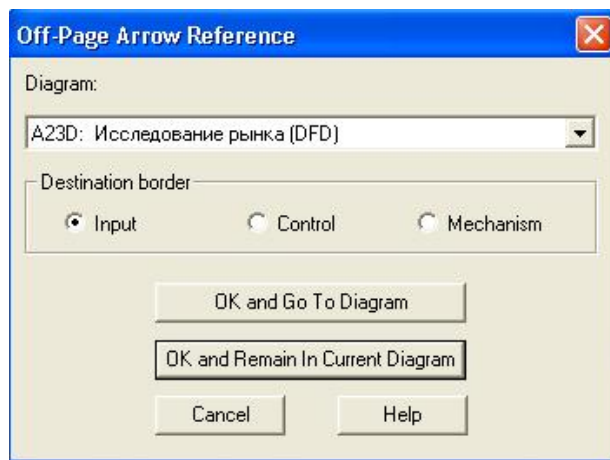


Рис. 2.54. Диалог Off-page arrow reference

16. Перейдите в меню *Model/Model Properties*, далее - во вкладку *Display*. Установите опцию *Off-Page Reference label - Node number*.

17. Перейдите на диаграмму A23D "Исследование рынка" и направьте стрелку "Информация о новом клиенте" на вход работы "Привлечение новых клиентов". Результат представлен на рис. 2.55.



Рис. 2.55. Межстраничная ссылка на диаграмме A23

Контрольные вопросы

1. Чем отличаются диаграммы *DFD* от диаграмм *IDEF0* и *IDEF3*?
2. Что такое хранилище данных?
3. Какие процессы описывают диаграммы *DFD*?
4. Как изображаются работы на диаграммах *DFD*?
5. Что такое межстраничная ссылка?

3. ИНСТРУМЕНТАЛЬНАЯ СРЕДА ERWIN

Лабораторная работа №1. Построение модели

Цель работы: Ознакомиться с CASE-средством *ERwin*. Получить навыки в построении *ERD* модели. Ознакомиться с добавлением сущностей и атрибутов в модель *ERD*.

Теоретические сведения

Диаграмма

Окно диаграммы содержит заголовок диаграммы, рабочую область и соответствующие возможности.

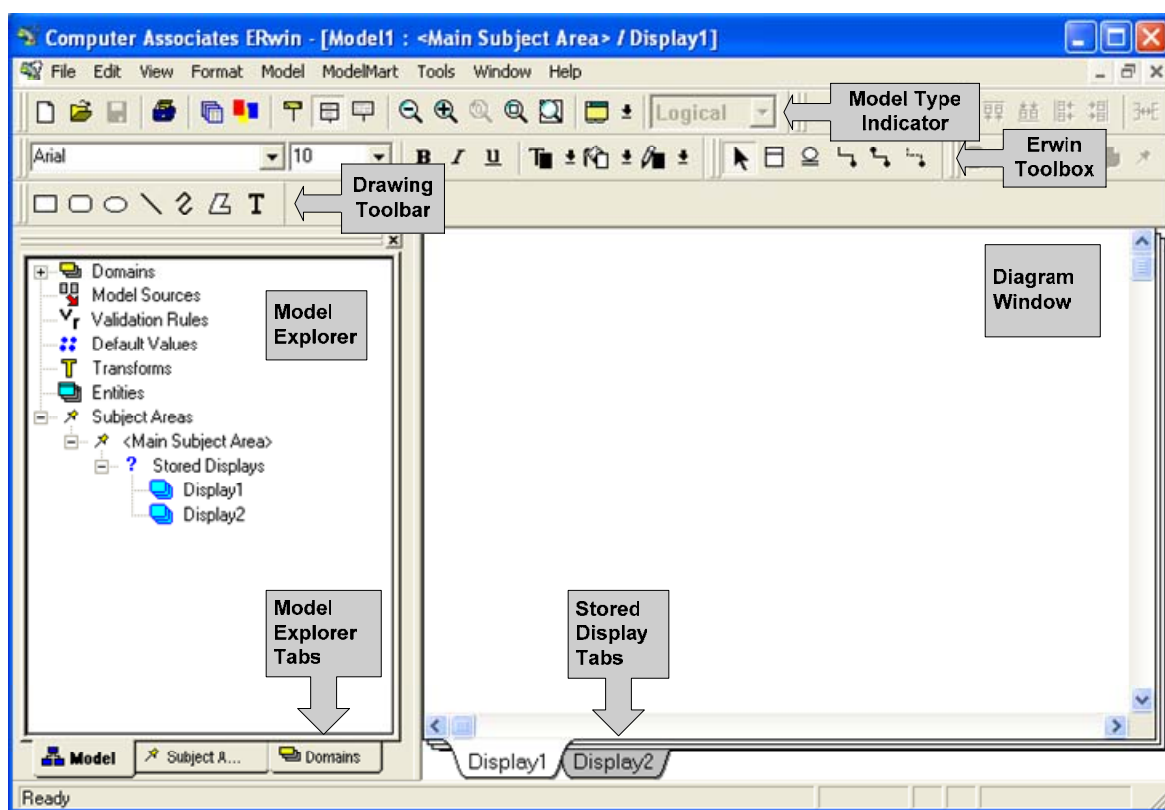


Рис. 3.1. Стандартное окно диаграммы

- *Diagram Window* – окно диаграммы, отображает графическое представление модели данных;
- *Model Explorer* – обеспечивает иерархию базового текстового представления модели данных, которая отображена в окне диаграммы; обеспечивает структурное представление модели данных и содержания. В *Model Explorer* можно увидеть другие виды модели;
- *ERwin Toolbars* – панель инструментов; установив курсор мыши над кнопкой панели инструментов, можно узнать описание выполняемой задачи;

- *Stored Display* – графическое представление области, содержащее координаты сущностей и связей и графические режимы демонстрации изображения; каждая модель данных в **ERwin** имеет загруженный дисплей, который называется "*Display1*". Его можно переименовать. Можно создать другие дисплеи для модифицирования представления модели данных.

Демонстрация окон на экране осуществляется тремя способами: *Cascade*, *Horizontal Tile* или *Vertical Tile*. Режим расположения окон можно выбрать в меню "*Windows*".

Существующий файл **ERwin** открывается командой "*Open*" из меню "*File*".

Примечание: открывать и сохранять можно только файлы с определенными расширениями. Можно использовать механизм "*Drag and Drop*" в *File Manager* для открытия одной или нескольких ранее сохраненных диаграмм. "*Drag and Drop*" используется в случаях, когда окно приложения **ERwin** полного размера или **ERwin** минимизирован до пиктограммы. **ERwin** открывает новое окно диаграммы для каждого выбранного файла.

Примечание: в сетевой среде при открытии диаграммы **ERwin** автоматически ставит на файл блокировку "только для чтения" для того, чтобы остальные пользователи не могли изменять этот файл. Однако если **ERwin** неожиданно прерывает операцию, то блокировка остается. Блокировка снимается в директории расположения файла командой DOS "*Attrib*" с параметром "*-r*".

Диаграмма сохраняется командой "*Save*" или "*Save As*". При сохранении диаграммы **ERwin** автоматически добавляет выбранное расширение к имени файла. Предыдущая версия этого файла записывается в файл с этим же именем и расширением "*BK*". Диаграмма закрывается командой "*Close*" из меню "*File*".

Выход из приложения **ERwin** выполняется командой "*Exit*" из меню "*File*".

В нижней части окна **ERwin** находится *status bar* (строка состояния), где содержится информация о выбранной функции или команде меню. Если статус активного окна – "*Ready*", то это означает, что **ERwin** готов к новой функции.

Status bar можно включать и выключать при помощи переключателя "*Status Bar*", который находится в меню "*View*". Статус активного окна, описание кнопок отображаются в *status bar*.

Примечание: **ERwin** поддерживает печать диаграмм только в полном размере. Если активное окно диаграммы изображено на экране в уменьшенном виде или в виде пиктограммы, то команда "*Print*" недоступна и изображается бледным тоном.

Модели данных

Существует два разных способа мышления и моделирования – логический уровень и физический уровень. Понятие логический уровень подразумевает, что мы мыслим в понятиях реального мира и непосредственно из него берем объекты для моделирования. Например, люди, столы, подразделения, собаки и компьютеры – это реальные вещи. Объекты, на которые мы ссылаемся на логическом уровне, должны получать имена из естественного языка, с использованием таких разделителей (пробелов, черточек и т.п.), которые имеют смысл. На логическом уровне не имеет значения, например, какой СУБД мы будем пользоваться, является ли некоторое число целым или действительным, как лучше индексировать таблицу.

То, как логические модели будут выражаться в терминах физической базы данных, включая выбор СУБД, типов данных по умолчанию и эффективных схем индексирования, принято называть физическим уровнем модели **ERwin**. **ERwin** поддерживает построение и организацию работы на этих двух уровнях для одной диаграммы.

Используя одну и ту же модель, можно говорить с конечными пользователями в понятной им терминологии и в то же время генерировать схему базы данных на языке, который может быть обработан СУБД пользователя.

Модели подразделяются на:

- *Logical* – логическая модель, которая содержит сущности, атрибуты и ключевые группы);
- *Physical* – физическая модель (база данных), которая содержит столбцы таблиц, и типы данных;

Logical/Physical – стандартная модель **ERwin**, включает логическую и физическую модели.

Model Type Indicator (указатель на тип модели) в *Toolbar* идентифицирует текущий тип модели:



Рис. 3.2. Тип модели

Указатель на тип модели может быть переключен между логической и физической моделью только для логико-физической модели. Для только логической или только физической модели тип модели появляется тусклым в списке указателя типа модели, переключение в этом случае исключено.

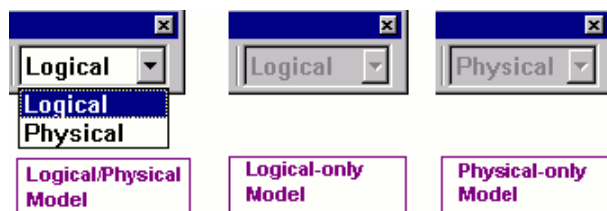


Рис. 3.3. Список указателя типа модели

Модель данных использует объекты и связи для представления логической структуры, а впоследствии будут физические таблицы в базе данных.

Команда "*Physical Order Level*" позволяет просматривать информацию о физической схеме в окне диаграммы **ERwin** (*Format/Display Level*).

Сущность

Сущность служит для представления набора реальных или абстрактных предметов (людей, мест, событий и т.п.), которые обладают общими атрибутами или характеристиками. Сущность – это "логический" объект, который в физической среде СУБД представлен таблицей.

ERwin Toolbox содержит два типа сущностей: независимые и зависимые. Независимая сущность – это сущность, экземпляры которой могут быть уникальным образом идентифицированы без определения ее связи с другой сущностью. Зависимая сущность – это сущность, экземпляры которой не могут быть уникальным образом идентифицированы без определения ее связи с другой сущностью или сущностями.

Новая сущность создается по следующему алгоритму:

1. Щелкнуть по пиктограмме независимой сущности;
2. Передвинуть крест в то место, где будет находиться новая сущность, и щелкнуть кнопкой мыши. Появится новая сущность с меткой "*E/#*", где "*E*" означает сущность, а "*#*" – уникальный номер. Номера сущностей используются только один раз и последовательно увеличиваются, начиная с единицы.

Быстрое открытие редактора

Когда нужно ввести большой объем информации о сущностях и связях, может потребоваться более быстрый способ входа в различные редакторы **ERwin** для ввода данных. **ERwin** позволяет быстро открывать редактор с помощью объектно-ориентированных сокращенных меню.

Объектно-ориентированные сокращенные меню **ERwin** – это меню, содержащие только те команды, которые применимы в данный момент для того объекта, на который указывает курсор. Есть сокращенное меню для диаграмм, которое позволяет изменить режим демонстрации объектов на диаграмме. Для входа в сокращенное меню нужно установить курсор на объект и затем нажать правую кнопку мыши. При нажатии на правую кнопку мыши появляется меню в месте нахождения курсора. Выбрать несколько объектов как группу можно следующим образом:

1. Выбрав инструмент "указатель", установить курсор на диаграмму слева от первого объекта – сущности или текстового блока – который следует включить в группу.

2. Нажать левую кнопку мыши и, не отпуская ее, передвигать мышь вправо вдоль группы сущностей, которые нужно выбрать. Сущности и текстовые объекты, которые полностью попали в выделенную часть, будут выбраны.

3. Когда все объекты, которые нужно выбрать, попали в прямоугольник, следует отпустить кнопку мыши. **ERwin** выделит имена сущностей и текстовых блоков, включенных в группу.

Для отмены выбора отдельного объекта следует установить курсор на этот объект, нажать "**CTRL**" и щелкнуть кнопкой мыши. Для отмены выбора всех объектов следует щелкнуть кнопкой мыши по любому месту фона диаграммы.

Вставка

Сущности можно вставить в ту же самую диаграмму, из которой они были скопированы, в существующую диаграмму, в новую пустую диаграмму.

Если не требуется вставка скопированных объектов в ту же самую диаграмму, то следует открыть ту диаграмму, в которую нужно вставить сущности.

При вставке объекта из одной диаграммы в другую, он располагается на том же месте (физически), на котором он был в исходной диаграмме. Это означает, что если скопируете сущность, которая находилась в правом нижнем углу страницы 1 диаграммы, и затем скопируете ее на диаграмму, находящуюся в другом окне **ERwin**, то она окажется в правом нижнем углу на странице 1 этой новой диаграммы. Если на новой диаграмме в данный момент не видна нижняя часть страницы 1, то вставленную сущность увидеть будет нельзя, пока не прокрутите диаграмму или не уменьшите ее размер.

Копирование сущностей

Если возникнет потребность в копировании сущности на новое место в этой же диаграмме, то можно использовать "*Drag and Drop*" для дублирования выбранной сущности или множества сущностей. Эту возможность можно использовать при создании представлений для презентации, когда не нужно применять правила нормализации.

Функция "Go To"

Функция позволяет задать имя сущности и предоставить возможность **ERwin** автоматически прокрутить диаграмму до заданной сущности и вывести ее на экран.

Для поиска сущности с использованием "*Go To*" следует выполнить следующие действия:

1. Дать команду *"Go To"* в меню *"Edit"*. **ERwin** откроет окно-диалог *"Go To"*, где находится окно-список со списком имен всех сущностей в текущей области.

2. Выбрать нужную сущность. Поставьте метку в окне *"Open Editor"* для того, чтобы прокрутить список до искомой сущности и автоматически открыть текущий редактор для этой сущности.

3. Нажмите *"OK"*, чтобы закрыть окно-диалог. При закрытии редактора *"Go To"* **ERwin** прокручивает диаграмму так, что выбранная сущность оказывается точно в окне.

4. Примечание: вы можете быстро в любой момент войти в окно-диалог *"Go To"*, нажав *"Ctrl"+"g"*.

Для того чтобы вырезать объект на диаграмме **ERwin** следует выполнить следующие действия:

1. Откройте диаграмму, содержащую объекты, которые надо вырезать.

2. Выберите сущность, которую нужно скопировать (можно выбрать группу сущностей).

3. Дайте команду *"Cut"* в меню *"Edit"* или нажмите *"Ctrl"+"X"*.

При удалении объектов из диаграммы **ERwin** помещает их в *Windows Clipboard*. Можно вставить удаленные объекты в другую диаграмму или в другое приложение, используя команду *"Paste"* меню *"Edit"*.

Копирование

Если между копируемыми сущностями есть связи, то внешние ключи, мигрировавшие через эти связи, копируются вместе со связью. Если копируется только дочерняя сущность, а родительская не копируется, то связь между ними не копируется и внешние ключи, которые мигрировали в дочернюю сущность, становятся атрибутами-"сиротами". При вставке дочерней сущности без родительской эти атрибуты становятся собственными атрибутами вставленной сущности. Если после этого будет создана новая связь, для которой эта сущность является дочерней и через которую снова передаются те же самые атрибуты, **ERwin** распознает эту ситуацию и заменит собственные атрибуты дочерней сущности атрибутами внешнего ключа.

Примечание: вырезанные или скопированные объекты остаются в clipboard даже после вставки их на другое место. Это удобно, если требуется сделать несколько копий.

Функции **ERwin** *"Copy"* и *"Paste"* способствует повторному использованию моделей и ускоряет слияние моделей, созданных несколькими проектировщиками баз данных, в одну модель. Когда копируется объект, то вся информация об этом объекте "вспоминается" и тоже копируется. Поэтому при копировании двух сущностей, между которыми есть связь, в другую диаграмму копируются не только сами сущности и связь, но и другая связанная с ними информация: определения, *notes*, примеры экземпля-

ров, глагольные фразы, ограничения целостности, данные физической схемы и индексные данные.

Когда копируется родительская и дочерняя сущность связи, то сама связь (или связи) между этими сущностями тоже копируется. Для копирования сущности из одной диаграммы **ERwin** в другую следует выполнить следующие действия:

1. Выберите сущность, которую нужно скопировать, или, используя способы множественного выбора, выберите группу сущностей.

2. Дайте команду "*Copy*" в меню "*Edit*" или нажмите "*Ctrl*" + "*C*". Вы войдете в окно-диалог "*Copy*", который позволяет копировать выбранные сущности как изображение типа "*bitmap*" или как элементы модели **ERwin** (последнее делается по умолчанию).

3. Откройте диаграмму, в которую хотите скопировать.

4. Нажмите кнопку "*Paste*" в меню "*Edit*" для того, чтобы скопировать выбранные сущности со всей информацией и определениями **ERwin**.

Примечание: связи копируются только в том случае, когда и родительская и дочерняя сущности включены в множество, выбранное для копирования.

Изменение имени

При работе с диаграммой может возникнуть потребность в изменении логического имени атрибута или физического имени, которое было задано для атрибута. Можно сделать это для того, чтобы присвоить "более говорящее" имя, привести имя в соответствие с принятым соглашением, или исправить орфографическую ошибку. При обратном проектировании (*Reverse Engineering*) модели можно заменить логические имена чем-то более наглядным, чем те физические имена полей, которые были перенесены из схемы базы данных.

Если имя атрибута, которое подлежит изменению, передается через одну или более дочерних сущностей в качестве внешнего ключа, то этот процесс окажется непростым. Нежелательно изменять имя атрибута в редакторе "*Entities*", поскольку оно будет обработано таким образом, как если бы был удален исходный атрибут и задан новый; пропадут все *notes* и определения, испортятся имена ролей или любой специальный заданный порядок. Вместо этого можно просто изменить имя атрибута без изменения его других характеристик. Для безопасного изменения логического и (или) физического имени атрибута следует применять редактор "*Attributes*" (*Model/Attribute*).

Редактор "*Attributes*"

В редакторе имеются закладки:

- **General.** В блоке "*Domain*" можно задать режим сортировки: в алфавитном порядке (*Alphabetically*), иерархически (*Hierarchically*), выбрать общий тип атрибута (*Number*-числовой, *Datetime*-временной,

Blob-связанный с мультимедиа, *String*-строковый), иконку, определить атрибут как первичный ключ;

- **Datatype** – тип данных. Можно выбрать конкретный тип атрибута, правило ограничения целостности, значение по умолчанию

*Valid:** - ограничение целостности

*Default:** - значение по умолчанию;

- **Definition** – описание атрибута;
- **Note**. Редактор позволяет добавлять замечания об одном или нескольких атрибутах сущности. Назначение этого редактора – связывание с атрибутами дополнительных замечаний, которые не вошли в определения.
- **UDP** – определение пользовательских свойств;
- **Key Group** – для определения группового ключа;
- **History** – содержится история атрибута.

В редакторе имеются следующие кнопки:

- **New** – создание нового атрибута;
- **Rename** – переименование атрибута;
- **Delete** – удаление атрибута;
- **Reset** – восстановление значения домена по умолчанию для одной или более характеристик колонки.

Все сущности представлены в списке "*Entity*". Из этого списка можно выбрать любую сущность для редактирования, информация о выбранной сущности будет перенесена в редактор. Данные, введенные до этого, сохраняются.

Редактор "*Entities*"

Редактор используется для ввода определения сущности. Эти определения полезны на логическом уровне, поскольку они помогают людям, читающим модель, понять, что это за объект. Они полезны и на физическом уровне, поскольку их можно экспортировать как часть схемы и использовать в реальной базе данных.

Примечание: для отображения определения на экране следует просматривать диаграмму на уровне определений (*Definition*).

В строке выбора "*Entity*" следует выбрать одно из имен сущностей для переноса информации о сущности в редактор. В поле "*Name*" отражается имя выбранной сущности. Для изменения имени сущности следует изменить поле "*Name*". По мере перехода от одной сущности к другой, все предыдущие данные сохраняются.


В редакторе имеются закладки:

- **Definition** – определение сущности;
- **Note, Note2, Note3** – можно добавлять замечания об одном или нескольких атрибутах сущности. Назначение – связывание с атрибутами дополнительных замечаний, которые не вошли в определения;

- **UDP** – пользовательские определения свойств;
- **Icon** – выбор иконки для сущности;
- **History** – история сущности.

Присвоить иконку сущности выполняется по следующей схеме:

1. Щелкнуть правой кнопкой мыши по сущности и дать команду "Definition" из меню *Entities Properties*.

2. В *Large Icon* нажмите на кнопку . Откроется окно "Icons". Нажмите на кнопку "Import". Откроется окно-диалог *Open File*. Выберите дисковод и директорию, в которой находится рисунок, и затем щелкните по имени *bitmap*, который хотите изобразить на выбранной сущности (например, *C:\WINDOWS\Movie.BMP*). Можно выбрать иконку из выпадающего списка *Small Icon*.

Порядок выполнения работы

1. Запустите **ERwin**. Выберите "Create a new mode".
2. Создание модели. В меню "File" **ERwin** выберите "New". Появится диалог "Create Model – Select Template". Диалог будет выглядеть следующим образом:

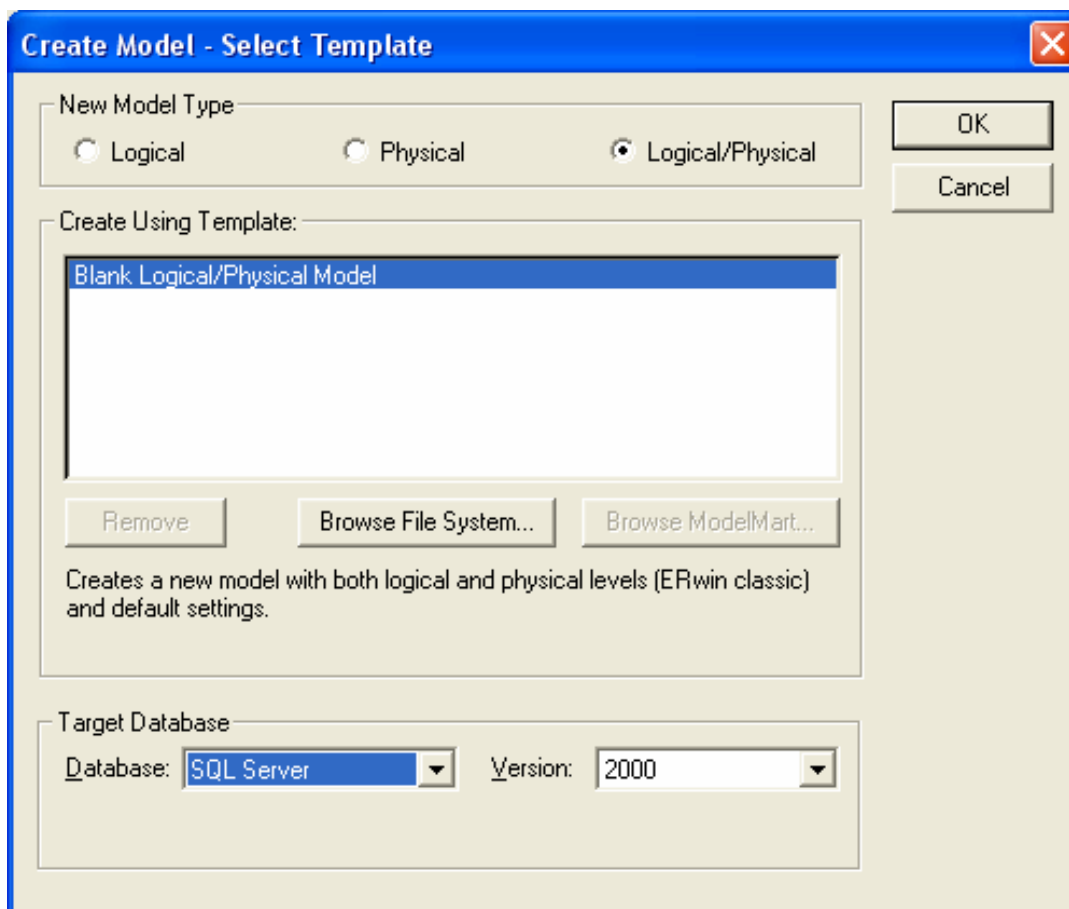


Рис. 3.4. Диалог создания модели

В качестве типа для новой модели выберите *"Logical/Physical"*. Мы будем использовать шаблон по умолчанию *"Blank Logical/Physical Model"* и базу данных *"SQL Server 2000"* для физической модели. Щелкните *"OK"* и **ERwin** откроет новое окно диаграммы.

3. Зайдите в меню *"File"* **ERwin** и выберите *"Save As dialog"*. Введите имя для модели *"My ERwin Model"*. **ERwin** добавляет автоматически расширение *".er1"* для модели данных.

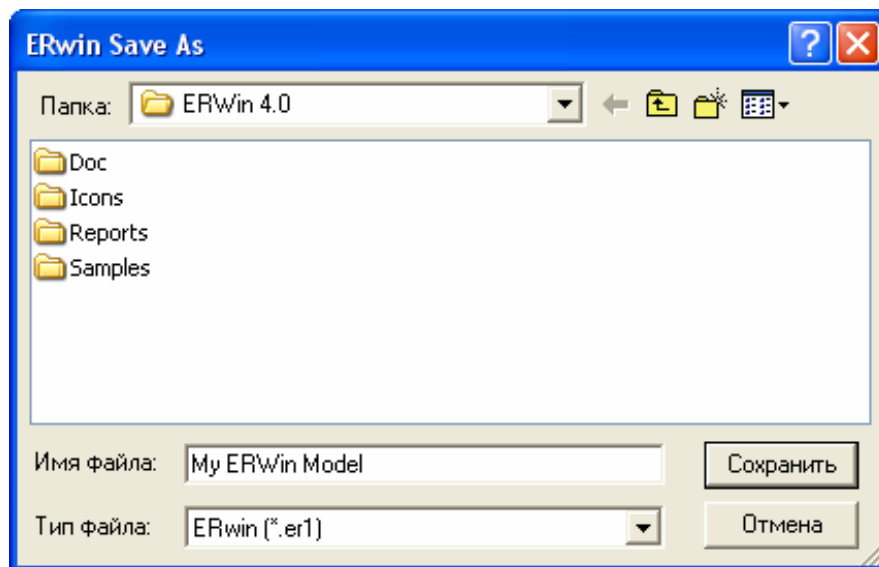



Рис. 3.5. Сохранение модели

Щелкните *"Save"* для возврата в окно диаграммы. Имя, которое указали в диалоге *"Save As dialog"* появляется в области названия **ERwin**.

4. Добавление объектов. Кликните по объектному средству . Затем кликните в любом месте окна диаграммы, чтобы установить первый объект. По умолчанию этот объект будет назван *"E/1"*. Добавьте еще два объекта (они будут названы *"E/2"* и *"E/3"*)

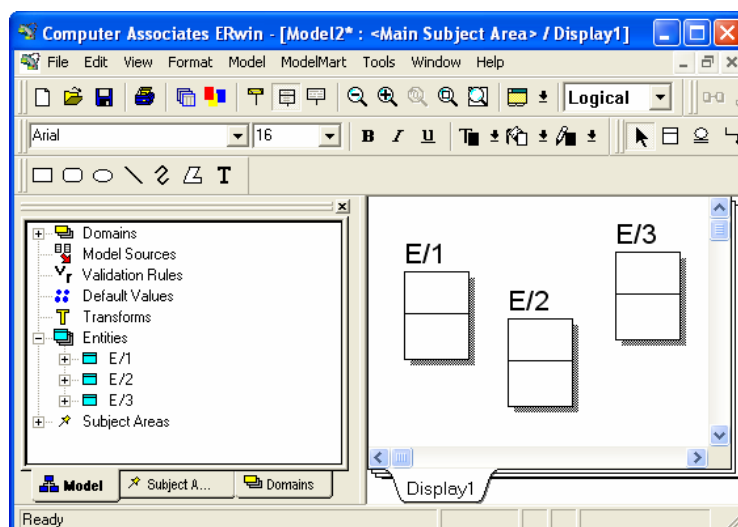


Рис.3.6. Модель с добавленными объектами

5. Сохраните модель. Сущности, которые были созданы в окне диаграммы, появятся в *Model Explorer*.

6. Задание имени сущностям. Можно назвать сущность непосредственно в окне диаграммы или в *Model Explorer*.

1 способ: В окне диаграммы расположите сущность с именем "E/1". Сделайте двойной щелчок по имени сущности. Появится блок редактирования вокруг имени. Наберите "CUSTOMER" над именем сущности по умолчанию и кликните мышкой вне блока редактирования.

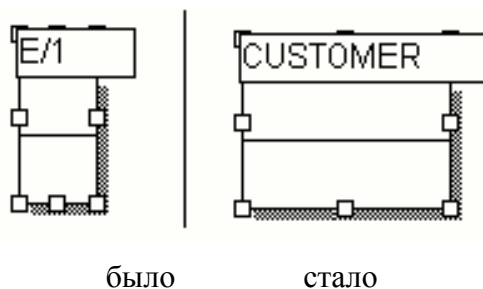


Рис. 3.7. Редактирование имени сущности

2 способ: В *Model Explorer* щелкните правой кнопкой мыши по "E/2" и из контекстного меню выберите "Rename" (переименовать). Наберите "ORDER". Щелкните мышкой вне блока редактирования.

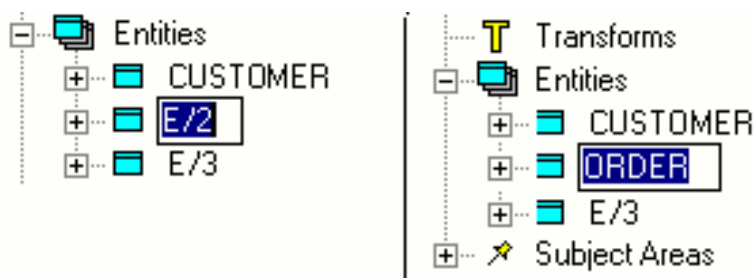


Рис. 3.8. Переименование имени сущности

Одним из вышеприведенных способом переименуйте последнюю сущность на "PRODUCT". После сделанных изменений диаграмма должна быть схожа с нижеприведенной:

Сохраните модель.

7. Добавление атрибута. Можно добавлять атрибуты к сущностям непосредственно в окне диаграммы или в *Model Explorer*. Применить оба метода:

1 метод: В окне диаграммы щелкните по "CUSTOMER". Появится редактор "Attributes" с именем сущности "Entity". Добавьте первый атрибут, являющийся первичным ключом. Для этого нажмите "New", наберите "Number" вместо имени по умолчанию. Поставьте галочку "Primary key". Нажмите OK.

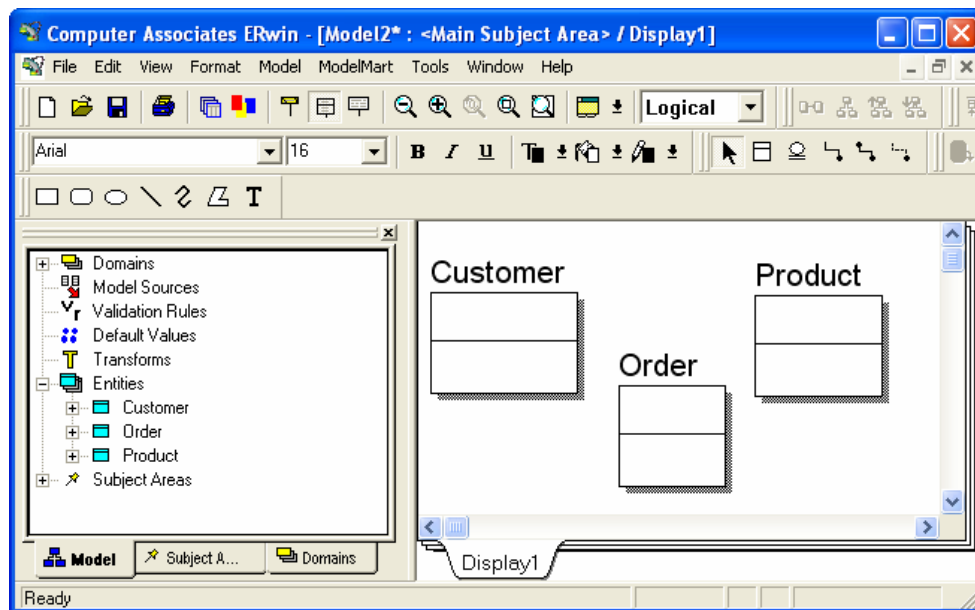


Рис. 3.9. Диаграмма с именованными сущностями

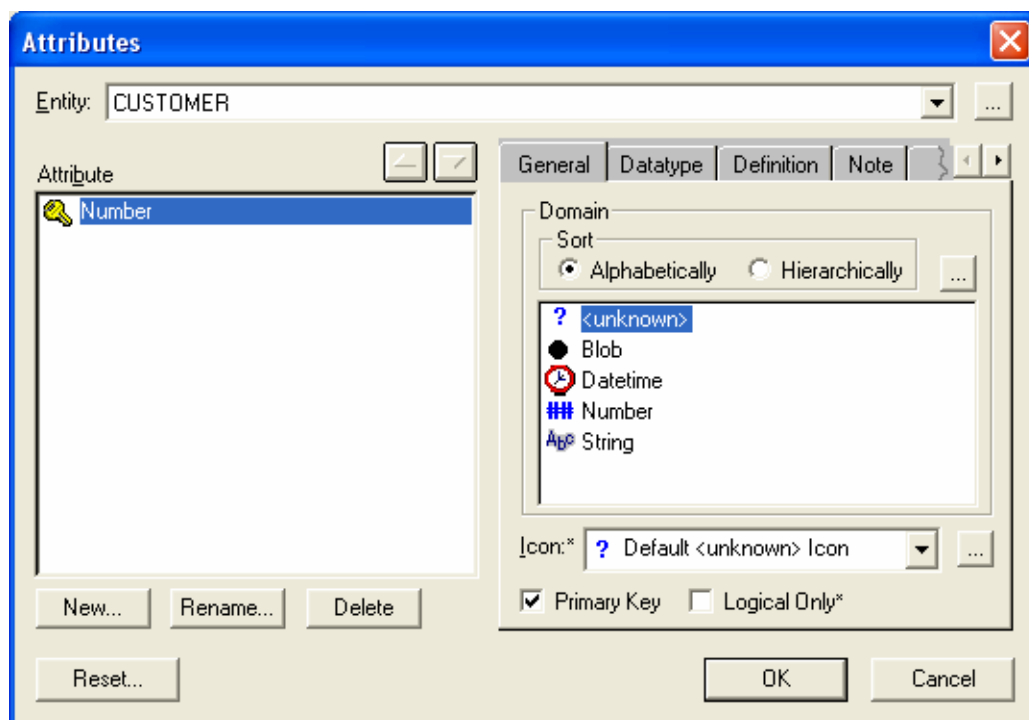


Рис. 3.10. Добавление нового атрибута (1-ый метод)

Чтобы добавить другой первичный ключ, нажмите *"Enter"* и выше строки на объекте появится блок редактирования. Для того, чтобы добавить не первичный ключ, нажмите *"Tab"* и блок редактирования появится ниже.

2 метод: Добавьте атрибут в *Model Explorer*. Для этого щелкните в плюсе около *"CUSTOMER"*. Это расширит объектный список. Щелкните

правой кнопкой мыши по *"Attributes"* и выберите *"New"* из контекстного меню. Атрибут *"New Attribute"* появится под *"Number"*. Переименуйте *"New Attribute"* на *"Name"*.

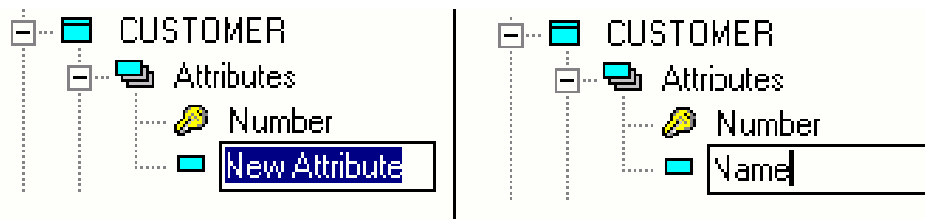


Рис. 3.11. Добавление нового атрибута (2-ой метод)

После сделанных изменений диаграмма должна быть похожей на нижеприведенную диаграмму:

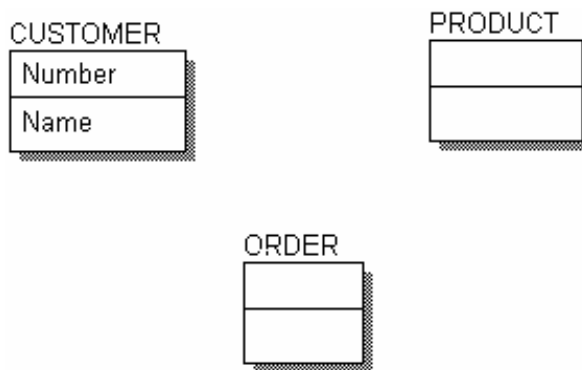



Рис. 3.12. Измененная диаграмма

8. Добавьте к сущности *"CUSTOMER"* атрибут *Price, Phone Number*. Определите этот атрибут как первичный ключ. Для этого в *Model Explorer* двойным щелчком по имени атрибута (или двойным щелчком по сущности *"CUSTOMER"*) откройте редактор атрибутов *"Attributes"*. В закладке *"General"* отметьте меткой *Primary Key*. Задайте новый тип данных для атрибута *"PRICE"*. Для этого откройте вкладку *"Datatype"* (тип данных) и измените тип данных на любой другой, который присутствует в списке *"Datatype"*. В правом нижнем углу нажмите *"OK"* для выхода из редактора.

9. Удаление атрибута. Удалите атрибут *"PRICE"*. Для этого воспользуйтесь одним из способов:

1 способ: В *Model Explorer* выделите атрибут *"PRICE"* и нажмите кнопку *"Delete"* (или щелкните правой кнопкой мыши по атрибуту и в сокращенном меню выберите *"Delete"*). Далее будет запрошено подтверждение на удаление атрибута. Нажмите кнопку *"Yes"*.

2 способ: В окне диаграммы выделите атрибут *"PRICE"* (при первом щелчке выделится сущность, а при втором – нужный атрибут) и нажмите кнопку *"Delete"*. Далее будет запрошено подтверждение на удаление атрибута. Нажмите кнопку *"Yes"*.

10. Создание иконки для атрибутов сущности. Создадим иконку для атрибутов сущности "CUSTOMER". В редакторе "Attributes" откройте закладку "General". Справа от поля "Icon:*" нажмите на . Откроется окно-редактор "Icons". Нажмите на кнопку "Import". Появится окно "ERwin Open File", из которого откройте файл с расширением "bmp". В окне "Icons" появится выбранная иконка. Ее можно переименовать, нажав на кнопку "Rename". Нажмите на кнопку "OK" для выхода из редактора "Icons". После выхода из редактора "Attributes" атрибуту будет присвоена иконка. Иконку можно увидеть в режиме просмотра "Attribute" (в меню "Format/Display Level") и при метке "Attribute Icon" в меню "Format/Entity Display".

11. Описание атрибутов сущностей. Откройте закладку "Definition" редактора "Attributes". В окне "Definition" введите описание для двух атрибутов сущности "CUSTOMER" соответственно: Номер (ID) и Телефонный номер.

12. Описание для сущностей. Зайдите в редактор "Entities". Выберите сущность "CUSTOMER" из списка "Entity". Откройте закладку "Definition" и введите описание для сущности: "Таблица клиентов". Затем введите описание для сущностей "ORDER" (Заказы) и "PRODUCT" (Продукты), выбрав нужную сущность из списка "Entity".

13. Включение в группу объектов сущностей. Включим все объекты в группу:

- выбрав инструмент "указатель", установите курсор на диаграмму слева от первого объекта (сущности);
- Нажмите левую кнопку мыши и, не отпуская ее, передвигайте мышь вправо вдоль группы сущностей: "CUSTOMER", "ORDER".

Примечание: сущности и текстовые объекты, которые полностью попали в выделенную часть, будут выбраны.

Когда все объекты, подлежащие выбору, попали в прямоугольник, отпустите кнопку мыши. **ERwin** выделит имена сущностей и текстовых блоков, включенных в группу.

Примечание: для того чтобы отменить выбор отдельного объекта, установите курсор на этот объект, нажмите "CTRL" и щелкните кнопкой мыши. Для отмены выбора всех объектов щелкните кнопкой мыши по любому месту фона диаграммы.

14. Добавление объекта "PRODUCT" к уже выбранной группе. Для добавления объекта к уже выбранной группе следует выполнить следующие действия:

- Выбрав инструмент "указатель", установите курсор на объект;
- После этого нажмите "CTRL" или "SHIFT" и щелкните кнопкой мыши. **ERwin** выделит имена сущностей и текстовых блоков, включенных в группу.

Примечание: при множественном выборе символы связи подтипа (*subtype*), рассматриваются, как сущности и должны быть включены в выбранное множество. В этом случае следует нажать "*SHIFT*" и одновременно щелкнуть кнопкой мыши по символу связи подтипа для добавления его к выбранному множеству.

15. С помощью функции "*Go To*" найдите объект "*ORDER*".

16. Скопируйте сущность "*Customer*" и вставьте в ту же самую диаграмму.

17. Измените логическое и физическое имена скопированной сущности "*Customer*" на "*NEW*". Сохраните модель и закройте ее.

Контрольные вопросы

1. Основная цель использования *ERWin*.
2. Какие бывают виды представления модели данных?
3. Что такое сущность?
4. Что такое атрибут?
5. Для чего существуют разделительная полоса в изображении сущности?

Лабораторная работа №2. Хранимые изображения

Цель работы: Ознакомится с *Model Explorer*, с хранимым изображением.

Теоретические сведения

Model Explorer обеспечивает структурное представление модели данных и содержания. *Model Explorer* позволяет создавать, отображать, управлять и модифицировать модель, используя "*Subject Areas*" – подчиненную область модели или "*Domains pane*" – подокно областей.

Специфические объекты из модели появляются под общим объектным типом. Для переключения на другое подокно нужно щелкнуть в нижней части *Model Explorer*.

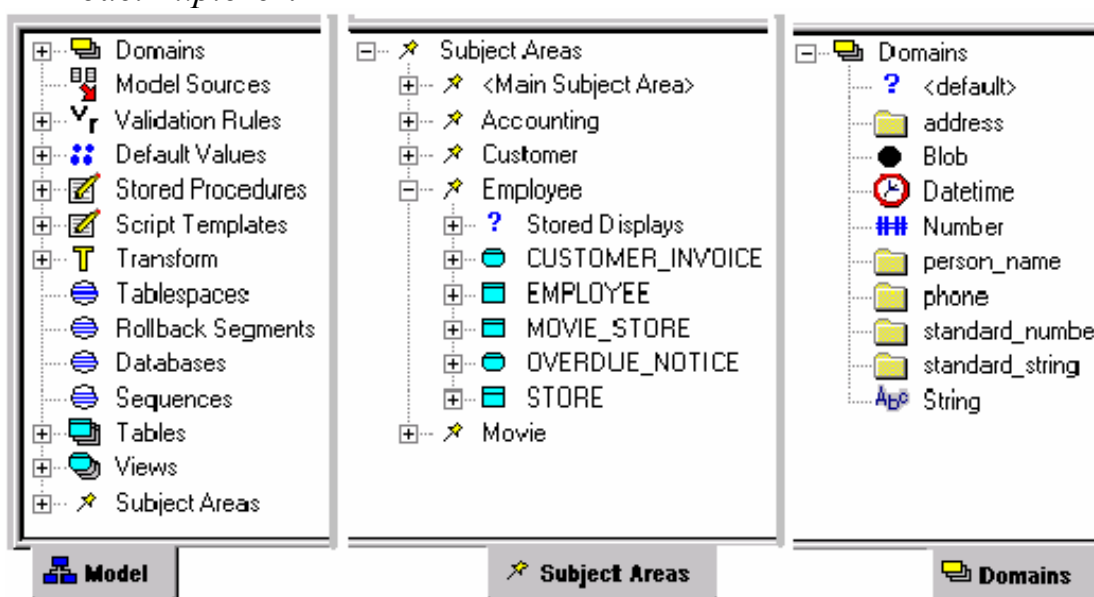


Рис. 3.13. Окна в *Model Explorer*

При совершении изменений в объекте логической модели, графическое представление модели немедленно корректируется.

Model Explorer обеспечивает целый диапазон полезных характеристик, которые направлены на создание и модифицирование модели данных как, например:

- создание новых объектов;
- переименование существующих объектов (*Rename*);
- переход к объектам в окне диаграммы (*Go To*);
- открытие редакторов для просмотра или изменения свойств объектов (*Properties*);
- изменение свойств объектов в *Model Explorer* для создания их в окне диаграммы;
- перемещение, копирование и удаление объектов.

Примечание: для работы с *Model Explorer* дисплейная опция *Model Explorer* должна быть отмечена в меню *View* (Вид).

При щелчке правой кнопкой мыши по объекту *Model Explorer* отображается контекстное меню, которое включает доступные опции для этого объекта. Тип объекта определяет доступные опции в контекстном меню:

- *New* – создается новый объект, который появляется в окне диаграммы;
- *Delete* – удаляется объект из модели данных;
- *Rename* – для изменения имени объекта ;
- *Properties* – свойства объекта *Model Explorer*, которые открываются для редактирования.

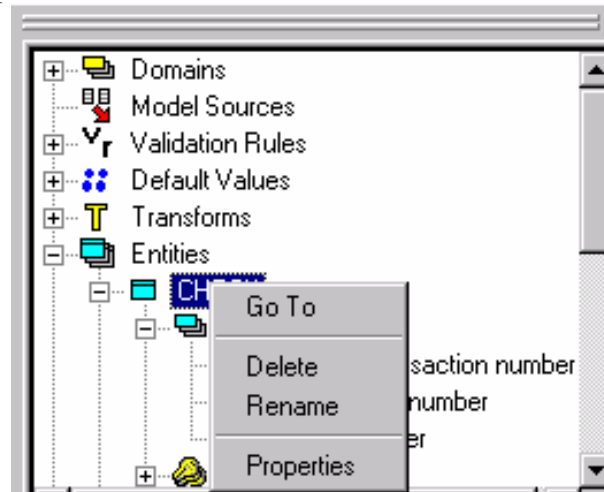


Рис. 3.14. Подокно *Model*

Подокно модели *Model Explorer* включает все объектные типы, которые появляются в текущей модели данных, основанной на типе модели и целевом сервере. Например, если тип модели логический, то *Model Explorer* не содержит физических объектов. Аналогично, если тип модели физический, то *Model Explorer* не может хранить логические объекты.

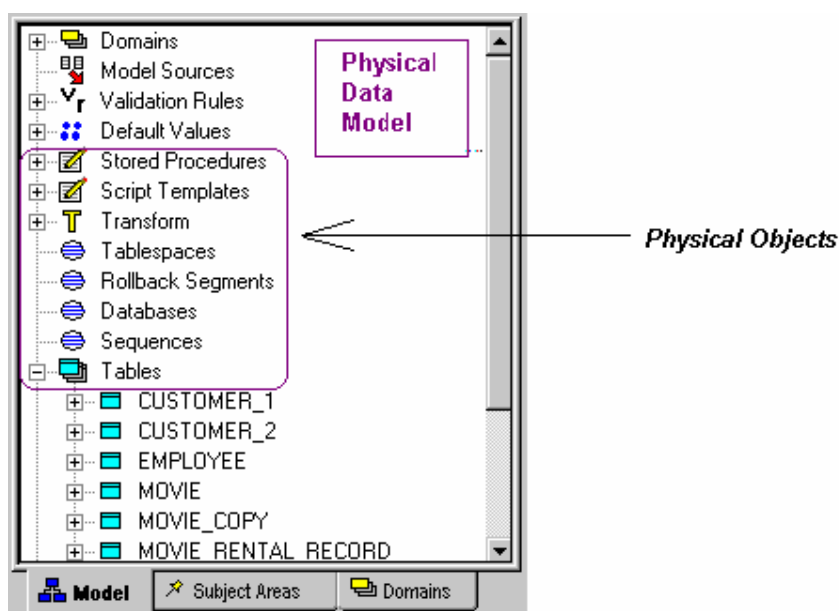


Рис. 3.15. Физические объекты

Подокно *Subject Areas* (подокно объектной области)

По умолчанию, каждая модель данных **ERwin** имеет главную предметную область, которая включает в себя все объекты модели. Есть возможность создать другие подчиненные области для разделения модели на меньшие управляющие части. В *Model Explorer* подокно объектной области отображает сортировку моделей предметной областью. Каждая предметная область может быть расширена для того, чтобы видеть список компонентов предметной области. Можно создать новую предметную область непосредственно в *Model Explorer*:

1. Щелкнув правой кнопкой мыши по "*Subject Areas*", выбрать опцию "*New*" из контекстного меню.

2. В блоке редактирования внизу списка ввести имя новой предметной области в блоке.

По умолчанию исходная модель данных является главной областью. При создании другой области выбираются объекты, которые должны быть включены в нее, и присваивается ей имя, которое описывает ее назначение. Предметная область может быть создана из меню "*Subject Areas*" (*Model/Subject Areas/*) с помощью нажатия на кнопку "*New*".

Для добавления элементов в новую предметную область следует перетащить объект или таблицу из главной предметной области в новую предметную область.

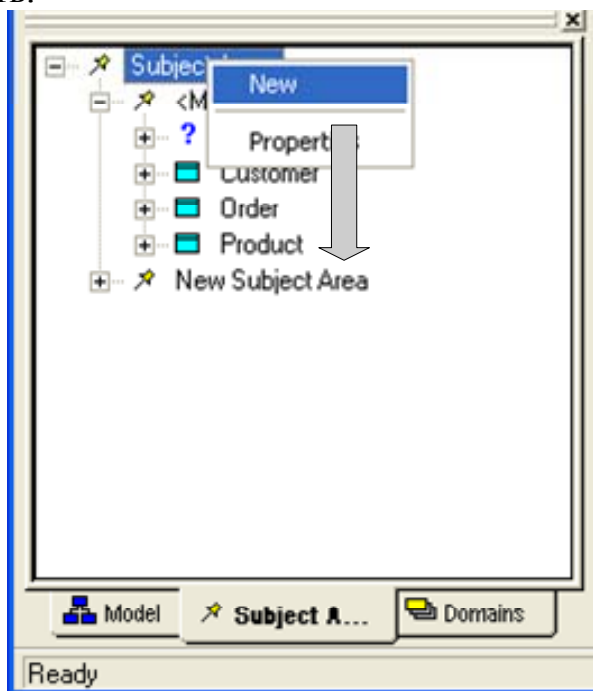


Рис. 3.16. Создание предметной области

Примечание: элементы предметной области просто ссылаются на объекты в главной предметной области, поэтому изменения автоматически

относятся к объекту в каждой предметной области, к которой объект относится (является частью главной предметной области).

Подокно областей (Domains)

В **ERwin** область является объектом, который помогает ускорять создание атрибутов и колонок. Область подобна шаблону, поскольку она определяет все свойства, которые будут унаследованы любым атрибутом или колонкой, созданной из области. **ERwin** включает набор встроенных областей, которые можно модифицировать по требованию пользователя. Можно добавлять новые области для моделирования.

Подокно областей включает все области для текущей модели, которые включают все встроенные и созданные пользователем области. Можно перетащить область от *Model Explorer* в сущность для добавления атрибута или столбца.

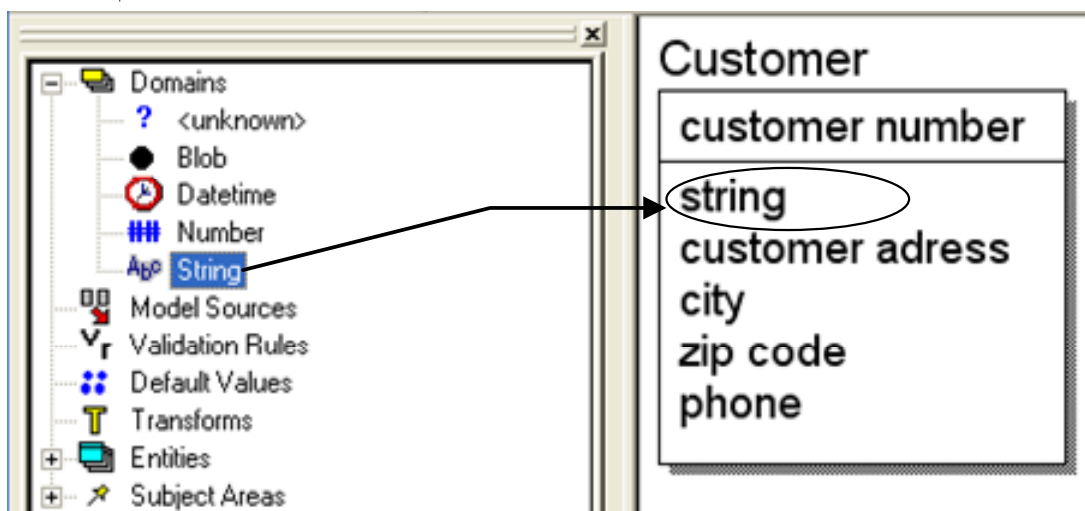


Рис. 3.17. Добавление атрибута или столбца

При создании новой области или изменении имеющейся области, работая в редакторе "*Subject Area*", **ERwin** сохраняет эти изменения при нажатии "OK" при выходе из редактора.

Примечание: **ERwin** не разрешает изменять имя главной области. Нельзя удалять из нее сущности в редакторе "*Subject Area*".

При наличии нескольких областей для одной модели данных можно переходить из одной области в другую. Переход выполняется следующим образом: находясь в окне "*Subject Areas*" диаграммы **ERwin**, в блоке "*Subject Area*" щелкнуть по области, с которой нужно работать.

ERwin показывает выбранные области в окне диаграммы.





Редактор "*Subject Areas*"

В редакторе "*Subject Areas*" присутствуют следующие закладки:

- *General* – можно ввести имя автора (*Author*), или изменить;
- *Members* – содержит объекты, до и после включения.

Список с правой стороны редактора показывает, какие сущности и текстовые блоки доступны из исходной диаграммы, но не были присвоены той области, с которой идет сейчас работа.

Список с левой стороны редактора показывает, какие сущности и текстовые блоки присвоены той области, с которой идет сейчас работа.

Нажатием на кнопку  выбираются все объекты в новую область из окна "*Available Objects*" в окно "*Included Objects*". Чтобы выбрать один или несколько объектов, нужно в окне "*Available Objects*" выделить объект, нажать на кнопку  и выбранный объект будет включен. Чтобы исключить из области объект, нужно в окне "*Included Objects*" выбрать объект, который нужно исключить, нажать на кнопку  и объект будет исключен из области. Чтобы исключить все объекты из области, нужно нажать на кнопку  и все объекты будут исключены. Для создания или обновления области нажмите на кнопку "*OK*".

Примечание: Когда сохраняется файл модели данных, *ERwin* сохраняет все связанные с ним области вместе, как один файл.

- *Definition* – можно ввести определение (описание) области, или изменить;
- *UDP* – определение пользовательских свойств.

ERwin сохраняет все области вместе с диаграммой главной области, когда сохраняется файл модели данных. Нельзя сохранить область как отдельный файл.

Хранимое изображение

Хранимые изображения создаются, модифицируются и удаляются в редакторе "*Stored Displays*" (*Format/Stored Displays Settings*).

Хранимое изображение создается посредством нажатия на кнопку "*New*" с последующим вводом имени в окне "*New Stored Displays*" в строке "*Name*". Присвоенное имя переносится на закладку хранимого изображения в окне диаграммы.

Редактор "*Stored Displays*"

Редактор имеет следующие вкладки:

- *General*;
 - Поле "*Author*" – для ввода имени автора.

Можно установить вид линий связи в блоке "*Relationship lines*":

- *Orthogonal* – линии связи ортогональные;
- *Diagonal* – линии связи диагональные

Хранимое изображение может содержать связи (называемые висящими связями), для которых не включается либо родительская, либо дочерняя сущность. По умолчанию *ERwin* не показывает линию связи, если это висящая связь. При создании области с меньшим числом сущностей, чем в главной области, может появиться одна или несколько висящих связей. В этом случае можно установить режим "*Show Dangling Relationship*" для по-

каза висящих связи на хранимом изображении, или отключить этот режим для скрытия висящих связей (в том числе и на хранимом изображении с именем *"Display1"*, которое автоматически создает **ERwin**).

Примечание: когда создается хранимое изображение, содержащее висящие связи, может оказаться полезным показать висящие связи на одном хранимом изображении и спрятать их на другом. Поскольку главная область всегда содержит полный набор сущностей, она не может содержать висящих связей.

- *Logical* – содержит установки для логической модели;
- *Physical* – содержит установки для физической модели;
- *Definition* – описание;
- *UDP* – определение пользовательских свойств.

Для изменения характеристик хранимого изображения нужно в редакторе отредактировать их.

Можно создавать, изменять или удалять одно или несколько хранимых изображений в одном и том же сеансе редактирования. При нажатии на кнопку *"OK"* все изменения будут сохранены с последующим выходом из редактора.

При сохранении файла модели данных **ERwin** сохраняет все связанные с ним хранимые изображения (и области) как один общий файл.

Удаление хранимого дисплея выполняется по схеме:

1. Войти в окно-диалог редактора *"Stored Displays"*.
2. Нажать кнопку *"Delete"*. **ERwin** удаляет хранимое изображение из списка *"Stored Display"*.
3. Нажмите *"OK"* для удаления хранимого изображения. **ERwin** удалит выбранное хранимое изображение, закроет редактор и возвратится в диаграмму. Остальные связанные с изображением хранимые изображения и области удалены не будут.

Порядок выполнения работы

1. Откройте модель, сохраненную в первой лабораторной работе.
2. Переименуйте объекты *"CUSTOMER"*, *"ORDER"* и *"PRODUCT"* на *"Customer"*, *"Order"*, *"Product"* соответственно.
3. Переименуйте объект *"NEW"* в *"Apple"*.
4. Создайте новую предметную область *"Array2"* непосредственно в *Model Explorer*.
5. Добавьте объекты *"Customer"*, *"Order"*, *"Apple"* в предметную область *"Array2"*.
6. Из предметной области *"Array2"* удалите объект *"Apple"*. Для этого в появившемся диалоговом окне выберите *"Current Subject Area"*.
7. В редакторе *"Subject Areas"* включите объект *"Product"* в предметную область *"Array2"*.

8. Введите определение главной области: "Главная область" и "Array2": "Дополнительная область".
9. Для главной и дополнительной области введите свое имя как автора.
10. Создайте хранимое изображение с именем "Stor_Display".
11. Введите имя автора в хранимое изображение.

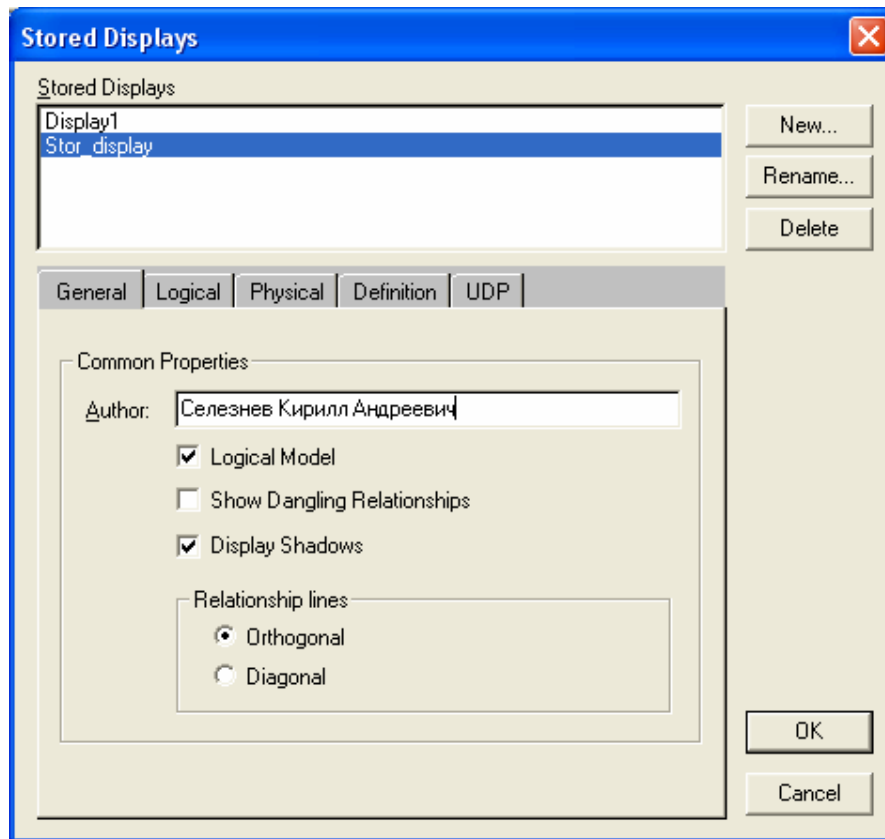


Рис. 3.18. Диалог "Stored Displays"

12. Сохраните и закройте модель.

Контрольные вопросы

1. Что такое хранимое изображение?
2. Что такое *Model Explorer*?
3. Что такое предметная область?
4. Для чего используются дополнительные предметные области?
5. Как удалить объект только из дополнительной предметной области?

Лабораторная работа №3. Отношения

Цель работы: Научиться создавать связи. Получить навыки в редактировании графических объектов. Получить навыки в преобразовании объектов *ERwin*

Теоретические сведения

Изменение этой панели инструментов основано на типе модели (логической или физической) и нотации (*IDEFIX* или *DM*-пространственная модель), как показано на рис. 3.19.

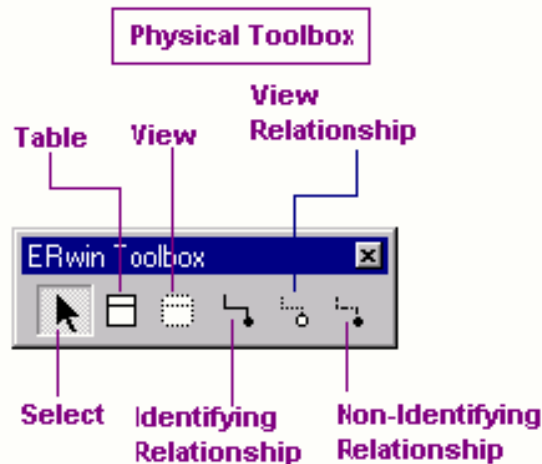


Рис. 3.19. Панель связи

При добавлении сущности *ERwin* определяет ее тип, основанный на отношении, в которое он включен. Например, когда первоначально добавляется сущность в модель, она представляется как независимая. Когда сущность подключается к другой сущности, использующей отношение, *ERwin* определяет сущность (независимая или зависимая), базируясь на типе отношения.

В модели данных отношение показывает связи между двумя объектами или таблицами. *ERwin* представляет отношение как линию, соединяющую два объекта или две таблицы, как видно на рис. 3.20.

В зависимости от выбора нотации, символы и окончание линии могут измениться.

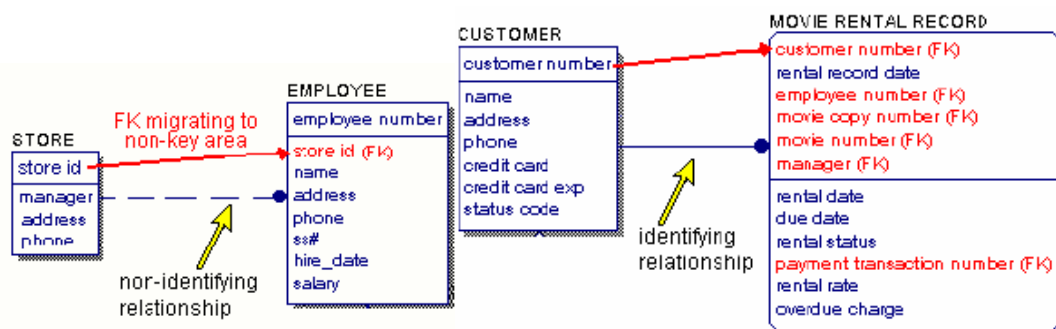


Рис. 3.20. Идентифицирующая и не идентифицирующая связи

Дополнительно к устанавливающей и не определяющей связи можно создать другую связь: определяющая, неопределяющая, рекурсивная, отношение многое-ко-многим, отношение супертип-подтип, которые показаны на рис.3.21:

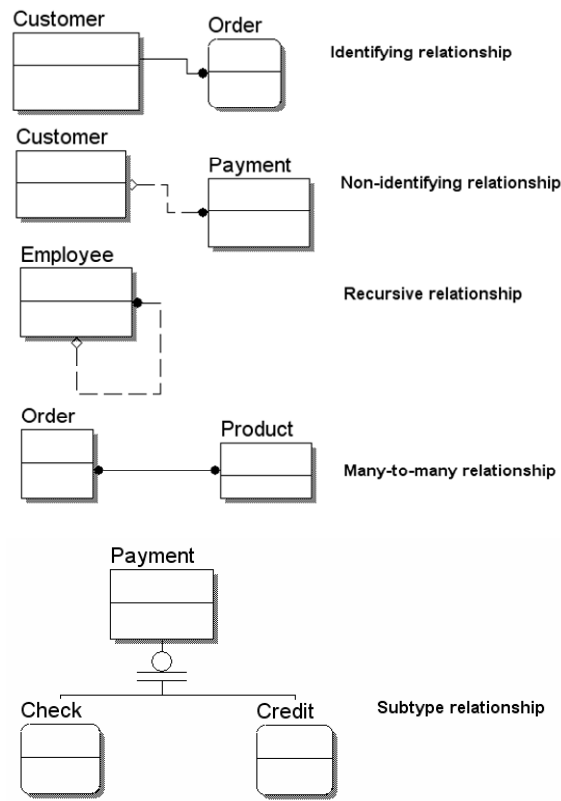


Рис. 3.21. Дополнительные виды связей

Выравнивание, размещение и группировка средств панели инструментов.

Инструментальные средства для выравнивания (*Alignment Toolbar*) ускоряют процесс графического обустройства и группирования моделей объектов. Панель инструментов изображена на рис. 3.22.

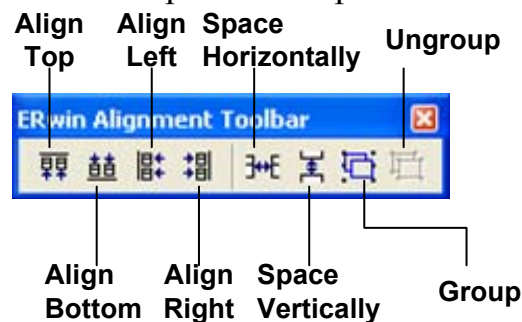


Рис. 3.22. Панель инструментов выравнивания, размещения и группировки

Можно выбрать многочисленные объекты диаграммы и при этом для них будет выделено пространство. Аналогично, групповые средства (*Group*) позволяют легко обрабатывать группы многочисленных диаграмм.

Уровни демонстрации изображения в ERwin

ERwin позволяет просматривать информацию о модели на разных уровнях:

- Уровень демонстрации сущности (*Entity Display Level*)

На уровне сущности имя каждой сущности выводится на экран, заключенное в окно сущности. Никакая другая информация о сущностях на экране не появляется. Для вывода на экран модели на уровне сущности следует в меню *Format/Display Level* дать команду "Entity".

- Уровень демонстрации определений сущности (*Format/Display Level/Definition*).

На этом уровне размер окна сущности увеличивается и в него включается определение сущности.

- Уровень демонстрации атрибутов (*Format/Display Level/Attribute*).

При переходе к процессу построения модели нужно сохранить и вывести на экран подробную информацию, которая и составляет сущность. Эти подробности называются атрибутами. На уровне демонстрации атрибутов каждая сущность в диаграмме открывается, показываются ее атрибуты, причем атрибуты, которые являются первичными ключами, расположены над чертой, а не ключевые атрибуты – под чертой в окне сущности.

- Уровень демонстрации первичных ключей (*Format/Display Level/Primary Key*).

Разновидностью уровня демонстрации атрибутов является уровень первичного ключа. На этом уровне демонстрируются только те атрибуты, которые являются первичными ключами (те, которые находятся над чертой в окне сущности) каждой сущности. Атрибуты, которые находятся под чертой, не выводятся на экран, но они будут выведены снова при переходе обратно на уровень атрибутов.

- Уровень пиктограммы (*Format/Display Level/Icon*)

Для пиктограммы сущности должен использоваться файл "Windows bitmap" (расширение "BMP"). Для каждой сущности можно задать свой "bitmap".

Примечание: нельзя распечатать диаграмму, когда она изображена на уровне *Icon*. В этом случае команда "Print" меню "File" не работает.

Теневой эффект

Команда "Show Shadows" меню "Format" позволяет добавить в диаграмму эффект трех мерности. Окно-диалог "Preference" (*Format/Preferences*) позволяет делать тени больше или меньше. Для этого нужно в окне *Format Preferences (Format/Preferences)* выбрать вкладку "Display".

Примечание: теневой эффект применяется ко всей модели данных. Его нельзя включить или убрать для какой-то отдельной сущности.

Внешний ключ

Когда создается отношение между объектами, **ERwin** автоматически передает первичный ключевой атрибут (*s*) родительского объекта, на объект-потомок. Обозначение (*FK*) указывает имя переданного ключевого атрибута (внешний ключ). Эта пересылка происходит автоматически в физической модели.

В определяющем отношении *FK* передается выше линии и становится частью первичного ключа объекта-потомка. В не определяющем отношении внешний ключ передается ниже линии и становится не ключевым атрибутом на объекте-потомке. В **ERwin** пунктирная линия представляет собой не определяющее отношение.

Если возникнет необходимость в миграции "*non-key*" (не ключевого атрибута) вместо первичного ключа, то можно воспользоваться альтернативным ключом для миграции.

Управление внешними ключами.

При создании связи между двумя сущностями **ERwin** автоматически производит миграцию ключевых атрибутов родительской сущности в дочернюю сущность, где они становятся внешними ключами. Поддерживается автоматическая миграция ключей, поэтому лучше добавлять первичные ключи в независимую сущность сразу после создания.

ERwin позволяет задать режим "*Unique Key*", который можно использовать для предупреждения пользователя о создании атрибута, который может автоматически мигрировать через связь, или может запретить пользователю создание таких атрибутов. При удалении связи **ERwin** автоматически удаляет соответствующие внешние ключи из дочерних сущностей.

Нарисуйте связь между двумя сущностями или между сущностью и ею же самой (рекурсивная связь). Внешний ключ автоматически мигрирует.

Примечание: чтобы видеть автоматическую миграцию внешних ключей, нужно установить режим просмотра диаграммы *Attribute (Format/Display Level)*; должны быть уже созданы атрибуты первичного ключа в родительской сущности.

Когда **ERwin** производит миграцию атрибута первичного ключа, то по умолчанию внешний ключ, который появляется в дочерней сущности, наследует имя, но не наследует определение атрибута первичного ключа.

Связи

Связь – это соотношение либо между двумя сущностями, либо между сущностью и этой же сущностью.

Идентифицирующей связью называется связь, которая добавляет признаки идентичности в дочернюю сущность путем миграции ключей родительской сущности в область ключевых атрибутов дочерней и, таким образом, делая дочернюю сущность зависимой от родительской в смысле своей идентичности.

Можно задать также и такую связь, которая не ставит дочернюю сущность в зависимость от родительской. Этот тип связи называется не идентифицирующей связью. В **ERwin** такая связь обозначается пунктирной линией с жирной точкой на конце, соответствующем дочерней связи. При не идентифицирующей связи атрибуты первичного ключа родительской сущности мигрируют в область данных (не ключевая область), которая расположена под чертой в дочерней сущности. Если атрибуты, которые мигрировали в не ключевую область дочерней сущности, не нужны в этой сущности, то связь называется не обязательной не идентифицирующей связью, что подразумевает, что мигрировавшие атрибуты не нужны дочерней сущности для ее идентификации и что она может существовать и без этих атрибутов. В **ERwin** необязательная не идентифицирующая связь обозначается пунктирной линией с жирной точкой на одном конце (дочернем) и ромбиком на другом (родительском).

Если связь уже создана на диаграмме, то можно изменить ее тип в редакторе "*Relationships*" в закладке "*General*".

Нулевые значения

По умолчанию для не идентифицирующей связи задается режим "*Nulls Allowed*", что означает, что дочерняя сущность может существовать без родительской, и связь называется необязательной; "*No Nulls*" означает, что существование дочерней сущности зависит от родительской, и связь называется обязательной. В случае необязательной связи (*Nulls Allowed*) на родительском конце не идентифицирующей связи **ERwin** ставит знак — ромбик.

Примечание: внешние ключи, которые мигрируют через не идентифицирующую связь, могут принимать значения *NULL*.

Создание связи

1. Установите курсор на нужный инструмент и нажмите левую кнопку мыши;

2. Щелкните по родительской, а затем по дочерней сущности.

Примечание: если установить курсор на линию связи, нажать "*SHIFT*" и дважды щелкнуть левой кнопкой мыши, то появится окно-диалог, в котором **ERwin** спросит о желании открыть активный редактор для родительской или для дочерней сущности. Выберите нажатием кнопки мыши родительскую или дочернюю сущность и нажмите "*OK*".

Изменение типа связи

1. Войдите в редактор "*Relationships*";

2. Нажмите нужную кнопку в окне "*Relationship Type*" ("*Identifying*" - идентифицирующая, "*Non-identifying*" — не идентифицирующая).

Примечание: рекурсивной связи в **ERwin** автоматически присваивается тип "*Non-identifying*", который изменить нельзя.

Работа в редакторе "Relationships"

Связи, как и сущности, могут иметь имена и метки. Эти метки называются глагольными фразами, они должны описывать, каким образом родительская сущность связана с дочерней. Редактор "Relationships" позволяет задать глагольную фразу, мощность и отношение к нулевым значениям для связи. Можно задать в редакторе "Relationships" имя роли для атрибута – внешнего ключа.

- Закладка "Rolename"

Во вкладке "Rolename" содержатся все внешние ключи, которые возникли в результате миграции при задании текущей связи. Чтобы связать имя роли с внешним ключом, в списке окна "Relationships" выберите нужную связь, во вкладке "Rolename" введите имя роли в текстовое окно "Rolename".

- Закладка "General"

Введите имя, которое идентифицирует связь, в окне "Verb Phrase". Редактор "Relationships" показывает первые четыре строки глагольной фразы и позволяет вводить и просматривать даже большее число строк.

Примечание: для отображения на диаграмме глагольной фразы следует поставить метку в соответствующее меню (*Format/Relationships Display/Verb Phrase*).

Можно задать здесь тип связи "Identifying" или "Non-identifying".

"Cardinality" и "Nulls" служат для мощности и режима нулевых значений для текущей связи.

Мощность связи служит для обозначения отношения числа экземпляров родительской сущности к числу экземпляров дочерней. Родительская сущность может связываться с дочерней одним из четырех способов. В *IDEFIX* мощность бинарных отношений (мощность связи) равна 1:n, где n может принимать значения:

- 0, 1 или более (*Zero, one or More*) – обозначается "пустым местом". Каждая родительская сущность связана с 0, 1 или более экземпляров дочерней сущностей.
- 1 или более (*One or More*) – обозначается буквой "P". Каждая родительская сущность связана с 1 или более экземпляров дочерней.
- 0 или 1 (*Zero or One*) – обозначается буквой "Z". Каждая родительская сущность связана с 0 или 1 экземпляром дочерней.
- ровно n (*Exactly*), где "n" – некоторое число. Каждая родительская сущность связана с ровно n экземплярами дочерней.

Чтобы показать мощность связи следует дать команду "Cardinality" в меню (*Format/Relationships Display*).

Влияние удаления связи на внешний ключ

Атрибуты внешнего ключа автоматически мигрируют с одной сущности на другую при задании связи. Их можно видеть, если между двумя

сущностями задана связь (для этого надо находиться в режиме просмотра *"Display Level Attribute"*, и у сущностей есть именованные атрибуты первичного ключа). При удалении связи происходит процесс, обратный предыдущей миграции, и перенесенный внешний ключ автоматически удаляется из дочерней сущности.

Использование имен ролей атрибутов

Когда атрибут первичного ключа мигрирует из родительской сущности в дочернюю, то там он становится атрибутом внешнего ключа. У внешнего ключа может быть иная роль, отличающаяся от роли связанного с ним первичного ключа. В этом случае следует связать с внешним ключом имя роли. Имя роли – это алиас, присваиваемый атрибуту внешнего ключа.

При рекурсивной связи, когда сущность связана сама с собой, один и тот же атрибут может быть и первичным, и внешним ключом. Имя роли помогает прояснить вторую роль атрибута.

Имя роли состоит из трех частей: присваиваемое имя роли, точка – разграничитель и базовое имя. Имя роли присваивается атрибуту внешнего ключа в редакторе *"Relationships"* и *"Subtype Relationshipss"*. **ERwin** вставляет точку после имени роли, а затем ставит базовое имя атрибута. Базовое имя – это имя атрибута, которое мигрировало из родительской сущности в дочернюю.

Изображение имен ролей на диаграмме

Для того чтобы показать на экране или убрать имя роли с присоединенным базовым именем, следует дать команду *"Rolename/Attribute" (Format/Entity Display)*. Если режим *"Rolename/Attribute"* включен, то **ERwin** показывает на экране атрибуты с именами ролей в формате *"имя роли базовое имя"*. Если этот режим выключен, то **ERwin** показывает на экране атрибуты с именами ролей в формате *"имя роли"*.

Понятие унификации

Зависимая сущность может унаследовать один и тот же внешний ключ из нескольких родительских сущностей, а также сущность может унаследовать один и тот же внешний ключ несколько раз от одного и того же родителя через несколько разных связей. Когда **ERwin** обнаруживает одно из этих событий, то он распознает, что два атрибута одинаковы, и предполагает, что пользователь хочет видеть атрибут в зависимой сущности только один раз. В редакторе *"Attributes"* эта сущность показана с обоими атрибутами, но на диаграмме атрибут показывается только один раз. Это комбинирование, или объединение идентичных атрибутов, называется унификацией. Унификация - слияние двух или более атрибутов внешнего ключа в один атрибут внешнего ключа на основе предположения, что значения исходных атрибутов внешнего ключа, вероятно, идентичны. Эта унификация производится, поскольку правила нормализации запрещают существование в одной сущности двух атрибутов с одинаковыми именами.

Есть случаи, когда унификация нежелательна. Например, когда два атрибута имеют одинаковые имена, но на самом деле они немного отличаются, и пользователь хочет, чтобы это отличие отражалось в диаграмме.

Имена ролей используются для сохранения отличий между двумя разными атрибутами, имеющими одинаковые имена.

Перенос или копирование атрибута

1. Щелкнуть по инструменту управления атрибутами на *Toolbox*.
2. Выбрать один или несколько атрибутов исходной сущности. Можно щелкнуть мышью для выбора одной сущности или нажать "*SHIFT*" или "*CTRL*" одновременно с кнопкой мыши для выбора нескольких атрибутов.
3. Для переноса атрибута нажмите левую кнопку мыши и, не отпуская ее, перенесите атрибут в другую сущность или на новое место в той же самой сущности или, если нужно скопировать атрибут, установите курсор на тот атрибут, который стоит ниже того места, в которое нужно вставить атрибут. При копировании или переносе атрибута **ERwin** переносит информацию и определения исходного атрибута (например, значения доменов, определение, замечания и т.д.) на новое место. При копировании атрибута **ERwin** не контролирует соблюдение уникальности имен, если оно не задано.

Примечание: при выборе рядом стоящих атрибутов нажимайте "*SHIFT*"-Click, а при выборе изолированных атрибутов – "*CTRL*"-Click. Чтобы вставить атрибут в конец первичного ключа или не ключевой области сущности, вставьте его на вторую с конца позицию, а затем перенесите последний атрибут так, чтобы он оказался над новым атрибутом.

Продублировать атрибут в этой же сущности нельзя. Нельзя удалить внешний ключ или перенести его из той сущности, в которой он находится. Нельзя перенести внешний ключ, который мигрировал в результате задания идентифицирующей связи, из не ключевой области, в которой он находится. Если внешний ключ мигрировал в не ключевую область в результате задания не идентифицирующей связи, то его можно перенести в область первичного ключа этой же сущности, за исключением случаев, когда не идентифицирующая связь, в результате которой он мигрировал, была рекурсивной.

Преобразование объектов

Transform – метод для создания и сохранения записей проектирования после изменения объектов или свойств в пределах проектного слоя (логический, физический уровни); изменение установки объектов от одного состояния до другого с целью нормализации, очистки, или денормализации модели.

Основные преимущества использования преобразования:

- **Автоматизация.** **ERwin** упрощает очистку логической и физической моделей. Можно использовать мастеров (*Wizard*) для автоматизи-

ческого изменения приложения в проектном слое вместо ручного изменения.

- **Трассируемость (Traceability).** *ERwin* поддерживает историю каждого объекта модели, созданного преобразованием. Можно проследить историю преобразованных объектов.
- **Сохранение объектных свойств.** *ERwin* сохраняет свойства преобразованных объектов. Вручную войти повторно нельзя.

Преобразование панели инструментов обеспечивает набор инструментальных средств для помощи в преобразовании. Тип модели и объекта, которые нужно преобразовать, доступны на панели инструментов "*Transforms*" (рис.3.23):

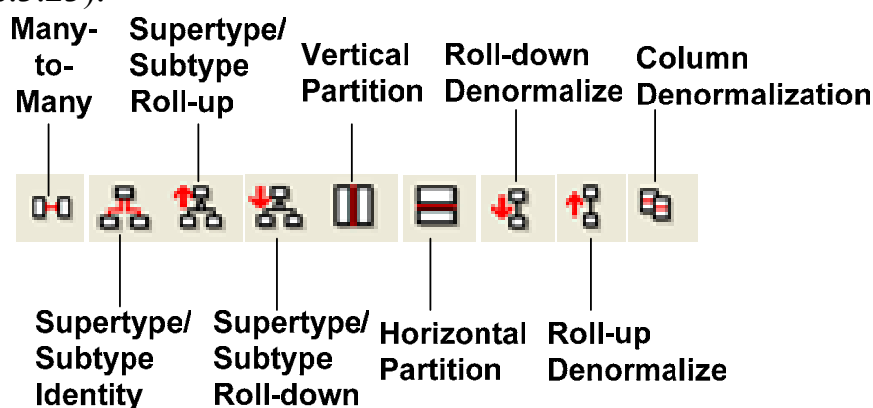


Рис. 3.23. Преобразование

Виды преобразований:

- ***Many-to-Many Transform*** (преобразование отношения многие-ко-многим через дополнительную сущность)

Для преобразования нужны две сущности, соединенные связью "*Many-to-Many*" (рис.3.24). При преобразовании можно задать имя дочерней сущности двух родительских сущностей (во втором диалоговом окне), определение дочерней сущности (в третьем диалоговом окне), имя преобразования (в третьем диалоговом окне).

Пример:

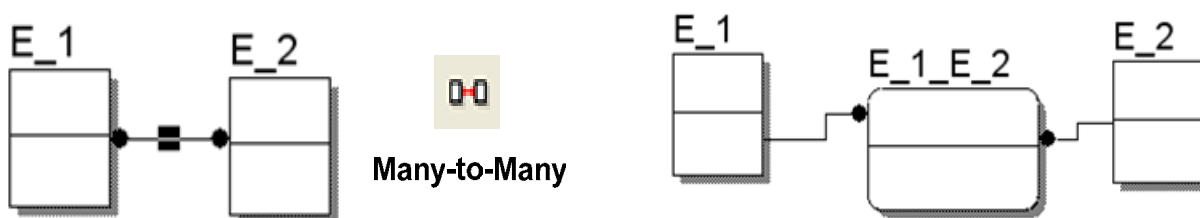


Рис. 3.24. Преобразование *Many-to-Many*

- **Supertype-Subtype Identity** (замещение отношения супертип-подтип)

При преобразовании можно задать имя преобразования (во втором диалоговом окне).

- **Supertype-Subtype Roll-up** (преобразование отношения супертип-подтип в одну сущность - супертип)

При преобразовании можно задать имя преобразования и описание преобразования (во втором диалоговом окне).

- **Supertype-Subtype Roll-down** (преобразование отношения супертип-подтип в две сущности - подтипы)

При преобразовании можно задать имя преобразования и описание преобразования (во втором диалоговом окне).

Пример (рис. 3.25):

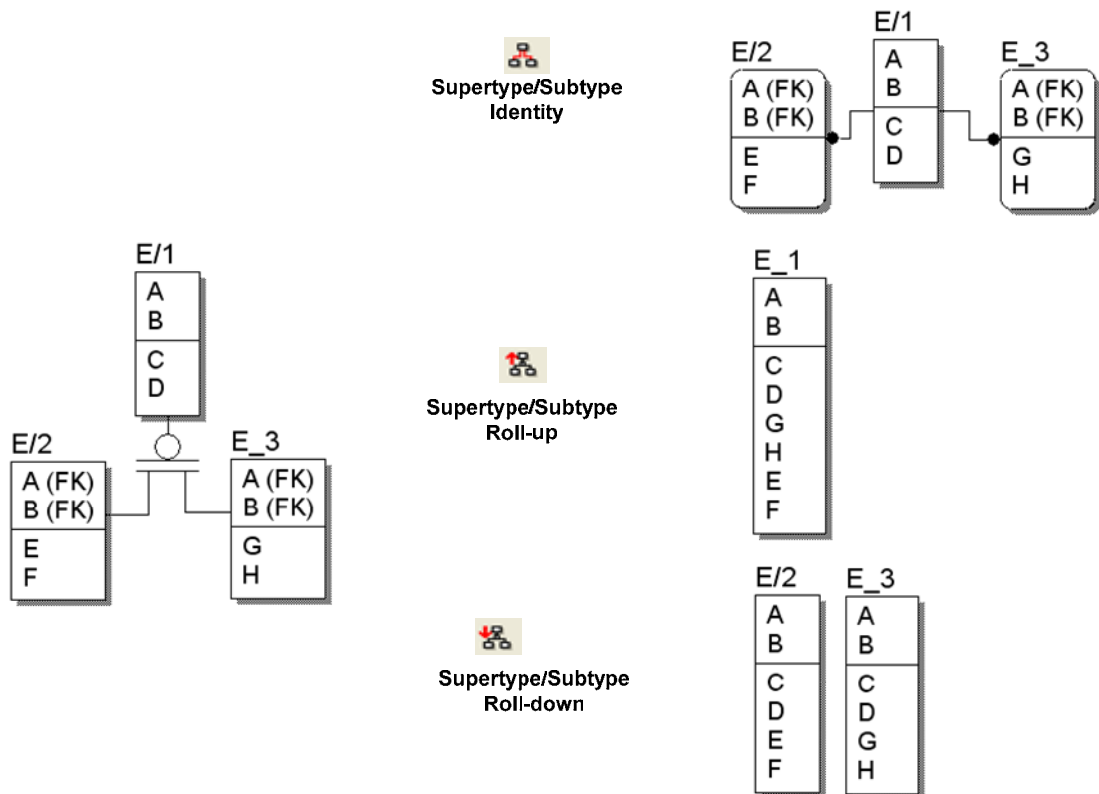


Рис. 3.25. Преобразование *Supertype-Subtype*

- **Vertical Partition** (вертикальное разделение)

При преобразовании во втором диалоговом окне можно задать число производных сущностей (в окне "Number Of Partitions"). На 4-м шаге можно определить атрибуты, включаемые в производные сущности, для сущности. Сущность можно выбрать из списка "Partition Name". Атрибуты можно переносить с помощью кнопок ">" (внести один атрибут в производную сущность), ">>" (внести все атрибуты в производную сущность).

- **Horizontal Partition** (горизонтальное разделение)

При преобразовании на 4-м шаге можно задать имя преобразования и описание преобразования.

Пример (рис. 3.26):

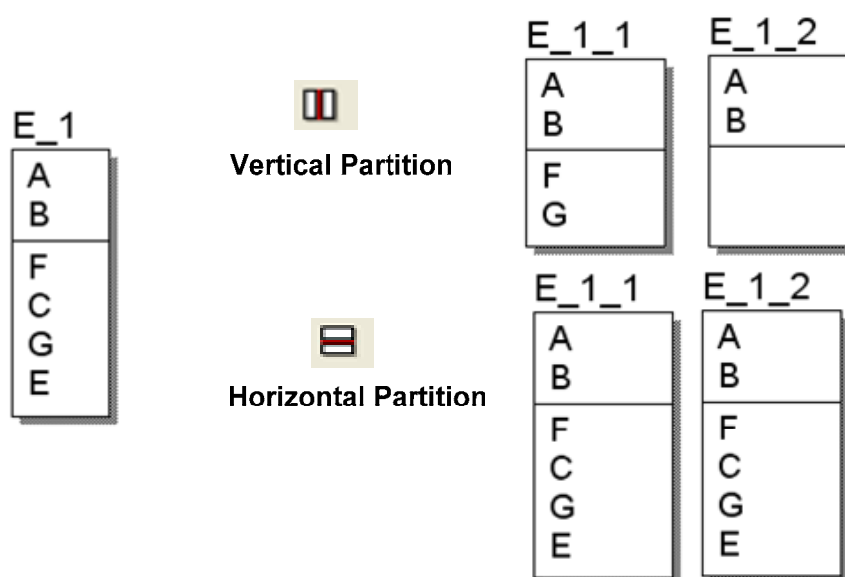


Рис. 3.26. Преобразование *Partition*

- **Roll-down Denormalization** (перенос атрибутов в дочернюю сущность)

Во втором диалоговом окне можно выбрать имя преобразования и описание преобразования;

- **Roll-up Denormalization** (перенос атрибутов в родительскую сущность)

Во втором диалоговом окне можно выбрать имя преобразования и описание преобразования. В 3-ем диалоге в окне "*Number of records*" можно задать число повторяющихся записей.

Пример (рис. 3.27):

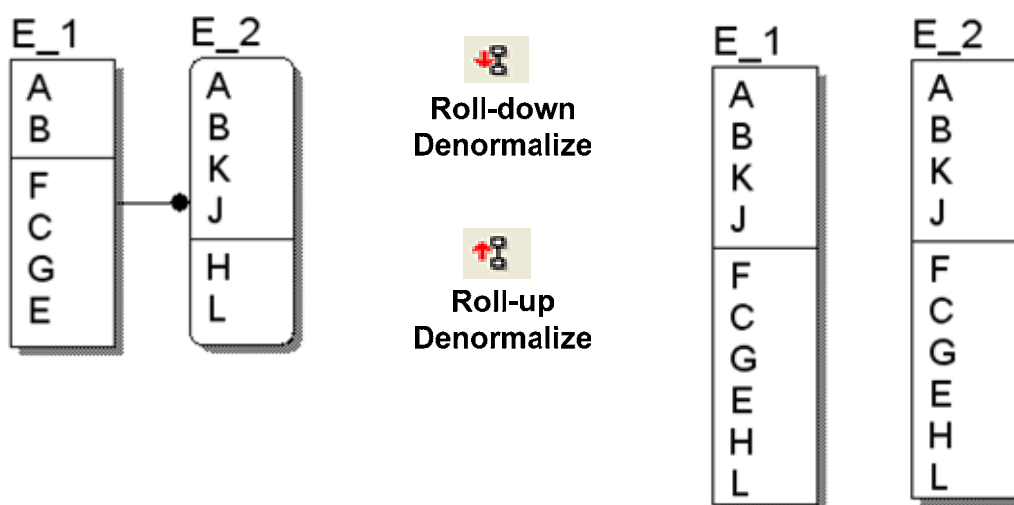


Рис. 3.27. Преобразование *Denormalization*

Column Denormalization (копирование атрибута в любую сущность) (рис. 3.28). До преобразования нужно выделить атрибут. При преобразовании во 2-м диалоговом окне можно указать имя и комментарий столбца. В 3-ем диалоговом окне следует выбрать таблицу, в которую атрибут будет включен. В 4-м диалоговом окне можно выбрать имя преобразования и описание преобразования.

Пример:

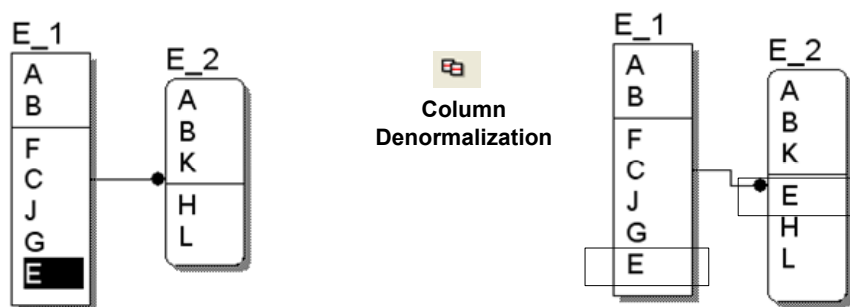



Рис. 3.28. Преобразование *Column Denormalization*

При преобразовании *"Model Explorer"* корректирует важную информацию под папкой *"Transform"*. Информация включает преобразованное имя и исходное и целевые объекты, включенные в преобразование.


При преобразовании исходные объекты будут удалены. Для изменения преобразования зайдите в *"Model Explorer"* и щелкните правой кнопкой мыши на *"Transform"*. Из контекстного меню выберите *"Delete"* и затем - *"Resolve"* (в случае, если возврат к модели-источнику не нужно) или *"Reverse"* (для возврата к модели-источнику).

Примечание: если модель-источник связана с преобразованной моделью, то после обратного преобразования связь будет разорвана. После возобновления преобразования *ERwin* сохраняет исходные объекты и удаляет преобразованные и создается преобразованная модель объекта.

Порядок выполнения работы

1. Откройте модель *"My ERwin Model"*, сохраненную во второй лабораторной работе. В инструментарии *ERwin* кликните  определяющее средство (*Identifying Relationship*) отношения.

2. В окне диаграммы добавьте определяющее отношение между *"Customer"* и *"Order"*, щелкнув по объекту *"Customer"*, чтобы назначить объект-родитель. Затем щелкните по *"Order"*, чтобы определить его как объект-потомок.

3. В инструментарии *ERwin* кликните  "не определяющее" (*Non-identifying*) средство отношения. Для того, чтобы добавить "не определяющее" отношение между *"Order"* и *"Product"*, щелкните по *"Order"*, а потом

по "Product". Модель должна выглядеть подобно нижеприведенной модели: на рис. 3.29.

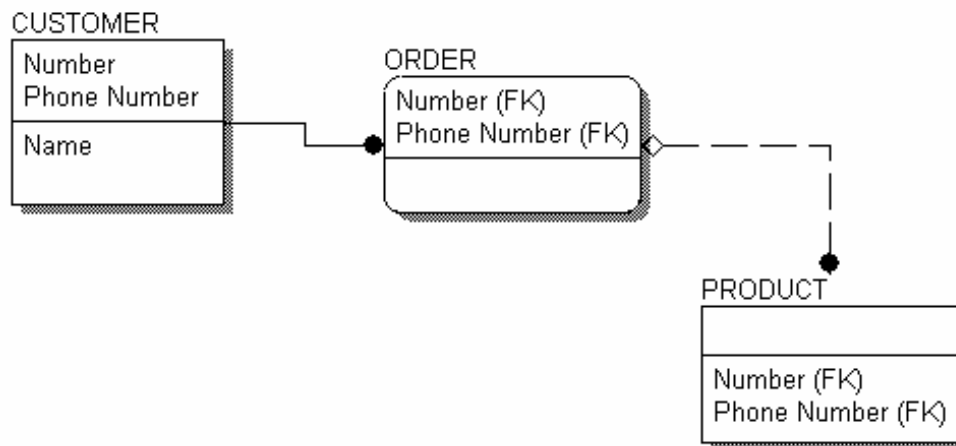


Рис. 3.29. Модель после внесения связей

4. Выводите и сгруппируйте объекты вашей диаграммы с помощью средств для выравнивания (*Alignment Toolbar*) как показано на рис. 3. 30.

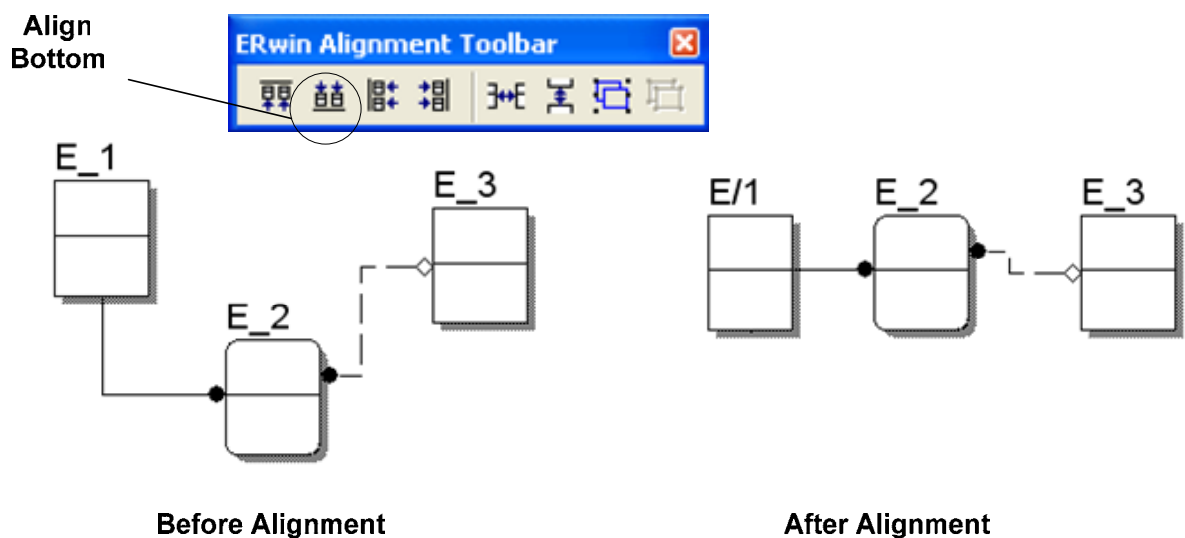


Рис. 3.30. Выравнивание *Align bottom*

5. Добавьте теневой эффект к модели данных. Для этого откройте окно *Format Preferences*, закладку *Display*. В полях *Right* и *Bottom* закладки *Shadow Offset* установите значения "15".

6. Свяжите внешние ключи с соответствующими именами ролей: "Номер ключа клиента" и "Номер телефона клиента" для определяющей связи (закладка "Rolename" редактора "Relationships"). Включите отобра-

жение базового имени атрибута (*Entity Display/Rolename/Attribute*) как показано на рисунке 3.31.

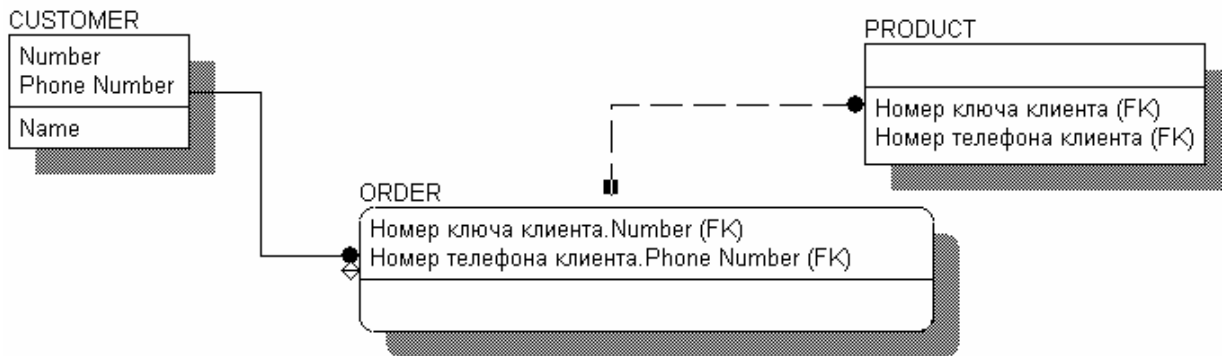


Рис. 3.31. Определение имени роли для внешних ключей

7. Задайте глагольную фразу для связей: <Клиент>(*Customer*) размещает <Заказ>(*Order*) , и <Заказ> выписывает <Товар>(*Product*).

8. Задайте для связей "*Customer*" - "*Order*" и "*Order*" – "*Product*" мощность "каждая родительская сущность связана с 0,1 и более экземпляров дочерней" (обычно установлено по умолчанию).

9. Поменяйте местами атрибуты "*Name*" и "*Phone Number*" сущности "*Customer*".

10. Сохраните модель.

11. Создайте новую логико-физическую модель *lab_3_2*.

12. Создайте две сущности с именами "*L1*" и "*L2*". Определите для них связь "*Many-to-Many*". Примените преобразование "*Many-to-Many Transform*". Сохраните модель под именем "1".

13. Создайте три сущности с именами "*L3*", "*L4*" и "*L5*": одна родительская и две дочерние – со связью "*Complete sub-category*". Добавьте к родительской сущности два ключевых атрибута и два не ключевых. К двум другим добавьте по два не ключевых атрибута (имена добавляемых атрибутов не должны совпадать) как показано на рисунке 3.32.

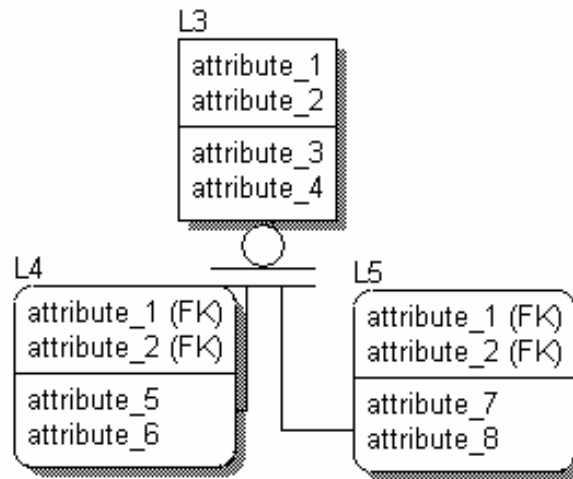


Рис. 3.32. Связь *Complete sub-category*

14. Примените преобразование "*Supertype-Subtype Identity*" (рис. 3.33).
Вернитесь к модели-источнику .

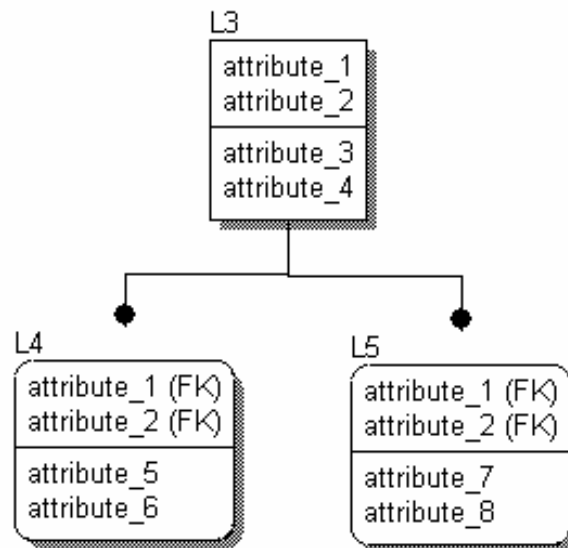


Рис. 3.33. Результат преобразования *Supertype-Subtype Identity*

15. Примените преобразование "*Supertype-Subtype Rollup*" (рис. 3.34).
Вернитесь к модели-источнику .

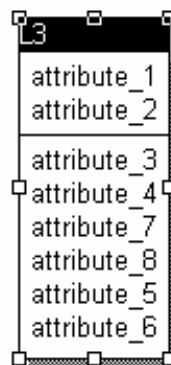


Рис. 3.34. Результат преобразования *Supertype-Subtype Rollup*

16. Примените преобразование *"Supertype-Subtype Rolldown"* (рис. 3.35). Вернитесь к модели-источнику.

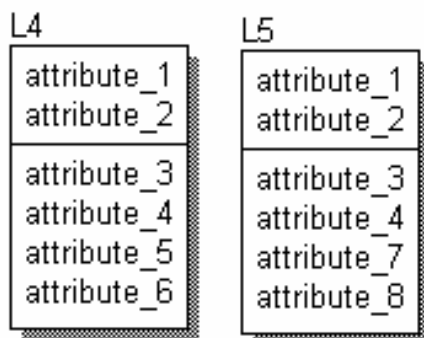


Рис. 3.35. Результат преобразования
Supertype-Subtype Rolldown

17. Создайте сущность с именем *"L6"* с шестью атрибутами, два из которых ключевые. Переключите вид отображения меню на физический (*physical*). Примените преобразование *"Vertical Partition"* для созданной сущности. Вернитесь к модели-источнику.

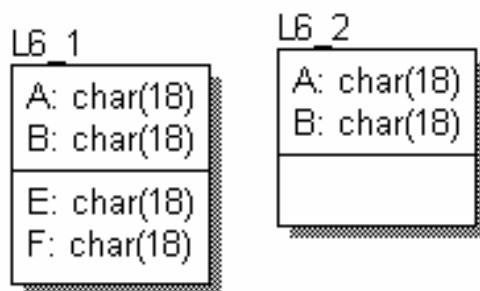


Рис.3.36. Результат преобразования
Vertical Partition

18. Примените преобразование *"Horizontal Partition"*. Вернитесь к модели-источнику.

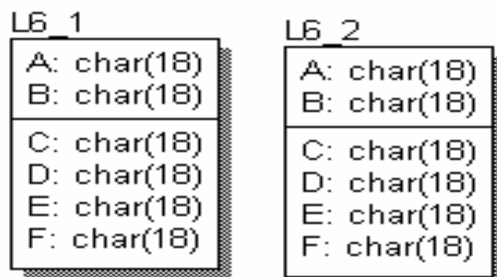


Рис. 3.37. Результат преобразования
Vertical Partition

19. Создайте две сущности с именами *"L7"* и *"L8"* с идентифицирующей связью. К родительской сущности добавьте 6 атрибутов, два из которых ключевые. К дочерней сущности добавьте два ключевых и два не

ключевых атрибута. Примените преобразование "*Column Denormalization*" скопировав один из атрибутов "L7" в "L8".

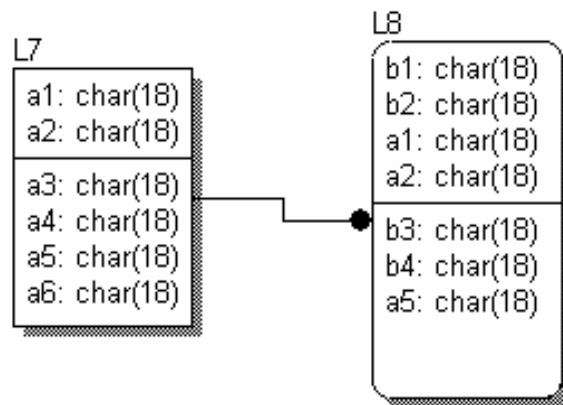


Рис. 3.38. Результат преобразования
Column denormalization

Контрольные вопросы

1. Что такое связь?
2. Какие существуют виды связей?
3. Что такое мощность связи?
4. Что такое роль атрибута?
5. Что такое внешний ключ?

Лабораторная работа №4. Индексация базы данных

Цель работы: Получить навыки в индексации баз данных. Получить навыки в редактировании шрифта и цвета объектов на диаграмме.

Теоретические сведения

В целях улучшения работы базы данных в таблице обычно имеется несколько индексов. Преимущество множественных индексов в том, что есть несколько точек доступа к данным в таблице. В **ERwin** атрибуты первичного ключа автоматически индексируются. Кроме этого, индексируются альтернативные ключи и "*Inversion Entry*" (Инверсные вхождения).

Альтернативный ключ

Альтернативным ключом называется атрибут или группа атрибутов, уникальным образом определяющие экземпляр сущности. Если у сущности есть несколько атрибутов, уникальным образом определяющих каждый экземпляр, то можно назначить любой из этих атрибутов, за исключением атрибутов первичного ключа, альтернативным ключом, и **ERwin** создаст дополнительные индексы.

ERwin создает уникальный индекс для каждого альтернативного ключа.

Назначение атрибута или группы атрибутов альтернативным ключом:

1. Зайти в редактор "*Key Groups*" (*Model/Attributes*).
2. Нажмите кнопку "*New*". Появится диалоговое окно "*New Key Group*".
3. В поле "*Key Group*" введите имя альтернативного ключа.
4. В поле "*Index*" введите индекс альтернативного ключа.

Например, если три атрибута вместе составляют альтернативный ключ, то они должны заканчиваться одинаково. Они будут по очереди проиндексированы вместе.

У сущности может быть несколько альтернативных ключей. Они нумеруются последовательно (напр., *AK1*, *AK2*, *AK3* и т.д.). Если атрибут - часть нескольких альтернативных ключей, то следует отделить метки разных ключей, стоящих в скобках, запятыми (*AK1*, *AK2*).

При генерации схемы индексы создаются в определенном порядке. Сначала создается индекс первичного ключа, затем индексы альтернативных ключей (напр., *AK1*, *AK2*, *AK3*), затем индексы "*Inversion Entry*" (напр., *IE1*, *IE2*, *IE3* и т.д.).

Если в меню у "*Alternate Key Designator*" (*Format/Entity Display*) стоит метка, то индикаторы альтернативных ключей будут показаны на диаграмме (показывается *AKn*, где *n* – номер).

Назначение инверсивного вхождения (*Inversion Entries*)

На основе атрибутов, назначенных альтернативными ключами, строятся индексы, дополняющие индексы, основанные на первичном ключе. Можно назначить атрибуты, которые будут участвовать в неуникальных

индексах. Атрибуты, участвующие в неуникальных индексах, называются *Inversion Entries*. ***Inversion Entry*** – это атрибут, который не определяет экземпляр сущности уникальным образом, но часто используется для обращения к экземплярам сущности. ***ERwin*** генерирует неуникальный индекс для каждого "*Inversion Entry*".

Задание "*Inversion Entry*":

1. Войти в редактор "*Attributes*" для той сущности, для которой нужно создать "*Inversion Entry*".

2. Добавить к имени атрибута (атрибутов), которые будут составлять неуникальный индекс под номером, *n*, "*(IE_n)*". Например, если есть два атрибута, которые вместе составляют неуникальный индекс, то каждый из них должен заканчиваться на "*(IE1)*". Оба атрибута будут по очереди проиндексированы вместе как неуникальный индекс.

3. Нажмите "*OK*" для выхода с сохранением всех изменений.

У сущности может быть несколько разных неуникальных ключей. Они будут нумероваться: *IE1*, *IE2*, *IE3* и т.д.

Примечание: При переносе альтернативного ключа (*AK*) или инверсионного вхождения (*IE*) в рамках одной сущности он сохраняет свое назначение (*AK* или *IE*). При переносе его в другую сущность пометка *AK* или *IE* исчезает.

Если поставить метку рядом с "*Alternate Key Designator*" (*Format/Entity Display*), то будут показаны индикаторы "*Inversion Entry*" на диаграмме.

Примечание: Может возникнуть такая ситуация, что некоторые атрибуты сущности будут включены и в альтернативные ключи, и в "*Inversion Entry*".

При обратном проектировании базы данных ***ERwin*** импортирует информацию об индексах, поэтому можно просматривать и изменять ее как часть модели данных. При прямом проектировании базы данных ***ERwin*** автоматически создает индексы для каждой сущности на основе первичного ключа, альтернативных ключей и "*Inversion Enter*".

Индекс

В таблице реляционной базы данных данные хранятся в том же порядке, в котором их ввели в таблицу. В базе данных, если информация, хранящаяся в колонке, индексирована, индекс указывает на все строки, в которых хранится конкретное значение колонки.

Для каждого названия элемента индекс содержит ссылку, указывающую, в каком месте таблицы хранится этот элемент.

Значения в индексе хранятся в определенном порядке и нужно просматривать значительно меньший объем данных; использование индекса для поиска нужной информации гораздо быстрее, чем просмотр требуемых таблиц базы данных.

Можно создать свой индекс для каждой колонки в таблице. Можно создать единый индекс на основе нескольких колонок для поиска определенного набора значений. При генерации схемы физической базы данных **ERwin** автоматически создает отдельный индекс на основе первичного ключа каждой таблицы, а также на основе всех альтернативных ключей, внешних ключей и "*Inversion Entry*", поскольку эти колонки наиболее часто используются для поиска данных. При создании нового индекса можно присвоить ему разные колонки и изменить порядок сортировки значений в индексе для удовлетворения требований поиска конкретных приложений, работающих с базами данных.

Создание и модифицирование индексов в *ERwin*

При генерации схемы на основе модели данных **ERwin** автоматически создает индекс для первичного ключа (*PK*) и отдельный индекс для каждого альтернативного ключа (*AK*), внешнего ключа (*FK*), "*Inversion Entry*" (*IE*). Если у сущности не было назначено альтернативных ключей и "*Inversion Entry*", то **ERwin** создает индексы только для первичного ключа и внешних ключей.

Когда **ERwin** создает индекс, он автоматически присваивает все атрибуты одного и того же ключа индексу. Например, если в сущности "*CUSTOMER*" два атрибута назначены как "*AK*", **ERwin** автоматически создает индекс "*AK1*" и выбирает в качестве колонок для нового индекса оба атрибута, составляющие ключ "*AK1*".

Можно создать новый индекс на основе альтернативного ключа или "*Inversion Entry*", используя в качестве источника существующий индекс. Можно неявным образом создать новый индекс в редакторе "*Attributes*", добавив к атрибуту новый значок ключа, например "*AK1*" или "*IE1*".

После создания индекса можно изменить его характеристики в редакторе "*Index*".

Примечание: нельзя создать новый индекс на основе первичного ключа (*PK*) или внешнего ключа (*FK*). Индексы *PK* и *FK* может создать только **ERwin**.

Имена индексов

ERwin формирует имя индекса следующим образом: "*X*" + "*PK*" (или "*AK*", "*IF*", "*IE*" + "*n*") + "Имя таблицы сущности", где *n* – целое число ≥ 1 , используемое для различения нескольких индексов одного типа, обычно внутри одной таблицы.

Примечание: **ERwin** последовательно присваивает номера индексам *IF*, исходя из порядка, в котором создаются внешние ключи в диаграмме.

Имя индекса связывается с именем физической таблицы при первом входе в редактор "*Index*" после создания индекса.

Чтобы изменить имена таблиц в физической модели в соответствии с именами сущностей в логической модели, следует в *<DB> Tables (Model/Tables...)* из списка "*Table*" выбрать таблицу, имя которой нужно

изменить, и в поле "*Name*" ввести нужное имя (в поле "*Owner*" можно ввести имя владельца).

Примечание: редактор *Index* будет отражать то табличное имя сущности, которое у нее было в момент связывания, а не новое логическое имя. Имя индекса связывается с именем таблицы при нажатии кнопок "*Preview*", "*Report*" и "*Print*" в редакторе *<DB> Schema Generation*.

Колонки индекса

ERwin автоматически выбирает атрибут в качестве индексной колонки, если знак ключа (то есть *PK*, *AK*, *IF* или *IE*) совпадает с типом ключа для индекса. Т.е., когда **ERwin** создает индекс по первичному ключу, то все атрибуты со знаком *PK* выбираются в качестве индексных колонок.

Ниже приводится пример автоматического создания индексов в **ERwin** для сущности *CUSTOMER* и колонки, выбранные для индекса по первому альтернативному ключу (*AK1*).

По умолчанию колонки индекса по внешнему ключу включают в себя все атрибуты, мигрировавшие из одной и той же сущности (один индекс по внешнему ключу – для одной родительской сущности).

Когда **ERwin** создает индекс, он автоматически располагает колонки в индексе в порядке, противоположном тому, в котором расположены атрибуты соответствующей сущности. Если присвоить индексу разные колонки, то порядок расположения колонок в редакторе "*Index*" непредсказуем.

Примечание: нельзя изменить порядок расположения колонок в индексе по первичному ключу или по внешним ключам в редакторе "*Index*". Чтобы изменить порядок расположения колонок в этих индексах, следует пользоваться редактором "*Attributes*".

Работа в редакторе "*Index*"

Редактор "*Index*" выводит на экран информацию об индексах каждой сущности в модели данных. **ERwin** вставляет имя выбранной СУБД в качестве префикса перед заголовком редактора и настраивает режимы редактора в соответствии с СУБД. Режимы редактора "*Index*" адаптируются для каждой конкретной СУБД.

Чтобы создать новый индекс по альтернативному ключу или "*Inversion Entry*", выберите имеющийся индекс, который послужит источником, и нажмите кнопку "*New*". Если в окне "*Unique*" для исходного индекса стоит метка, то **ERwin** создает индекс по альтернативному ключу. Если окно "*Unique*" пустое, то **ERwin** создает индекс по "*Inversion Entry*".

Когда **ERwin** создает индекс, он присваивает очередной порядковый номер имени индекса, а также присваивает значения по умолчанию всем характеристикам нового индекса, таким как порядок сортировки, кластеризация и т.д. Колонки с новым индексом связываются путем выбора атрибутов из вложенной диаграммы.

Чтобы изменить какие-то характеристики индекса, нужно выбрать индекс из списка наверху редактора и ввести новые значения. Во вложенной

диаграмме можно выбрать один или несколько атрибутов (с помощью методов "*SHIFT-Click*" и "*CTRL-Click*") в качестве колонок для индекса. С помощью окна вложенной диаграммы можно назначить альтернативные ключи и "*Inversion Entry*" или изменить существующие назначения.

ERwin автоматически создает индекс по внешнему ключу, когда внешний ключ создается путем миграции. По умолчанию индексы по внешним ключам не видны в редакторе "*Index*". Чтобы они были видны, поставьте метку в окне "*FK*", которое находится в верхней части редактора.

Примечание: если щелкнуть по атрибуту во вложенном окне, то **ERwin** будет считать, что нужно добавить атрибут к тому индексу, который в данный момент выделен. Для отмены изменения индекса следует нажать кнопку "*Cancel*". Нажмите ее сразу же, ДО ТОГО, как щелкните по другой сущности. Для индекса по первичному ключу требуются уникальные значения. Изменить это нельзя. Для индекса по альтернативному ключу по умолчанию задается режим "*Unique*", но его можно изменить. Для индекса по *Inversion Entry* должны быть разрешены неуникальные значения. Если изменяется режим на "*Unique*", то **ERwin** изменяет индекс на индекс по альтернативному ключу.

С помощью кнопки "*Rename*" можно изменить имя индекса, предварительно выделив индекс.

Индекс по первичному ключу удалить нельзя. При удалении индекса по альтернативному ключу или по "*Inversion Entry*" атрибуты, составляющие ключ, не удаляются. Убирается только знак альтернативного ключа или "*Inversion Entry*".

Создание индексов по альтернативным ключам и "*Inversion Entry*"

Можно с помощью редактора *Index* создать новый альтернативный ключ или "*Inversion Entry*", используя в качестве источника ранее созданный индекс. Если поставить метку в окне "*Unique*" для индекса-источника, то **ERwin** создаст индекс по альтернативному ключу. Если окно "*Unique*" для индекса-источника останется пустым, то **ERwin** создаст *Inversion Entry*. В обоих случаях **ERwin** создает новый индекс и увеличивает префикс индекса так, что он становится равен очередному доступному порядковому номеру.



Чтобы создать индекс по "*Inversion Entry*", можно выбрать любой из имеющихся индексов и убрать метку из окна "*Unique*". Если выбрать индекс по другой "*Inversion Entry*", то **ERwin** создаст индекс и увеличит префикс так, что он будет равен очередному доступному порядковому номеру.



Когда **ERwin** создает индекс, он автоматически по умолчанию запрещает повторяющиеся значения в индексах по первичным и альтернативным ключам и разрешает повторяющиеся значения в индексах по "*Inversion Entry*".

Когда в окне "*Unique*" стоит метка, то в индексированной колонке (колонках) разрешается хранить только уникальные значения.

Сортировка значений в индексе

По умолчанию **ERwin** автоматически сохраняет значения в индексе в порядке возрастания (значения сортируются по алфавиту от *A* до *Z*, а числа от 0 до 9). Если нужно изменить порядок сортировки для колонки и СУБД поддерживает режим сортировки по убыванию (*Z-A*, 9-0), выберите колонку и затем поставьте метку в окне *check box* "*Descending*".

В окне "*Index Members*" (закладка *Members*) с помощью стрелок  и  можно изменять порядок расположения колонок.

Выделив атрибут в левом списке, можно его добавить в индекс можно с помощью кнопки . Удалить атрибут из индекса можно с помощью кнопки , выделив атрибут в правом списке.

Использование кластеризованного индекса

Кластеризованный индекс – это специальная техника индексирования, при которой данные в таблице физически располагаются в индексированном порядке. Использование кластеризованного индекса значительно ускоряет выполнение запросов, которые производят выборку строк с одинаковыми значениями в индексированной колонке (колонках). Поскольку данные физически расположены в индексированном порядке, для каждой таблицы может существовать только один кластеризованный индекс. Если ваша СУБД поддерживает использование кластеризованного индекса, то **ERwin** автоматически создает индекс первичного ключа кластеризованным. Если нужно произвести кластеризацию индекса не по первичному ключу, а каким-то другим способом, то **ERwin** автоматически снимает кластеризацию с индекса по первичному ключу.

В некоторых СУБД редактор **ERwin** "*Index*" позволяет присвоить сущности кластеризованный хешированный индекс (*clustered hashed index*). Хеширование - альтернативный способ хранения данных в заранее заданном порядке с целью ускорения поиска, но физически это более сложно, чем простое сохранение строк в алфавитном порядке или в соответствии с числовыми значениями.

Поскольку данные хранятся в таком порядке, как в индексе, только один индекс может быть кластеризован или хеширован для каждой таблицы, и нельзя задать для одной таблицы кластеризованный и хешированный индекс одновременно. При задании для таблицы кластеризованного или хешированного индекса добавление и удаление строк для этой таблицы замедляется, поскольку для соответствия индексу необходимо реорганизовать данные.

Изменение физических характеристик индекса

При обратном проектировании базы данных **ERwin** импортирует информацию относительно физической памяти для таблиц и индексов, которая определяет, в каком месте эти объекты физически хранятся на сервере.

Можно изменять любые физические параметры, выводимые на экран в редакторе "Index".

Таблица 3.1

Режимы физических параметров, доступные в редакторах
Index для СУБД *ORACLE*, *SYBASE* и *SQL Server*

| Режим физического хранения | Для чего используется режим |
|----------------------------|---|
| <i>ORACLE</i> | |
| <i>PCTFREE</i> | Задаёт размер пространства, которое нужно оставить свободным для обновлений и вставок в каждом блоке данных. |
| <i>NO SORT</i> | Ускоряет создание индекса, если данные расположены физически по порядку. Если в окне стоит <i>X</i> , то значения индекса не сортируются. Если окно пустое, то значения индекса сортируются. |
| <i>INITTRANS</i> | Задаёт параметры для команды СУБД <i>CREATE TABLE</i> . |
| <i>MAXTRANS</i> | Задаёт параметры для команды СУБД <i>CREATE TABLE</i> . |
| <i>SQL и SYBASE</i> | |
| <i>IGNORE_DUP_KEY</i> | Разрешает или запрещает использование повторяющихся значений ключа в таблице с уникальным индексом (кластеризованным или некластеризованным). Если в окне стоит <i>X</i> , то повторяющиеся значения не допускаются. Если окно пустое, то повторяющиеся значения разрешаются. |
| <i>SORTED DATA</i> | Ускоряет создание индекса, если данные расположены физически по порядку. Если в окне стоит <i>X</i> , то значения индекса не сортируются. Если окно пустое, то значения индекса сортируются. |
| <i>DUP ROW</i> | Разрешает или запрещает использование повторяющихся значений ключа в таблице с кластеризованным индексом. Если в окне стоит <i>X</i> , то повторяющиеся значения не допускаются. Если окно пустое, то повторяющиеся значения разрешаются. |
| <i>FILL_FACTOR</i> | Задаёт, сколько данных можно добавить к странице данных при создании индекса. |

Режимы индексирования для разных СУБД

ERwin позволяет изменять имя индекса, но при этом длина имени ограничивается конкретной СУБД. Можно изменить порядок сортировки по умолчанию с возрастающего на убывающий. Можно задать кластеризованный индекс.

Список возможностей и режимов, которые поддерживаются
ERwin для каждой СУБД

| СУБД | Максимальная длина имени индекса | Поддерживается ли сортировка по убыванию | Поддерживаются ли кластеризованные индексы | Поддерживаются ли характеристики физического хранения |
|--------------------|----------------------------------|--|--|---|
| <i>DB2</i> | 18 | ДА | ДА | НЕТ |
| <i>SQL Server</i> | 30 | НЕТ | ДА | ДА |
| <i>Rdb</i> | 30 | ДА | ДА | НЕТ |
| <i>ORACLE</i> | 30 | ДА | НЕТ | ДА |
| <i>SQL Base</i> | 18 | ДА | ДА (хешированные) | НЕТ |
| <i>WATCOM</i> | 128 | ДА | ДА | НЕТ |
| <i>Ingres</i> | 24 | ДА | ДА | НЕТ |
| <i>SYBASE</i> | 30 | НЕТ | НЕТ | ДА |
| <i>AS/400</i> | 30 | ДА | ДА | НЕТ |
| <i>NetWare SQL</i> | 20 | НЕТ | НЕТ | НЕТ |
| <i>INFORMIX</i> | 18 | ДА | ДА | НЕТ |
| <i>Progress</i> | 30 | ДА | ДА | НЕТ |

Редактирование объектов

Drawing Tools – панель инструментов для рисования.

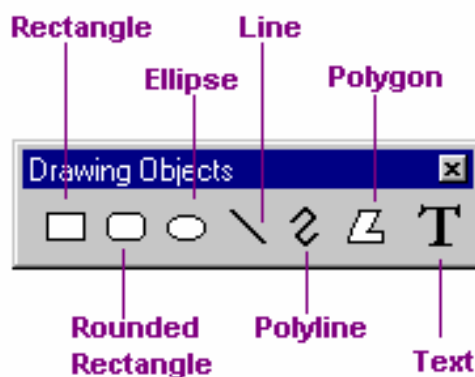


Рис. 3.39. Панель редактирования

После добавления объекта для рисования можно воспользоваться шрифтом и изменить цвет этих добавляемых объектов.

Можно определить шрифт и цвет объектов для рисования. В меню "Format" установите шрифт и цвет по умолчанию. Щелкните "Drawing Objects" в "Default Fonts and Colors" для подтверждения установки по умолчанию для создания объектов.

При расположении объекта на диаграмме, **ERwin** автоматически присваивает объекту – сущности, атрибуту, связи и тексту – цвет и шрифт в соответствии со значениями, присваиваемыми по умолчанию. Можно изменить цвет и шрифт любого индивидуального объекта или группы в любое время, войдя в соответствующий редактор и изменив текущие значения.

По умолчанию атрибут внешнего ключа автоматически наследует цвет и шрифт связанного с ним первичного ключа.

Обычно диаграмма в **ERwin** содержит три типа графических характеристик: текст, объекты заполнения и контур.

Графические характеристики каждого объекта распространяются на шрифты, цвет текста, цвет линии и цвет заполнения.

Некоторые объекты обладают множеством графических характеристик, другие – только одной характеристикой. Например, можно использовать разные шрифты и цвета для улучшения внешнего вида текстового блока. Но по отношению к фону диаграммы или окну сущности можно использовать только один цвет.

Редакторы шрифта и цвета

ERwin позволяет редактировать шрифты и цвета объектов на диаграмме в двух редакторах: "*Object Font&Color*" и "*Default Font&Color*". Войти в редактор "*Default Font&Color*" можно из меню "*Format*", выбрав команду "*Fonts&Color*".

Примечание: шрифтовые и цветовые установки применяются как к логической модели, так и к физической схеме.

Редактор "*Object Font&Color*" позволяет изменять шрифты и (или) цвета одного объекта. Изменения, сделанные в этом редакторе для одного объекта, не распространяются на другие объекты, находящиеся на диаграмме сейчас, и на объекты, которые будут добавлены потом.

Редактор "*Default Font&Color*" позволяет изменять шрифты и цвета для всех объектов из одной группы объектов. Изменения, сделанные в этом редакторе по отношению к группе объектов, могут отразиться на других объектах, находящихся на диаграмме, а также и на тех, которые будут добавлены потом.

Редактор "*Object Font&Color*"

Редактор позволяет изменять шрифтовые и цветовые характеристики одного объекта на диаграмме (имя сущности, один или несколько атрибутов, линия связи или текстовый блок).

Редактор является контекстно-чувствительным. Строка-заголовок, различные метки и варианты выбора изменяются в зависимости от выбранного объекта (при выборе связи заголовок будет выглядеть так: "*Relationship Font&Color*")

Можно пользоваться закладками характеристик, которые находятся в нижней части редактора, для задания шрифтов и (или) цветов для характе-

ристик объекта. Например, если была выбрана сущность, то можно выбрать следующие закладки:

- *Color* – для задания типа, стиля, размера шрифта и цветов для текста имени сущности и атрибута;
- *Fill Color* – для задания цвета окна сущности;
- *Outline Color* – для задания цвета контуру окна сущности.

Варианты выбора, предлагаемые в окне-списке, также меняются в зависимости от выбранного объекта. Например, если был выбран "*Color*", то появляются варианты шрифтов и цветов. Если же был выбран "*Outline Color*", появляются только варианты цветов.

Редактор "*Default Fonts&Colors*"

В редакторе имеются следующие закладки (для логического и физического представлений):

- *Entities*

Блок *Name* – блок имени сущности. Можно выбрать стиль шрифта (*Font*), размер шрифта (*Font Size*), шрифт можно сделать жирным (*Bold*), шрифт можно сделать курсивом (*Italic*), можно выбрать цвет шрифта (*Color*). Для этого нужно поставить соответствующую метку. Метка в "*Strikeout*" для перечеркивания выбранного объекта. Метка в "*Underline*" для подчеркивания выбранного текста.

Блок *Definition* – определение сущности. Можно выбрать стиль шрифта в определении сущности (*Font*), размер шрифта (*Font Size*), шрифт можно сделать жирным (*Bold*), шрифт можно сделать курсивом (*Italic*), можно выбрать цвет шрифта (*Color*). Для этого нужно поставить соответствующую метку. Метка в "*Strikeout*" для перечеркивания выбранного объекта. Метка в "*Underline*" для подчеркивания выбранного текста.

Блок *Box*. Можно определить цвет заполнения (*Fill Color*) и цвет контура (*Outline Color*).

Блок *Apply To*. С помощью кнопок "*Apply To*" можно контролировать порядок присваивания шрифтовых и цветовых значений в *ERwin* к существующим объектам и к новым объектам, которые будут добавляться в диаграмму.

- *All Objects* – новые установки будут применяться как к уже существующим объектам, так и к тем, которые будут добавляться в диаграмму;
- *New Objects* – новые установки будут применяться к тем объектам, которые будут добавляться в диаграмму. Установки для уже существующих объектов не изменятся;
- *Current Object* – новые установки будут применяться как к уже существующим объектам, так и к тем, которые будут добавляться в диаграмму.

Если текущий набор объектов – область, отличная от главной области, и выбирается режим "*Current Object*", то новые установки будут примене-

ны как к уже существующим, так и к новым объектам этой области. Но объекты главной области, которые не входят в эту область, не попадут под влияние новых установок для текущей области.

- *Attributes*

Блок *Owned*: в блоке можно определить цвет, стиль, тип, размер шрифта.

Блок *Foreign Key* – атрибут внешнего ключа.

Стиль шрифта может наследоваться из отношения (*Inherit Font From PK*) и первичного ключа (*Inherit Font From Relation*).

Цвет шрифта может наследоваться из отношения (*Inherit Color From PK*) и первичного ключа (*Inherit Color From Relation*).

Убрав метки "*Inherit Font From PK*" и "*Inherit Color From Relation*" можно вручную определить характеристики шрифта.

- *General*

Background – блок фона диаграммы.

Блок "*All Fonts*" позволяет изменить цветовые и шрифтовые характеристики всех объектов (включая контуры сущностей и линии связи) на диаграмме, за исключением фоновой заливки, заливки сущности и заливки подтипа.

- *Relationships*

Метка "*Inherit From Parent Entity Name*" означает наследование характеристик из родительского имени сущности. При отсутствии метки можно вручную определить характеристики.

Блок *Line Colors* – цвет линии сущности

- *Subtypes*

Блок *Discriminator*: применение шрифта применительно к связи "*Subtypes*".

Блок *Colors*: заливка значка связи (*Subtype Symbol Fill Color*) и цвет линии связи (*Subtype Symbol Line Color*).

- *Drawing Object Text* – блок описания текстовых характеристик нарисованных объектов.

Блок *Horizontal Alignment* – выравнивание по горизонтали.

Блок *Vertical Alignment* – выравнивание по вертикали.

- *Drawing Object Color* – блок описания цветовых характеристик нарисованных объектов.

Блок *Fill* - заливка объекта.

Блок *Line* - линия объекта.

Порядок выполнения работы

1. Откройте модель "*lab_3_1*", сохраненную в третьей лабораторной работе. Удалите сущность "*Apple*".

2. В сущность "*Customer*" добавьте следующие атрибуты (не определяйте атрибуты как первичные ключи): "*C_name*", "*C_pol*", "*C_god*", "*C_country*".

3. В сущность "Order" добавьте следующие атрибуты: "O_name", "O_pol", "O_god", "O_country".

4. В сущность "Product" добавьте следующие атрибуты: "P_name", "P_pol", "P_god", "P_country".

5. Задайте для атрибутов "C_name", "O_name", "P_name", "C_pol", "O_pol", "P_pol", "C_country", "O_country", "P_country" текстовый тип; для атрибутов "O_god", "P_god", "C_god" тип *DATE*. Для атрибутов "C_god", "O_god", "P_god" задайте значения по умолчанию: *My_default* - 23.06.1982 (logical only) и правило коррекции введенных значений: *My_validation* (диапазон значений: от 01.01.1800 до 31.12.2100).

6. Назначьте атрибуты "C_god" альтернативным ключом, "O_god" альтернативным ключом, "P_god" альтернативным ключом. Назначьте группу атрибутов "P_name" и "P_country" альтернативным ключом, "C_name" и "C_country" альтернативным ключом, "O_name" и "O_country" альтернативным ключом. Включите отображение альтернативных ключей: *Entity Display/Alternate Key Designator (AK)*.

7. Для атрибутов "O_pol", "P_pol" создайте неуникальные индексы.

8. Перенесите атрибуты "C_god" и "C_pol" из сущности "Customer" в сущность "Product".

9. Измените имя индексов атрибута "C_god" и "C_pol" в сущности "Product" на "P_age" и "P_child" соответственно.

10. Задайте правило проверки введенных значений для атрибутов "P_age": *Age_validation* (от 0 до 150) и "P_child": *Child_validation* (да и нет).

11. Сущностям "Customer", "Order", "Product" присвойте кластеризованный индекс первичного ключа. Для отображения редактора "Index" необходимо перейти на физический вид модели.

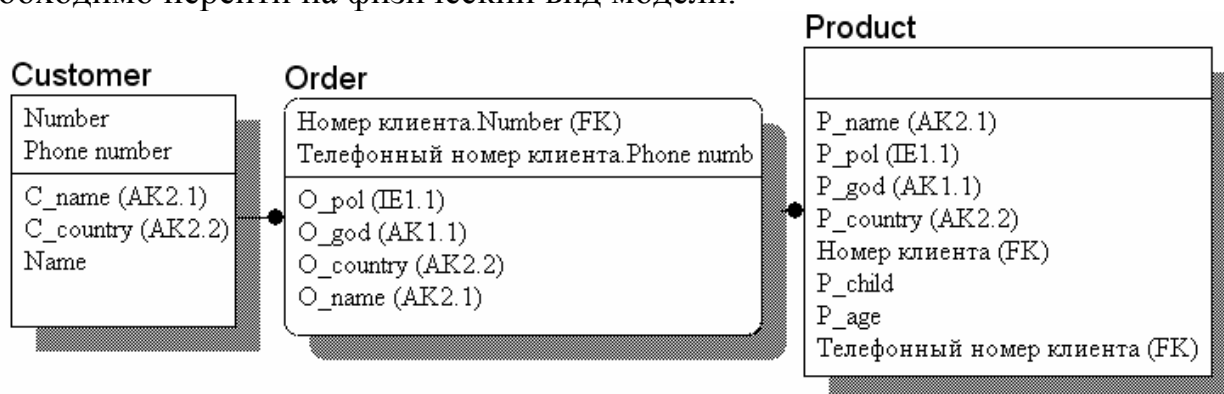


Рис. 3.40. Результат выполнения первой части лабораторной работы

12. Сохраните модель как "lab_4_1.er1".

13. В *Drawing Objects* выберите средство "закругленный прямоугольник" (*Rounded Rectangle*).

14. Поместите закругленный прямоугольник вокруг "Customer" и "Product". Щелкните по стрелке "Apply Background Color" панели инструментов для изменения цвета заливки прямоугольника.



**Apply
Background Color**

Рис. 3.41. Панель инструментов для
изменения цвета заливки

Выберите любой оттенок синего цвета, который будет фоном прямоугольника.

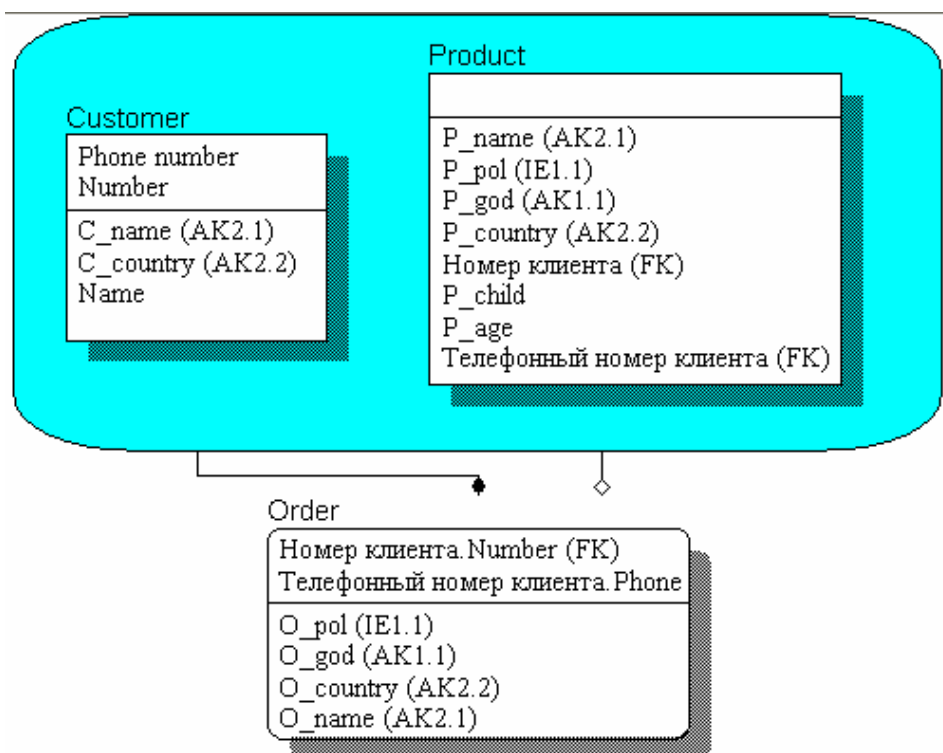


Рис. 3.42. Создание фона прямоугольника

15. Аналогичным образом сделайте желтый фон вокруг "Order".

16. Двойным кликом по желтому фону откройте редактор "Drawing Objects". Откройте закладку Text и введите образец "A product is part of an order".

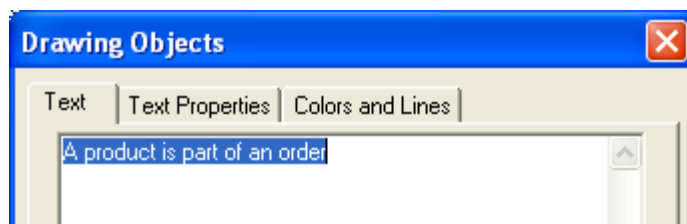


Рис. 3.43. Внесение текста

17. Откройте закладку "Text Properties" (свойства текста) и выберите любой темный цвет для текста. Щелкните *OK* для закрытия диалога и увидите текст в круге.

18. Задайте красный фон для модели.

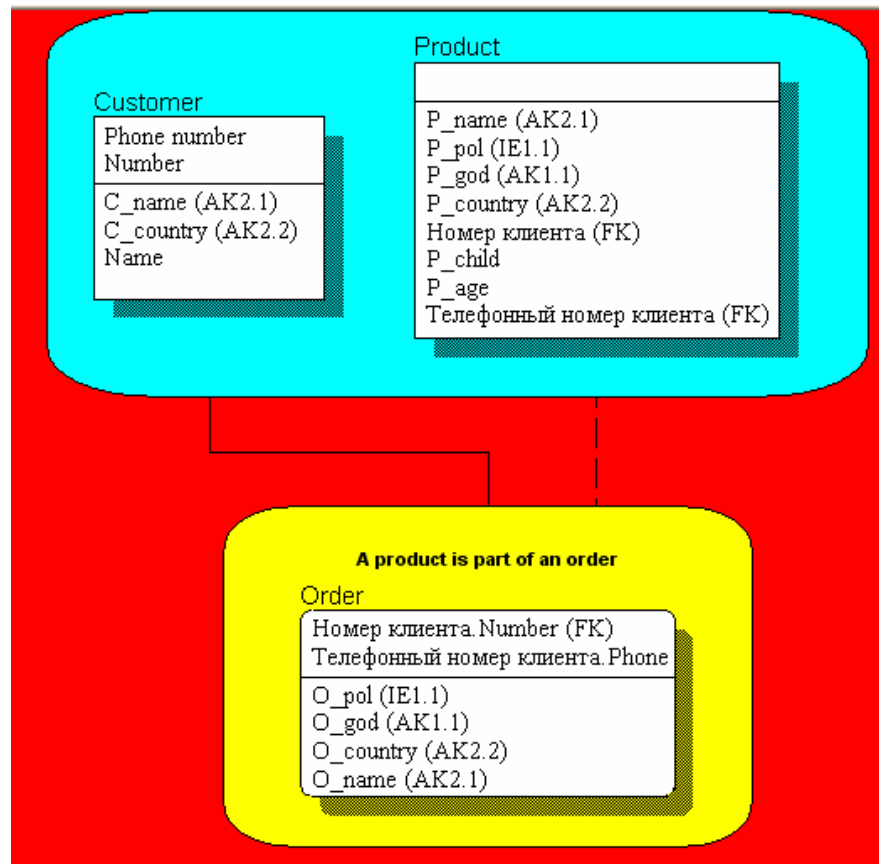


Рис. 3.44. Модель после редактирования

19. Сохраните модель как "lab_4_2.erl".

Контрольные вопросы

1. Что такое альтернативный ключ?
2. Что такое инверсное вхождение?
3. Что такое индекс?
4. Что такое кластеризованный индекс?
5. Для чего производится индексирование баз данных?

Лабораторная работа №5. Прямое и обратное проектирование

Цель работы: Получить навыки в генерации схемы базы данных, файла сценария.

Теоретические сведения

ERwin поддерживает обратное проектирование существующих баз данных и обеспечивает физическую и логико-физическую модель данных, поэтому можно поддерживать созданную базу данных или осуществлять передачу от вашего текущего целевого сервера к другому.

Распределение **ERwin's Complete Compare** автоматизирует модель и базу данных синхронизации; можно сравнить модели с базой данных с отображением различий. Можно выборочно перемещать различия в модель или вносить их в базу данных.

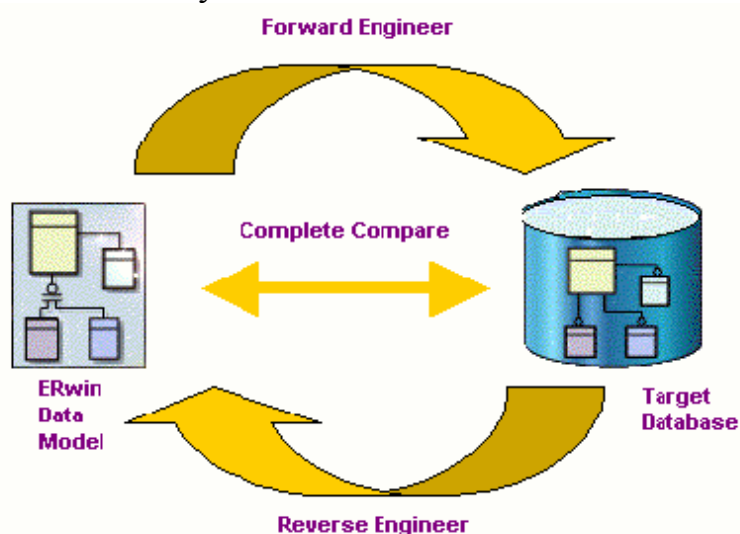


Рис. 3.45. Проектирование

Процесс генерации физической схемы базы данных из логической модели данных называется прямым проектированием (*Forward Engineering*).

Процесс генерации логической модели из физической базы данных называется обратным проектированием (*Reverse Engineering*). После создания модели можно произвести обратное проектирование структуры базы данных, а затем легко перенести его в другой формат базы данных.

Когда есть физическая модель данных, **ERwin** автоматически генерирует схему для целевого сервера при формировании модели. Обратное проектирование является процессом, который используется **ERwin**, для передачи схемы от модели данных к целевому серверу. Когда пересылается проект модели данных, можно сгенерировать файл сценария (*script*), который будет использоваться для корректировки базы данных. Корректировка базы данных будет вестись с помощью административного средства (*database administration tool*) или пересылкой проекта непосредственно для подключения к базе данных каталога.

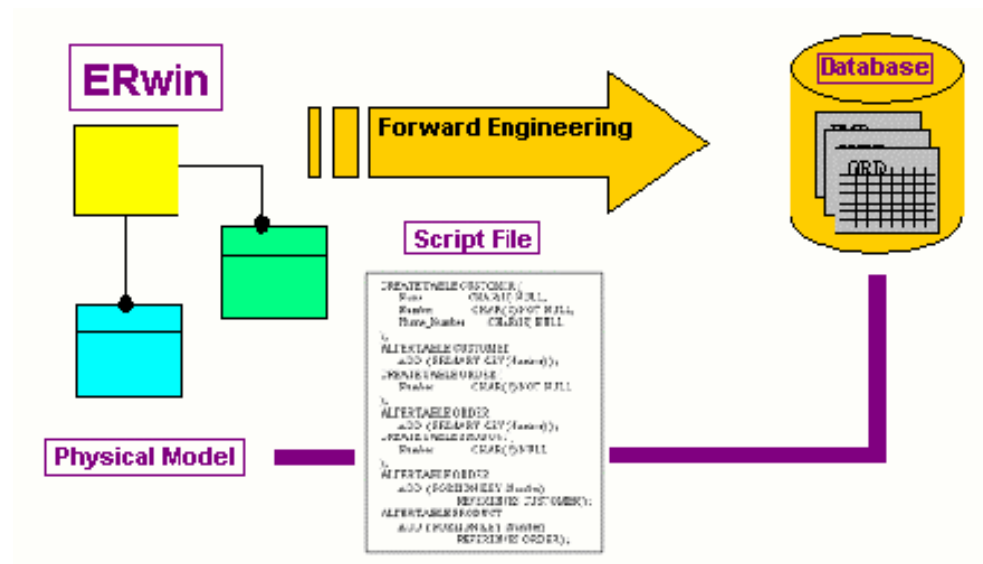


Рис. 3.46. Корректировка базы данных

Перед пересылкой проекта можно рассмотреть схему, которая является базовым текстовым представлением базы данных объектов, которые будут созданы в базе данных из сценария (*script*). **ERwin** использует язык определения данных (*DDL*) для целевой базы данных, чтобы записывать *script*. При каждом добавлении объекта или свойства в модель данных, **ERwin** автоматически корректирует файл сценария, чтобы отображения изменений в модели данных.

ERwin может произвести обратное проектирование существующей базы данных *SQL*, считывая определения схемы и автоматически создавая соответствующую диаграмму модели данных. При обратном проектировании базы данных, **ERwin** автоматически генерирует расположение объектов на диаграмме по умолчанию. После того как диаграмма будет сгенерирована в результате обратного проектирования, можно, используя инструменты и редакторы **ERwin**, добавлять новые объекты, создавать системную документацию и перепроектировать структуру базы данных, основываясь на изменениях технических и (или) организационных требований.

Разные СУБД на разных уровнях обеспечивают синтаксическую поддержку связей, индексов, ссылочной целостности и других свойств.

ERwin автоматически создает новое окно диаграммы главной области и показывает на экране схему в виде графической модели данных.

Помимо импорта информации, явным образом определенной в физической схеме, **ERwin** извлекает значительный объем информации из схемы и встраивает ее в диаграмму в процессе создания новой модели данных путем обратного проектирования.

Запуск процесса проектирования данных

При создании модель данных обратным проектированием можно ускорять проектировку новой модели данных с последующей поставкой новых систем.

ward Engineer/Schema Generation". Возможности, доступные в редакторе, различаются в зависимости от того, какие возможности поддерживаются пользовательской СУБД.

Для использования части сущностей текущей области для генерации схемы следует воспользоваться кнопкой "*Filter*" в редакторе.

Кнопки "*Preview*", "*Print*" и "*Report*" в нижней части редактора позволяют просматривать отчет на экране, распечатывать его или сохранять на диске в текстовом файле.

Кнопка "*Generate*" в редакторе *Schema Generation Report* служит для запуска процесса генерации схемы. При нажатии на кнопку "*Generate*", **ERwin** выводит на экран диалог *<DB> Connection*, который позволяет подключиться к базе данных и связать **ERwin** с системным каталогом базы данных.

Для генерации схемы из окна *Preview* следует выполнить следующие действия:

1. Нажмите кнопку "*Preview*", расположенную в нижней части редактора, для входа в окно *Schema Generation Preview*.

2. По умолчанию **ERwin** генерирует всю схему полностью. Для того чтобы сгенерировать часть схемы, нажмите левую кнопку мыши и, не отпуская ее, передвигайте мышь вниз, выделяя текст схемы, который следует выбрать. Отпустите кнопку мыши, когда будет достигнут конец генерируемой части.

3. Нажмите кнопку "*Generate*", расположенную в нижней части окна *Preview*. **ERwin** генерирует схему.

4. Если при генерации схемы возникнет ошибка, **ERwin** выдает сообщение об ошибке.

- Чтобы игнорировать ошибку и продолжить работу по генерации схемы, нажмите кнопку "*Continue*".
- Чтобы остановить процесс генерации схемы, нажмите кнопку "*Abort*". **ERwin** возвращается в редактор *<DB> Schema Generation*.

5. После того как **ERwin** завершит процесс генерации схемы, он возвращается в окно "*Preview*".

- Для выхода из окна "*Preview*" в редактор "*<DB> Schema Generation*" нажмите кнопку "*Close*".
- Чтобы распечатать отчет по схеме в том виде, в котором он демонстрируется в окне "*Preview*", нажмите кнопку "*Print*". **ERwin** закрывает редактор "*<DB> Schema Generation*" и распечатывает отчет. При редактировании отчета в окне *Preview* **ERwin** распечатывает отчет с изменениями.
- Для сохранения файла отчета по схеме можно нажать кнопку "*Report*". **ERwin** открывает окно-диалог "*Generate <DB> Schema Report*".

Примечание: если поставить метку в окне "*Stop If Failure*", **ERwin** приостановит работу в случае ошибки. Нажатием кнопки "*Continue*" про-

должается генерация схемы. Нажатием кнопки "*Abort*" отменяется генерация схемы. Можно отредактировать коды в окне "*Preview*", а затем нажать кнопку "*Generate*", чтобы сгенерировать отредактированную версию.

Сохранить файл отчета по схеме можно с помощью нажатия на кнопку "*Report*". В диалоге "*Generate <DB> Schema Report*" выбрать тип.

Примечание: при сохранении с расширением "*ERS*" можно войти в "*Schema Generation*" из *File Manager*, дважды щелкнув по имени файла.

Режимы генерации схемы

При работе в редакторе *Schema Generation* доступны различные режимы, в зависимости от СУБД. Поддерживаемые режимы находятся в соответствующем групповом окне. Список групповых окон: *Referential Integrity*, *Trigger*, *Table*, *Index*, *Column*, *Schema* и *Other Options*.

Referential Integrity – режимы *RI* (ссылочной целостности) позволяют указывать, как поступать со связанными записями, если значение в поле ключа изменяется или удаляется. Возможные режимы:

- *Primary Key* – для усиления уникальности определения каждой строки в таблице.
- *Foreign Key* – для усиления заданного правила ссылочной целостности в случае, когда значение во внешнем ключе изменяется.
- *On Delete* – для усиления заданного режима ссылочной целостности в случае, если значение удаляется в поле первичного или внешнего ключа.
- *Unique (AK)* – для усиления правила ссылочной целостности, требующего, чтобы значения альтернативных ключей были уникальными.
- *Create/PK* – для включения системной процедуры, создающей первичный ключ в каждой таблице.
- *Create/FK* – для включения системной процедуры, создающей внешние ключи.

Trigger – режимы триггера позволяют переопределить шаблоны *RI*, устанавливаемые ***ERwin*** по умолчанию, с целью усиления ссылочной целостности.

Возможные режимы:

- *RI Type Override* – для переопределения шаблона, устанавливаемого по умолчанию, для всех связей, которые были присвоены определенному типу правила ссылочной целостности;
- *Relationship Override* – для переопределения шаблона, устанавливаемого по умолчанию, для какой-то конкретной связи;
- *Table* – режимы для таблиц позволяют указать, какие операторы языка определения данных будут использованы при создании схемы.

Возможные режимы:

CREATE TABLE – для выполнения операторов *SQL CREATE TABLE* в процессе генерации схемы.

DROP TABLE – для выполнения операторов *SQL DROP TABLE* перед выполнением операторов *CREATE TABLE* при генерации схемы.

- *Physical Storage* – для включения в схему объектов и параметров физической памяти;
- *Table Validation* – для включения операторов *SQL*, создающих правила, которые накладывают ограничения, для каждой сущности;
- *Pre-Script* – для включения в схему пре-скриптов (скриптов, выполняемых непосредственно перед генерацией схемы);
- *Post-Script* – для включения в схему пост-скриптов (скриптов, выполняемых непосредственно после генерации схемы).

Index – режимы индексирования указывают, каким образом будут создаваться и храниться индексы и какие из ключевых атрибутов будут индексированы. Возможные режимы:

- *Primary Key (PK)* – для создания индекса по первичному ключу в каждой сущности;
- *Alternate Key (AK)* – для создания индекса по альтернативным ключам в каждой сущности;
- *Foreign Key (FK)* – для создания индекса по внешним ключам в каждой сущности;
- *Inversion Entry (IE)* – для создания индекса по инверсионным ключам в каждой сущности.

CLUSTERED – для создания в схеме индекса *CLUSTERED*.

- *Physical Storage* – для включения в схему информации, относящейся к объектам физической памяти;
- *Column* – режимы для колонок позволяют добавлять ограничения в операторы *SQL CREATE TABLE*. Выберите один или несколько возможных режимов;
- *Default* – для включения значения колонки по умолчанию в оператор схемы;
- *Physical Order* – для сохранения физического порядка расположения колонок при генерации новой схемы;
- *User Datatype* – для включения типа данных в оператор схемы, заданного пользователем для колонки;
- *Validation* – для включения в оператор схемы правило проверки введенных значений для колонки.

Other Options – другие доступные режимы поддерживают специальные возможности, предоставляемые выбранной СУБД. Возможные режимы:

- *Constraint Name* – для включения в схему имен ограничений;
- *Quote Names* – для заключения имен таблиц и колонок в кавычки;
- *Owner* – для включения владельца

Примечание: При генерации схемы на сервере все изменения табличных характеристик, сделанные в *ERwin* не распространяются на базу

данных, если не будет удалена измененная таблица (*DROP*) и не будет создана заново (*CREATE*). Чтобы заменить старую таблицу на новую, поставьте метку в окна режимов "*DROP TABLE*" и "*CREATE TABLE*" в "*Schema Generation*".

Порядок выполнения работы

1. Откройте модель "*lab_4_1.er1*", которая была сохранена в четвертой лабораторной работе. С помощью индикатора типа модели переключитесь на физическую модель.
2. Из меню "*Tools*" выберите *Forward Engineer/Schema Generation*.

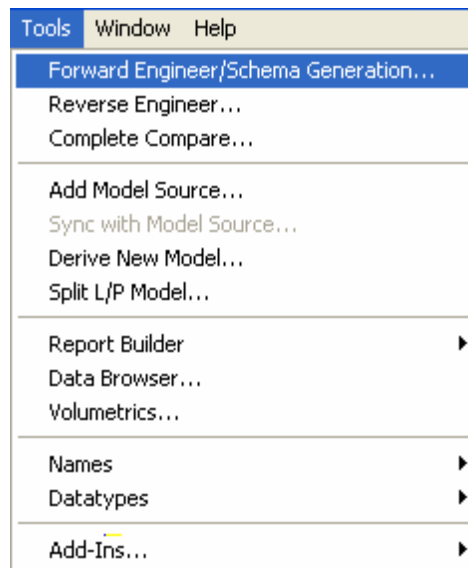


Рис. 3.48. Генерация схемы

Появится диалог "*SQL Server Schema Generation*".

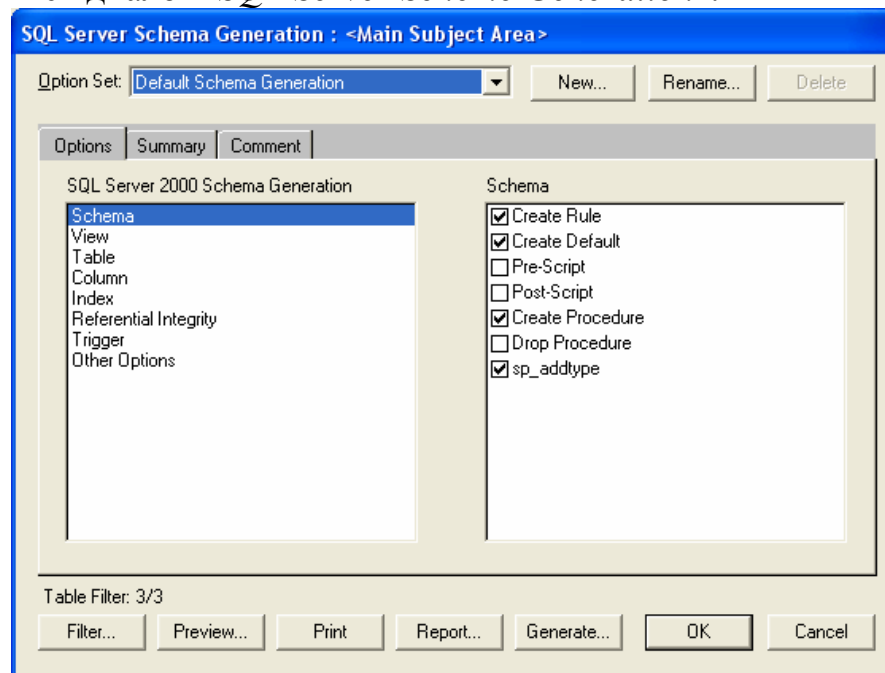


Рис .3.49. Схема генерации

3. Щелкните кнопку *Preview*, расположенную внизу диалога, для предварительного просмотра. Появится *SQL Server Scheme Generation Preview*.

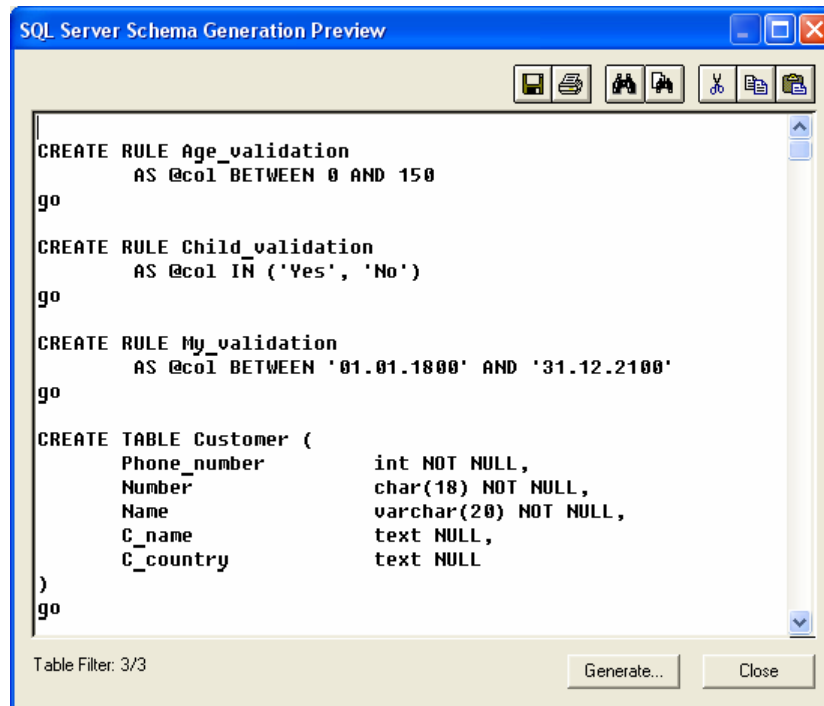


Рис. 3.50. Предварительный просмотр сгенерированной схемы

4. Для завершения просмотра схемы и возврата в диалог "*SQL Server Scheme Generation*", нажмите кнопку "*Close*". Целевым сервером является *SQL Server 2000* (см. лабораторную работу №1). Для того чтобы изменить физическую модель данных для открытия в целевом сервере, нужно в меню базы данных (*Database*) выбрать *Choose Database*. Появится диалог *Computer Associates Erwin – Target Server* (целевой сервер).

5. Щелкните кнопку *Report*, которая расположена в нижней части диалога.

6. В диалоге "*Save As*" в блоке "*File Name*" в качестве имени файла укажите "*My ERwin Model.sql*" и затем щелкните "*Save*".

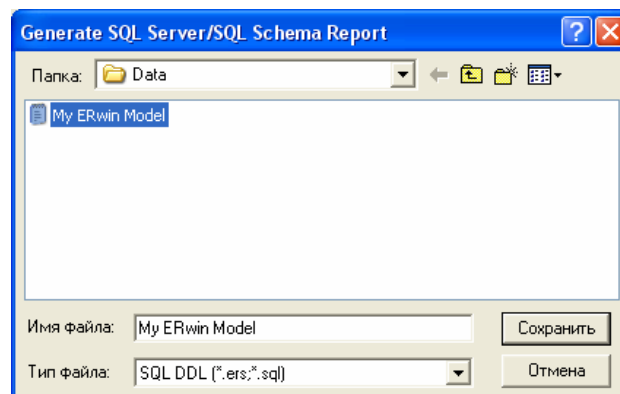


Рис. 3.51. Сохранение отчета сгенерированной схемы

7. Сохраните модель как "lab_5_1.erl".
8. Из меню панели инструментов "Tools" выберите *Reverse Engineer*. Появится диалог "Reverse Engineer – Select Template".
9. В диалоге выбираем "Physical" в качестве нового типа модели (блок *New Model Type*). В качестве шаблона выберите "Blank Physical Model" и "SQL Server 2000" в качестве целевой базы данных.

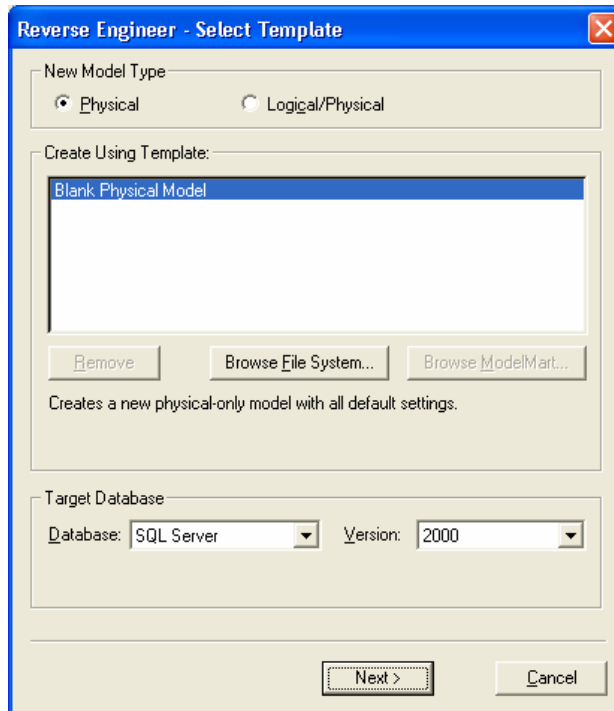


Рис. 3.52. Выбор базы данных

10. Затем щелкните "Next". Появится диалоговое окно "Reverse Engineer Set Options" (установка опций обратного проектирования):

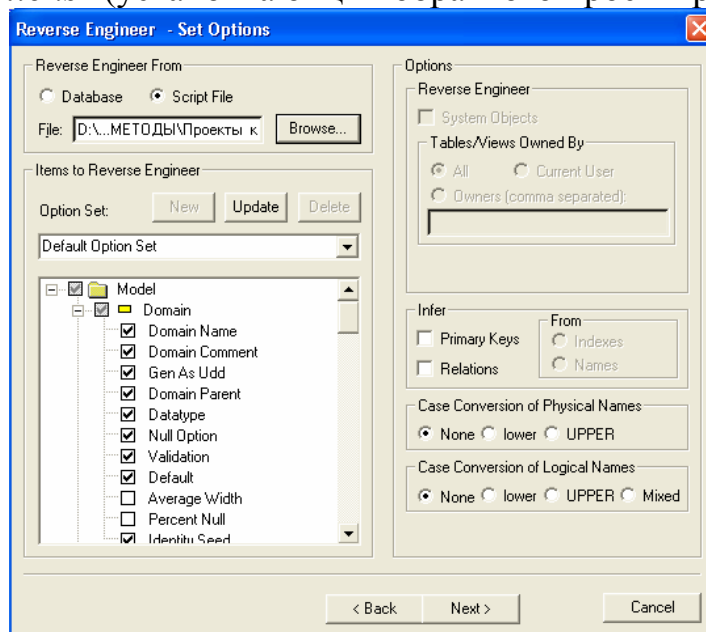


Рис. 3.53. Установка опций обратного проектирования

11. В *Reverse Engineer From* выберите "Script File" и щелкните "Browser" (просмотр), чтобы расположить файл "My **ERwin** Model.sql", который был сохранен выше. Для данного упражнения следует принять по умолчанию установки в остальных областях диалога и щелкать "Next". После проведенных операций будет виден небольшой диалог с текстом, который описывает структуру базу данных, это обратное проектирование **ERwin**. При завершении процесса новая модель данных появляется в окне диаграммы.

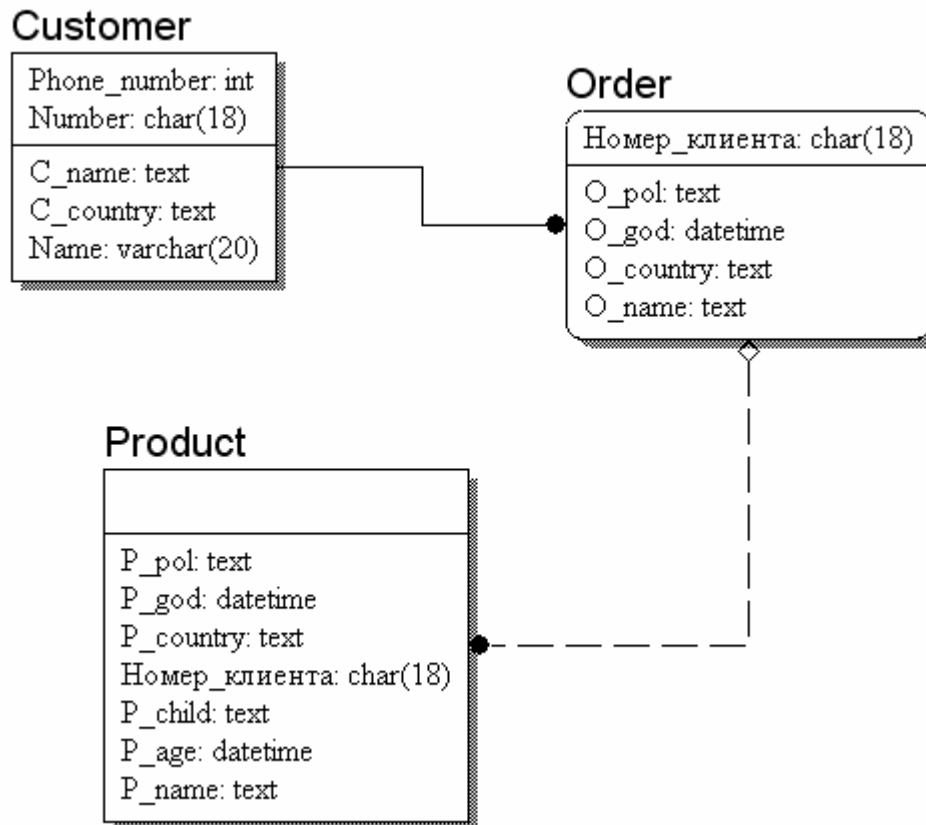


Рис. 3.54. Диаграмма в **ERwin**

12. В меню "File" выберите "Save" и сохраните модель как "lab_5_2.erl".

13. В *Model Explorer* в закладке *Model*, щелкните по плюсу у *Tables* для расширения таблицы. Затем щелкните по плюсу у *CUSTOMER* для его расширения. Затем расширьте столбцы (*Columns*) под *CUSTOMER* и удалите столбец "phone_number".

14. Сохраните модель.

15. В меню "Tools" выберите *Complete Compare*. Появится диалог *Complete Compare – Set Options*. В групповом блоке "Compare Type" выберите "Database Level Compare".

- *Database level compare* – сравнение на уровне базы данных;
- *Model level compare* – сравнение на уровне модели;
- Модель можно сравнивать с базой данных (*Database*), со сценарием файла (*Script File*) с файлом с расширением "erl".

16. В групповом блоке "*Compare Current Model With*" выберите "*Script File*", щелкните "*Browse*" (чтобы найти сценарий файла), определите файл "*My ERwin Model.SQL*" (с полным путем к нему).

17. В групповом блоке "*Sync Action*" выберите "*Update Current*" и щелкните "*Next*".

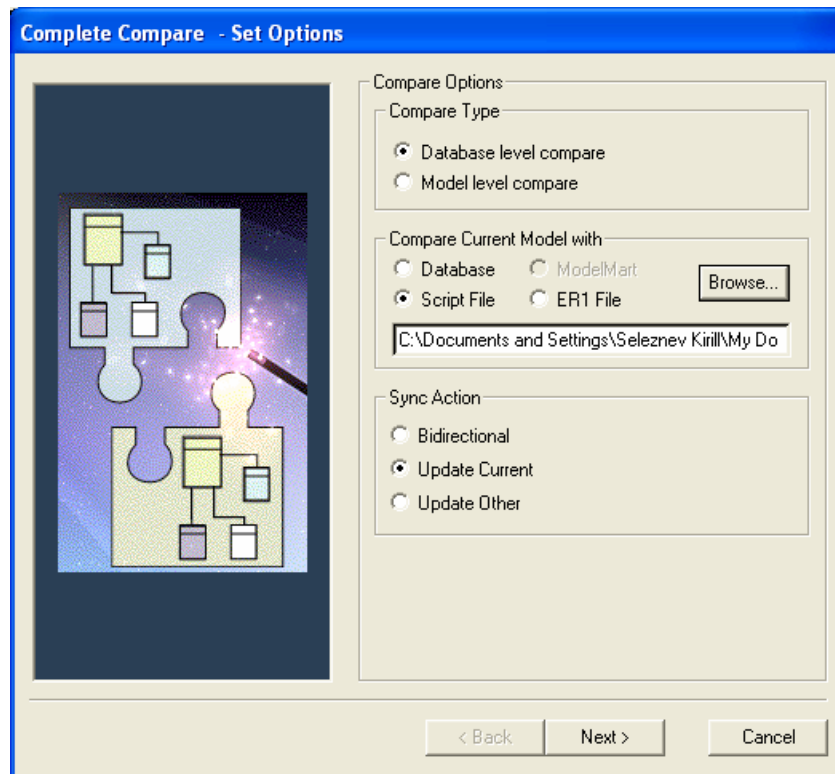


Рис. 3.55. Диалог выбора

- *Update Current* – синхронизировать текущие изменения;
- *Bidirectional* – двунаправленная синхронизация;
- *Update Other* – синхронизировать нетекущие изменения.

18. В диалоговом окне *Complete Compare – Items to Compare* примите все по умолчанию и щелкните "*Next*".

19. В левой части окна можно отметить галочкой объекты, подлежащие синхронизации (или можно принять установки по умолчанию).

20. В диалоговом окне *Complete Compare – Object Filter Options* примите все по умолчанию (можно установить фильтр на объект установкой метки, что означает, что объект не будет сравнен) и щелкните "*Next*". В этой точке **ERwin** начинает обрабатывать *Script File* (файл сценария).

21. В диалоговом окне *Complete Compare – Other Model Filter Options* примите все по умолчанию и щелкните "*Next*".

22. Далее в диалоговом окне *Complete Compare – Resolve Differences* появятся два дерева друг возле друга.

23. В них представлены различия в **ERwin** между текущей моделью и файлом сценария. Левое дерево помечено как "*lab_5_2*", а правое дерево помечено как "*My ERwin Model.sql*".

24. На странице *Complete Compare – Resolve Differences* просмотрите различия между моделью и сценарием *SQL*.

25. В правом дереве выберите столбец *Phone_Number* таблицы *CUSTOMER*, щелкнув по линии для его выделения.

26. Щелкните по кнопке "*Import*" в правой части диалога. Затем щелкните "*Next*". Появится диалоговое окно *Complete Compare – Import Changes*. Нажмите кнопку *Start Import*. В окне появится сообщение *Done Importing*. Для просмотра результатов на *View Results*. Появится окно *Complete Compare – Import Summary*. Нажмите кнопку *Close* для закрытия окна просмотра. Нажмите *Finish* для завершения импорта столбца *Phone_number* таблицы *CUSTOMER*.

27. В *Model Explorer* проверьте *phone_number*, который был повторно вставлен в модель после прежде удаленного атрибута.

28. Сохраните и закройте модель перед выходом.

Контрольные вопросы

1. Как произвести слияние и расщепление моделей?
2. Каким образом можно скопировать работу?
3. Какое надо задать имя новой модели при расщеплении?
4. Что означает опция *Cut/Paste entire dictionaries*?
5. Какие условия необходимо выполнить для слияния моделей?

Лабораторная работа №6. Проектные слои

Цель работы: Получить навыки в разделении моделей. Получить навыки в задании объектов физической памяти.

Теоретические сведения

Работа с проектными слоями.

Как правило, в проектной иерархии слоя другие типы модели используются для четкой цели проектирования жизненного цикла. К примеру, логическая модель может представлять деловые требования и правила. Из этой модели может быть создана общая физическая модель, в которой физические конструкции разработаны для общей базы данных. Как только общая физическая модель будет считаться прочной, можно будет отойти от этой многочисленной базы данных специфических моделей. Общая физическая модель становится стандартной моделью.

Ряд характеристик **ERwin** необходимо поддерживать для разделения типов модели и для сохранения связанных моделей соединениями (*linked*) и в синхронизации (*sync*).

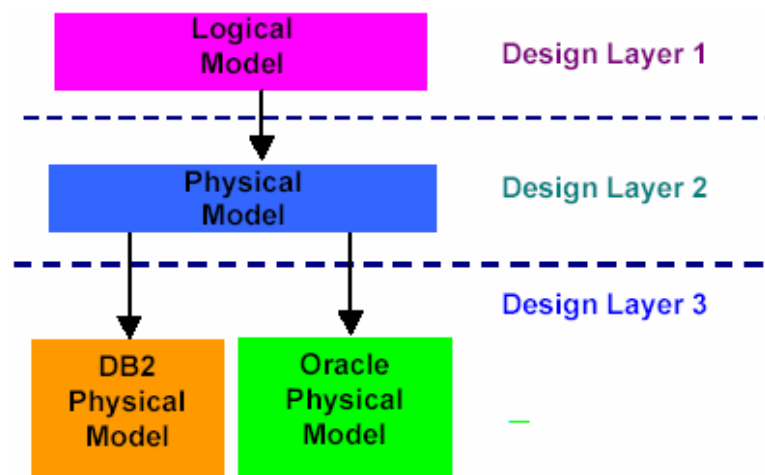


Рис. 3.56. Проектные слои

Обычно невыполнимо использование одной и той же модели для размещения всех фаз проектного процесса. Взамен можно разработать и соединить связанные модели в другие проектные слои. В пределах каждого проектного слоя можно сделать и записать проектные решения, которые превращают структуру от одного слоя до другого. Можно поддержать связи между моделями в других проектных слоях и для синхронизации сделанных изменений в других слоях поддержать подходящие структуры в каждом слое. В **ERwin** комбинация связи моделей в других проектных слоях через источники модели и применения превращений в пределах модели обеспечивать эту возможность.

- **Первый проектный слой:** концептуальная логическая модель данных (*Conceptual Logical Data Model*)

ERwin предоставляет наилучший метод представления структур баз данных, облегчает создание логической (*Logical*) и физической (*Physical*) баз данных. Этот структурный, систематический метод направлен на информационное управление и прикладную разработку, начиная с концептуальной логической модели, первого из нескольких проектных слоев, для захвата деловых специфических требований (включая общие объекты и структурный *supertype/subtype*).

- **Второй проектный слой:** общая физическая модель

Здесь производится установка структур таблиц и столбцов и общее (главное) присваивание имен представленным деловым приложениям. Но в основной физической модели данных объекты и свойства независимы от базы данных. Другая база данных – специфические модели данных, она может быть производной от общей физической модели данных.

- **Третий проектный слой:** база данных – специфические физические модели (*specific Physical Models*)

Здесь можно создать базу данных – специфический физический проектный слой. Каждое приложение может работать на нескольких платформах баз данных; конечный проектный слой необходим для моделей данных специфической базы данных.

Иерархия склада данных

Склад данных (*warehouse*) требует дополнительных слоев для моделей целого склада и данных. В **ERwin** имеются опции физической модели для мерной (пространственной *dimensional*) нотации (*Model/Model Propertie*, закладка *Notation*, блок *Physical Notation*) и для характеристик, имеющих отношение к перемещению данных (информационные исходные данные, правила преобразования данных); это позволяет оптимизировать модель склада.

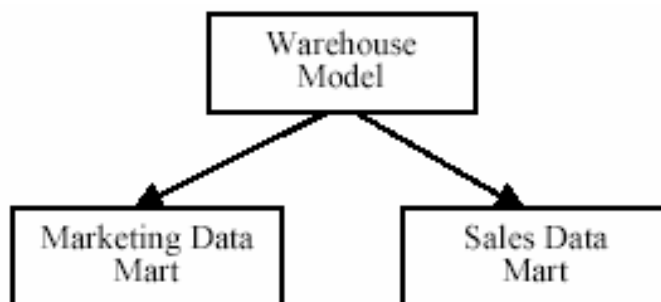


Рис. 3.57. Иерархия склада данных

Разделение модели

Для разделения модели на две модели: *logical-only* и *physical-only* можно использовать опцию *Split Model*, которая находится в меню *Tools*.

После разделения модели будет предложено сохранить по отдельности логические и физические данные модели с другими именами. При сохранении новой модели логическая модель становится источником физической модели, которая необходима для синхронизации изменений между двумя типами моделей. Начальная логико-физическая модель должна быть сохранена под своим изначальным именем.

Производная модель

Объекты не копируются от одной модели другой или начинается с набросков, *Derive New Model wizard* проходит постепенно для создания новой модели от модели-источника. **ERwin** соединяет начальную модель-источник с новой моделью.

Для получения новой модели служит команда "*Derive New Model*" в меню панели инструментов *Tools*.

Derive Model wizard позволяет определить объекты, которые нужны новой модели от исходной.

Дополнение к источнику модели (Model Source)

Источник модели является родительской моделью, которая связана с другой моделью для синхронизации изменений. **ERwin** автоматически назначает скрытые идентификаторы на объекты в модели-источнике и соединенной модели. **ERwin** контролирует процесс изменения в обеих моделях.

Модели используют объектный идентификатор. Затем можно синхронизировать изменения, даже если объектное имя изменилось.

Иногда построение проектной иерархии слоя требует соединения двух моделей, которые уже существуют, а не создавая новую модель из существующей модели. Например, может быть общая модель для определения в качестве источника модели другой базы данных – специфические модели. В этом случае можно добавить общую модель как источник модели базы данных специфической модели. При добавлении источника модели определяются объекты и свойства, чтобы модель-источник содействовала целевой модели.

Для добавления источника модели следует выбрать опцию *Add Model Source* в меню *Tools*.

Add Model Source wizard использует пошаговый метод, чтобы помочь в определении объектов, которые хотите добавить к целевой модели. **ERwin** добавляет объекты к целевой модели и связывает объекты для возможности синхронизации любых изменений позже.

Можно добавить многочисленные источники модели для внесения объектов многочисленным моделям. После добавления источника модели можно использовать диалог "*Model Sources Properties*" для просмотра и редактирования информации об источнике модели.

ERwin позволяет создавать объекты физической памяти, например – базы данных, табличные пространства и сегменты, и задавать параметры для этих объектов в **ERwin**. Можно определять и изменять важные пара-

метры физической базы данных из пользовательской среды построения модели данных.

При обратном проектировании базы данных информация о физической памяти автоматически импортируется в **ERwin**, поэтому можно использовать существующее распределение памяти для каждой таблицы или изменить параметры по мере развития модели данных. Закончив обратное проектирование модели данных, можно сгенерировать физическую схему, которая будет включать новую информацию о физических параметрах, несколькими щелчками мыши.

Поддержка объектов физической памяти в различных СУБД

ERwin поддерживает объекты физической памяти для "DB2" (*Tablespace, Database, Storage Group*), "Informix" (*DBSpace*), "Ingres" (*Location*), "ORACLE" (*Tablespace, Rollback Segments, Databases, Sequences*), "Red Brick" (*Segment*), "SQL Server" (*Filegroup*), "SYBASE" (*Segment*), "Teradata" (*Segment*), "WATCOM/SQL Anywhere" (*DBSpace*). Прямое проектирование (генерация) возможно только для объектов "ORACLE".

Типы объектов физической памяти

Чтобы создать или изменить базу данных, надо иметь привилегию системного администратора для работы в базе данных.

- *Tablespace* (табличное пространство): именованный сегмент в базе данных, состоящий из одного или более файлов данных. Создав табличное пространство, можно затем использовать его для хранения таблиц, индексов или сегментов отката.
- *Rollback Segment* (сегмент отката): зарезервированный объем пространства, который используется для хранения "снимка" данных в том виде, в котором они находились до выполнения транзакции. Если транзакция не завершилась, все изменения данных откатываются и образ данных, хранящийся в сегменте отката,
- *Segment* (сегмент): Именованный набор из одного или нескольких устройств, зарезервированный для использования конкретной базой данных *SQL*. Создав сегмент, можно использовать его для хранения объектов базы данных, например таблиц и индексов.

Задание параметров физической памяти для сущностей ERwin

Когда производится обратное проектирование базы данных, **ERwin** импортирует все имеющиеся объекты физической памяти (поддерживаемого типа) как часть модели данных. Можно просмотреть объект памяти, импортированный в **ERwin** путем обратного проектирования, и изменить его параметры таким же способом, как просматривается или изменяется объект физической памяти, созданный в **ERwin**.

Когда производится прямое проектирование базы данных, каждая сущность **ERwin** создает отдельную таблицу в физической базе данных. При генерации схему базы данных, **ERwin** может создать объекты физической памяти как часть схемы, а затем сгенерировать таблицы и индексы

физической базы данных в конкретных объектах памяти, которые задаются в **ERwin**.

ERwin позволяет контролировать место хранения таблиц базы данных, которые он генерирует. Используя возможность **ERwin** определять новые объекты памяти и импортировать существующие объекты путем обратного проектирования, задавать параметры для этих объектов, а затем связывать сгенерированные таблицы базы данных с конкретными объектами памяти, можно лучше контролировать использование памяти на сервере, и при необходимости можно отрегулировать работу базы данных, изменяя параметры, связанные с конкретным объектом памяти.

Связывание сущностей **ERwin** с объектами физической памяти

Редактор **ERwin Table Property** позволяет просматривать и изменять различные физические характеристики, включая параметры физической памяти, связанные с таблицами, соответствующими сущностям, для каждой таблицы.

Чтобы просмотреть характеристики физической памяти, связанные с конкретной сущностью **ERwin**, щелкните правой кнопкой мыши по сущности, дайте команду "*Table Properties*" в всплывающем меню, а затем — команду "*Physical Property*". **ERwin** демонстрирует список физических характеристик для текущей сущности.

Чтобы просмотреть физические характеристики для другой сущности, выберите сущность из списка справа вверху в редакторе. Обратите внимание, что **ERwin** помещает имя соответствующей таблицы в текстовое окно в левом верхнем углу окна-диалога. В центре диалога находятся параметры табличного пространства, в котором хранится выбранная таблица.

Редактор *Table Property* позволяет увеличивать или уменьшать пространство, доступное для ввода выбранной сущности, создавать пространство для хранения дополнительных сущностей в этом же физическом объекте, или изменять размер будущих объектов памяти в момент их создания.

Обратное проектирование объектов памяти (*Reverse Engineering*)

Когда производится обратное проектирование базы данных, **ERwin** может импортировать имена и определения объектов физической памяти, определенных на сервере, таким же образом, каким он импортирует физические таблицы, индексы и другую информацию по физической схеме.

Когда импортируется информация об объектах физической памяти из сервера, **ERwin** использует информацию об адресе каждой таблицы базы данных, поэтому впоследствии можно заново создать базу данных, используя то же самое распределение памяти. Не нужно будет заново вручную распределять память под таблицы.

После импортирования объектов физической памяти в **ERwin**, можно просматривать или изменять определения объекта и связи таблиц в редак-

торе *Table Property* физического объекта так же, как при работе с объектами физической памяти, созданными в **ERwin**.

Прямое проектирование объектов физической памяти

Если генерируете физическую схему в *ORACLE*, то Вы можете включить любую базу данных, табличное пространство или сегменты отката, которые Вы определили в **ERwin**, как часть схемы. **ERwin** автоматически транслирует определения физических объектов в команды *CREATE TABLE*, *CREATE TABLESPACE*, *CREATE ROLLBACK SEGMENT* и вставляет информацию о заданных параметрах с соблюдением синтаксиса *SQL*.

При генерации физической схемы **ERwin** сначала создает заданные родительские объекты памяти, затем – дочерние объекты памяти и физические таблицы, расположенный по заданным адресам в памяти.

В процессе генерации схемы **ERwin** выдает на экран сообщение о состоянии каждой таблицы, которую он пытается создать. После того как генерация схемы закончена, **ERwin** выводит на экран сообщение, в котором содержится число успешно созданных таблиц и число неудачно завершившихся попыток создания таблиц.

Примечание: после того как сгенерировали физический объект как часть схемы, можно изменить имя объекта, но нельзя изменить значения параметров, присвоенные этому объекту в **ERwin**. Если нужно изменить параметры физического объекта, то следует создать новый объект и заново связать все таблицы **ERwin** с новым объектом памяти. После этого следует удалить старый объект и произвести повторную генерацию схемы.

Синхронизация физических объектов

После того как произведено обратное проектирование базы данных, если добавляются или изменяются объекты в **ERwin** или создаются, удаляются и изменяются объекты непосредственно на сервере базы данных, то возникнут расхождения между моделью данных и физической схемой. Когда это происходит, **ERwin** позволяет синхронизировать определения для соответствия хранимой в **ERwin** информации с хранящейся на сервере.

Во время процесса синхронизации **ERwin** сравнивает имена логической модели и имена физической схемы, хранящиеся на сервере, и показывает, какие из объектов схемы не синхронизированы. Чтобы добавить объект **ERwin** к определениям, хранящимся в данный момент на сервере, нажмите кнопку "DB Sync" в "<DB> Table" (*Model/Tables*).

Чтобы создать новый объект табличного пространства **ERwin** на сервере, щелкните по строке "THRILLER" в окне "Unsynched **ERwin** Tablespace", а затем - по пунктирной (----) верхней строке в окне "Unsynched <DB> Tablespace". После этого нажмите кнопку "Export", чтобы переслать информацию о табличном пространстве **ERwin** на сервер. После нажатия на кнопку "Execute", **ERwin** создаст табличное пространство на сервере и добавит "THRILLER" в список синхронизированных объектов, который находится в центре окна-диалога синхронизация

Изменение объекта физической памяти

Если изменить значение параметра (параметров) объекта физической памяти в **ERwin**, то новая информация не может быть экспортирована на сервер с использованием диалога *DB Sync*. Вместо этого, для того чтобы изменить информацию, хранящуюся на сервере, удалите старое определение физического объекта, используя для этого режимы *DROP TABLE* и *CREATE TABLE* в редакторе *Schema Generation Report*, а затем заново сгенерируйте объект с новыми значениями параметров.

Обратное проектирование объектов физической памяти

Если Вы выбирали таблицы, чтобы импортировать их в процессе обратного проектирования, **ERwin** создаст новую диаграмму модели данных. Если Вы выбирали только объекты физической памяти, то окно диаграммы будет пустым. **ERwin** выводит на экран объекты памяти, которые он импортировал из сервера СУБД, для физической модели данных.

База данных

Определение: Базой данных называется зарезервированный объем пространства на одном или нескольких устройствах хранения, используемые для хранения данных и определений объектов базы данных, таких как таблицы и индексы. Чтобы создать базу данных в *Oracle*, надо иметь привилегию *DBA* для работы с базой данных.

Таблица 3.3

Параметры базы данных

| Имя параметра | Что определяет параметр | Значение по умолчанию | Диапазон |
|---------------------------|---|-----------------------|-------------------------|
| <i>ARCHIVE LOG</i> | Состояние автоматического архивирования. Поставьте метку в этом окне, чтобы включить автоматическое архивирование информации <i>log</i> , используемой при восстановлении. Оставьте окно пустым, если Вы не хотите использовать автоматическое архивирование. | <i>Off</i> | <i>On</i> <i>Off</i> |
| <i>CHARACTER SET</i> | Набор символов, используемый базой данных. Все данные в колонках типов <i>CHAR</i> , <i>VARCHAR2</i> , <i>LONG</i> хранятся в заданном наборе символов. После того как база данных создана, набор символов не может быть изменен. | | |
| <i>CONTROL FILE REUSE</i> | Статус повторного использования управляющего файла. Поставьте метку в этом окне, чтобы позволить <i>Oracle</i> переписать информацию в управляющих файлах, определенных в параметре <i>INIT. ORA CONTROL_FILES</i> . Оставьте окно пустым, если Вы не хотите, чтобы <i>Oracle</i> повторно использовал управляющие файлы. | <i>Off</i> | <i>On</i> <i>Off</i> |

| Имя параметра | Что определяет параметр | Значение по умолчанию | Диапазон |
|----------------------|--|--------------------------|--------------------------------------|
| <i>DATAFILE</i> | Имена всех файлов данных в БД. Просмотрите список, чтобы увидеть, какие файлы данных есть в базе. | | |
| <i>EXCLUSIVE</i> | Статус совместного использования данных. Поставьте метку в этом окне, чтобы указать, что в любой момент времени только один экземпляр может получить доступ к базе данных. Оставьте окно пустым, чтобы разрешить одно-временный доступ нескольких пользователей. | <i>Off</i> | <i>On</i> <i>Off</i> |
| <i>LOGFILE</i> | Имена всех <i>log</i> файлов в базе данных. | | |
| <i>MAXLOGFILES</i> | Максимальное число <i>log</i> групп, которые можно создать для базы данных. | | 2-56 |
| <i>MACLOGMEMBERS</i> | Максимальное число членов в каждой <i>log</i> группе (поддерживается <i>Oracle7</i> и более поздними версиями). | | |
| <i>MAXLOGHISTORY</i> | Объем памяти, который должен быть зарезервирован в управляющем файле для имен групп архивных <i>log</i> файлов транзакций (поддерживается <i>Oracle7</i> и более поздними версиями). | | |
| <i>MAXDATAFILES</i> | Максимальное число полей данных, которое можно назначить для базы данных. | зависит от конкретной ОС | 1 верхний предел зависит от ОС |
| <i>MAXINSTANCES</i> | Максимальное число экземпляров, для которых одновременно может быть установлена база данных. | 3 | 1-255 |

Табличное пространство

Определение: табличным пространством называется именованный сегмент базы данных, состоящий из одного или более файлов данных. После создания табличного пространства можно использовать его для хранения таблиц, индексов или сегментов отката. Чтобы создать табличное пространство, надо иметь привилегию *DBA* для работы с базой данных.

Параметры табличного пространства

| Имя параметра | Что определяет параметр | Значение по умолчанию | Диапазон |
|------------------------|---|-----------------------|---|
| <i>DATAFILE</i> | Имена всех файлов данных в табличном пространстве. | | |
| <i>FREELISTS</i> | Число списков, управляемых Oracle и определяющих, какие блоки данных располагают доступным пространством для вставки новых строк. Увеличив это значение, можно повысить скорость работы, если приложение требует выполнения множества команд <i>INSERT</i> в параллельном режиме. | | |
| <i>FREELIST GROUPS</i> | Максимальное число групп <i>FREELIST</i> , которое можно связать с таблицей. | | |
| <i>INITIAL</i> | Размер начального экстента в байтах. | 5 блоков данных | 2 блока данных - верхний предел зависит от ОС |
| <i>MAXEXTENTS</i> | Максимальное число экстентов, которое можно связать с таблицей, индексом или кластером табличного пространства. | Зависит от ОС | 1 - верхний предел зависит от ОС |
| <i>MINEXTENTS</i> | Минимальное число экстентов, которое автоматически распределяется при создании таблицы, индекса или кластера табличного пространства. | 1 | 1 - верхний предел зависит от ОС |
| <i>NEXT</i> | Размер следующего экстента в байтах. | 5 блоков данных | 1 блок данных - верхний предел зависит от ОС |
| <i>OFFLINE</i> | Состояние доступности таблицы. Поставьте метку в этом окне, чтобы перевести табличное пространство в <i>offline</i> . Оставьте окно пустым, если хотите работать в <i>online</i> . | <i>Off</i> | <i>On</i> <i>Off</i> |
| <i>OPTIMAL</i> | Оптимальный размер каждого экстента в байтах. | | |
| <i>PCTINCREASE</i> | На сколько процентов этот экстент может быть больше предыдущего по размеру. | 50% | 0% - верхний предел зависит от ОС |
| <i>TABLESPACE</i> | Имена всех табличных пространств в базе данных. | | |

Сегмент отката

Определение: Сегмент отката – зарезервированный объем пространства в табличном пространстве, используемый для хранения "снимка" состояния данных до выполнения транзакции. Если транзакция не будет завершена, все изменения данных откатываются, и восстанавливается образ данных, хранящийся в сегменте отката. Чтобы создать или изменить сегмент отката, надо иметь привилегию *CREATE ROLLBACK SEGMENT* для работы с табличным пространством.

Таблица 3.5

Параметры сегмента отката

| Имя параметра | Что определяет параметр | Значение по умолчанию | Диапазон |
|-------------------------|---|-----------------------|---|
| 1 | 2 | 3 | 4 |
| <i>PUBLIC</i> | Состояние доступности сегмента отката. Поставьте метку в этом окне, чтобы сделать сегмент отката доступным для каждого экземпляра. Оставьте окно пустым, если Вы хотите сделать сегмент отката доступным для одного конкретного экземпляра. | | |
| <i>FREELISTS</i> | Число списков, управляемых Oracle и определяющих, какие блоки данных располагают доступным пространством для вставки новых строк. Увеличив это значение, можно повысить скорость работы, если приложение требует выполнения множества команд <i>INSERT</i> в параллельном режиме. | | |
| <i>FREELIST GROUPS</i> | Максимальное число групп <i>FREELIST</i> , которое можно связать с сегментом отката. | | |
| <i>INITIAL</i> | Размер начального экстенда в байтах. | 5 блоков данных | 2 блока данных - верхний предел зависит от ОС |
| <i>MAXEXTENTS</i> | Максимальное число экстендов, которое можно связать с сегментом отката. | Зависит от ОС | Зависит от ОС |
| <i>MINEXTENTS</i> | Минимальное число экстендов, которое автоматически распределяется при создании сегмента отката. | 1 | 1 |
| <i>NEXT</i> | Размер следующего экстенда в байтах. | 5 блоков данных | 1 блок данных - верхний предел зависит от ОС |
| <i>ROLLBACK SEGMENT</i> | Имена файлов всех сегментов отката в базе данных. Прокручивая список, выберите сегмент, который Вы хотите посмотреть или изменить. | | |

| Имя параметра | Что определяет параметр | Значение по умолчанию | Диапазон |
|--------------------|--|-----------------------|------------------------------------|
| <i>OPTIMAL</i> | Оптимальное число управляемых экстен-тов для каждого сегмента отката. Oracle автоматически восстанавливает <i>OPTIMAL</i> размер сегментов отката после успешного завершения транзакции. | | |
| <i>PCTINCREASE</i> | На сколько процентов этот экстен-т может быть больше предыдущего по размеру. | 50% | 0% - верхний предел зави-сит от ОС |
| <i>TABLESPACE</i> | Имена всех табличных пространств в базе данных. Прокручивая список, выберите табличное пространство, содержащее сег-мент отката. | | |

Сегмент

Определение: Сегмент – именованное множество из одного или более устройств, зарезервированное для использования конкретной базой данных *SQL Server*. После создания сегмента можно использовать его для хранения объектов базы данных, таких как таблицы и индексы. Чтобы создать сегмент, надо иметь полномочия *SA* (системного администратора) для работы с базой данных.

Примечание: Нельзя реально создать сегмент в *ERwin*. Чтобы создать сегмент в *SQL Server*, следует использовать хранимую процедуру *SQL Server sp_addsegment*. После того как сегмент создан в *SQL Server*, можно связать с ним сущности *ERwin* в редакторе *ERwin Table Property*.

Порядок выполнения работы

1. Откройте модель "*lab_5_1*", сохраненную в пятой лабораторной работе. Из панели инструментов меню "*Tools*" выберите "*Split L/P Model*" (вид – *logical*).

После завершения процесса разделения нужно сохранить сначала логическую модель, а затем физическую модель. Сохраните логическую модель как "*lab_6(log).er1*". Сохраните физическую модель как "*lab_6(phys).er1*".

2. Сохраните и закройте все модели.

3. Для создания новой модели *ERwin* на основе существующей модели *ERwin* и перехода из одного проектного слоя к другому следует воспользоваться *Derive Model Wizard*.

4. Откройте модель "*lab_6(phys).er1*". В меню "*Tools*" выделите "*Derive New Model*".

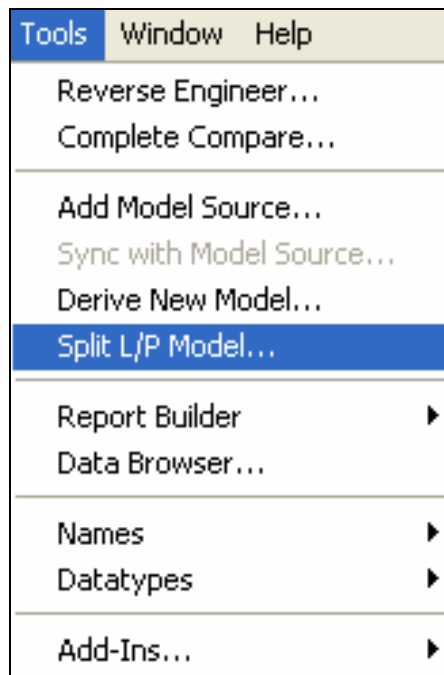


Рис. 3.58. Разделение модели

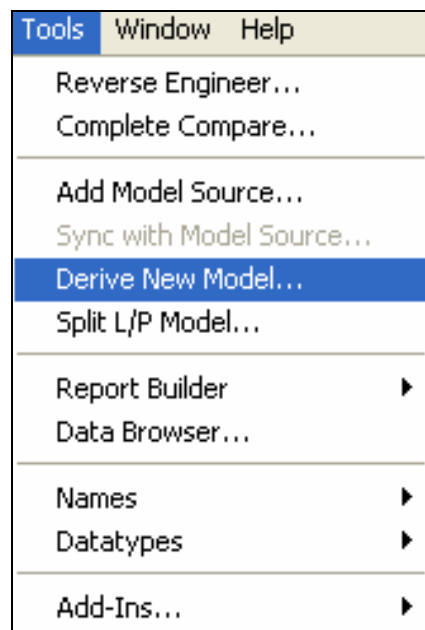


Рис. 3.59. Создание новой модели **ERwin** на основе существующей

5. На первой странице *Derive New Model* в качестве типа модели группового блока выберите "*Physical*". Для создания временного шаблона воспользуйтесь шаблоном "*Blank Physical Model*", который используется по умолчанию. В списке специальной базы данных выберите "*SQL Server*". Затем щелкните "*Next*".

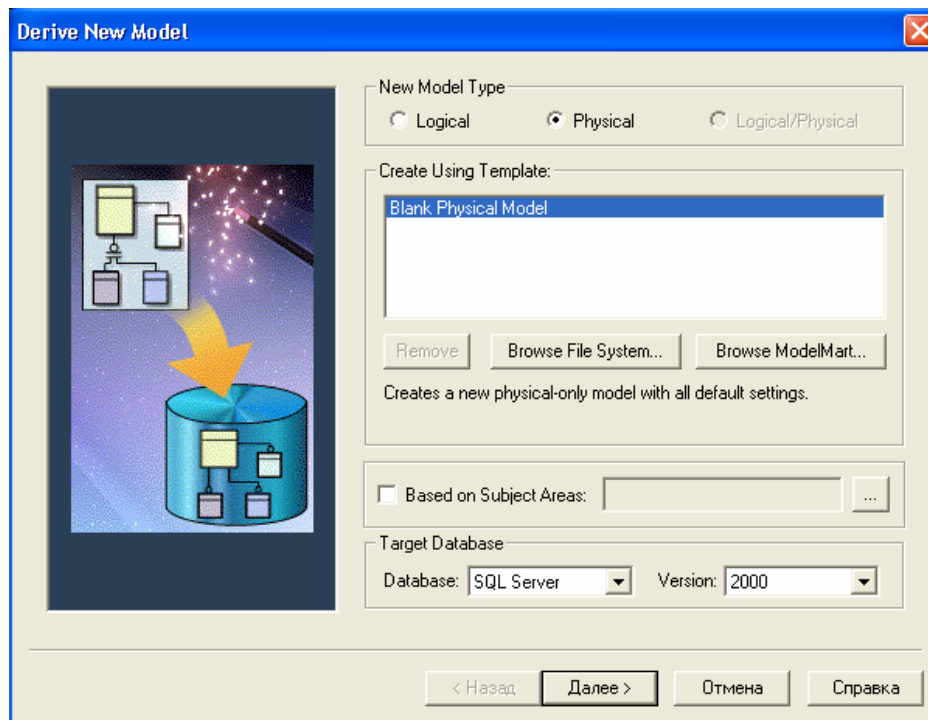


Рис. 3.60. Диалог создания модели

6. На следующей странице примите параметры установки по умолчанию (там выбраны все объекты: домены, правило проверки введенных значений, значения по умолчанию, свойства пользователя, хранимые процедуры, шаблоны скриптов, таблицы, группы файлов, представления).

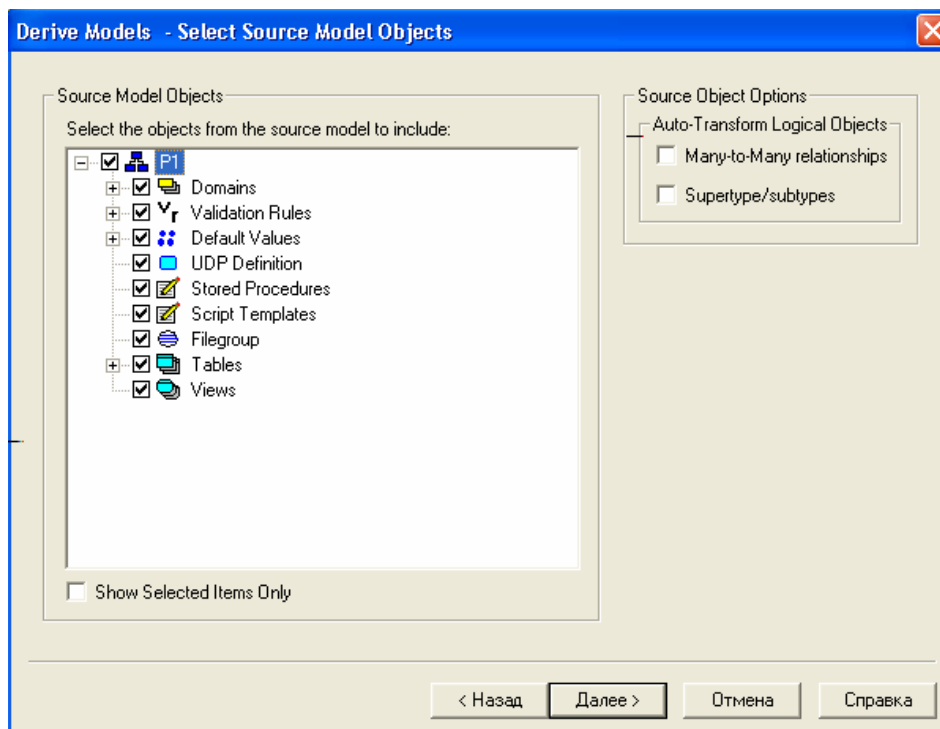


Рис. 3.61. Выбор объектов для создания новой модели

7. Далее, на следующей странице примите установки, заданные по умолчанию, и щелкните "Готово".

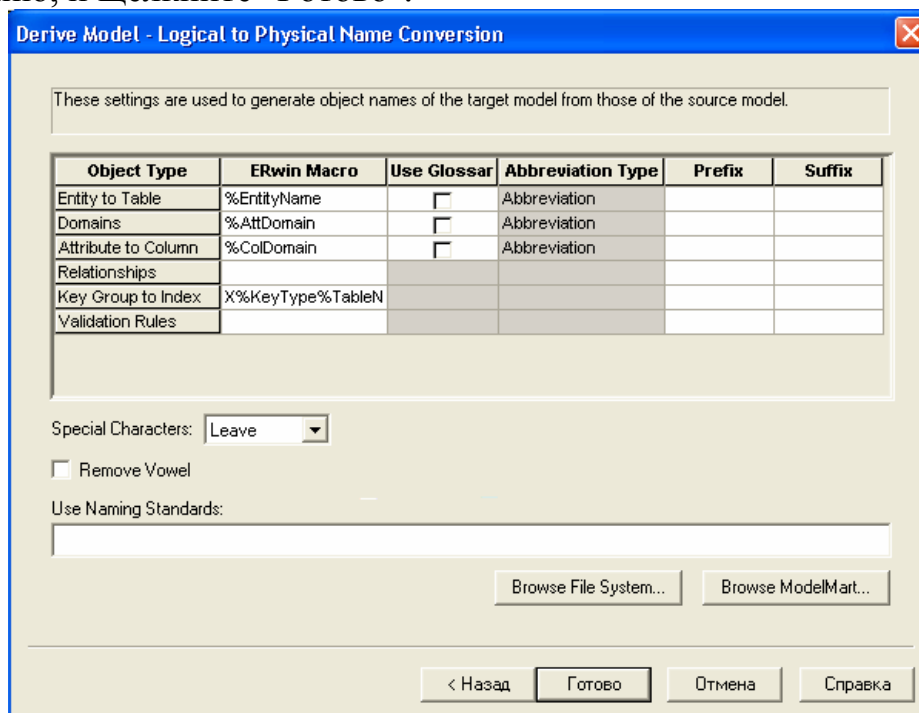


Рис. 3.62. Диалог задания установок для создания модели

Создана новая физическая модель.

8. Сохраните ее как "lab_6_1.er1", закройте "lab_6(log).er1" и "lab_6(phys).er1" модели.

9. Откройте модель "lab_5_2", сохраненную в пятой лабораторной работе.

10. Связывание сущности **ERwin** с объектом физической памяти.

10.1. Щелкните по сущности правой кнопкой мыши, дайте команду "Table Property", а затем – команду "Physical Property". **ERwin** откроет редактор *Table Property* и продемонстрирует на экране список физических характеристик для выбранной сущности.

10.2. Создайте новый *FileGroup*: *My_FileGroup*.

10.3. Нажмите кнопку "OK" для выхода из редактора и сохранения связанного с сущностью физического объекта и параметров памяти

11. Изменение значения параметров памяти для сущности:

11.1. Измените тип рассматриваемой базы данных на *Oracle*.

11.2. Щелкните по сущности правой кнопкой мыши, дайте команду "Table Property", а затем – команду "Physical Property". **ERwin** откроет редактор *Table Property* и продемонстрирует на экране список параметров для объекта физической памяти выбранной сущности.

11.3. Для изменения значений параметров щелкните по нужному окну или кнопке и выделите текущее значение. Введите новое значение.

11.4. Нажмите кнопку "OK" для выхода из редактора и сохранения новых значений параметров.

Примечание: если нужно связать с объектом памяти несколько сущностей, или изменить параметры для нескольких объектов, можно выбрать другую сущность из списка в правом верхнем углу редактора. При открытии списка "*Entity*" выберете другую сущность, **ERwin** сразу же сохранит изменения, сделанные в редакторе *Table Property*.

Контрольные вопросы

1. Какие бывают виды проектных слоев?
2. Каким образом можно разбить модель?
3. Что выполняет команда *Derive New Model*?
4. Что такое объекты физической памяти?
5. Что такое сегмент отката?

Лабораторная работа №7. Отчеты и сообщения

Цель работы: Получить навыки в создании шаблона сообщений, работы с отчетами в *ERwin*.

Теоретические сведения

Создание сообщений о моделях в *ERwin*

Report Template Builder создает встроенные шаблоны сообщения в *HTML*, *RTF* и *Text format*, что может быть использовано повторно для сообщения о любой модели. Например, если будет выбран формат *RTF*, то выход из сообщения автоматически ведет к текстовой обработке локальным программным обеспечением. Если будет выбран формат *HTML*, то выход из сообщения автоматически запускает локально включенный *web browser*. Если выбирается *Text format*, то выход из сообщения автоматически запустит *Microsoft Excel*.

Создание отчетов

При проектировании модели данных может потребоваться представить информацию из графической диаграммы в виде отчета в текстовом формате. Можно просматривать, сортировать и изменять информацию до создания отчета.

В редакторе отчетов "*Report*" можно выбрать нужные режимы для содержания и формата. *ERwin* позволяет создать отчет по всей диаграмме (используя главную область), по части диаграммы (используя любую другую область) или по части области (выбирая в области одну или несколько сущностей). В процессе проектирования отчета можно просмотреть его непосредственно в окне "*Data Browser*". Можно сохранить определение отчета, которое указывает на режимы, которые были выбраны при задании содержания и формата отчета, и использовать его, чтобы генерировать этот же отчет в дальнейшем.

"*Data Browser*" является разносторонним и хорошо настраиваемым средством для просмотра и генерации сообщений о диаграммах *ERwin* и информации о "*ModelMart*".


Если используете *ERwin* без соединения с "*ModelMart*", то "*Data Browser*" обеспечивает установку встроенных типов отчетов для диаграмм *ERwin*. Каждый тип отчета содержит установку связанных опций, которые можно включить или исключить в отчете. Некоторые типы отчетов обеспечивают встроенный фильтр и сортировку опций. Выбранная категория отчета используется как база для создания отчета, для включения нужных опций. После создания отчет сохраняется, так что открыв модель, можно просмотреть отчет снова.

"*Data Browser*" показывает результат сгенерированного отчета и добавляет результат в управляющее дерево с иконкой (📄) под иконкой сообщения. Можно модифицировать содержимое и изменить вид результата.




Можно осуществлять поиск нужной информации в результате. Можно определить выражение поиска, которое может включить строки, числа, или даты, для одного или более столбцов, чтобы "Data Browser" находил результат нужной колонки для удовлетворения всем выражениям поиска. Если **ERwin** используется со связью с "ModelMart", "Data Browser" обеспечивает два комплекта встроенных сообщений организованных в дереве папки под двумя папками с названием "General" и "ModelMart <Name>".


Можно создать отчет для столбцов (*Column reports*), для атрибутов (*Attribute reports*), для сущностей (*Entity reports*), для доменов (*Domain reports*), для таблиц (*Table reports*), для предметной области (*Subject area reports*), для правил проверки введенных значений (*Model validation reports*), для хранимых процедур (*Stored procedure reports*), для связей (*Relationship reports*), для просмотра (*View reports*).

Алгоритм создания отчета

1. Нажмите  на панели инструментов для открытия "Data Browser" (можно открыть из меню "File", выбрав "New **ERwin** Report").
2. Откройте "Report". Введите имя в блок "Name". Выберите категорию в списке "Category".
3. Если отчет составляется для логической модели, то выберите "Logical", если отчет составляется для физической модели, то выберите "Physical".
4. В закладке "Options" установите нужные опции.
5. Для задания описания отчета в закладке "Definition" введите описание.
6. В закладке "Note" введите примечание. Нажмите "OK".


Алгоритм редактирования отчета


1. Войдите в "Data Browser". Выделите область () для запуска "Data Browser".
2. Выберите сообщение, которое хотите отредактировать. Нажмите  для открытия **ERwin** Report Editor.
3. Выберите информацию, которую нужно включить в отчет. Иконка () около опции указывает, что соответствующий столбец в установленном сгенерированном результате может быть редактируемым.
4. Щелчок "Filter By", чтобы рассматривать фильтрующие опции. Нажмите одну или более кнопок фильтра, чтобы включить только нужные колонки в отчет.
5. Нажмите "Sort By" для сортировки колонок отчета.
6. Нажмите "Definition" и отредактируйте определение для отчета.
7. Нажмите "Note" и отредактируйте примечание для отчета.
8. Нажмите "OK", чтобы корректировать сообщение.


9. Нажмите , чтобы открыть сообщение напротив активной диаграммы **ERwin**.

Алгоритм генерации отчета

1. Зайдите в "Data Browser". Выделите область () для последующей генерации отчета **ERwin**.

2. Двойным щелчком по иконке отчета () сгенерируйте отчет. **ERwin** отобразит результат в правой области окна "Data Browser" и добавит результат под отчетом в контрольном дереве.

3. Двойным щелчком отредактируйте нужную ячейку. ячейка редактируется под колонкой с . Отредактируйте текст в ячейке.

4. Нажмите  в "Data Browser" для сохранения изменений в диаграмме с расширением ".er1".

5. При редактировании **ERwin** сразу же вносит изменения одновременно в "Browser" и в диаграмму.

Сортировка информации в ERwin Browser.

По умолчанию **ERwin** сортирует информацию, содержащуюся в *Attribute Browser*, в следующем порядке:

- Имя сущности
- Базовое имя
- Имя роли
- Статус

Чтобы отсортировать данные в другом порядке, нажмите правую кнопку мыши по названию колонки отчета и выберите сортировку по возрастанию (*Sort ascending*) или по убыванию (*Sort descending*). **ERwin** сразу же вносит в диаграмму все изменения, которые производятся в окне браузера.

Порядок задания режимов определяет порядок появления в отчете заголовков.

Если отчет нужно получить один раз, то поставьте метки в окнах нужных режимов и нажмите кнопку "Print". Если нужно сохранить выбранные режимы в качестве спецификации, которую можно было бы потом еще использовать для повторной генерации этого же отчета, введите новое имя определения отчета в текстовое окно "Report" в верхней части окна-диалога и нажмите кнопку "New" для сохранения определения отчета на диске, как части текущей диаграммы.

Можно изменить имя отчета и (или) задать новые или другие режимы.

Чтобы удалить определение отчета, выберите определение отчета, которое нужно удалить, из списка "Report" и нажмите кнопку "Delete". **ERwin** сразу же удаляет определение отчета. Файлы отчетов, созданные с использованием удаленного определения отчета, сохраняются.

Кнопка "Preview" служит для просмотра содержимого отчета.

Сохранение файла отчета

Отчет в **ERwin** состоит из двух элементов: определение отчета, в котором заданы режимы содержания и форматирования отчета, и выходной файл отчета, содержащий реальные данные, сгенерированные определением отчета. Определение отчета можно сохранить, как часть текущей диаграммы (подобно области). Выходной файл можно сохранить в отдельном текстовом файле в формате *ASCII*, для этого нужно нажать кнопку "*Report*" в редакторе *Report*, а затем задать имя файла и директорию в окне-диалогее "*Save As*".

Можно записать выходной файл отчета на диск, используя для этого кнопку "*Report*" в редакторе "*Report*" или в окне "*Preview*". Когда при записи отчета на диск выходной файл отчета, или данные, сохраняются в формате *ASCII*. **ERwin** сохраняет данные отчета в отдельном файле. Можно открыть отчет **ERwin** из "*Microsoft Word*", "*WordPerfect*", "*Excel*" или любого другого приложения, занимающегося обработкой текстов или таблиц, которое может читать файлы *ASCII*.

Чтобы изменить определение отчета следует открыть отчет, а потом произвести необходимую корректировку.

Использование отчетов **ERwin** с другими приложениями

Если задается режим "*DDE Table*" в редакторе *Report*, то можно экспортировать выходной отчет **ERwin** в любое приложение обработки текстов или таблиц, например, в "*Word for Windows*", "*WordPerfect for Windows*", "*AmiPro*" и "*Microsoft Excel*".

Если установлен этот режим, приложение автоматически строит таблицу и вставляет данные отчета **ERwin**, структурированные в виде таблицы. Когда открывается этот отчет из приложения, то можно использовать режимы форматирования, чтобы изменить внешний вид отчета **ERwin**.

Алгоритм пересылки отчета **ERwin** в приложение *DDE server*

1. Задайте режимы содержания и форматирования отчета в редакторе *Report* для создания отчета.

2. Задайте режим "*DDE Table*" в групповом окне *Report Format*, а также предпочтительный для Вас режим *multi-value*. После этого нажмите на кнопку "*Report*". **ERwin** выводит на экран список доступных серверов *DDE*.

3. Выберите приложение.

- Если приложение уже открыто, Вы можете выбрать, будете ли Вы вставлять Ваш отчет в новый документ или в уже существующий.
- Если Вы выбрали уже существующий документ, **ERwin** вставляет отчет в этот документ, после курсора. Если приложение не было открыто, **ERwin** запускает приложение, открывает новый документ и вставляет отчет с первой строки окна документа.

Режимы форматирования

Редактор *Report* поддерживает несколько режимов форматирования, позволяющих Вам указать, каким образом **ERwin** форматирует отчет, когда одно значение может быть связано с несколькими значениями в другой колонке отчета.

При редактировании **ERwin** сразу же вносит изменения одновременно в *Browser* и в диаграмму.

Порядок выполнения работы

Создание шаблона сообщений

1. Откройте модель "*lab_6(log)* ", сохраненную в шестой лабораторной работе.

2. Перейдите в *Tools*, выберите *Report Builder*. До сохранения первого шаблона в папке сообщений (*Report*) можно увидеть сообщение, указывающее, что папка пустая, или можно увидеть сообщение, указывающее, что не нужно щелкать кнопку *Browser*, и выбирается папка, в которой нужно будет сохранить ваши сообщения.

3. Создайте новый шаблон. Для этого нажмите "*New*".

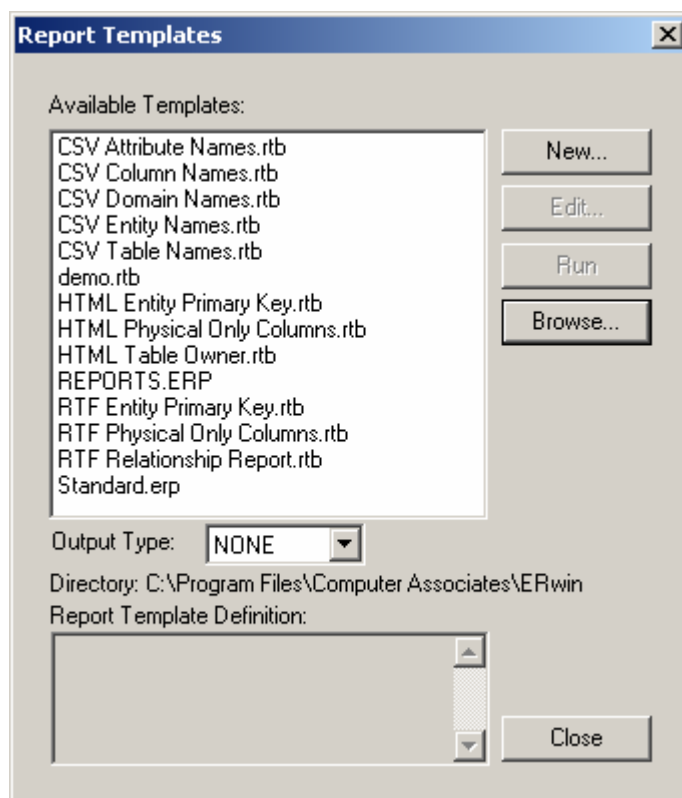


Рис. 3.63. Создание шаблона

4. В *Report Template Builder* в *Report Layout pane* сделайте двойной щелчок по "*Document Untitled*".

| |
|----------------------------|
| Report Layout: |
| Document Untitled |
| Document Properties |
| Has Table Of Contents |
| Export As |
| HTML |

Рис. 3.64. Создание шаблона отчетов

Примите по умолчанию "*Property Tree*". Выберите закладку "*Title*". Оставьте пространство в конце названия перед выполнением следующего шага.

5. Добавьте имя модели в названии сообщения. Для этого нажмите по кнопке *Add Macro*.

6. Нажмите *Export* и проверьте, что *HTML* – выходной тип сообщения. Если нет, то выберите *HTML* из списка "*Export AS*". Затем в *HTML Export Properties* сгруппируйте блоки. Выберите *Picture Reports as Pop-Up Windows* в качестве управляющего окна.

7. Закройте окно и вернитесь в подокно *Report Template Builder*.

8. Добавление частей сообщения к шаблону сообщения:

8.1. В *Report Template Builder* (конструктор шаблона сообщений) в левом подокне *Available Sections* выберите *Picture* и нажмите по стрелке, находящейся справа, чтобы добавлять этот раздел к подокну *Report Layout pane*.

8.2. В подокне *Available Sections* выберите *Entity* (в *Logical Section*) (в логическом разделе), нажмите по стрелке справа для добавки этого раздела к подокну *Report Layout*.

8.3. В подокне *Report Layout* с помощью двойного щелчка по сущности (*%Model*) откройте *Properties dialog* (диалог свойств) и определите детали сообщения в этом разделе.

8.4. Нажмите по плюсу, чтобы расширить сущность (*Entity*), затем выберите имя (*Name*) и определение (*Definition*).

8.5. Нажмите по плюсу, чтобы расширить *Attribute*, выберите имя (*Name*) и закройте окно для возврата в *Report Template Builder*.

8.6. Выберите *File/Save As* и, введя "*MyReport*", как имя шаблона, сохраните этот шаблон сообщения.

Запуск HTML-сообщения:

9. Нажмите кнопку *Run* в toolbar . Выберите ваш шаблон и нажмите "*Run*", *Report Template Builder* запускает ваше окно просмотра (*web-browser*) с отображаемым сообщением.

10. В левом *frame* окна просмотра расположите связи в *Report Components* (компоненты сообщения).

11. При щелчке связываются в *Picture* и разделы сущностей (*Entity*) для просмотра каждого раздела.

12. В разделе сущности (*Entity*) измените представление табличное (*Tabular*) на иерархическое (*Hierarchical*).

13. Сообщение, создаваемое в *HTML*, основано на модели "*My ERwin Model.er1*", и выглядит похожим на это:

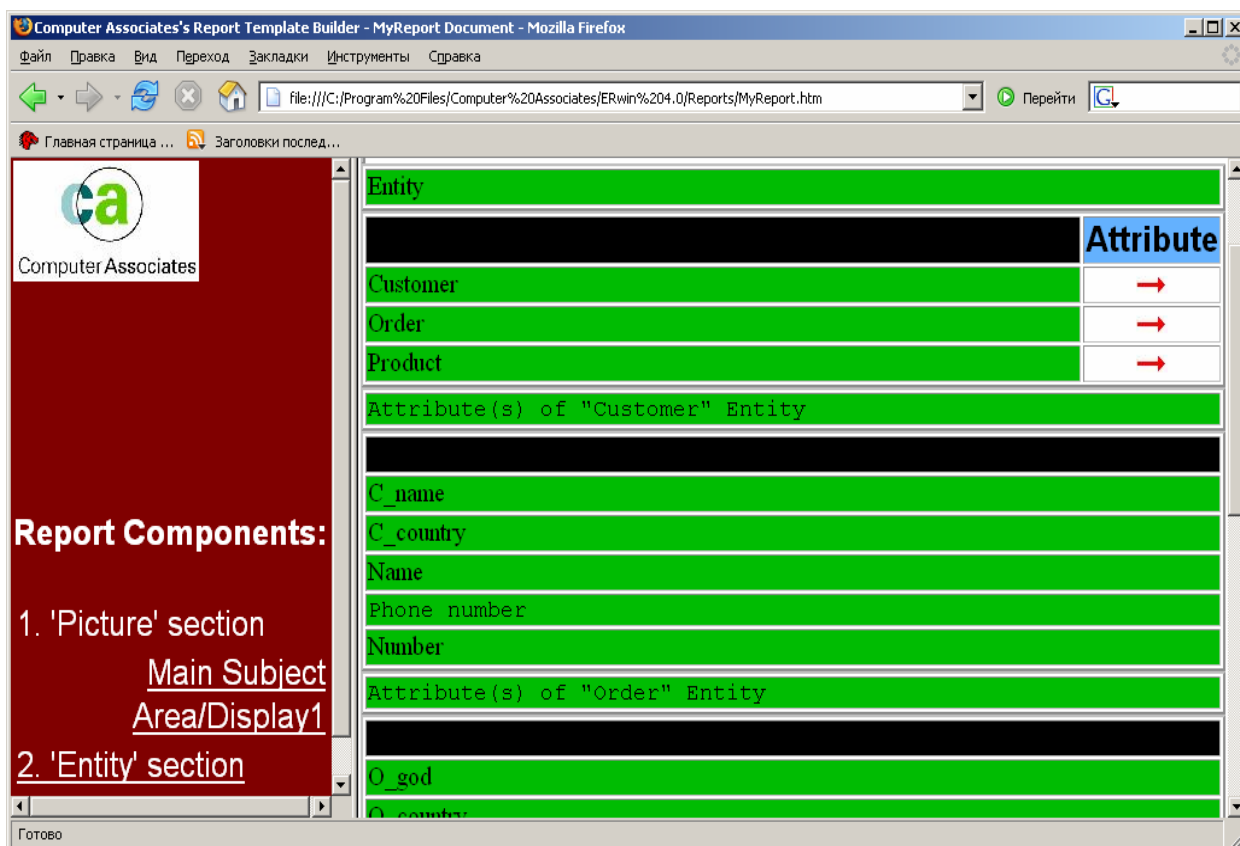


Рис 3.65. Сообщение, основанное на модели

14. Закройте окно просмотра (*web browser*).

Применение шаблона для другой модели:

15. Откройте "*lab_3_1.er1*".

16. Нажмите кнопку *Report Template Builder* в toolbar .

17. В диалоге *Report Templates* из списка *Available Templates* выбираем "*MyReport*".

18. Из списка *Output Type* выберите *TXT* (текстовый документ).


19. Нажмите *Run*, чтобы запустить ваш локальный текстовый процессор и отобразить сообщение в окне документа.

20. После просмотра сообщения закройте ваш текстовый процессор, а затем закройте "*lab_3_1.er1*".

21. Закройте модель.


Создание отчета

22. Откройте модель "lab_6(log)", сохраненную в шестой лабораторной работе. Составим отчет для физической модели.

23. Нажмите  в *Standard toolbar* для открытия *Data Browser File/New Erwin Report*.

24. Откройте "Report". Введите имя "Report Attribute" в блок *Name*.

25. Выберите категорию "Attribute" в списке *Category*. Выбор категорий отчета зависит от типа: *logical* или *physical*.

26. В закладке *Options* установите опции в поле *name* и в поля, где есть иконка .

27. Для задания описания отчета в закладке *Definition* введите описание.


28. В закладке *Note* введите примечание. Нажмите *OK*.

29. Аналогичным образом создайте отчеты для сущностей и физических (*physical*) категорий: столбцов, таблиц, правил проверки введенных значений с именами "Report Entity", "Report Column", "Report Table", "Report Valid" соответственно. Сохраните все отчеты.

Редактирование отчета

30. Выберите отчет, который нужно отредактировать.

31. Нажмите  для открытия *ERwin Report Editor*.


32. Выберите информацию, которую нужно включить в отчет. Иконка  около опции указывает, что соответствующий столбец в установленном сгенерированном результате может быть редактируемым.

33. Нажмите *Sort By* для просмотра сортирующих опций.



34. Нажмите *Definition* и отредактируйте определение для отчета.

35. Нажмите *Note* и отредактируйте примечание для отчета.


36. Нажмите *OK*.

37. Нажмите , чтобы запустить генерацию отчета напротив активной диаграммы *ERwin*.

Генерация отчета "Report Entity"

38. Для генерации отчета нажмите . Можно также сгенерировать отчет двойным щелчком по иконке отчета . *ERwin* отобразит результат в области справа и добавит результат под отчетом в контрольном дереве.

39. Отсортируйте данные во колонке *Name* по возрастанию (*Sort ascending*). Для этого выберите в контекстном меню *Edit report format* и зайдите во вкладку *Sort*.

40. Экспортируем отчет: нажмите правой кнопкой мыши по сгенерированному отчету (рядом с таким отчетом есть иконка ). Для предварительного просмотра отчета следует выбрать в контекстном меню "*Preview result set* <имя отчета>". Выберите из меню "*Export result set* <имя отчета>". В появившемся окне "*Export from Data Browser*" выберите в качестве формата экспорта "*CSV*" (текстовый файл с разделенными точкой с запятой значениями). Нажмите кнопку "*Export*" и сохраните как "*Report Attribute*".

41. Импортируйте ранее сохраненный отчет следующим образом: дайте команду "*Import...*" в меню "*File*". Выберите формат источника для импорта: "*CSV file*". Нажмите кнопку "*Import*" с последующим выбором файла.

42. Закройте *Data Browser* и модель.

Контрольные вопросы

1. Что такое шаблон сообщения?
2. Что такое отчет по модели?
3. Какие существуют форматы отчетов?
4. Как сгенерировать отчет?
5. Каким образом можно экспортировать отчет в другие приложения?

Лабораторная работа №8. Работа с доменами

Цель работы: Изучение информации, ориентированной на СУБД. Получить навыки при работе с доменами в **ERwin**.

Теоретические сведения

Просмотр колонок в порядке физического следования

В режиме "*Physical Order*" (*Format/Display Level/*) атрибуты каждой сущности демонстрируются на диаграмме в том порядке, в котором соответствующие колонки располагаются в физической таблице. Этот режим позволяет Вам производить обратное проектирование таблиц базы данных, у которых первичные ключи находятся не в первой колонке, а в других.

Связывание физических имен

Физическое имя сущности, атрибута или связи в **ERwin** автоматически связывается с логическим именем объекта при просмотре диаграммы на уровне "*Physical Schema*" или при открытии отчета **ERwin**. Физическое имя связывается и при выборе объекта.

Примечание: физическое имя колонки связывается с атрибутом **ERwin**, когда:

1. Вы открываете редактор *Database Schema* для сущности, содержащей атрибут;
2. Вы открываете редактор *Database Schema* для сущности, которая является родителем другой сущности, содержащей атрибут, и включаете режимы "*Field Name Inheritance*" при выходе из редактора.

Работа в редакторе "Columns"

Редактор "*Columns*" позволяет просматривать и модифицировать характеристики, присваиваемые по умолчанию каждой колонке в таблице базы данных. Колонке могут быть присвоены характеристики: имя колонки, тип данных, режим нулевых значений, правило проверки введенных значений, значение по умолчанию, домен и правила миграции характеристик колонки.

Редактор "*Columns*" демонстрирует имя выбранной сущности и соответствующей физической таблицы в верхней части диалога. Прямо под именем сущности/таблицы находится список, содержащий информацию о каждой колонке выбранной таблицы, включая имя колонки, тип данных, имя атрибута и текущий режим нулевых значений.

Для просмотра колонок в другой таблице нажмите стрелку вниз, которая находится рядом с окном "*Column*" и разверните список сущностей, в котором выберите другую сущность, щелкнув по ней. **ERwin** сразу же выводит на экран информацию о колонках для новой сущности.

Кнопка "*Reset*" открывает окно-диалог для повторной установки значения одной или нескольких характеристик в соответствии со значениями, устанавливаемыми по умолчанию и определенными в домене.

Кнопка "*DB Sync*" открывает окно-диалог "*DB Sync*" для синхронизирования имен колонок, определенных в модели данных, с информацией, хранящейся на сервере СУБД.

При использовании редактора "*Columns*" для задания информации схемы базы данных лучше задать все характеристики колонок независимых сущностей перед вводом информации в дочерние сущности. В этом случае **ERwin** может произвести автоматическую миграцию имени колонки, домена и типа данных для внешних ключей через связи.

Характеристики колонок по умолчанию

ERwin автоматически создает имена таблиц и колонок на основе имен соответствующих сущностей и атрибутов, учитывая максимальную длину имени и другие синтаксические ограничения, накладываемые СУБД. Когда **ERwin** создает имя таблицы или колонки по умолчанию, то он автоматически преобразует все пробелы и тире в символы подчеркивания и обрезает имя колонки до максимальной длины, допустимой для конкретной СУБД. Эти изменения не отражаются на именах сущностей и атрибутов, поскольку информация на логическом и физическом уровнях в **ERwin** хранится отдельно.

ERwin автоматически присваивает каждой колонке при ее создании тип данных по умолчанию, определенный в редакторе *Target Server*, и этот тип данных сохраняется, если Вы не измените его в редакторе *Columns*.

ERwin также автоматически присваивает режимы нулевых значений всем неключевым колонкам, исходя из значений по умолчанию, устанавливаемых в редакторе "*Target Server*" (*Database/Choose Database*). Поскольку ключевые колонки по определению не могут быть "*NULL*", **ERwin** устанавливает режим "*NOT NULL*" для каждой колонки первичного ключа и (или) альтернативных ключей. Режим "*NOT NULL*" не присваивается автоматически "*Inversion Entry*".

Миграция характеристик колонок

При создании связи **ERwin** автоматически производит миграцию первичных ключей родительской сущности в дочернюю сущность в качестве внешних ключей. Режимы "*Migrate*" в редакторе "*Column Property*" позволяют определить, какие характеристики колонок первичного ключа будут автоматически мигрировать во внешние ключи на основе этой колонки.

По умолчанию **ERwin** переносит все характеристики колонки, за исключением имени колонки. Если нужно перенести характеристики в колонки внешнего ключа, то следует поставить метку.

При отмеченном режиме "*Migrate*" при входе в редактор "*Column Property*" вся информация о соответствующих характеристиках колонки автоматически переносится во внешние ключи дочерних сущностей во всей диаграмме. Мигрирующие характеристики записываются на место старых характеристик колонки, которые были присвоены внешним ключам, включая характеристики, присвоенные по умолчанию в соответствии со значениями, заданными в редакторе "*Target Server*", связанный с колон-

кой домен или некоторое специальное переопределение для конкретной колонки.

При использовании различных имен колонок в качестве внешних ключей лучше поставить метку в "*Col Name*" в окне *Migrate*, когда впервые вводятся характеристики колонок для зависимых сущностей. При внесении изменений в имена колонок для дочерних сущностей не нужно, чтобы в "*Col Name*" стояла метка, иначе имена внешних ключей будут заменены на новые.

Примечание: Миграция характеристик колонки, определенных в редакторе "*Column Property*", производится только в одну сторону - от родительских сущностей к дочерним. Миграция никогда не производится от дочерних сущностей к родительским.

Ограничения

ERwin поддерживает ввод правил проверки введенных значений для колонок (в зависимости от выбранной СУБД), а также информации, присваиваемой колонкам по умолчанию.

Значение по умолчанию - значение, которое нужно ввести в колонку, если никакое другое значение не задано явным образом во время ввода данных. С каждой колонкой (атрибутом) можно связать значение по умолчанию. Правило проверки введенных значений задает список допустимых значений для конкретной колонки или использует выражение для задания какого-то вида кода для проверки данных.

Правила проверки введенных значений и значения колонок по умолчанию создаются с использованием подходящих выражений языка определения данных для конкретной СУБД, либо активным образом через подключение к системному каталогу, либо через скрипт файла *ASCII*. **ERwin** не производит проверки на непротиворечивость данных в доменах и значений, которые присваиваются колонкам по умолчанию.

По умолчанию **ERwin** создает выражение – команду языка СУБД, используя значения, связанные с правилами проверки введенных значений, и разделяя их запятыми (например, *C, D, M*). В некоторых случаях правила синтаксиса данной СУБД требуют, чтобы каждое значение в команде заключалось в апострофы ("*C*", "*D*", "*M*"). Чтобы автоматически заключить каждое значение в апострофы, нажмите по "*Quote Value*" и поставьте в него метку.

Чтобы изменить правило проверки введенных значений, выберите правило в списке "*Validation Name*" в верхней части редактора, измените соответствующие характеристики и нажмите кнопку "*Update*". Правилу проверки введенных значений можно присвоить одной или нескольким колонкам в редакторе "*Column Property*". Чтобы присвоить правило, войдите в редактор "*Column Property*", выберите колонку для связывания и выберите правило из списка "*Valid*".

Работа в редакторе "Valid Value"

Редактор "Valid Value" позволяет создавать список всех допустимых значений, которые можно хранить в колонке, и связать его с правилом валидации. Если будет задан список допустимых значений и присвоен колонке, то в этой колонке будет можно хранить только те значения, которые есть в списке.

Список в верхней части редактора "Valid Value" содержит все имеющиеся валидационные правила. Чтобы выбрать правило, с которым нужно связать список допустимых значений, нажмите по стрелке "вниз", а затем — по имени правила в списке. При добавлении нового значения **ERwin** автоматически заносит его в конец списка. Вместе с тем редактор "Valid Value" позволяет расположить значения в списке.

Вставьте новое допустимое значение в список

1. Поставьте метку *X* в *check box* "Insert" в верхней части редактора *Valid Value*.

2. Выделите значение, прямо над которым Вы хотите вставить новое значение в список. Например, если Вы хотите вставить *Foreign* между *Drama* и *Horror*, то выбирайте *Horror*.

3. Нажмите по текстовому окну "Data Value" и введите имя нового значения.

4. Нажмите по окну "Value Definition" и введите определение значения.

5. Нажмите кнопку "New". **ERwin** вставит новое значение над тем, которое Вы выбрали, и выделит его.

Примечание: Если Вы не поставите *X* в окне "Insert", то **ERwin** автоматически добавит новое значение в конец списка.

Сортировка списка допустимых значений

При нажатии кнопки "Sort" (справа внизу в редакторе *Valid Value*) **ERwin** автоматически отсортирует список допустимых значений по возрастанию (например, A-Z, 0-9).

Работа с доменами

Домен определяет набор характеристик колонки вместе под одним именем.

Если используется стандартная версия **ERwin** (**ERwin/ERX**), то домен может включать в себя одну или более характеристик колонки, ориентированных на СУБД, таких как тип данных, режим нулевых значений, значение по умолчанию и правило валидации. Если используется версия **ERwin**, ориентированная на конкретный инструмент разработки клиентских систем, типа "**ERwin/ERX for PowerBuilder**" или "**ERwin/ERX for SQLWindows**", то домен может также включать характеристики колонки, ориентированные на среду клиента, например форматы изображения и стили редактирования "**PowerBuilder**".

Использование доменов ускоряет процесс проектирования базы данных и упрощает работу с моделью данных. Вместо того чтобы задавать ограничения для каждой колонки в отдельности, можно создать домен и использовать его для задания нескольких характеристик одновременно. Для последующего изменения характеристик колонки можно изменить домен, и все связанные с ним колонки будут автоматически изменены.

Редактор "*Domain*" позволяет задавать имя домена, родительский домен, имя колонки, тип данных, режим нулевых значений, значение по умолчанию, правило валидации и другие характеристики, которые можно скомбинировать и сохранить в домене. Режимы и возможности, доступные в редакторе *Domain*, различаются в зависимости от выбранной СУБД.

Работа в редакторе *Domain*

Редактор подобен редактору "*Column Property*", только он позволяет просматривать и изменять характеристики домена, а не колонки.

Значение по умолчанию и родительский домен.

В редакторе на экран выводятся имя домена, тип данных и определение для всех доменов из списка, который находится в верхней части окна-диалога. При выделении домена в этом списке **ERwin** сразу же показывает в остальных окнах редактора все значения характеристик колонки, связанные с выбранным доменом. Чтобы просмотреть характеристики колонки, связанные с другим доменом, просто выделите имя домена в верхнем списке, щелкнув по нему.

Чтобы создать домен, выберите домен в списке "*Domain Name*" и выберите родительский домен в списке "*Inheritance Hierarchy*". Установите курсор в окно "*Name*" и введите имя домена. По желанию можете ввести определение домена в текстовое окно "*Domain Definition*".

После этого нажмите кнопку "*New*" для создания домена.

Для изменения характеристик колонки, связанной с выделенным доменом, нажмите по нужным значениям в текстовом окне или нажмите нужные кнопки, а затем нажмите кнопку "*Update*". Например, чтобы изменить режим нулевых значений, присвоенный домену "<default>", с "*NULL*" на "*NOT NULL*", нажмите кнопку "*NOT NULL*" в групповом окне "<DB> *Null Option*" и нажмите кнопку "*Update*".

Для удаления домена выделите удаляемый домен в верхнем списке и нажмите кнопку "*Delete*".

Кнопки "*Default*" и "*Validation*" в редакторе "*Domain Dictionary*" служат для входа в редакторы "*Default*" и "*Validation Rules*", поэтому можно перейти в определенный редактор, задать новую характеристику колонки, например значение по умолчанию, а затем вернуться в редактор "*Domain*" и присвоить ее существующему домену, не обращая при этом к главному меню.

Кнопка "*Reset...*" открывает окно-диалог, в котором можно восстановить значения домена по умолчанию для одной или более характеристик

колонки. Например, при изменении режима нулевых значений для домена "<default>" с "NULL" на "NOT NULL" можно использовать кнопку "Reset..." для восстановления начального значения "NULL".

Можно изменить характеристики нескольких доменов в течение одного сеанса, не выходя из редактора "Domain Dictionary". При изменении характеристик, связанных с доменом, изменение происходит при выделении другого домена.

Примечание: при выборе СУБД: *Rdb*, *SYBASE* или *SQL Server* - в редакторе "Domain" появляется дополнительное окно "UDP". При метке в этом окне присваиваемые домену характеристики будут созданы как тип данных, определяемый пользователем (*User Defined Datatype*) при генерации схемы. Если в окне не будет метки, то **ERwin** использует синтаксис полного типа данных для создания присвоенных ограничений в операторе *CREATE TABLE*.

Домен "Default"

Характеристики, присваиваемые новой колонке доменом *default*, изначально основываются на значениях типов данных и режима нулевых значений, присваиваемых по умолчанию, которые задаются в редакторе "Target Server". Домен *default* использует макрокоманду "AttName" для присваивания колонке имени логического атрибута в качестве имени колонки. Можно изменить в редакторе "Domain Dictionary" любые характеристики, присвоенные домену *default*, за исключением его имени.

Наследование домена

Новый домен можно создать только из уже существующего домена. При создании нового домена автоматически наследуются все характеристики, присвоенные родительскому домену.

Для создания нового домена в **ERwin** следует выбрать родительский домен, на котором будет основан новый домен, из списка "Inheritance Hierarchy" с правой стороны диалога. Это означает, что домен "<default>" должен быть использован в качестве родителя первого домена. При нажатии на кнопку "New" для создания нового домена **ERwin** добавляет имя домена в список "Inheritance Hierarchy" и вставляет его после родительского домена с более глубоким уровнем вложенности, чтобы показать иерархию доменов.

Создание домена

1. Новый домен можно создать только из уже существующего домена.
2. Войдите в редактор "Domain Dictionary". Выберите способ редактирования в "Edit Mode" "Logical" (может быть "Logical" или "Physical"). Задайте порядок сортировки в алфавитном порядке ("Alphabetically"-алфавитный порядок, "Hierarchically"-иерархический порядок).
3. Нажмите кнопку "New" для создания домена. Выделите родительский домен в списке "Domain" (при первом создании домена список будет

таким: *Blob, Datetime, Number, String*). Введите логическое и физическое имена доменов "*Logical Name*" и "*Physical Name*". Новый домен автоматически унаследует все характеристики, присвоенные родительскому домену.

Изменение имени домена

4. Войдите в редактор "*Domain Dictionary*".
5. Выберите домен, имя которого нужно изменить, в списке в верхней части редактора "*Domain Dictionary*".
6. Нажмите кнопку "*Rename*" для изменения имени. Максимальная длина имени определяется конкретной СУБД.


Изменение типа данных домена

1. Войдите в редактор "*Domain Dictionary*".
2. Выберите домен, тип данных которого нужно изменить, в списке в верхней части редактора "*Domain Dictionary*". **ERwin** покажет тип данных, который сейчас присвоен выбранному домену, в списке "<DB> Datatype" с правой стороны редактора.
3. Нажмите по новому типу данных, который следует присвоить выбранному домену, в списке "<DB> Datatype". Если для задания выбранного типа требуется задать длину (например, для типа *CHAR()*), то в скобках следует ввести нужное число, затем снова щелкните по домену в верхнем списке, чтобы вставить параметр в заново присвоенный тип данных.

Изменение режима нулевых значений домена


1. Войдите в редактор "*Domain Dictionary*".
2. Выберите домен, режим нулевых значений которого следует изменить, в списке в верхней части редактора "*Domain Dictionary*".
3. Нажмите одну из кнопок в групповом окне "*Null Option*" для изменения режима нулевых значений домена.

Присвоение домену значения по умолчанию

1. Войдите в редактор "*Domain Dictionary*".
2. Выберите домен, которому следует присвоить значение по умолчанию, в списке в верхней части редактора "*Domain Dictionary*".
3. Щелкните по стрелке "вниз" для открытия списка "*Default*", а затем выберите значению "*Default Value*", которое следует присвоить выбранному домену.
4. Для задания нового значения по умолчанию и присвоения его домену следует нажать кнопку  рядом с "*Default*" для входа в редактор "*Default/Initial Values*". Нажать на кнопку "*New*" и задать логическое и физическое имена значения. Затем нажать "*OK*" и в окне закладки <DB> ввести значение по умолчанию. Нажать "*OK*" для возврата в редактор.

Если нужно убедиться, что никакое значение по умолчанию не было присвоено домену, то выберите из списка верхнюю строку - пунктирную линию (----).

Присвоение домену правила валидации

1. Войдите в редактор "*Domain Dictionary*".
2. Выберите домен, которому нужно присвоить правило валидации, в списке в верхней части редактора "*Domain Dictionary*".
3. Щелкните по стрелке "вниз" для открытия списка "*Valid*", а затем, если правило валидации есть в наличии, щелкните по правилу валидации, которое нужно присвоить выбранному домену.
4. Для задания нового правила валидации и присвоения его домену следует войти в редактор "*Validation Rules*", нажав на кнопку  рядом с "*Valid*". Нажмите кнопку "*New*" для создания нового правила валидации. Задайте логическое и физическое имена правила валидации. Можно задать три типа правил валидации: *User-Defined* (в окне *Validation* задается описание правила), *Min/Max* (можно задать границы правила), *Valid Values List* (можно задать список валидации). Можно переименовать или удалить правило с помощью кнопок *Rename* и *Delete* соответственно.

Если нужно убедиться, что никакое правило валидации не было присвоено домену, то выберите из списка верхнюю строку – пунктирную линию (----).

Связывание с доменом новых ограничений

Войдите в редактор "*Domain Dictionary*", а затем выберите домен, с которым следует связать новое ограничение.

Если после этого нажать кнопку "*Default*" или "*Validation*" для задания нового ограничения на колонку, **ERwin** автоматически использует имя выбранного домена в качестве имени ограничения и связывает новое ограничение с выделенным доменом.

Восстановление характеристик домена с присвоением им значения, задаваемого по умолчанию

1. Войдите в редактор "*Domain Dictionary*", а затем выберите домен, характеристики которого следует восстановить.
2. Нажмите кнопку "*Reset*" для входа в диалог "*Reset Domain Property*".
3. Отметьте один или несколько пунктов в диалоге "*Reset Domain Properties*", чтобы показать, какие из характеристик следует восстановить, присвоив им значения по умолчанию.

Нажмите "*OK*" для восстановления помеченных характеристик или "*Cancel*" - для выхода из диалога без изменения характеристик.

Удаление домена

1. Войдите в редактор "*Domain Dictionary*".
2. Выделите имя домена, который следует удалить, в списке, который находится в верхней части редактора "*Domain Dictionary*".

3. Нажмите кнопку "*Delete*" для удаления домена. **ERwin** удаляет выбранный домен, не запрашивая подтверждения на удаление.

Если удаляемый домен связан с колонкой, то **ERwin** выдает окно-диалог, в котором содержится имя домена и связанная с ним колонка (колонки) и запрашивается подтверждение на удаление. При удалении домена все характеристики колонки, определенные этим доменом, восстанавливаются в соответствии с содержимым родительского домена.

Порядок выполнения работы

1. Откройте модель "*lab_5_1*", сохраненную в пятой лабораторной работе (модель должна иметь возможность быть рассмотренной как с логической, так и физической точки зрения).

2. Сохраните ее как "*lab_8.er1*".

3. Войдите в редактор "*Domain Dictionary*".

4. Выберите способ редактирования в "*Edit Mode*" "*Logical*". Задайте порядок сортировки в алфавитном порядке.

5. Нажмите кнопку "*New*" для создания домена. Выделите родительский домен *String* в списке "*Domain*". Введите логическое (*Dom*) и физическое (*Dom*) имена доменов "*Logical Name*" и "*Physical Name*".

6. Измените имя домена с помощью команды "*Rename*" на "*My_Dom*".

7. Изменение типа данных домена:


- Выберите домен "*My_Dom*", тип данных которого нужно изменить, в списке в верхней части редактора "*Domain Dictionary*";
- Щелкните по новому типу данных (*char(22)*), который следует присвоить выбранному домену, в списке "<DB> Datatype".

8. Присвойте новую иконку созданному домену.

9. Изменение режима нулевых значений домена;

- Поменяйте вид модели на физический (*physical*);
- Выберите домен "*My_Dom*", режим нулевых значений которого следует изменить, в списке в верхней части редактора "*Domain Dictionary*";
- Нажмите кнопку "*NOT NULL*" в групповом окне "*Null Option*" для изменения режима нулевых значений домена.

10. Присвоение домену значения по умолчанию

- Выберите домен "*My_Dom*", которому следует присвоить значение по умолчанию, в списке в верхней части редактора "*Domain Dictionary*";
- Для задания нового значения по умолчанию и присвоения его домену нажмите кнопку  рядом с "*Default*" для входа в редактор "*Default/Initial Values*". Нажмите на кнопку "*New*" и задайте логическое и физическое имена значения "*DefVal*" и "*DefVal*". Затем нажмите "*OK*" и введите значение по умолчанию: "*Hello*". В колонке "*Comment*" введите описание значения по умолчанию. Нажмите "*OK*" для возврата в редактор.

11. Связывание с доменом новых ограничений
 - Войдите в редактор "*Domain Dictionary*", а затем выберите домен, с которым следует связать новое ограничение;
 - Создайте новое ограничение "*New_valiation*";
 - Выберите "*Valid Value List*" и задайте три возможных значения: "*Hello*", "*Goodbye*" и "*Thank you*".
12. Измените имя таблицы "*Customer*" на "*My_Customer*".
13. Установите домен "*My_Dom*" для атрибутов "*Name*", "*C_Name*" и "*C_country*".
14. Сохраните модель.

Контрольные вопросы

1. Каким образом производится работа с информацией, ориентированной на СУБД?
2. Что такое правило валидации?
3. Как задается значение по умолчанию?
4. Как производится наследование доменов?
5. В каких целях используют домены?

Лабораторная работа №9. Триггеры

Цель работы: Ознакомиться с триггерами в *ERWin*.

Теоретические сведения

Триггеры - это процедуры базы данных, которая автоматически вызывается SQL сервером при возникновении определенных событий (добавление, удаление, обновление записей). Триггеры хранятся на сервере для того, чтобы быстро производить выполнение запросов, валидацию данных и выполнять другие, часто вызываемые функции. Если эти команды сохраняются на сервере, то нужно создавать код только один раз, а не в каждом приложении, работающем с базой данных. Это экономит время при написании программ. Поскольку коды хранятся на сервере, то их не требуется пересылать по сети из клиентского приложения, что значительно снижает сетевой трафик. При сохранении кодов на сервере гарантируется целостность данных и правила (*Business Rules*) поддерживаются единым образом, независимо от того, какое клиентское приложение обращается к данным.

Триггером называется именованный набор прекомпилированных команд *SQL*, хранящийся на сервере, который автоматически выполняется, когда происходит заданное событие. Например, триггер может выполняться при вставке, изменении или удалении строки в существующей таблице. Триггер сообщает СУБД, как нужно выполнять команды *SQL INSERT*, *UPDATE* или *DELETE*, чтобы выполнялись нормальные правила (*Business Rules*) организации.

Триггер ссылочной целостности – особый вид триггера, используемый для поддержания целостности между двумя таблицами, которые связаны между собой. Если строка в одной таблице вставляется, изменяется или удаляется, то триггер ссылочной целостности (*RI*-триггер) сообщает СУБД, что нужно делать с теми строками в других таблицах, у которых значение внешнего ключа совпадает со значением первичного ключа вставленной (измененной, удаленной) строки.

Если СУБД поддерживает *RI*-триггеры, то команда *DELETE* языка *SQL* может быть обработана одним из следующих способов:

- Правило ссылочной целостности, запрещающее вставку, изменение или удаление строки, называется *RESTRICT*.
- Правило ссылочной целостности, передающее изменение от одной таблицы к другой, называется *CASCADE*.
- Правило ссылочной целостности, изменяющее текущее значение данных на нулевое, называется *SET NULL*.
- *ERwin* предоставляет альтернативный способ усиления ссылочной целостности в случае СУБД "*Fox Pro*".

Примечание: *ERwin* располагает шестью *RI*-триггерами, устанавливаемыми по умолчанию, которые можно связывать с сущностями, чтобы

указывать СУБД ссылочную целостность для усиления. В особых ситуациях можно переопределить код, генерируемый **ERwin** по умолчанию, изменяя эти шаблоны триггеров в соответствии с конкретной ситуацией.

RI-триггеры, устанавливаемые ERwin по умолчанию

Для того чтобы создать триггер, в обычном случае требуется ввести нужный код *SQL* и поместить его на сервер. **ERwin** предлагает набор шаблонов RI-триггеров, устанавливаемых по умолчанию, которые используют для автоматической генерации кода *SQL* predefined макроккоманды.

Макроккоманды **ERwin** содержат скелетный код языка *SQL*, в который вставляются при генерации физической схемы базы данных имена таблиц и другие переменные. Ниже приводятся примеры макроккода триггера **ERwin** и расширенного кода, который экспортируется из **ERwin** на сервер в процессе генерации схемы.

Просмотр кода триггера, устанавливаемого по умолчанию, для сущности

Щелкните правой кнопкой мыши по сущности и дайте команду меню "*Triggers*". **ERwin** открывает окно, предназначенное только для чтения, в котором показан расширенный код для каждого RI-триггера, связанного с этой сущностью.

Создание триггеров ERwin

ERwin создает RI-триггеры автоматически. Когда строится модель данных, **ERwin** автоматически связывает шаблон RI-триггера, устанавливаемый по умолчанию, с каждой сущностью связи. На то, какой именно шаблон триггера будет присвоен связи и на генерируемый им код *SQL*, влияют три критерия:

- Правило ссылочной целостности, которое он применяет к связи (*RESTRICT, CASCADE, SET NULL, SET DEFAULT, NONE*);
- Тип связи, с которым он связан (идентифицирующая или неидентифицирующая);
- Роль сущности в связи (родительская или дочерняя).

Правила ссылочной целостности

Когда триггер связывается с сущностью, он автоматически устанавливается так, чтобы усиливать одно из следующих правил ссылочной целостности, в зависимости от типа связи и роли сущности в этой связи.

Примечание: **ERwin** создает RI-триггеры автоматически, связывая шаблоны триггеров со всеми связями. Можно переопределить код *SQL*, генерируемый **ERwin**, адаптируя эти шаблоны для своей ситуации.

RI-триггеры и типы связей

Шаблоны RI-триггеров в **ERwin** связываются с сущностями, исходя из типа связи и роли сущности в этой связи. Тип связи и роль сущности определяют, какое правило ссылочной целостности будет, по умолчанию, усилено присвоенным шаблоном триггера.

Таблица 3.6

Правила ссылочной целостности

| Правило ссылочной целостности | Что оно делает |
|-------------------------------------|--|
| <i>RESTRICT</i> | Запрещает СУБД производить требуемое изменение (<i>INSERT</i> , <i>UPDATE</i> или <i>DELETE</i>). |
| <i>CASCADE</i> | Производит требуемое изменение в первой таблице и распространяет его на связанные с ней таблицы. |
| <i>SET NULL</i> | Производит требуемое изменение в первой таблице и устанавливает нулевые (пустые) значения внешнего ключа в связанных с ней таблицах. |
| <i>SET DEFAULT</i> | Работает как <i>SET NULL</i> , с той разницей, что вместо нулевого значения присваивает внешним ключам значение по умолчанию. |
| <i>NONE</i> | Ничего не делает (ERwin не усиливает ссылочную целостность). |

Роль сущности в связи может быть – родительская (*Parent*) или дочерняя (*Child*) сущность. Если сущность является родительской в данной связи, то **ERwin** присваивает ей шаблон триггера для родительской сущности. Если сущность является дочерней в данной связи, то **ERwin** присваивает ей шаблон триггера для дочерней сущности. Код триггера, который генерируется шаблоном триггера для родительской сущности, указывает СУБД, что нужно делать при вставке, изменении или удалении строки в родительской таблице связи. Код триггера, который генерируется шаблоном триггера для дочерней сущности, указывает СУБД, что нужно делать при вставке, изменении или удалении строки в дочерней таблице связи.

Таблица 3.7

Описание присваивания связи правила ссылочной целостности

| Роль сущности | Тип связи | | | |
|----------------------|------------------------|---|--|-----------------|
| | Идентифицирующая связь | Неидентифицирующая (<i>nulls allowed</i>) | Неидентифицирующая (<i>no nulls</i>) | Связь подтипа |
| <i>Child Delete</i> | <i>None</i> | <i>None</i> | <i>None</i> | <i>None</i> |
| <i>Child Insert</i> | <i>Restrict</i> | <i>Set Null</i> | <i>Restrict</i> | <i>Restrict</i> |
| <i>Child Update</i> | <i>Restrict</i> | <i>Set Null</i> | <i>Restrict</i> | <i>Cascade</i> |
| <i>Parent Delete</i> | <i>Restrict</i> | <i>Set Null</i> | <i>Restrict</i> | <i>Cascade</i> |
| <i>Parent Insert</i> | <i>None</i> | <i>None</i> | <i>None</i> | <i>None</i> |
| <i>Parent Update</i> | <i>Restrict</i> | <i>Set Null</i> | <i>Restrict</i> | <i>Cascade</i> |

Примечание: с одной сущностью можно связать до шести шаблонов *RI*-триггеров, в зависимости от ее роли в разных связях. Если сущность является в какой-то связи родительской, то с ней можно связать триггеры

parent insert, update и delete. Если сущность является в какой-то связи дочерней, то с ней можно связать триггеры *child insert, update и delete*.

Изменение режима *RI*-триггера для связи

Редактор *Referential Integrity* позволяет изменять правило ссылочной целостности, связанное с конкретной связью.

Работа в редакторе *Trigger Template*

Редактор *Trigger Template* позволяет изменять шаблон, связанный с любым типом *RI*-триггера, а также просматривать и модифицировать макрокod, используемый конкретным шаблоном. При изменении шаблона триггера можно использовать другой встроенный шаблон или пользовательский (*User Override* - переопределенный пользователем) шаблон.

Для входа в редактор дайте команду "*Global Trigger Templates...*" (*Database/RI Triggers*). **ERwin** открывает редактор и показывает на экране встроенные и пользовательские шаблоны триггеров, связанных с сущностями диаграммы.

Использование *Trigger Toolbox* и макрокоманд

Редактор *Trigger Toolbox* предоставляет набор предопределенных макрокоманд, которые помогают адаптировать встроенные шаблоны триггеров, создавать переопределенные шаблоны триггеров или писать новые триггеры и хранимые процедуры на *SQL*. Предопределенные макрокоманды, которые начинаются с символа процента (%), генерируют псевдокод, который в процессе генерации схемы расширяется в специальный синтаксис *SQL*, поддерживаемый СУБД.

Если щелкнуть по имени макрокоманды и выделить ее, то **ERwin** выведет на экран, в центральное окно, вспомогательную информацию о выделенной макрокоманде.

Если при редактировании макрокода в окне *Template Code* будет выполнен вход в редактор *Trigger Toolbox* и дважды щелкнуто по имени макрокоманды, то **ERwin** вставит макрокоманду в то место окна кода, где стоял курсор в последний раз.

Переопределение шаблонов триггеров в *ERwin*

При генерации физической схемы базы данных **ERwin** по умолчанию использует для генерации кода триггера на языке *SQL* встроенные шаблоны *RI*-триггеров, которые автоматически присваиваются каждой связи. Поскольку с каждым типом правила ссылочной целостности связан какой-то встроенный шаблон, можно сгенерировать код триггера для всей модели и больше в этом отношении ничего не делать. Если используются шаблоны, присваиваемые **ERwin** по умолчанию, то **ERwin** составляет коды триггера по умолчанию, используя фиксированную внутреннюю схему для комбинирования различных встроенных шаблонов.

Если нужно изменить коды триггера, генерируемые на основе встроенных шаблонов, **ERwin** позволяет изменить шаблон и указать, что при ге-

нерации модифицированная версия должна заменить встроенный шаблон. **ERwin** позволяет переопределить триггер, устанавливаемый по умолчанию, тремя способами:

1. *RI Type Override* – Переопределение типа *RI*. Для каждой комбинации правил ссылочной целостности **ERwin** позволяет создать переопределенный шаблон и использовать этот шаблон вместо шаблона, используемого по умолчанию, для всех связей диаграммы, которым был присвоен этот тип правила ссылочной целостности. Используя в качестве отправного пункта встроенный код шаблона, можно, пользуясь этим способом, производить глобальные изменения в отношении ссылочной целостности, изменяя коды триггера только в одном месте. Шаблоны *RI Type Override* используются вместо стандартных шаблонов **ERwin**, если при генерации схемы задается режим *RI Type Override*.

2. *Relationship Override* – переопределение связи. Если нужно переопределить шаблон, задаваемый по умолчанию, для какой-то конкретной связи, можно модифицировать встроенный шаблон и связать новую версию только с этой связью. Шаблоны *Relationship Override* используются вместо стандартных шаблонов **ERwin** (а также вместо шаблонов *RI Type Override*, если они есть), если при генерации схемы задается режим *Relationship Override*.

3. *Entity Override* – переопределение сущности. **ERwin** позволяет создавать собственные триггеры *Entity Override* для любой сущности в диаграмме. Шаблоны *Entity Override* используются вместо стандартных шаблонов **ERwin**, а также вместо созданных Вами шаблонов *RI Type Override* и *Relationship Override*, если при генерации схемы задаете режим *Entity Override*.

Создание шаблона *RI Type Override*

Шаблон *RI Type Override* используется для изменения поведения встроенного шаблона для типа *RI*-триггера. Предположим, например, что нужно, чтобы **ERwin** всякий раз при удалении строки в родительской таблице *ORDER*, использовал код триггера *Parent-Delete CASCADE*, присваиваемый по умолчанию, для удаления соответствующих дочерних строк в таблице *Order Line*, но при этом Вы хотите также автоматически захватывать удаляемую информацию и вставлять удаляемые строки в архивную таблицу, так чтобы эту информацию потом при желании можно было найти.

Один из способов, как можно это сделать – изменить встроенный триггер *Parent-Delete CASCADE* так, чтобы он производил архивирование, создавая *RI Type Override*. Имеется несколько достоинств этого способа:

- к триггеру требуется добавить только пару строк кода;
- изменение производится только в одном месте;
- оно распространяется на всю базу данных;
- оно не отражается на исходном коде приложений;

- при решении больше не архивировать эту информацию можно будет быстро восстановить исходный встроенный шаблон *Parent-Delete CASCADE*.

Создание шаблона *RI Type Override* облегчает создание собственных триггеров ссылочной целостности, поскольку все связи, которым присваивается переопределенный тип *RI*-триггера, автоматически используют связанный с ним шаблон *User Override*.

Создание шаблона триггера *Relationship Override*

Чтобы облегчить работу с базой данных, целесообразно сделать так, чтобы триггеры работали идентично на всей модели данных. Однако в некоторых случаях может понадобиться изменить встроенный триггер только для одной связи. Шаблон *Relationship Override* используется для изменения поведения встроенного шаблона только для одной связи, а не для всех связей, для которых задан некоторый тип триггера.

Например, можно, чтобы **ERwin** использовал код триггера, установленный по умолчанию, – *Parent-Delete RESTRICT* – всякий раз при удалении строки из родительской таблицы *ORDER*, чтобы проверить, нет ли в этом заказе невыполненных пунктов. Если часть заказа не выполнена, то Вы, наверное, захотите использовать код триггера, устанавливаемого по умолчанию, который запрещает удалять заказ. Предположим, что в данной ситуации нужно также, чтобы триггер ссылочной целостности автоматически изменил значение в колонке *Order-Status* на "*Outstanding*".

Если изменяете встроенный триггер так, чтобы он мог выполнить это действие, создавая для этого *Relationship Override* для триггера *Parent-Delete RESTRICT*, то этот шаблон будет применяться только к связи между таблицами *ORDER* и *ORDER LINE*. Хотя изменение режимов ссылочной целостности для отдельных связей может усложнить работу с базой данных, использование режима *Relationship Override* позволяет вам поддерживать правила, которые распространяются на какие-то отдельные связи.

Примечание: режим *Relationship Override* изменяет действие *RI*-триггера для одной конкретной связи.

Создание триггера *Entity Override*

Во многих случаях бывает нужно, чтобы триггер выполнял действия, выходящие за рамки стандартных действий по усилению правил *RI*, например, производил вычисления, работал с колонкой, содержащей выведенные значения, или изменял содержимое колонки, исходя из текущего значения этой колонки. Эти триггеры расширенных правил *Business Rules* можно применить к связи или к сущности.

Если нужно усилить какие-то правила для отдельных сущностей, **ERwin** позволяет изменить поведение встроенного триггера для отдельной сущности. Шаблон *Entity Override* используется для изменения поведения встроенного шаблона по отношению только к одной сущности.

Предположим, например, что нужно, чтобы **ERwin** каждый раз при добавлении новой строки в таблицу *ORDER LINE* добавлял количество заказанного товара *order-quantity* из таблицы *ORDER LINE* к количеству проданного товара *product-sold-quantity* в таблице *PRODUCT*. Этого можно добиться, создав триггер *Entity Override* для сущности *ORDER LINE*, который изменяет действия, выполняемые встроенным шаблоном *Child-Insert RESTRICT*.

Примечание: *Entity Override* изменяет поведение триггера для какой-то одной сущности.

Использование ссылочной целостности для усиления *Business Rules*

Триггер – функции поддержания ссылочной целостности (*RI*) сообщают СУБД, какое действие предпринять при вставке, изменении или удалении строки в таблице. Обычно *RI*-триггер либо не дает изменению произойти (это называется *RESTRICT* – ОГРАНИЧЕНИЕМ), либо допускает изменение и распространяет его влияние на другие таблицы (*CASCADE* – КАСКАД).

Как и в случае кардинальности связи, режимы ссылочной целостности **ERwin** можно использовать для формулирования бизнес-утверждений.

Список макрокоманд

Данная таблица содержит имена всех макрокоманд **ERwin**, их синтаксис, описание расширенного кода, генерируемого макрокомандой и список СУБД, поддерживающих ее использование. Информация, которую Вы вводите вручную, заключена в угловые скобки (например, *<macro code 1>*).

Вы можете использовать эти макрокоманды в шаблонах, которые Вы создаете для триггеров, хранимых процедур и скриптов (если эти возможности поддерживаются Вашей СУБД). Если в графе "СУБД" стоит слово "Все" – это означает, что любая СУБД поддерживает использование макрокоманды хотя бы в одном типе шаблонов **ERwin**. Обратите внимание, что некоторые макрокоманды, например *%Fire*, позволяющие задавать, когда будет выполняться триггер, зависят от возможностей, которые не поддерживаются всеми СУБД.

Таблица 3.8

Список макрокоманд

| Макрокоманда | Описание | СУБД |
|---|--|------|
| <i>%!=(<i><macro code1></i>, <i><macro code 2></i>)</i> | Оператор сравнения, <i>!=</i> , сравнивает расширения <i>macro code1</i> и <i>macro code 2</i> | Все |
| <i>%%</i> | Используйте два символа <i>"%"</i> , если расширенный текст триггера должен содержать один символ <i>"%"</i> . | Все |

| Макрокоманда | Описание | СУБД |
|--|---|------|
| <code>%+(<macro code1>, <macro code 2>)</code> | Складывает расширения <i>macro code1</i> и <i>macro code 2</i> | Все |
| <code>%-(<macro code1>, <macro code 2>)</code> | Вычитает одно из другого расширения <i>macro code1</i> и <i>macro code 2</i> | Все |
| <code>%/(<macro code1>, <macro code 2>)</code> | Делит одно на другое расширения <i>macro code1</i> и <i>macro code 2</i> | Все |
| <code>%:<variable></code> | Возвращает значение <i><variable></i> . | Все |
| <code>%<(<macro code1>, <macro code 2>)</code> | Оператор сравнения, <i><</i> , сравнивает расширения <i>macro code1</i> и <i>macro code 2</i> | Все |
| <code>%<=(<macro code1>, <macro code 2>)</code> | Оператор сравнения, <i><=</i> , сравнивает расширения <i>macro code1</i> и <i>macro code 2</i> | Все |
| <code>%=(<variable>, <macro code>)</code> | Присваивает расширение <i><macro code></i> переменной <i><variable></i> . | Все |
| <code>%==(<macro code1>, <macro code 2>)</code> | Оператор сравнения, <i>==</i> , сравнивает расширения <i>macro code1</i> и <i>macro code 2</i> | Все |
| <code>%>(<macro code1>, <macro code 2>)</code> | Оператор сравнения, <i>></i> , сравнивает расширения <i>macro code1</i> и <i>macro code 2</i> | Все |
| <code>%>=(<macro code1>, <macro code 2>)</code> | Оператор сравнения, <i>>=</i> , сравнивает расширения <i>macro code1</i> и <i>macro code 2</i> | Все |
| <code>%Action</code> | Действие, до или после которого выполняется триггер (напр., <i>INSERT</i> , <i>UPDATE</i> , <i>DELETE</i>). | Все |
| <code>%Actions(<separator>)</code> | Разделенный список действий, до или после которых выполняется триггер (напр., <i>INSERT</i> or <i>UPDATE</i>). | Все |
| <code>%And(<macro code1>, <macro code2>)</code> | Выполняет операцию "логическое И" над булевскими предикатами, заданными в <i><macro code1></i> и <i><macro code2></i> . | Все |
| <code>%AttDatatype</code> | Создает строку, представляющую тип данных текущего атрибута. | Все |
| <code>%AttFieldname</code> | Создает строку, представляющую физическое имя поля текущего атрибута. | Все |
| <code>%AttFieldWidth</code> | Генерирует целое число, представляющее длину типа данных текущего атрибута (напр., <i>varchar(50)</i> -- <i>>50</i>). | Все |
| <code>%AttIsFK</code> | Булевский предикат, который может быть использован как условие в выражении <i>%If</i> . Он определяет, входит ли текущий атрибут во внешний ключ. | Все |

| Макрокоманда | Описание | СУБД |
|--|--|------|
| <i>%AttIsPK</i> | Булевский предикат, который может быть использован как условие в выражении <i>%If</i> . Он определяет, входит ли текущий атрибут в первичный ключ | Все |
| <i>%AttName</i> | Создает строку, представляющую логическое имя текущего атрибута. | Все |
| <i>%AttNullOption</i> | Создает строку, представляющую режим нулевых значений для текущего атрибута. | Все |
| <i>%AttPhysDatatype</i> | Генерирует физический тип данных текущего атрибута независимо от того, является ли этот тип данных типом данных, определенным пользователем. | Все |
| <i>%Atts(<separator>, <function>, <prefix>)</i> | Выдает список всех атрибутов сущности триггера, выполняя заданную функцию для каждого элемента. | Все |
| <i>%AttValidation</i> | Возвращает имя правила валидации, связанного с данным атрибутом; может быть использован в <i>ForEachAtt</i> или <i>ForEachFKAtt</i> . | Все |
| <i>%Cardinality</i> | Кардинальность связи. | Все |
| <i>%Child</i> | Физическое имя таблицы дочерней сущности связи. | Все |
| <i>%ChildAtts(<separator>, <function>, <prefix>)</i> | Выдает список всех атрибутов дочерней сущности связи, выполняя заданную функцию для каждого элемента. | Все |
| <i>%ChildFK(<separator>, <function>)</i> | Выдает список внешних ключей дочерней сущности связи, выполняя заданную функцию для каждого элемента. (<i>update(customer_number)</i> или <i>update(customer_name)</i> и т.д.). | Все |
| <i>%ChildFKDecl(<old prefix>, <new prefix>, <separator>)</i> | Выдает список внешних ключей дочерней сущности связи с их типами данных (См. <i>%ParamDecl</i>). | Все |
| <i>%ChildNK(<separator>, <function>, <prefix>)</i> | Генерирует разделенный список функций для каждого неключевого элемента дочерней сущности (напр., <i>update(customer_number) or update(customer_name) or</i>). | Все |
| <i>%ChildNKDecl(<old prefix>, <new prefix>, <separator>)</i> | Выдает список неключевых атрибутов дочерней сущности связи с их типами данных (См. <i>%ParamDecl</i>). | Все |

| Макрокоманда | Описание | СУБД |
|---|--|--|
| <i>%ChildParamDecl(<old prefix>, <new prefix>, <separator>)</i> | Выдает список атрибутов дочерней сущности связи с их типами данных (См. <i>%ParamDecl</i>). | Все |
| <i>%ChildPK<separator>, <function>, <prefix>)</i> | Генерирует разделенный список функций для каждого элемента первичного ключа дочерней сущности (напр., <i>update(customer_number) or update(customer_name) or</i>). | Все |
| <i>%ChildPKDecl(<old prefix>, <new prefix>, <separator>)</i> | Выдает список атрибутов первичного ключа дочерней сущности связи с их типами данных (См. <i>%ParamDecl</i>). | Все |
| <i>%Concat(<value1>, <value2>)</i> | Производит конкатенацию <i><value1></i> и <i><value2></i> . Возвращает результат. | Все |
| <i>%CustomTriggerDefaultFooter</i> | Часть триггера, определенного пользователем - <i>default footer</i> , которая содержится в <i>diagram-wide</i> сегменте шаблона <i>CUSTOM TRIGGER FOOTER</i> . | Все |
| <i>%CustomTriggerDefaultHeader</i> | Часть триггера, определенного пользователем - <i>default header</i> , которая содержится в <i>diagram-wide</i> сегменте шаблона <i>CUSTOM TRIGGER HEADER</i> . | Все |
| <i>%Datetime</i> | Создает строку, представляющую текущую дату и время. | Все |
| <i>%DBMS</i> | Возвращает имя СУБД. | Все |
| <i>%DBMSDelim</i> | Возвращает разделитель операторов СУБД. | Все |
| <i>%Decl(<arg>, <initial value>)</i> | Объявляет <i><arg></i> как переменную и, если это задано, присваивает ей значение <i><initial value></i> . | Все |
| <i>%Fire</i> | Задаёт условие, когда выполняется триггер (напр., <i>BEFORE</i> , <i>AFTER</i>). | <i>INFORMIX</i> <i>Ingres</i> <i>ORACLE7</i> <i>Rdb</i> |
| <i>%ForEachAtt(<table>, <separator>) {\015\n<macro code>\015\n}</i> | Расширяет макрокод для каждого из атрибутов заданной таблицы. | Все |
| <i>%ForEachChildRel (<separator>) {\015\n<relationship code>\015\n}</i> | Расширяет <i><relationship code></i> для каждой связи, в которой сущность триггера является дочерней. | Все |
| <i>%ForEachFKAtt(<separator>) {\015\n<macro code>\015\n}</i> | Расширяет макрокод для каждого из атрибутов внешнего ключа, мигрировавших через текущую связь. | Все |
| <i>%ForEachParentRel (<separator>) {\015\n<relationship code>\015\n}</i> | Расширяет <i><relationship code></i> для каждой связи, в которой сущность триггера является родительской. | Все |

| Макрокоманда | Описание | СУБД |
|--|--|------|
| <code>%If(<predicate>){<macro code>}</code> <code>%Else {<macro code>}</code> | В зависимости от условия, расширяет макрокод <i>if</i> или <i>else</i> . Часть <i>else</i> не является обязательной. | Все |
| <code>%include("path name")</code> | Позволяет Вам включать макрокоды триггера в файлы. | Все |
| <code>JoinFKPK([<child table>,<parent table>,<comparison op>,<separator>)</code> | Часть условия поиска оператора <i>Where</i> , присоединяющая внешний ключ дочерней сущности к первичному ключу родительской сущности связи. | Все |
| <code>JoinPKPK(<table>,<correlation>,<comparison op>,<separator>)</code> | Часть условия поиска оператора <i>Where</i> , соединяющая первичные ключи двух корреляций или таблицы и корреляции. | Все |
| <code>%Len(<macro code>)</code> | Возвращает длину строки <i><macro code></i> . | Все |
| <code>%Lower(<macro code>)</code> | Преобразует расширение <i><macro code></i> в нижний регистр. | Все |
| <code>%Max(<value1>,<value2>)</code> | Возвращает максимальное значение - <i><value1></i> или <i><value2></i> . | Все |
| <code>%Min(<value1>,<value2>)</code> | Возвращает минимальное значение - <i><value1></i> или <i><value2></i> . | Все |
| <code>%NK(<separator>,<function>,<prefix>)</code> | Выдает список всех неключевых атрибутов сущности триггера, выполняя заданную функцию для каждого элемента. | Все |
| <code>%NK'Decl(<old prefix>,<new prefix>,<separator>)</code> | Выдает список неключевых атрибутов сущности триггера с их типами данных (См. <i>%ParamDecl</i>). | Все |
| <code>%Not(<macro code>)</code> | Выполняет операцию "логическое НЕ" над булевым предикатом, определенным в <i><macro code></i> . | Все |
| <code>%NotNullFK(<child table>,<not null expression>,<prfix>,<separator>)</code> | Часть условия поиска оператора <i>Where</i> , сравнивающая внешний ключ дочерней сущности связи с <i>null</i> . Эта макрокоманда расширяется тогда и только тогда, когда связь является неидентифицирующей, <i>nulls allowed</i> . | Все |
| <code>%Or(<macro code1>,<macro code2>)</code> | Выполняет операцию "логическое ИЛИ" над булевыми предикатами, определенными в <i><macro code1></i> и <i><macro code2></i> . | |

| Макрокоманда | Описание | СУБД |
|---|---|------|
| <i>%ParamDecl(<old prefix>, <new prefix>, <separator>)</i> | Выдает список всех атрибутов сущности триггера с их типами данных. Имя каждого атрибута имеет формат: <i><old/new prefix>_<att_name></i> . Если заданы и старый и новый префикс, то длина списка удваивается. В первой половине списка содержится <i><old prefix>_<att_name></i> , во второй - <i><new prefix>_<att_name></i> . | |
| <i>%ParamPass(<old prefix>, <new prefix>, <param/value separator>, <param separator>)</i> | Присваивает значения параметрам процедур, заданным в <i><old prefix></i> и (или) в <i><new prefix></i> для всех атрибутов сущности триггера. | |
| <i>%Parent</i> | Физическое имя таблицы родительской сущности связи. | |
| <i>%ParentAtt (<attribute macro>)</i> | Расширяет любую макрокоманду атрибута (напр., <i>%AttFieldName</i> , <i>%AttDatatype</i>) для атрибута родительского первичного ключа, который, мигрировав, сформировал текущий атрибут. | |
| <i>%ParentAtts(<separator>, <function>, <prefix>)</i> | Выдает список всех атрибутов родительской сущности связи, выполняя заданную функцию для каждого элемента. | Все |
| <i>%ParentNK(<separator>, <function>, <prefix>)</i> | Выдает список всех неключевых атрибутов родительской сущности связи, выполняя заданную функцию для каждого элемента. | Все |
| <i>%ParentNKDecl(<old prefix>, <new prefix>, <separator>)</i> | Выдает список неключевых атрибутов родительской сущности связи с их типами данных (См. <i>%ParamDecl</i>). | Все |
| <i>%ParentParamDecl(<old prefix>, <new prefix>, <separator>)</i> | Выдает список неключевых атрибутов родительской сущности связи с их типами данных (См. <i>%ParamDecl</i>). | Все |
| <i>%ParentPK(<separator>, <function>)</i> | Выдает список всех атрибутов первичного ключа родительской сущности связи, выполняя заданную функцию для каждого элемента. | Все |
| <i>%ParentPKDecl(<old prefix>, <new prefix>, <separator>)</i> | Выдает список атрибутов первичного ключа родительской сущности связи с их типами данных (См. <i>%ParamDecl</i>). | Все |
| <i>%PhysRelName</i> | Физическое имя связи. | Все |

| Макрокоманда | Описание | СУБД |
|--|--|--------------------------------------|
| %PK(<separator>, <function>) | Выдает список первичных ключей сущности триггера, выполняя заданную функцию для каждого элемента. | Все |
| %PKDecl(<old prefix>, <new prefix>, <separator>) | Выдает список атрибутов первичного ключа сущности триггера с их типами данных (См. %ParamDecl). | Все |
| %RefClause | Оператор ссылок; расширяется: <i>REFERENCES OLD as <old name> new as <new name></i> . | INFORMIX Ingres ORACLE7 Rdb |
| %RelTemplate | Расширяет связь "Template Code", присоединенную к текущей связи. Если нет присоединенного кода, то расширяется соответствующий <i>diagram-wide</i> шаблон ссылочной целостности. | Все |
| %Scope | Задаёт, каким образом будет выполняться триггер (напр., один раз для всей таблицы, для каждой строки и т.д.). | ORACLE7 |
| %SetFK(<child table>, <value>) | Выдает список внешнего ключа дочерней сущности связи, в котором каждому элементу присвоено заданное значение. | Все |
| %SetPK(<table>, <value>) | Выдает список первичного ключа заданной таблицы, в котором каждому элементу присвоено заданное значение | Все |
| %Substitute(<value>, <pattern>, <substitute>) | Заменяет строку <pattern> в строке <value> на строку <substitute>. | Все |
| %Substr(<macro code>, <initial pos>, <length>) | Создает подстроку для расширения заданного <macro code>. | Все |
| %Table Name | Физическое имя таблицы сущности триггера. | Все |
| %Template Name | Возвращает имя шаблона триггера, хранимой процедуры или скрипта; может быть использовано в редакторе <i>Entity Trigger</i> . | Все |
| %Trigger Name | Физическое имя триггера. | Все |
| %TriggERelRI(<action>, <type>, <integrity>) | Булевский предикат, принимающий значение "истинно", если заданный триггер и связь относятся к заданному действию (<i>Update/Delete/Insert</i>), типу (<i>Child/Parent</i>) и целостности (<i>Cascade/Restrict/Set Null/Set Default</i>). | Все |

| Макрокоманда | Описание | СУБД |
|--|--|-----------------------------------|
| <i>%UpdateChildFK()</i> | Выдает список внешнего ключа дочерней сущности связи, выполняя функцию <i>update</i> для каждого элемента. | <i>ORACLE7, SQL Server SYBASE</i> |
| <i>%UpdateParentPK()</i> | Выдает список первичного ключа родительской сущности связи, выполняя функцию <i>update</i> для каждого элемента. | <i>ORACLE7, SQL Server SYBASE</i> |
| <i>%UpdatePK()</i> | Выдает список первичного ключа сущности триггера, выполняя функцию <i>update</i> для каждого элемента. | <i>ORACLE7, SQL Server SYBASE</i> |
| <i>%Upper(<macro code>)</i> | Преобразует расширение <i><macro code></i> в верхний регистр. | |
| <i>%ValidationHasValidValues (<arg>)</i> | Возвращает <i>"TRUE"</i> , если заданное правило валидации <i><arg></i> имеет допустимые значения, иначе - <i>"FALSE"</i> . | |
| <i>%ValidationRule(<validation name>)</i> или <i>%ValidationRule</i> | Возвращает правило валидации для сервера; может быть использовано в любом месте с аргументом <i><validation name></i> или в рамках действия правила, без аргументов. | |
| <i>%ValidValue</i> | Возвращает значение допустимого значения; используется в рамках действия допустимого значения | Все |
| <i>%ValidValueDef</i> | Возвращает определение допустимого значения; используется в рамках действия допустимого значения | Все |
| <i>%VerbPhrase</i> | Возвращает глагольную фразу связи. | Все |

Порядок выполнения работы

1. Изменение режима *RI*-триггера для связи:
 - 1.1. Откройте модель *"lab_8.er1"*, сохраненную в восьмой лабораторной работе. Установите физический вид модели.
 - 1.2. Щелкните правой кнопкой мыши по связи и выберите *"Relationships Properties"*.
 - 1.3. Выберите триггер, который хотите изменить, из списка *"RI actions"*.
 - 1.4. Выберите режим, который Вы хотите задать для правила ссылочной целостности.
 - 1.5. Нажмите *"OK"* для выхода из диалога в диаграмму.

2. Просмотр макрокда шаблона триггера. Войдите в редактор *"Global Trigger Templates..."*. Выберите шаблон триггера для просмотра из списка *"Built-in Trigger Templates"* или из списка *"User Override"*.
3. Вставка макрокоманды в окно кода шаблона
 - 3.1. Находясь в редакторе шаблона триггера или хранимой процедуры, установите курсор в то место, в которое следует вставить макрокоманду, и щелкните кнопкой мыши, чтобы указать точку, в которую будет вставлена макрокоманда.
 - 3.2. Нажмите кнопку *"Macro Toolbox"* для входа в редактор *Macro Toolbox*.
 - 3.3. Найдите ту макрокоманду, которую следует вставить, и щелкните по ней. После команды *Insert Macro ERwin* вставляет макрокоманду в ту точку в окне кода шаблона, в которой был установлен курсор перед входом в *Macro Toolbox*.
 - 3.4. Нажмите кнопку *"Close"* в редакторе *Macro Toolbox*.
 - 3.5. Нажмите *"Close"* для выхода без изменений из редактора
4. Изменение шаблона, связанного с *RI*-триггером
 - 4.1. Откройте редактор *Global Trigger Templates*.
 - 4.2. Выделите тип *RI*-триггера, который следует изменить, в списке, который находится в верхней части редактора *Trigger Template*.
 - 4.3. Нажмите кнопку *"Detach ->"*, чтобы отсоединить тот шаблон, который в настоящий момент связан с выбранным *RI*-триггером.
 - 4.4. Прокручивая список *"Built-in Template"* или *"User Override"*, найдите шаблон, который следует связать с выбранным *RI*-триггером. Выделите имя шаблона, а затем нажмите кнопку *"Attach"* прямо над списком. **ERwin** свяжет выбранный шаблон с триггером и покажет новую комбинацию в окне-списке, который находится в верхней части редактора *Trigger Template*. Можно снова связать исходный встроенный шаблон триггера с выбранным типом *RI*-триггера. Для этого выделите тип триггера в списке и нажмите кнопку *"<- Rebind"*.
5. Создание своего шаблона триггера
 - 5.1. Войдите в редактор *Global Trigger Templates* и выберите встроенный шаблон, который будете настраивать.
 - 5.2. Отредактируйте выбранный шаблон в окне *Template Code*, используя для этого макрокоманды **ERwin** и стандартные клавиши, применяемые при редактировании.
 - 5.3. Закончив редактировать код шаблона, щелкните по окну *"Template Name"* и введите новое имя для шаблона.
 - 5.4. Нажмите кнопку *"<- Add"*, чтобы добавить новый шаблон в список *"User Override"*.
 - 5.5. Нажмите кнопку *"Close"* для выхода из редактора шаблона в диаграмму.

6. Удаление созданного шаблона

6.1. Войдите в редактор *"Trigger Template"* и выберите шаблон, который Вы будете удалять, в списке *"User Override"*, т.е. в списке шаблонов, созданных пользователями **ERwin**.

6.2. Нажмите кнопку *"Delete ->"* для удаления шаблона.

6.3. Нажмите кнопку *"Close"* для выхода из редактора шаблона в диаграмму.

Примечание: нельзя удалить шаблоны, поставляемые в составе **ERwin**.

7. Создание шаблона *RI Type Override*

7.1. Войдите в редактор *Trigger Template* и выберите шаблон, который нужно изменить, из списка *"Built-in Trigger Template"*, так что код шаблона для этого триггера появится в окне *Template Code*.

7.2. Измените исходное имя *"Template Name"* на "говорящее" имя шаблона, который создаете.

7.3. Войдите в окно *Template Code* и измените код шаблона так, чтобы он удовлетворял некоторым требованиям. Например, можно добавить к шаблону новые коды, которые будут автоматически вставлять строку в архивную таблицу каждый раз, когда активизируется этот триггер. Можно использовать макрокоманды из *"Trigger Toolbox"*, чтобы ускорить процесс написания кода.

7.4. Закончив редактирование кода, нажмите кнопку *"Add"*, чтобы добавить шаблон в список *"User Override"*.

7.5. Чтобы присвоить новый шаблон в качестве переопределяющего шаблона, выделите Ваш шаблон, а также встроенный шаблон, который следует переопределить, а затем нажмите кнопку *"Attach"*, которая находится над списком *"User Override"*. **ERwin** заменит встроенный шаблон новым и покажет тип триггера и связанный с ним шаблон в списке наверху редактора *Trigger Template*.

7.6. Нажмите кнопку *"Close"* для выхода из редактора в диаграмму.

Примечание: Чтобы **ERwin** мог использовать шаблоны *RI Type Override* вместо встроенных, включите режим *"RI Type Override"* в редакторе *Schema Generation Report* при генерировании физической схемы базы данных.

8. Создание шаблона *Relationship Override Trigger*

8.1. Выделите линию связи, для которой следует создать новый триггер, нажмите правую кнопку мыши для входа в *pop-up* меню *Editor* и дайте команду *"Relationship Templates"* для входа в редактор *"Relationship Templates"*.

- 8.2. Выберите шаблон, который нужно изменить, из списка *"Built-in Trigger Template"*, так что код шаблона для этого триггера появится в окне *"Template Code"*.
- 8.3. Измените исходное имя *"Template Name"* на "говорящее" имя шаблона
- 8.4. Откройте окно *"Template Code"* и измените код шаблона так, чтобы он удовлетворял Вашим требованиям. Например, можно добавить в шаблон новый код, который будет автоматически изменять значение в строке вместо того, чтобы удалять эту строку. Можно использовать макрокоманды из *"Macro Toolbox"*, чтобы ускорить процесс написания кодов, и просмотреть расширенный код в окне *"Expanded Code"*.
- 8.5. Закончив редактирование кода, нажмите кнопку *"Add"*, чтобы добавить шаблон в список *"User Override"*.
- 8.6. Чтобы присвоить новый шаблон в качестве переопределяющего, выделите Ваш шаблон, а также встроенный шаблон, если он есть, который следует переопределить, а затем нажмите кнопку *"Attach"*, расположенную над списком *"User Override"*. **ERwin** заменяет встроенный шаблон новым и показывает тип триггера и связанный с ним новый шаблон в списке, расположенном наверху в редакторе *Trigger Template*.
- 8.7. Нажмите кнопку *"Close"* для выхода из редактора в диаграмму.

Примечание: Чтобы **ERwin** начал использовать шаблоны *"RI Type Override"* вместо встроенных, включите режим *"Relationship Override"* в редакторе *"Schema Generation Report"* при генерировании физической схемы базы данных.

9. Создание шаблона *Entity Override*

- 9.1. Зайдите в окно *"Triggers"* сущности *Order*, предназначенное только для чтения.
- 9.2. Щелкните по окну *"Trigger"* и введите имя нового шаблона триггера сущности, например *"Special order-line insert"*. Нажмите кнопку *"New"* и добавьте новый шаблон в список шаблонов триггеров сущности.
- 9.3. Щелкните по одному из окон *checkbox "Trigger On"*, чтобы указать, хотите ли Вы создать собственный триггер для *Insert*, *Update* или *Delete*. Когда Вы ставите метку в одно из этих окон, **ERwin** автоматически загружает код встроенного шаблона в окна кодов, которые расположены в нижней части редактора.
- 9.4. Щелкните по окну *Template Code* и измените код шаблона так, чтобы он удовлетворял определенным требованиям. Например, можно добавить в шаблон новый код, который будет автоматически изменять значение в строке вместо того, чтобы удалять эту строку. Можно использовать макрокоманды из *"Trigger"*

Toolbox", чтобы ускорить процесс написания кодов. Чтобы просмотреть расширенный код, щелкните по окну *Expanded Code* и используйте *scrollbars*, рамку окна и (или) кнопку *Maximize*, чтобы увеличить размер окна.

9.5. Нажмите кнопку "OK" для выхода из редактора в диаграмму.

Примечание: Чтобы *ERwin* начал использовать шаблоны "Entity Override" вместо встроенных, включите режим "Entity Override" в редакторе "Schema Generation Report", когда будете генерировать физическую схему базы данных.

При создании триггеров для сущностей и связывании их с конкретными сущностями *ERwin* не может использовать свою стандартную схему для комбинирования разных встроенных шаблонов. Вы должны взять на себя написание кода, который контролирует, каким образом комбинируются шаблоны для этих сущностей. *ERwin* предоставляет набор специальных управляющих макрокоманд, включая макрокоманды, которые просматривают все связи в поисках заданной сущности.

10. Проанализируйте пример триггера *Entity Override*

Ниже приводится исходный код шаблона для создания триггера *Special order-line insert* для сущности *Order*.

```
create trigger %TriggerName on %TableName
for %Actions(",")
as
/* ERwin Builtin %Datetime */
/* %Actions(",") trigger on %TableName */
/* default body for %TriggerName */
begin
    declare @numrows int,
           @nullcnt int,
           @validcnt int,

    %PKDecl(,@ins)%decl(bComma,0)%ForEachAtt() {%if(%AttIsPk) {%=(bComma,1)}
}%if (%==(bComma,1)) {,}
    @errno int,
    @errmsg varchar(255)
    select @numrows = @@rowcount
    %ForEachChildRel() {
        %RelTemplate
    }
    %ForEachParentRel() {
        %RelTemplate
    }
    return
error:
    raiserror @errno @errmsg
    rollback transaction
end
go
```

Код измененного шаблона, включающий в себя специальный код (он выделен), нужный для того, чтобы изменять значение *P_Name* в таблице *Product* каждый раз при добавлении новой строки *O_Name* в таблицу *Order*, показан ниже. Код вставляется сразу же после выражения *%ForEachChildRel*, поскольку сущность *Order* является родительской в связи между *Order* и *Product*. Для того чтобы просмотреть расширенный код после того, как шаблон, заданный по умолчанию, изменен, щелкните по окну *Expanded Code*.

```
create trigger %TriggerName on %TableName
  for %Actions(",")
  as
/* ERwin Builtin %Datetime */
/* %Actions(",") trigger on %TableName */
/* default body for %TriggerName */
begin
  declare @numrows int,
          @nullcnt int,
          @validcnt int,
          %PKDecl(,@ins)%decl(bComma,0)%ForEachAtt() {%if(%AttIsPk)
{%=(bComma,1)} }%if (%==(:%bComma,1)) {,}
          @errno int,
          @errmsg varchar(255)

  select @numrows = @@rowcount
  %ForEachChildRel() {
    %RelTemplate
    %if (%==(:%VerbPhrase,is requested on)){
      update %Parent from %Parent,inserted
      set P_Name = %Parent.P_Name
      +inserted.O_Name
    where
      %JoinFKPK(inserted,%Parent)
    }
  }

  %ForEachParentRel() {
    %RelTemplate
  }
  return
error:
  raiserror @errno @errmsg
  rollback transaction
end
go
```

ERwin автоматически присваивает каждой связи режим ссылочной целостности, устанавливаемый по умолчанию, прежде чем добавить ее в

диаграмму. Режимы *RI*, присваиваемые **ERwin** по умолчанию, могут быть изменены в редакторе *Referential Integrity Default*.

Изменить режимы *RI*, присваиваемые по умолчанию, можно в редакторе *Referential Integrity Default*.

11. Задание режимов *RI* по умолчанию

11.1. Войдите в редактор *Referential Integrity Default* в *Model Properties*

11.2. Задайте нужный режим *RI* по умолчанию для каждого типа действия триггера для каждого из четырех типов связи в **ERwin**.

11.3. Для каждого случая имеется *combobox*, содержащий только допустимые *RI*-режимы для данного действия триггера и для данного типа связи.

11.4. Если Вы хотите изменить *RI*-режимы для уже существующих связей, заменив их на новые, нажмите кнопку *"Rebind"*. **ERwin** попросит Вас подтвердить это. Нажмите *"Yes"* для обновления существующих значений *RI*-триггеров и возвращения в редактор.

11.5. Нажмите кнопку *"Reset"* для присвоения всем *RI*-режимам, присваиваемым по умолчанию, их системных значений. Как и большинство режимов, задаваемых **ERwin** по умолчанию, *RI*-режимы не имеют обратного действия. Они влияют только на новые связи, которые созданы после внесения изменений в *RI*-режимы.

11.6. Закончив работу, нажмите *"OK"* для выхода с сохранением изменений. После того как Вы задали режимы *RI*, присваиваемые по умолчанию каждому типу связи, выбранный *RI*-режим и связанный с ним *RI*-триггер автоматически присваиваются каждой новой связи, когда она добавляется на диаграмму.

Примечание: разные СУБД по-разному поддерживают ссылочную целостность.

Правила ссылочной целостности сообщают СУБД, как обрабатывать изменения данных в одной из таблиц связи. Редактор *Referential Integrity* позволяет Вам задать свою реакцию для каждой ситуации, когда данные добавляются, изменяются или удаляются из одной из таблиц связи.

12. Генерирование *RI*-триггеров

12.1. Войдите в редактор *"Schema Generation Report"*.

12.2. Поставьте метку в *checkbox* *"Trigger"* в групповом окне *Trigger* для того, чтобы задать генерацию всех триггеров, определенных в модели данных, используя встроенные шаблоны *RI*-триггеров.

12.3. Если Вы хотите, чтобы **ERwin** использовал переопределенные (*User Override*) шаблоны триггеров, а не встроенные триг-

геры, поставьте метку в одном или нескольких окнах *"User Override"* для включения режимов генерации триггеров: *"RI Type Override"*, *"Relationship Override"* и (или) *"Entity Override"*. Эти три режима переопределения можно использовать по отдельности или любую их комбинацию для достижения необходимых результатов при генерации триггеров.

- 12.4. Нажмите кнопку *"Preview..."*, чтобы просмотреть код *SQL* до того, как Вы начнете создавать триггеры. Нажмите кнопку *"Generate..."*, чтобы начать генерацию триггеров на сервере.
- 12.5. Когда Вы нажимаете *"Generate..."*, **ERwin** выводит на экран окно для подключения к базе данных, если Вы еще к ней не подключены. Введите, если это нужно, информацию, необходимую для подключения к базе данных.
- 12.6. После того как **ERwin** завершит генерацию всех триггеров, он выводит на экран сообщение, содержащее число созданных триггеров. Нажмите *"OK"*, чтобы закрыть окно-сообщение. После этого нажмите *"Close"* для выхода из редактора в диаграмму.

Контрольные вопросы

1. Что такое триггер?
2. Что такое триггер ссылочной целостности?
3. Что такое макрокоманда?
4. Каким образом используются шаблоны триггеров?
5. Что выполняет триггеры *Restrict* и *Cascade*?

Лабораторная работа №10. Хранимые процедуры

Цель работы: Ознакомиться с хранимыми процедурами в *ERWin*. История модели.

Теоретические сведения

Хранимые процедуры

Хранимой процедурой называется именованный набор прекомпилированных команд *SQL*, который работает точно так же, как и триггер, только он обычно вызывается из другой программы, а не выполняется автоматически, как реакция на событие.

Поскольку триггеры и хранимые процедуры имеют большое значение для ускорения работы и поддержания целостности, **ERwin** располагает специальными редакторами *Trigger* и *Stored Procedure* со встроенными шаблонами и мощными макрокомандами, которые могут существенно ускорить процесс создания этих процедур *SQL*.

Хранимой процедурой называется блок кода *SQL*, подобный триггеру, который хранится на сервере для быстрого выполнения. Хранимая процедура напоминает триггер, с той разницей, что она не откликается на какое-то событие, как *RI*-триггер. Вместо этого она вызывается из другой программы, которая передает на сервер имя процедуры. Хранимой процедуре можно передавать параметры и она может возвращать параметры, значения и сообщения, вызывать другие хранимые процедуры. Различают хранимые процедуры выбора и процедуры действия.

При создании хранимой процедуры можно использовать панель инструментов *Trigger* и предопределенные макрокоманды **ERwin**, чтобы вставить нужный код в окно *Template Code*. После того как код шаблона для хранимой процедуры написан, ее можно связать с отдельной сущностью или со всей схемой. Когда создается в физической схеме базы данных таблица, связанная с сущностью, **ERwin** автоматически расширяет код шаблона и создает все связанные с ним хранимые процедуры с соблюдением синтаксиса *SQL* для конкретной СУБД точно так же, как он создает триггеры, связанные с сущностями и связями **ERwin**.

В отличие от *RI*-триггера, который реагирует на определенное изменение данных в таблице, хранимая процедура может выполнять практически любой тип действий. Эта гибкость является причиной того, что надо задавать шаблон хранимой процедуры с нуля. Не существует встроенных шаблонов хранимых процедур, которые можно было бы использовать как отправной пункт при создании новой хранимой процедуры.

Можно вводить выражения *SQL* непосредственно в окна редакторов шаблонов **ERwin**. Если вводится код *SQL* для хранимых процедур в **ERwin**, то можно использовать возможности просмотра и генерации отчетов в **ERwin** для того, чтобы просмотреть все процедуры, связанные со схемой, а также те, которые связаны с отдельными сущностями. Задавая хранимые

процедуры в **ERwin**, можно управлять этой информацией в одной среде, поэтому легко можно изменять ссылки на схему в кодах шаблона процедуры.

Работа в редакторе *Stored Procedures*

Редактор позволяет создавать, изменять или удалять хранимую процедуру. Список "*Stored Procedure*" содержит имена всех хранимых процедур, связанных с выбранной сущностью.

Для изменения существующей хранимой процедуры щелкните по имени шаблона, который следует изменить, в списке "*Stored Procedure*". Когда **ERwin** покажет на экране код, щелкните по закладке "*Code*" и измените коды, пользуясь встроенными макрокомандами и стандартными клавишами редактирования.

Для удаления ранее определенной процедуры щелкните по имени удаляемого шаблона в списке "*Stored Procedure*", а затем нажмите кнопку "*Delete*".

Для того чтобы создать новую хранимую процедуру, нажмите кнопку "*New*" и введите имя. После этого создайте код шаблона. Можно вводить выражения *SQL* непосредственно в окно "*Code*".

Скрипты "до и после генерации схемы"

Скриптами "до и после генерации схемы" называются скрипты *SQL*, которые **ERwin** выполняет сразу же до или после генерации схемы. Например, когда Вы производите обратное проектирование базы данных из модели **ERwin**, можно создать скрипт "до генерации схемы", который удаляет старую базу данных и создает новую до того, как **ERwin** начнет генерацию таблиц и индексов, определенных в модели данных.

Скрипты уровня схемы связаны со схемой таким же образом, что и хранимые процедуры.

Создание истории модели

Присваивание имен стандартов

В меню *Tools*(инструментальные средства), выберите "*Names*", затем выберите в диалоге "*Model Naming Options*". В этом диалоге вы определите имена стандартов для присвоения текущей модели данных.

Вы можете определить максимальную длину для физических объектов и логических объектов. В меню *Tools* выберите "*Names*", а затем выберите *Edit Naming Standards*. После успешно произведенных действий **ERwin** откроет *Naming Standard Editor*. В этом редакторе можно определить отдельное присваивание имен стандартов для логических и физических объектов.

В окне "*Model Naming Options*" (*Tools/Names*) имеются следующие закладки:

- **General.** Можно использовать имена стандартов по умолчанию. Для этого надо отметить "*Do Not Use Naming Standards File*". Можно

подключить свой файл имен стандартов. Для этого надо выбрать *"Use File"* и нажать на кнопку *"Browser"* для открытия файла имен стандартов.

- **Logical.** Можно установить максимальную длину для объектов. Для этого в колонке *"Maximum Length"* ввести целое число. Можно определить уровень: высокий (*upper*), низкий (*lower*), начальный (*initial*).
- **Physical.** Можно установить максимальную длину для объектов. Для этого в колонке *"Maximum Length"* ввести целое число.

Можно определить уровень: высокий (*IPPER*), низкий (*lower*), начальный (*initial*). При метке *"Allow special characters"* в будут разрешены специальные характеристики, т.е. при высоком уровне имена объектов будут иметь заглавные буквы, при низком уровне – прописные, а при начальном – первая буква будет заглавной, а остальные прописными.

- **Name Mapping** – распределение имени. Назначать имена, моделировать объекты можно с помощью *Macro Toolbox*. **ERwin** включает макроразработки для облегчения задачи и отличия между приложениями в логической и физической типах моделей;
- **Duplicates Names** – реакция на повторные имена. Можно установить следующие предпочтения для повторяющихся имен:
 - *Allow duplicates names* – разрешить повторные имена;
 - *Automatically Rename duplicate names* – автоматически переименовывать повторяющиеся имена;
 - *Ask* – будет предложено переименовать;
 - *Disallow duplicate names* – не разрешаются повторяющиеся имена.

Можно определить процесс осуществления стандартов и создание данных с включением условий договора и сокращения. Для этого надо дать команду *"Edit Naming Standards"* (*Tools/Names*). **ERwin** сохраняет присвоенные имена стандартов информации с добавлением к файлу расширения (*.nsm). Для каждой модели **ERwin**, в которой используется присваивание имен стандартов, нужно подключить присваивание имен стандартов файла. Можно приложить присваивание имен стандартов файла к многочисленным моделям. Когда подключается присваивание имен стандартов файла к модели данных, **ERwin** автоматически прилагает стандарты и соглашения, определенные в файле. Можно использовать файл, чтобы проверить на соответствие имена объектов в модели. **ERwin** использует присваивание имен стандартов файла в качестве словаря и сравнивает имена в модели данных с именами в словаре.

Распределение типов данных

Datatype – встроенный набор характеристик для атрибута или колонки, которые определяют длину области, набор приемлемых символов, дополнительные и необходимые параметры.

В логической модели определяется областью, из которой атрибут наследует свои свойства или из назначаемого типа данных. В физической

модели тип данных определяется значением по умолчанию, которое определяется целевым сервером или назначаемым типом данных. **ERwin** имеет несколько инструментальных средств для автоматического назначения и поддержки согласованности типов данных.

Datatype Standards Editor (стандартный редактор типов данных) может быть использован для редактирования типа данных, заданного по умолчанию. Для физических моделей можно отредактировать тип данных, заданный по умолчанию; **ERwin** автоматически подключается к каждому столбцу. Для логических моделей можно добавить логические типы данных и назначить тип данных для атрибутов в логической модели. При определении стандарта распределения типов данных определяем как логический карту типов данных на типы данных, доступные для вашего целевого сервера. Если есть база данных приложений, работающая на многочисленных платформах сервера, то можно отобразить тип данных для всех целевых серверов. **ERwin** сохраняет тип данных, отображая информацию в файле стандартных типов данных (*.dsm). Для каждой модели **ERwin**, в которой используется стандартный тип данных, надо подключить файл стандартных типов данных (*datatype standards file*). Если при открытии **ERwin** не был указан стандартный тип данных, то **ERwin** использует тип данных по умолчанию.

В меню *Tools* выберите *Datatype*, затем выберите *Model Datatype Options*; можно определить файл распределения типа данных, который хотите отнести к текущей модели.

Можно указать допустимые характеристики моделирования (закладка "General"): "Dimensional" (данные в пространстве, т.е. не смещаются) и "Data Movement" (данные смещаются). Для "Dimensional" можно с помощью метки "Display Conformance warnings" указать: выводить или нет соответствующие ошибки на экран. В блоке "Auto-Transform Logical Objects" (автоматическое преобразование логических объектов) можно указать, чтобы преобразовывалась связь "Many-To-Many" с соответствующими таблицами, типы *Supertype/Subtype* с идентифицирующими связями.

Историю можно сохранять, когда:

- происходит создание в модели (*Created in the model*)
- происходит создание из модели-источника (*Created from a model source*)
- происходит связывание с моделью-источником (*Linked to a model source*)
- при преобразовании (*Transformed*)
- при миграции внешнего ключа (*Migrated from a foreign Key (FK)*)
- при обратном преобразовании из базы данных (*Reverse Engineering from a database*).

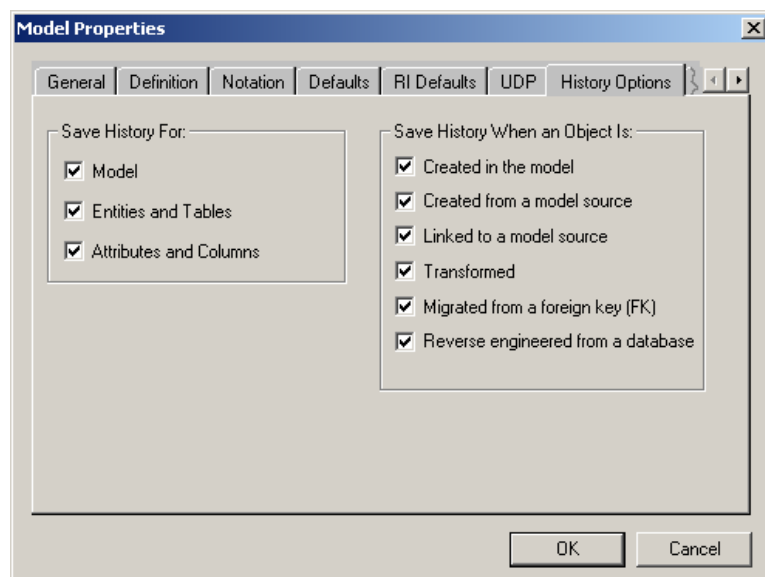


Рис. 3. 66. Диалог свойств модели

Порядок выполнения работы

6. Откройте модель "lab_9.erl", сохраненную в девятой лабораторной работе.

7. Создание хранимой процедуры.



7.1. В *Model Explorer* выполните двойной щелчок мыши по "Stored Procedures" (вид модели должен быть физический).

7.2. Появляется лист связей хранимых процедур *Stored Procedure*. Если СУБД не поддерживает хранимые процедуры, то *Stored Procedure* не появляется. Лист связей хранимых процедур - *Stored Procedure* редактора "Stored Procedures" содержит информацию о том, какая таблица связана с каждой из сущностей **ERwin** и какие хранимые процедуры в настоящий момент связаны с каждой из сущностей и таблиц. В этом редакторе можно просмотреть код шаблона каждой из хранимых процедур, а также связать хранимую процедуру с определенной сущностью или отсоединить ее от сущности.

7.3. Для того чтобы создать процедуру нажмите кнопку *New* и введите имя новой процедуры: "My_procedure".

8. Связывание хранимых процедур с сущностями

Лист связей хранимых процедур позволяет связывать хранимые процедуры с сущностями **ERwin** и соответствующими таблицами базы данных. Два списка в верхней части листа позволяют выбрать сущность или соответствующую ей таблицу, с которой следует связать хранимую процедуру. Список "Attached Table" (редактор *Stored Procedure Browser*) содержит все процедуры, связанные с выбранной сущностью или таблицей. Список "Un-attached Table" содержит все шаблоны хранимых процедур, которые не связаны с выбранной таблицей.

- 8.1. Во вкладке *General* выберите *Table-level*.
- 8.2. Нажмите кнопку *Browser*. Выберите сущность из списка "*Table*", а затем выделите имя шаблона процедуры, которую следует связать с сущностью, и нажмите кнопку "*Browser*". Для создания шаблона для новой процедуры выберите таблицы из "*Unattached Table*" и с помощью кнопок  и  переместите их в поле *Attached Table*.
- 8.3. Если вы хотите, чтобы хранимая процедура была автоматически связана с каждой вновь создаваемой вами сущностью поставьте метку в "*Attach To New Table*".
- 8.4. Нажмите кнопку "*Close*" для выхода.
9. Отсоединение хранимой процедуры от сущности
 - 9.1. Выделите сущность, от которой следует отсоединить хранимую процедуру, и щелкните по ней. **ERwin** автоматически выделяет соответствующую таблицу базы данных.
 - 9.2. Перенесите ее в список "*Unattached Table*".

10. Связывание хранимых процедур со схемой

ERwin позволяет связывать хранимые процедуры не только с отдельными таблицами, но и со всей схемой. Если создана хранимая процедура, которая выполняет административную функцию, например, определяет привилегии пользователей, можно связать процедуру со схемой, а не с отдельными таблицами.

Для создания хранимой процедуры на уровне схемы или для связывания хранимой процедуры со схемой дайте команду меню *Tools "Forward Engineer/Schema Generation..."*. Редактор позволяет просматривать все хранимые процедуры, а также скрипты "до и после генерации схемы", которые связаны со схемой. В редакторе можно просмотреть код шаблона любой процедуры, а также связать процедуру со схемой и отсоединить от нее. Редактор позволяет просматривать код шаблона для выбранной процедуры в окне "*<DB>Preview*" внизу редактора.

11. Создание новой хранимой процедуры или скрипта на уровне схемы

- 11.1. Войдите в редактор *<DB> Schema Generation (Forward Engineer/Schema Generation...)*.
- 11.2. В левом окне выберите "*Schema*". Чтобы **ERwin** автоматически выполнил скрипт перед генерацией схемы, поставьте метку в окно "*Pre-Script*". Чтобы **ERwin** автоматически выполнил скрипт после генерации схемы, поставьте метку в окно "*Post-script*".
- 11.3. Нажмите кнопку "*Preview*". Введите макрокод.
- 11.4. Нажмите кнопку "*Generate*".

Примечание: в редакторе *<DB> Schema Generation Preview* можно изменить хранимую процедуру.

12. Генерирование хранимых процедур
 - 12.1. Войдите в редактор *Schema Generation*.
 - 12.2. Поставьте в окне "*Schema*" метку "*Create Procedure*" для того, чтобы создать все хранимые процедуры, связанные со схемой. Поставьте в окне "*Table*" метку "*Create Procedure*" для того, чтобы создать все хранимые процедуры, связанные с сущностями.
 - 12.3. Для того чтобы выполнить скрипт, который Вы хотите запустить перед генерацией схемы, поставьте метки "*Pre-script - Schema*" или, чтобы выполнить скрипт, который следует запустить после генерации схемы, поставьте метки "*Post script - Table*".
 - 12.4. Нажмите кнопку "*Preview...* ", чтобы просмотреть код *SQL* до того, как Вы начнете создавать хранимые процедуры. Нажмите кнопку "*Generate...* ", чтобы начать генерацию хранимых процедур на сервере.
 - 12.5. Когда Вы нажимаете "*Generate...* ", **ERwin** выводит на экран окно для подсоединения к базе данных, если Вы еще к ней не подсоединены. Введите, если это нужно, информацию, необходимую для подсоединения к базе данных.
 - 12.6. После того как **ERwin** завершит генерацию всех хранимых процедур, он выводит на экран сообщение, содержащее число созданных хранимых процедур. Нажмите "*OK*", чтобы закрыть окно-сообщение. После этого нажмите "*Close*" для выхода из редактора в диаграмму.
13. Сохранение истории модели
 - 13.1. Войдите в редактор "*Model Naming Options*" и установите максимальную длину для атрибутов физической и логической моделей. Уровень шрифта (*CASE*) установите как начальный. В закладке "*Duplicates Names*" установите автоматическое переименование повторяющихся имен.
 - 13.2. Войдите в редактор **ERwin** *Datatype Standards Editor* (*Tools/Datatypes*).
 - 13.3. В редакторе присутствуют колонки: тип данных (*Datatype*), домен **Erwin** (*Erwin Dom*), длина (*Length*), точность (*Precision*), по умолчанию (*Default Len*). Параметры, которые можно изменить, выделены белым цветом.
 - 13.4. В колонке "*Datatype*" введите в качестве нового типа данных "*My_str*", в качестве домена "*String*", длину задайте как "*Optional*". Сохраните измененный файл как "*My_type*". Закройте окно.
 - 13.5. Откройте редактор "*Model Properties*" (*Model/Model Properties...*). Откройте закладку "*General*". В поле "*Author*" (блок

"Model Info") введите свое имя как имя автора модели. Откройте закладку *"Definition"* и введите описание модели. Откройте закладку *"Defaults"*. В блоке *"Default Datatype"* введите значение по умолчанию *"TEXT"* для логической и физической моделей. В закладке *"RI Defaults"* определяются действия при модифицировании типа связи. Откройте закладку *"History Options"*. Сохраните историю для модели, сущностей и таблиц, атрибутов и столбцов, установив соответствующие метки.

Контрольные вопросы

1. Что такое хранимая процедура?
2. Чем процедура отличается от триггера?
3. Что такое скрипт?
4. Каким образом различаются скрипты до генерации и после генерации?
5. Как работать с редактором типов данных?

Лабораторная работа №11. Связывание моделей

Цель работы: Связывание модели процессов и модели данных в *ER-Win*

Теоретические сведения

Соответствие модели данных и модели процессов

После разработки модели данных ее следует связать с моделью процессов. Такая связь гарантирует завершенность анализа, гарантирует, что есть источник данных (сущность) для всех потребностей данных (работа). Связи объектов способствуют согласованности, корректности и завершенности анализа.

Стрелки в модели процессов (*BPwin*) обозначают некоторую информацию, использующуюся в моделируемой системе. В *ERwin* на логическом уровне модели данных информация отображается в виде сущностей (соответствуют таблицам на физическом уровне), состоящих из атрибутов сущностей (соответствуют колонкам таблицы). Сущности состоят из совокупности отдельных записей – экземпляров сущностей (соответствуют записям в таблице). К модели данных предъявляются определенные требования, которые призваны обеспечить компактность и непротиворечивость хранения данных. Основная идея нормализации данных - каждый факт должен храниться в одном месте. Это приводит к тому, что информация, которая моделируется в виде одной стрелки в модели процессов, может содержаться в нескольких сущностях и атрибутах в модели данных. Кроме того, на диаграмме модели процессов могут присутствовать различные стрелки, изображающие одни и те же данные, но на разных этапах обработки (например, необработанные детали – обработанные детали – собранное изделие). Информация о таких стрелках находится в одних и тех же сущностях. Следовательно, одной и той же стрелке в модели процессов могут соответствовать несколько сущностей в модели данных и, наоборот, одной сущности может соответствовать несколько стрелок.

Стрелке в модели процессов может соответствовать отдельная сущность в модели данных. Так, стрелке "Части" на рис. 3.67. соответствует сущность "Часть", стрелке "Конечные продукты" – сущность "Продукт".

Информация о стрелке может содержаться только в нескольких атрибутах сущности. Разным атрибутам одной и той же сущности могут соответствовать разные стрелки. На рис. 3.68. стрелка "Новая часть" соответствует атрибутам "Номер части" и "Название части", стрелка "Наличное количество" – атрибутам "Количество".

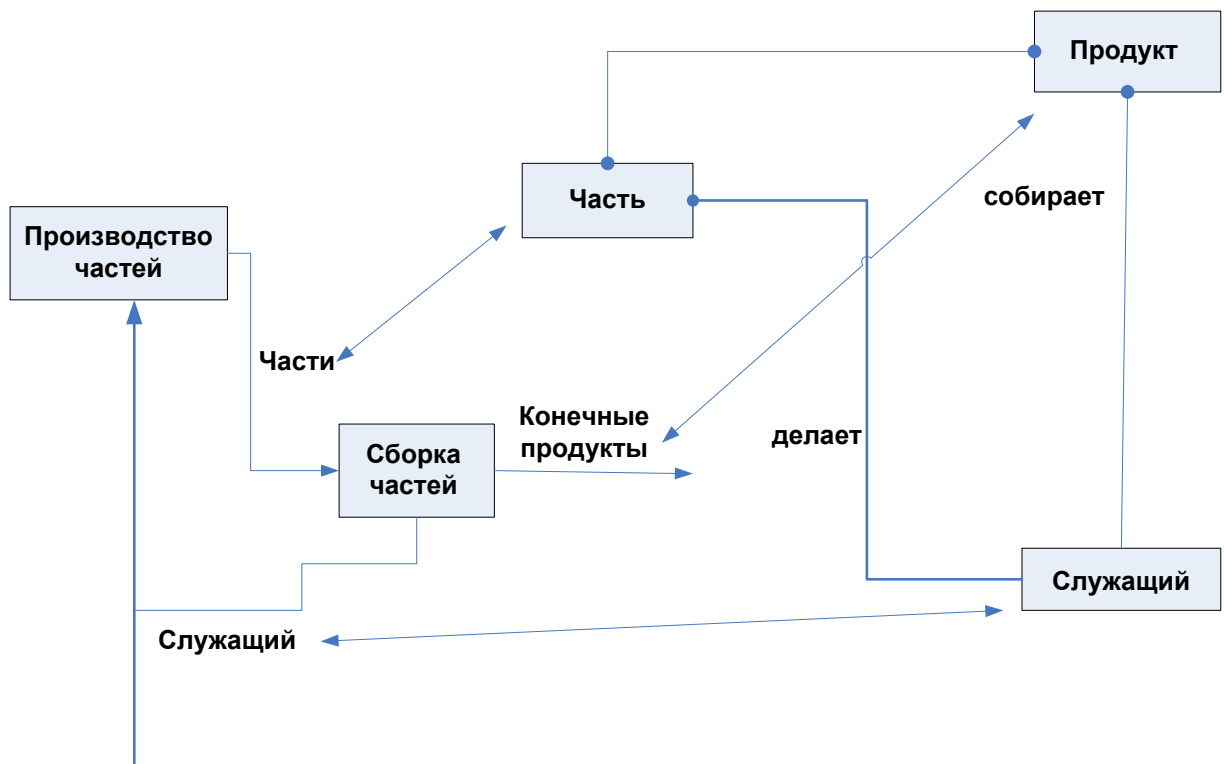


Рис. 3.67. Преобразование стрелки в сущность

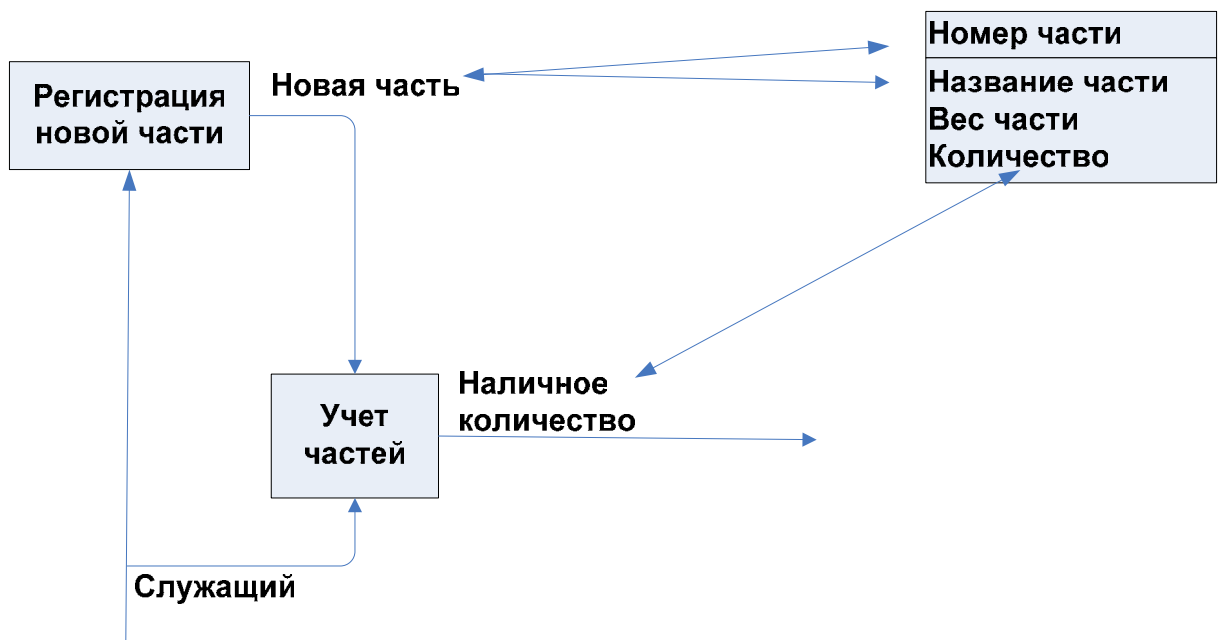


Рис. 3.68. Преобразование стрелки в атрибут

Работы в модели процессов могут создавать или изменять данные, которые соответствуют входящим или выходящим стрелкам. Они могут воздействовать как целиком на сущности (создавая или модифицируя экземпляры сущности, рис. 3.69), так и на отдельные атрибуты сущности (рис. 3.70).

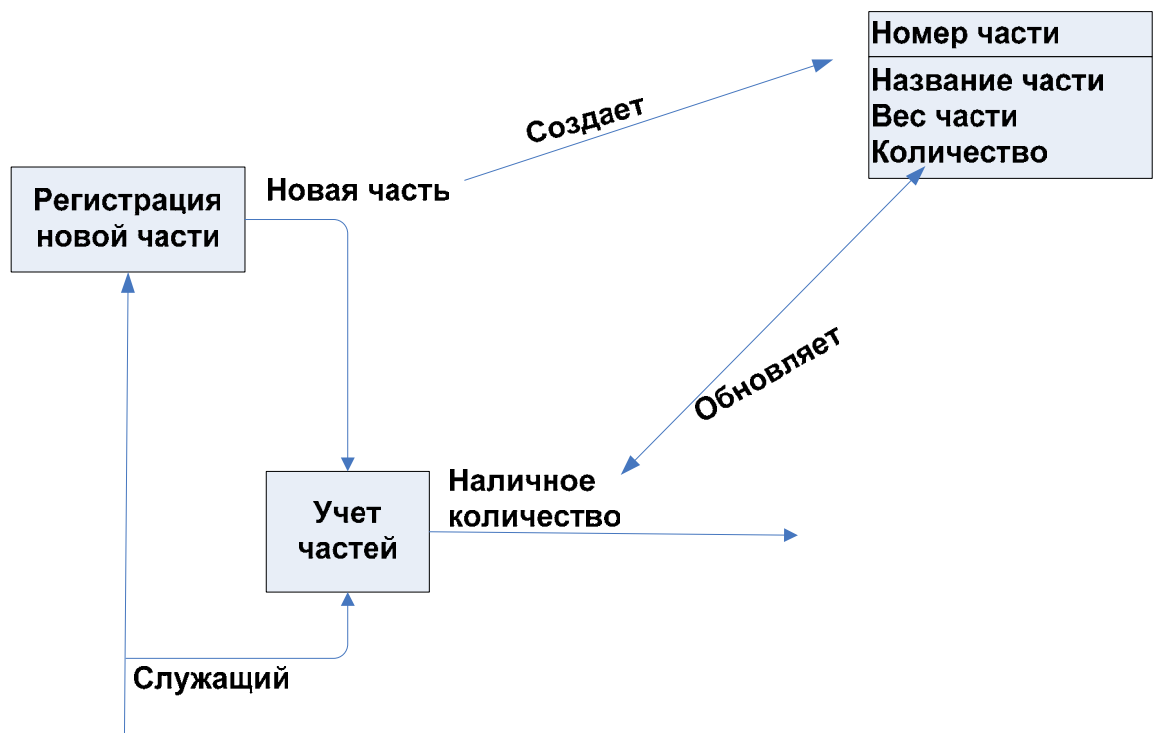


Рис. 3.69. Воздействие работы на сущность

BPwin позволяет связывать элементы модели данных, созданной с помощью **ERwin**, документировать влияние работ на данные и, тем самым, позволяет создать спецификации на права доступа к данным для каждого процесса (см. ниже).

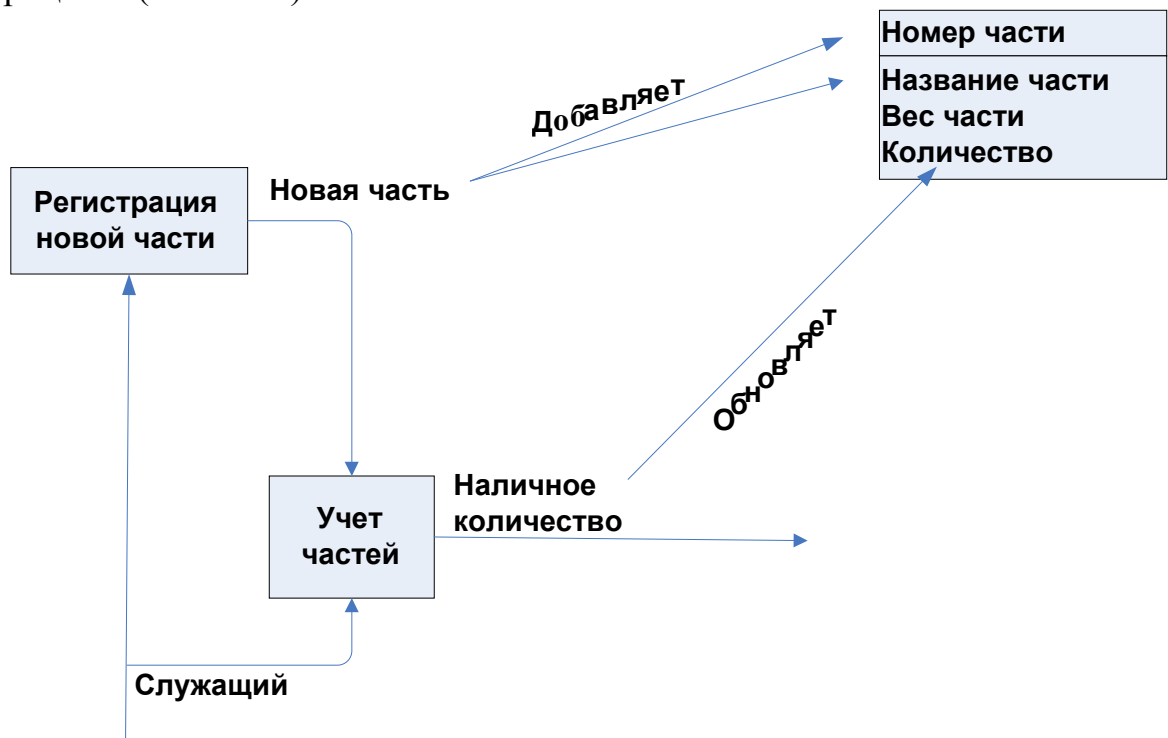


Рис. 3.70. Воздействие работы на атрибут

Экспорт данных из *ERwin* в *BPwin* и связывание объектов модели данных со стрелками и работами

Первым шагом связывания модели данных и модели процессов является экспорт данных из ***ERwin*** в ***BPwin***.

Существует три способа связывания объектов модели данных и модели процессов:

1. Экспорт через *.DBF*-файлы (реализован в ранних версиях ***ERwin*** и ***BPwin***).
2. Экспорт и импорт через файлы формата *.EAX* и *.BPX*.
3. Синхронизация моделей, хранящихся в репозитории *ModelMart* при помощи утилиты *ModelMart Synchronizer*.

Ниже будет рассмотрен второй способ связывания моделей. Для экспорта модели данных из ***ERwin*** в ***BPwin*** необходимо в ***ERwin*** открыть модель и выбрать пункт меню *File/Bpwin/Export*. В появившемся диалоге необходимо выбрать имя файла **.eax* и нажать *OK*.

Затем в ***BPwin*** нужно открыть модель процесса, выбрать в меню пункт *File/Import/Erwin (EAX)*, выбрать имя файла и нажать *OK*. Появится протокол импорта (рис. 3.71). Для внесения данных в модель процесса следует щелкнуть по кнопке *Accept*.

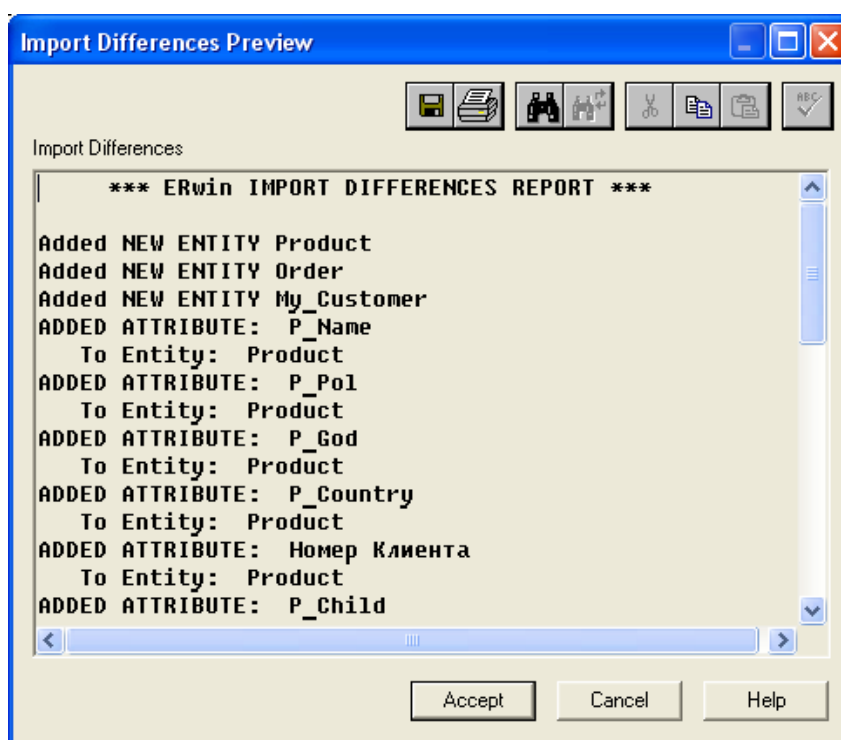


Рис. 3.71. Окно диалога *Import Differences Preview*

После внесения данных в модель процессов можно связать сущности и атрибуты со стрелками. Правой кнопкой мыши нужно щелкнуть по стрелке и выбрать в контекстном меню *Arrow Data*.

Появляется закладка *Arrow Data* диалога *Arrow Properties*.

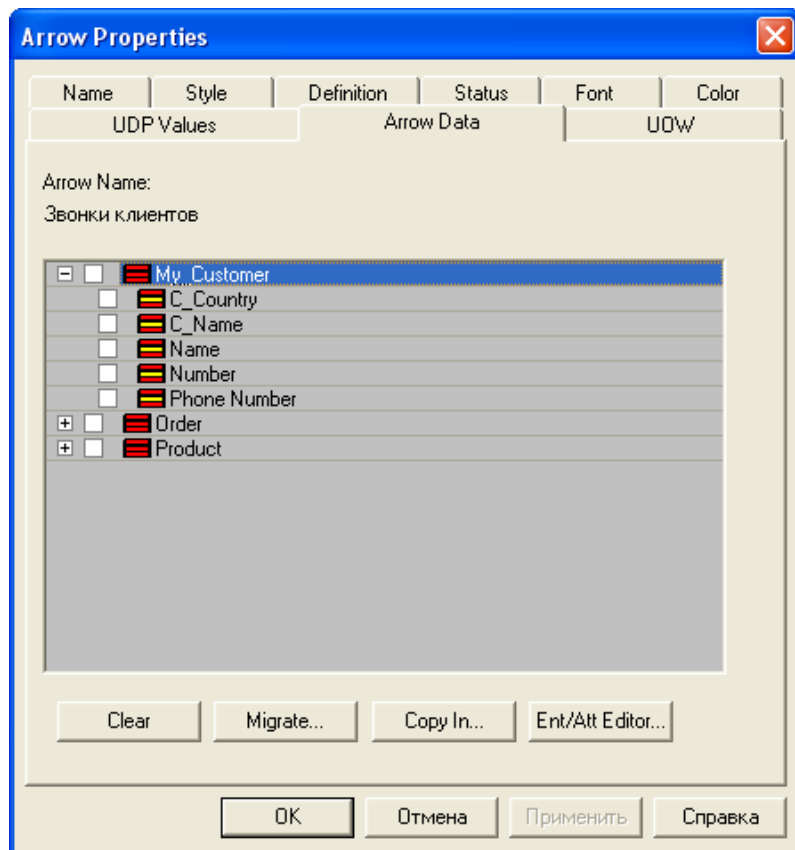


Рис. 3.72. Закладка *Arrow Data* диалога *Arrow Property*

Для связывания атрибута со стрелкой достаточно щелкнуть по иконке выбора в иерархическом списке атрибутов. При этом сущность автоматически связывается со стрелкой. Каждая стрелка в модели процессов может быть связана с несколькими атрибутами различных сущностей:

- Кнопка *Copy In* позволяет копировать связанные данные из другой стрелки.
- Кнопка *Clear* – все связи стрелки с данными.
- Кнопка *Migrate* вызывает диалог *Changes to Arrow Data Associations*, в котором отображаются данные, мигрирующие от дочерних к родительским стрелкам (для разветвляющихся и сливающихся стрелок). При миграции возможны изменения связывания данных:
 - *Deletions* – если данные связаны с родительской стрелкой, но не связаны с дочерней, связи с родительской стрелкой удаляются;
 - *Additions* – если данные связаны с дочерней стрелкой и не связаны с родительской, добавляется связь с родительской стрелкой.

Для подтверждения изменений в диалоге *Changes to Arrow Data Associations* следует щелкнуть по кнопке *Commit*. Миграция возможна только в моделях *IDEF0* и *DFD*.

Как было указано выше, работы могут воздействовать на данные. Для документирования такого воздействия необходимо щелкнуть правой кнопкой мыши по работе и выбрать пункт меню *Data Usage Editor* (рис. 3.73).

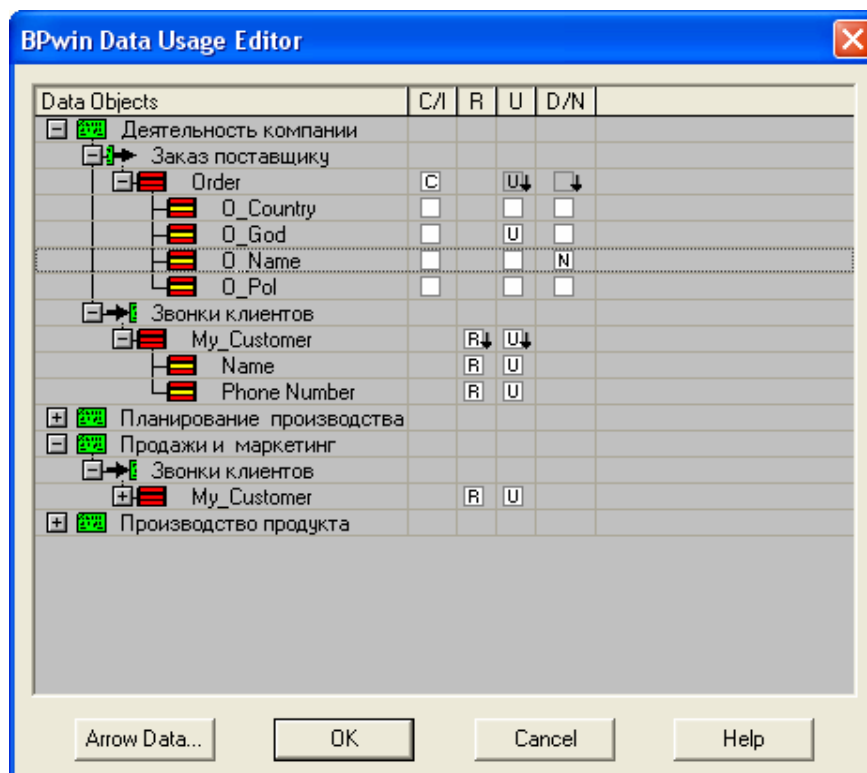


Рис. 3.73. Диалог *BPwin Data Usage Editor*

В появившемся диалоге *Data Usage Editor* в виде иерархического списка показываются все работы модели, стрелки, которые касаются работ, сущности и атрибуты, которые были связаны со стрелками. В верхнем списке нужно щелкнуть по имени стрелки, с которой были связаны сущности и атрибуты. Для задания ассоциации достаточно выбрать соответствующую ассоциацию в правой части окна.

Для сущностей задается ассоциация *CRUD* (*Create, Read, Update, Delete*), для атрибутов – *IRUN* (*Insert, Read, Update, Nullify*). Ассоциации *CRUD* и *IRUN* – это правила использования сущностей и атрибутов работами, т. е. то, что могут делать работы с входящими или исходящими данными. Данные не могут использоваться работами произвольно. Стрелки входа представляют данные, которые работа преобразует в выход или потребляет. Такие данные могут быть обновлены (*Update*) или удалены (*Delete*), но не могут быть созданы (*Create*). Данные, связанные со стрелками управления, могут быть только прочитаны (*Read*), но не могут быть изменены - процедуры и стратегии не могут изменяться в работе. Данные, связанные со стрелками выхода, могут быть обновлены (если им соответствуют данные стрелок входа), удалены (*Delete*) или созданы (*Create*). Для стрелок механизма ассоциации не устанавливаются.

Результат связывания объектов модели процессов можно отобразить в отчете *Data Usage Report* (меню *Report/Data Usage Report*). Ниже приведен пример такого отчета.

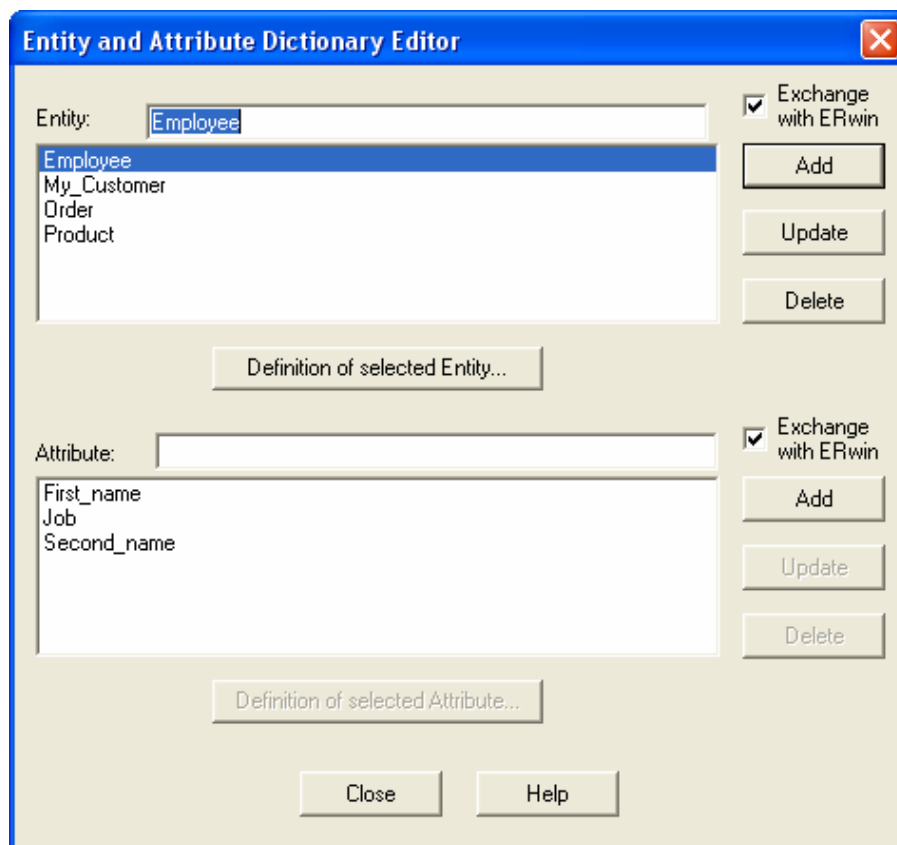
Создание сущностей и атрибутов в **BPwin** и их экспорт в **ERwin**

| <i>Arrow Name</i> | <i>Entity Name</i> | <i>C R U D</i> | <i>Attribute Name</i> | <i>I R U N</i> |
|-------------------|--------------------|----------------|-----------------------|----------------|
| Детали | Часть | RUD | Вес части | RUN |
| | | RUD | Количество | RUN |
| | | RUD | Название части | RU |
| | | RUD | Номер части | R |

Если в процессе связывания стрелок с объектами модели данных окажется, что каких-либо сущностей или атрибутов не хватает, их можно добавить прямо в **BPwin**, а затем экспортировать в **ERwin**.

Для редактирования сущностей и атрибутов следует выбрать пункт меню *Edit/Entity/Attribute Dictionary*. Появляется диалог *Entity and Attribute Dictionary* (рис. 3.11.8.).

Диалог *Entity and Attribute Dictionary* имеет два списка – в верхнем показывааются сущности, в нижнем – атрибуты. Для создания новой сущности следует в верхнем поле *Entity* задать имя сущности (на рис. 3.74. – "*Employee*") и щелкнуть по кнопке *Add*. Сущность будет добавлена в список. Если включить опцию **BPwin** only, созданная сущность при экспорте не будет передана в **ERwin**. Кнопки *Delete* и *Update* служат соответственно для удаления и обновления сущности. Каждой сущности можно дать определение (кнопка *Definition of selected Entity*).

Рис. 3.74. Диалог *Entity and Attribute Dictionary*

Список атрибутов отображается в нижнем окне. Полностью атрибуты создаются и редактируются аналогично.

После описания сущностей и атрибутов следует щелкнуть по кнопке *Close*.

Для экспорта данных в **BPwin** следует выбрать меню *File/Export/ ERwin(BPX)* и указать файл, в который будет выгружена информация о модели.

В **ERwin** следует выбрать меню *BPwin/Import* и указать файл *BPX*, в который была выгружена информация о модели.

Возникает диалог **ERwin/BPwin Import** (рис. 3.75), в котором отображаются:

- сущности, имеющиеся в модели **ERwin**, но отсутствующие в *BPX*-файле;
- сущности, имеющиеся в *BPX*-файле, но отсутствующие в модели **ERwin**;
- сущности, имеющиеся в *BPX*-файле, и соответствующие им сущности в модели **ERwin**, а также действия по синхронизации, которые будут проводиться **ERwin**.

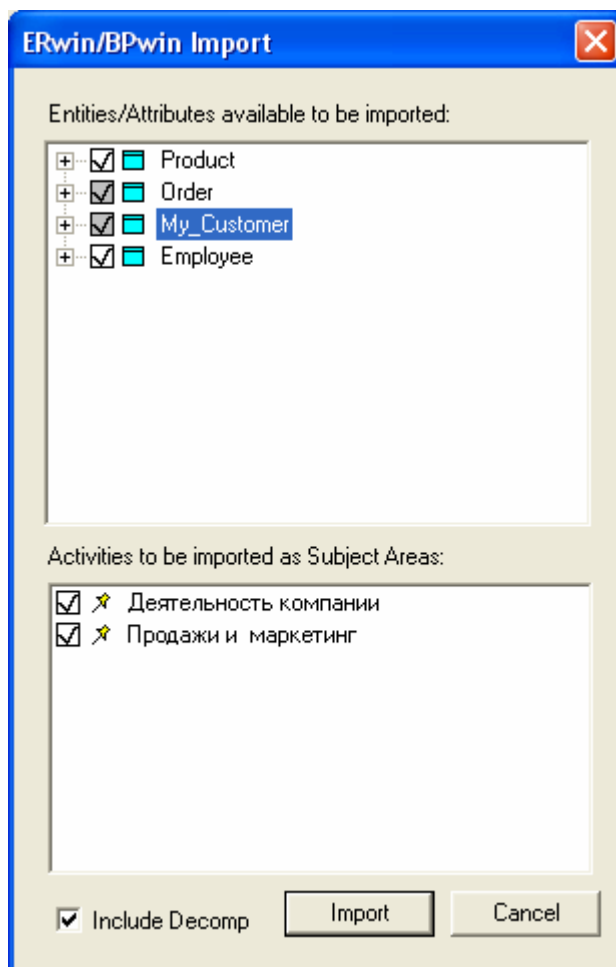


Рис. 3.75. Диалог **ERwin/BPwin Entity Editor**

В примере на рис. 3.75. сущность "*Employee*" будет импортирована из *BPX*-файла в модель *ERwin*.

Импортированная сущность (на рис. 3.76 – сущность "*Employee*") не имеет первичного ключа и не связана с другими сущностями. Назначение атрибутов первичным ключом и связывание сущностей можно провести только средствами *ERwin*; другими словами, сущности и атрибуты, созданные в *BPwin* и затем импортированные в *ERwin*, можно рассматривать как заготовку для создания полноценной модели данных, а не как готовую модель.

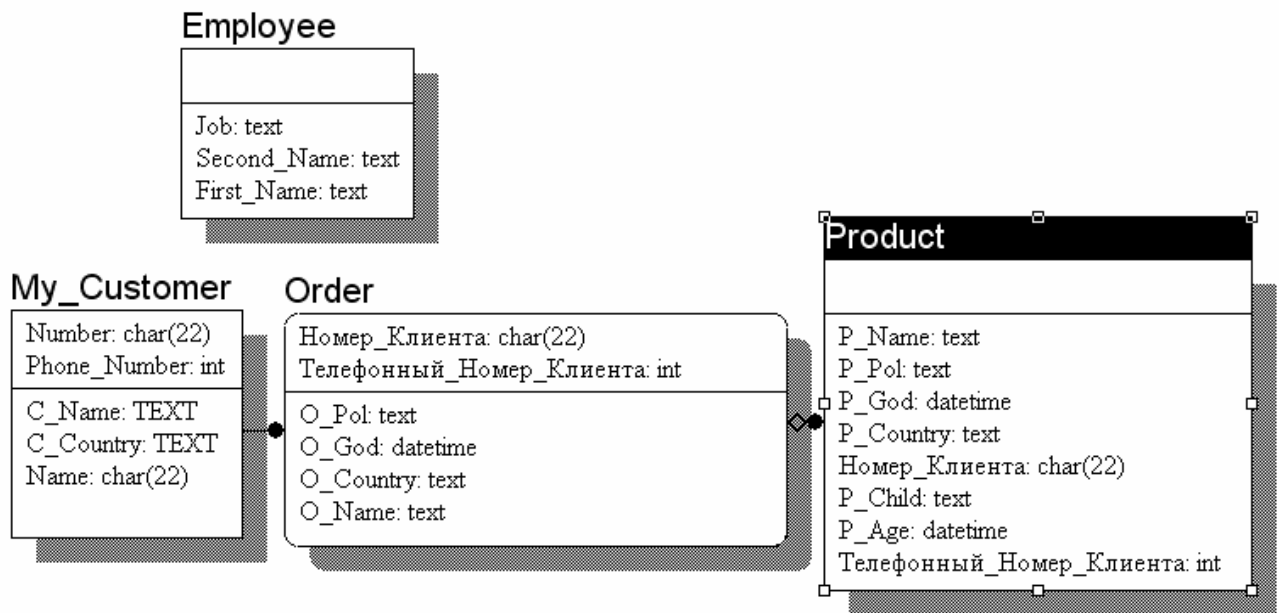


Рис. 3.76. Модель данных после импорта сущности "*Employee*"

Порядок выполнения работы

Выполните связку модели данных и модели процессов, основываясь на теоретическую часть данной лабораторной работы. Свяжите данные в *ERwin* и работы в *BPwin*.

Контрольные вопросы

1. Как экспортируются и импортируются данные между *BPWin* и *ERWin*?
2. Как связать сущности и атрибуты с процессами?
3. Что такое *CRUD* и *IRUN*?
4. Каким образом можно создать новую сущность в *BPWin*?
5. Как создать отчет по получившейся общей модели?

4. ПОРЯДОК ОТЧЕТНОСТИ ПО ЛАБОРАТОРНЫМ РАБОТАМ

Для выполнения последующего упражнения необходимо иметь результат выполнения предыдущего, поэтому следует сохранять модель, полученную в конце каждой лабораторной работы. Эта модель показывается преподавателю вместе с отчетом по проделанной работе.

Для защиты лабораторной работы требуется:

1. Отчет о проделанной работе, включая модель.
2. Знание теоретической части материала лабораторной работы.
3. Умение отвечать на контрольные вопросы.

5. ЗАДАНИЕ НА ИНДИВИДУАЛЬНУЮ РАБОТУ

1. Разработать модель бизнес-процессов организации (*BPWin*)

- Спроектировать функциональную структуру организации (глубина декомпозиции – 3, ширина – 5);
- Построить диаграммы узлов, *FEO* и *IDEF-3* диаграммы;
- Провести стоимостной анализ;
- Создать модель *TO-BE* (реинжиниринг процессов);
- Построить *DFD* – диаграмму;
- Оформить отчет о проделанной работе в виде системного проекта.

Примеры: киоск "Роспечатать", киоск "Овощи и фрукты", аптечный киоск, отдел продажи сотовых телефонов, видео-прокат, отдел продажи *DVD* и т.д.

2. Разработать информационную модель организации (*ERWin*)

- Спроектировать логическую модель базы данных (8-10 сущностей);
- Провести индексирование таблиц;
- Создать отчет;
- Оформить отчет о проделанной работе в виде технического проекта.

Примеры: юридическая фирма, туристическая компания, университет, библиотека, товарный склад, салон красоты, компания сотовой связи и т.д.

Замечание: Желательно, чтобы проектирование бизнес-процессов и создание информационной модели осуществлялось в комплексе для одной организации.

МОДЕЛЬ ПРЕДПРИЯТИЯ: "ОТДЕЛ ПРОДАЖ СОТОВЫХ ТЕЛЕФОНОВ"

Анализ первичных требований и планирование работ.

Данный этап предваряет инициацию работ над проектом. Его основными задачами являются: анализ первичных бизнес требований, предварительная экономическая оценка проекта, построение план-графика работ, создание и обучение совместной рабочей группы.

В данной работе необходимо решить следующие задачи:

- Общую схему организации и работы салона сотовой связи;
- Провести стоимостной анализ;
- Выявить “узкие” места;
- Сделать предложения по улучшении бизнеса;
- Скорректировать общую концепцию деятельности компании.

Длительность проекта такого объема составляет 3-4 недели.

Проведение исследования деятельности предприятия

В рамках данного этапа осуществляется:

- Предварительное выявление требований, предъявляемых заказчиком;
- Определение оргштатной и топологической структур предприятия;
- Определение перечня целевых задач;
- Анализ распределения функций по подразделениям и сотрудникам;
- Определения перечня применяемых на предприятии средств автоматизации.

При этом выявляются функциональные деятельности каждого из подразделений и функциональные взаимодействия между ними, информационные потоки внутри подразделений и между ними, внешние по отношению к предприятию объекты и внешние взаимодействия.

В качестве исходной информации при проведении исследования и выполнении дальнейших этапов служат:

- Данные по оргштатной структуре предприятия.

Описываемый отдел продаж сотовых телефонов состоит из трех отделов: продаж телефонов, технических работ, склада.

Штат сотрудников составляет 6-8 человек.

Отдел продаж сотовых телефонов располагает 2-3 помещениями. Одно представляет собой салон продаж и остальные технические помещения.

Штатное расписание:

- 2 продавца консультанта;
- 2-3 технических работника;

- 1 кладовщик;
- директор.
- Информация о принятых технологиях деятельности.

В отделе функционирует приобретенная система учета продаж, которая включает два модуля: бухгалтерский и материальный учет.

Нормативные документы: правила работы с клиентами, продажи телефонов, хранения. Отметим, что эти правила существуют больше для "наличия" и сотрудники не всегда их придерживаются (см. модель "как есть").

- Стратегические цели и перспективы развития:
 - Развить бизнес до специализированного магазина.
 - Увеличить оборот в 4 раза.

Длительность обследования составляет 1-2 недели. По окончании обследования строится и согласуется с заказчиком предварительный вариант функциональной модели предприятия, включающей идентификацию внешних объектов и информационных взаимодействий с ними, а также детализацию до уровня основных деятельности предприятия и информационных связей между этими деятельностью.

Построение модели деятельности предприятия.

Теоретические положения.

На данном этапе осуществляется обработка результатов обследования и построение моделей деятельности предприятия следующих двух видов.

- Модель *AS-IS*.

Данная модель представляет собой "снимок" положения дел на предприятии (организационная структура, взаимодействия подразделений, принятые технологии, автоматизированные и неавтоматизированные процессы, бизнес-процессы т.д.) на момент обследования и позволяющей понять, что делает и как функционирует данное предприятие с позиций системного анализа. Модель позволяет на основе автоматической верификации выявить ряд ошибок и узких мест, а также сформулировать ряд предложений по улучшению ситуации.

- Модель *TO-BE*.

Эта модель интегрирует перспективные предложения руководства и сотрудников предприятия, экспертов и системных аналитиков и позволяет сформировать видение новых рациональных технологий работы компании.

Переход от модели "как есть" к модели "как должно быть" осуществляется двумя способами.

1. Совершенствование технологий на основе оценки их эффективности. При этом критериями оценки являются стоимостные и временные затраты выполнения бизнес-процессов, дублирование и противоречивость выполнения отдельных задач бизнес-процесса, степень загруженности сотрудников.

2. Радикальное изменение технологий и переосмысление бизнес-процессов. Например, существует возможность вообще исключить бизнес-процесс, если потери от его отсутствия не составят существенных затрат.

ОПИСАНИЕ МОДЕЛИ "КАК ДОЛЖНО БЫТЬ"

Диаграмма верхнего уровня (А-0).

На диаграмме верхнего уровня обычно присутствует одна главная работа, которая представляет собой все предприятие в целом.

Таблица А.1

Список работ диаграммы верхнего уровня

| Название работы | Описание работы |
|--|---|
| Работа магазина по продаже сотовых телефонов | Работа представляет собой все функции магазина в целом. |

Таблица А.2

Список стрелок диаграммы верхнего уровня

| Название стрелки | Описание стрелки | Работа |
|-----------------------------------|---|--|
| Правила работы и торговли | Стрелка управления. Работа "Работа магазина по продаже сотовых телефонов" выполняется в соответствии с законом "О защите прав потребителей". | Работа магазина по продаже бытовой техники |
| Поставки телефонов, комплектующих | Стрелка входа. Телефоны поступают от поставщика в магазин. | |
| Звонки, обращения клиентов | Стрелка входа. Обращения клиентов принимаются продавцами магазина. | |
| Персонал магазина | Стрелка механизма. Персонал магазина осуществляет всю работу. | |
| Проданные телефоны | Стрелка выхода. Проданная бытовая техника. | |
| Пакеты подключения, SIM карты | Стрелка входа. SIM карты для подключения телефонов к сети оператора сотовой связи | |

Диаграмма верхнего уровня (А-0) приведена на рис. А.1.



Рис. 1. Диаграмма декомпозиции A-0

Диаграмма декомпозиции A0.

Эта диаграмма декомпозирует работу "Деятельность магазина по продаже сотовых телефонов".

Таблица A.3

Список работ диаграммы A0

| Название работы | Описание работы |
|--------------------|--|
| Продажа телефонов | Продажа телефонов клиентам, их консультирование |
| Технические работы | Технический сервис |
| Получение хранение | Получение, проверка документов, хранение телефонов |

Таблица A.4

Список стрелок диаграммы A0

| Название стрелки | Описание стрелки |
|-----------------------------------|--|
| Телефоны на тестирование | Телефоны после получения на предпродажное тестирование |
| Телефоны для продажи | Протестированные телефоны со склада на реализацию |
| Подготовленные к продаже телефоны | Телефоны после предпродажного тестирования на склад |
| Продавцы | Исполнители работ |
| Кладовщик | |

Диаграмма декомпозиции A0 приведена на рис. A.2.

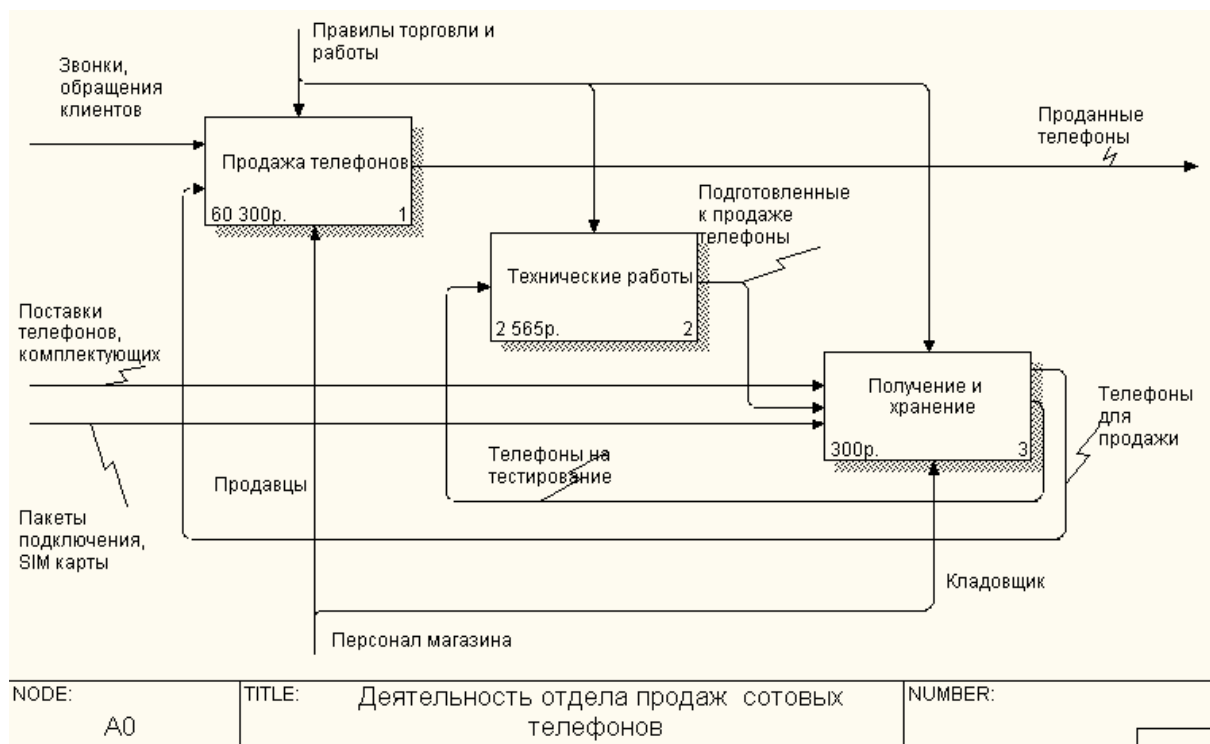


Рис. А.2. Диаграмма декомпозиции А0

Основные работы деятельности отдела продаж сотовых телефонов следующие: продажа телефонов, технические работы, получение и хранение. Важно отметить, что между "Техническими работами" и "Получение и хранением" существует круговая связь. Самой важной является работа "Продажа телефонов".

Диаграмма декомпозиции А2.

Эта диаграмма декомпозирует работу "Технические работы".

Таблица А.7

Список работ диаграммы А2

| Название работы | Описание работы |
|--|--|
| Распаковка телефонов, проверка комплектующих | Подготовка телефонов к предпродажному тестированию |
| Тестирование телефонов, комплектующих | Тестирование телефонов |
| Упаковка, наклейка этикеток | Упаковка, наклейка продажных этикеток |

Таблица А.8

Список стрелок диаграммы А2

| Название стрелки | Описание стрелки |
|--------------------------|---|
| Телефоны на тестирование | Телефоны после распаковки на предпродажное тестирование |
| Проверенные телефоны | Телефоны после предпродажного тестирования на склад |
| Диспетчер | Исполнители работ |
| Тестер | |

Диаграмма декомпозиции приведена на рис. А.3.

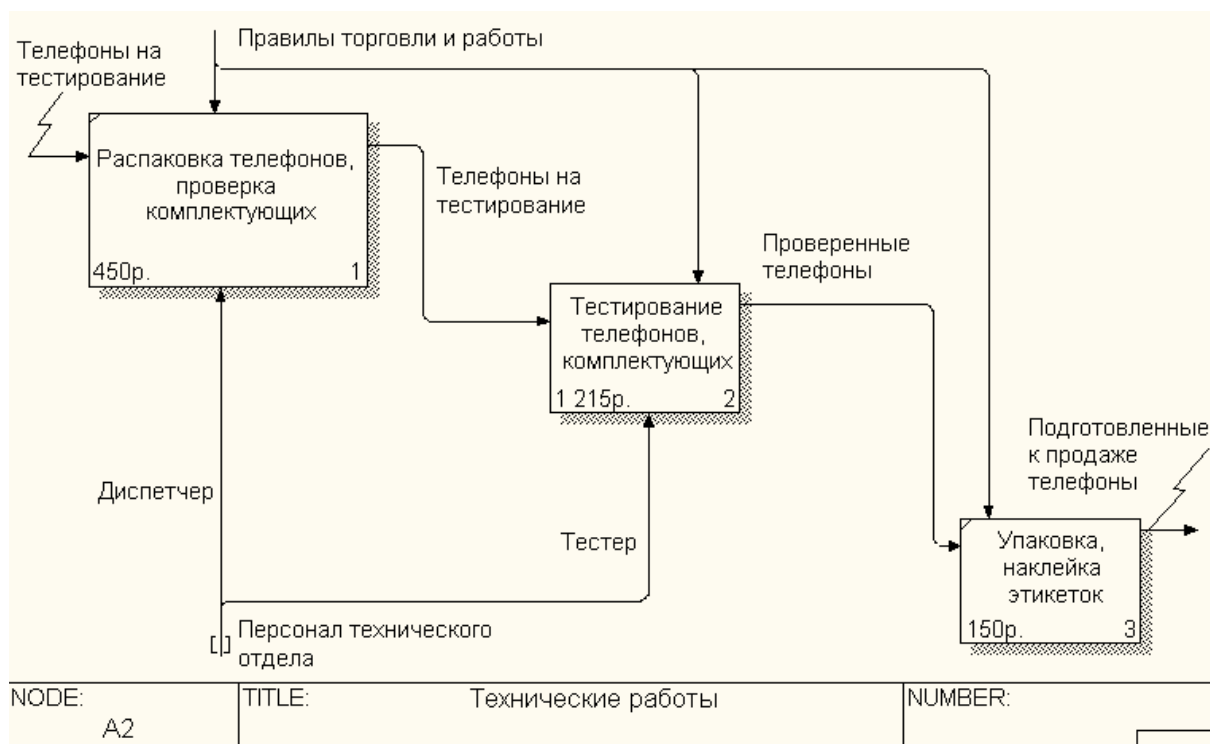


Рис. 3. Диаграмма декомпозиции A2

Диаграмма декомпозиции A3

Эта диаграмма декомпозирует работу "Получение, хранение".

Таблица А.9

Список работ диаграммы декомпозиции A3

| Название работы | Описание работы |
|-------------------------------|-----------------------------------|
| Получение, проверка накладных | Получение телефонов от поставщика |
| Хранение | Хранение телефонов |

Таблица 10

Список стрелок диаграммы A3

| Название стрелки | Описание стрелки |
|---------------------|-------------------|
| Полученные телефоны | Телефоны на склад |

Диаграмма декомпозиции приведена на рис. А.4.

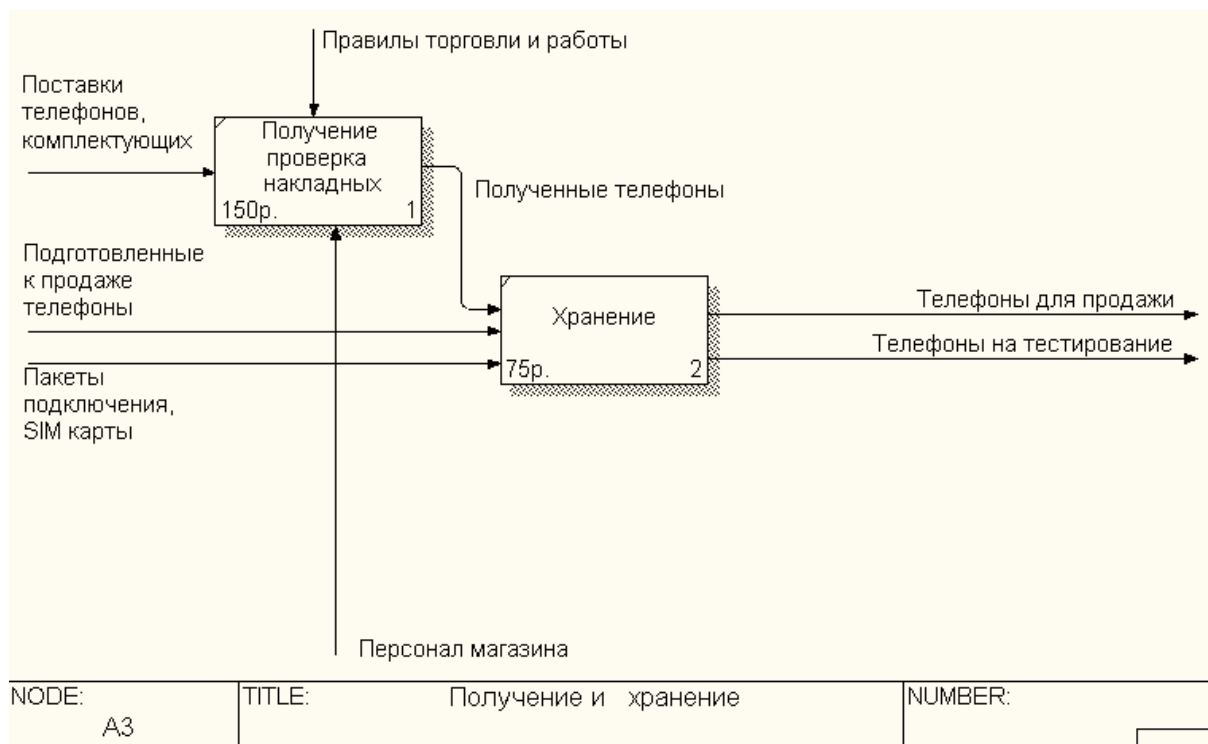


Рис. А.4. Диаграмма декомпозиции А3

Результаты анализа модели "как есть".

Основные для деятельности работы (технические работы) отработаны и организованы для успешного функционирования бизнеса. Недочетов в рамках принятой концепции ведения бизнеса до обращения в консалтинговую фирму нет.

Существуют недочеты в организации работы "Получение и хранение". А именно не документируется выдача телефонов менеджерам. Выдача телефонов должна быть выделена в отдельную работу и, соответствующим образом, документироваться. Эта деталь не заметна, пока бизнес не достиг особого развития, но она серьезно может сказаться в дальнейшем.

В данной ситуации будет разумным применить реинжиниринг процессов и, что самое важное, изменить концепцию работы компании. Смысл новой концепции состоит в следующем: "идти на встречу клиенту". Сделать все, чтобы клиент чувствовал себя комфортно.

На основе этого консалтинговая фирма предложила ввести новую услугу. "Закачка мелодий, игр, картинок" на телефон клиента при покупке телефона. Кроме того, эта услуга сейчас довольно популярна и будет окупать себя. Ее можно оказывать и покупателям телефонов в компании и другим клиентам. Также эта работа не потребует особых финансовых затрат, так вся техническая база уже имеется.

Модель "как должно быть"

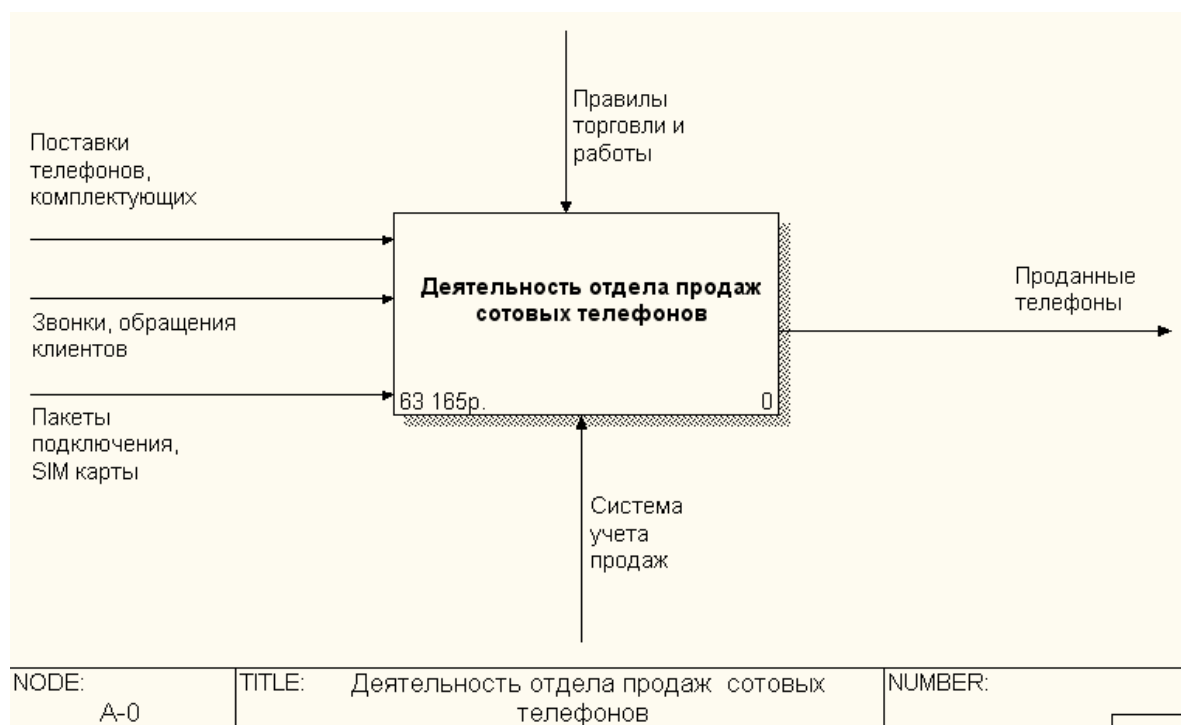


Рис. А.5. "Диаграмма А-0"

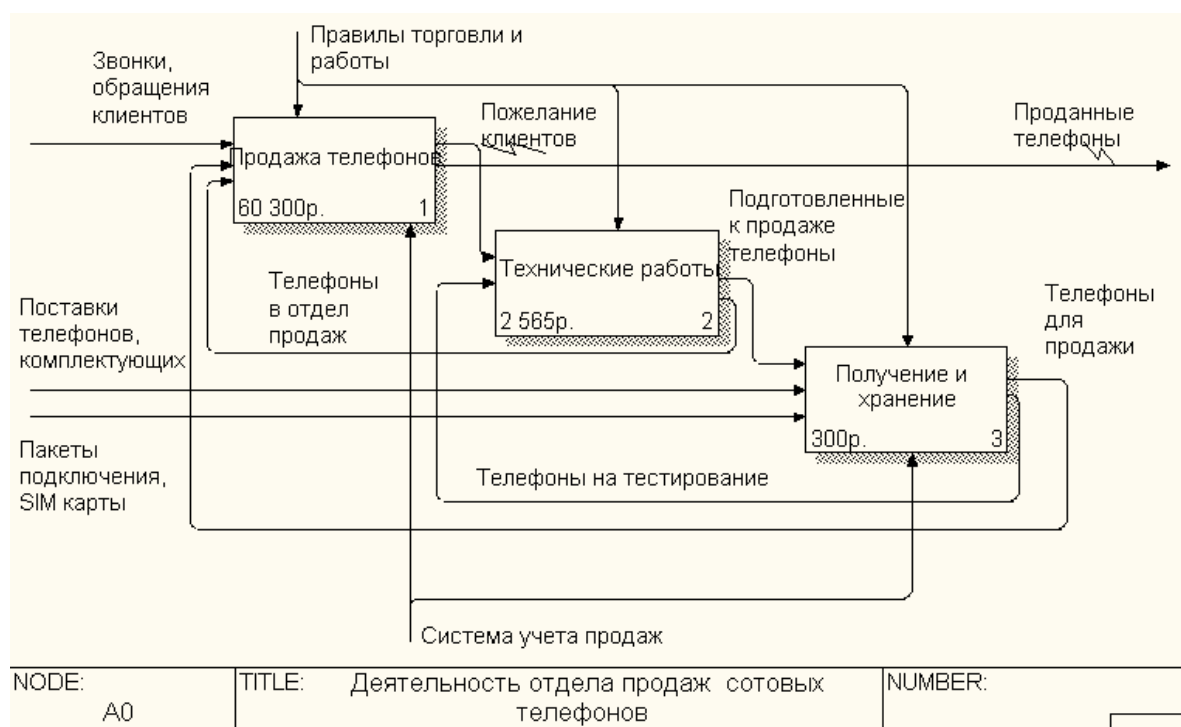


Рис. А.6. "Диаграмма декомпозиции А0"

Здесь мы добавили стрелку "Пожелания клиентов". Она отражает услугу "Установка ПО, игр, картинок по заказу клиента". Это сделано в соответствии с принятой концепцией.

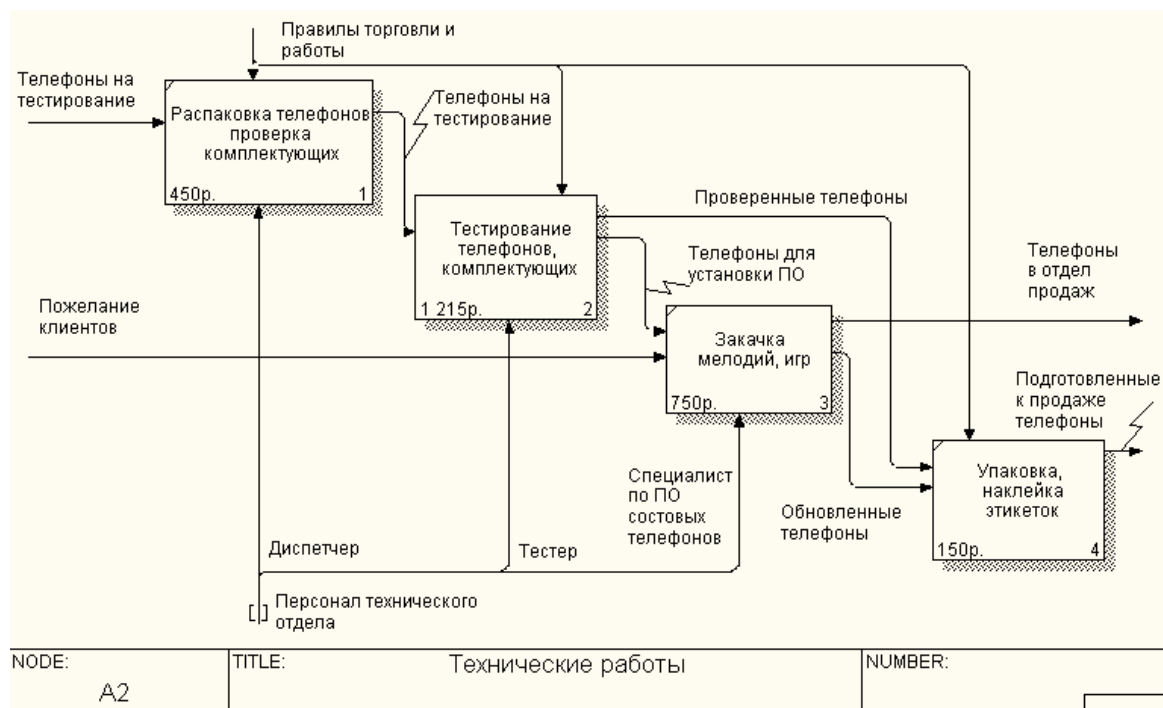


Рис. А.7. "Диаграмма декомпозиции А2"

"Закачка мелодий, игр" выделено в отдельную работу. От этой работы стрелка "Телефоны в отдел продаж" идет на вход работы "Продажа телефонов". Также появился исполнитель этой работы "Специалист по ПО сотовых телефонов".

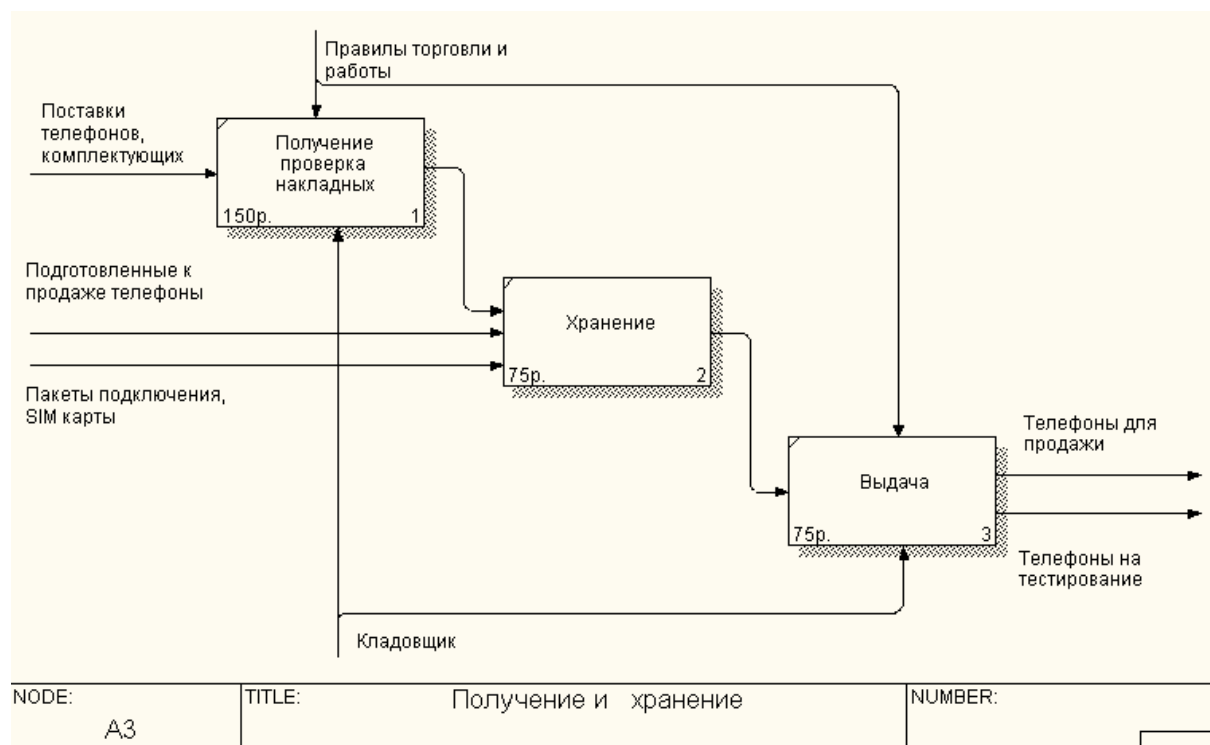


Рис. А. 8. "Диаграмма декомпозиции А3"

Реорганизация работы "Хранение и получение".

"Выдача" выделено в отдельную работу. Это важно для перспективы развития компании. Также эта работа отражает выдачу телефонов в системе учета продаж.

Стоимостной анализ

Стоимостной анализ проведен исходя из следующих начальных данных:

1. Средняя цена телефона от поставщика составляет 4000 руб.
2. Компания в среднем продает по 15 телефонов в день.
3. Стоимость телефонов и комплектующих отражена в работе "Продажа телефонов".
4. Основные направления расходов компании: управление, рабочая сила, сотовые телефоны, комплектующие от поставщика.

Таблица А.11

Стоимостной анализ проведен в рамках 1 рабочего дня

| <i>Activity Name</i> | <i>Activity Cost (p.)</i> | <i>Cost Center</i> | <i>Cost Center Cost (p.)</i> |
|--|---------------------------|---------------------------------|------------------------------|
| Деятельность отдела продаж сотовых телефонов | 63 165,00 | Рабочая сила | 2 265,00 |
| | | Сотовые телефоны, комплектующие | 60 000,00 |
| | | Управление | 900,00 |
| Продажа телефонов | 60 300,00 | Рабочая сила | 300,00 |
| | | Сотовые телефоны, комплектующие | 60 000,00 |
| Проверка и внесение клиента | 10,00 | Рабочая сила | 10,00 |
| Оформление телефона, выдача клиенту | 4 010,00 | Рабочая сила | 10,00 |
| | | Сотовые телефоны, комплектующие | 4 000,00 |
| Технические работы | 2 565,00 | Рабочая сила | 1 665,00 |
| | | Управление | 900,00 |
| Распаковка телефонов, проверка комплектующих | 30,00 | Рабочая сила | 10,00 |
| | | Управление | 20,00 |
| Тестирование телефонов, комплектующих | 81,00 | Рабочая сила | 41,00 |
| | | Управление | 40,00 |

| <i>Activity Name</i> | <i>Activity Cost (p.)</i> | <i>Cost Center</i> | <i>Cost Center Cost (p.)</i> |
|-------------------------------------|---------------------------|--------------------|------------------------------|
| Установка тестовой SIM-карты | 41,00 | Рабочая сила | 1,00 |
| | | Управление | 40,00 |
| Проверка характеристик аккумулятора | 15,00 | Рабочая сила | 15,00 |
| Проверка базовой функциональности | 15,00 | Рабочая сила | 15,00 |
| Оценка функциональности | 10,00 | Рабочая сила | 10,00 |
| Закачка мелодий, игр | 50,00 | Рабочая сила | 50,00 |
| Упаковка, наклейка этикеток | 10,00 | Рабочая сила | 10,00 |
| Получение хранение | 300,00 | Рабочая сила | 300,00 |
| Получение проверка накладных | 10,00 | Рабочая сила | 10,00 |
| Хранение | 5,00 | Рабочая сила | 5,00 |
| Выдача | 5,00 | Рабочая сила | 5,00 |

Итого

Сумма всех работ = 63 165,00.

Прибыль: при накрутке цены телефона 25% она составляет 75000-63165=11813.

Конечно, здесь отражены не все направления расхода (аренда помещения, коммунальные услуги, налоги), но это не входит в рамки задачи.

Вспомогательные диаграммы.

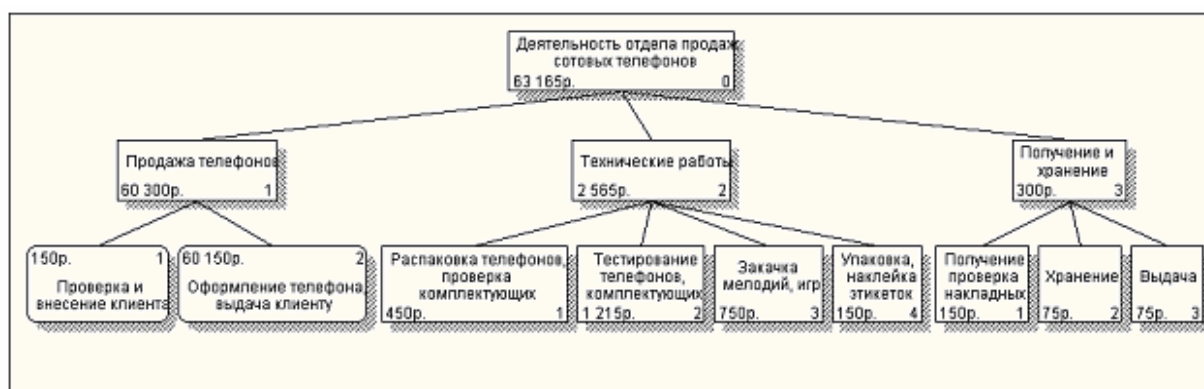


Рис. А.9. "Диаграмма дерева узлов"

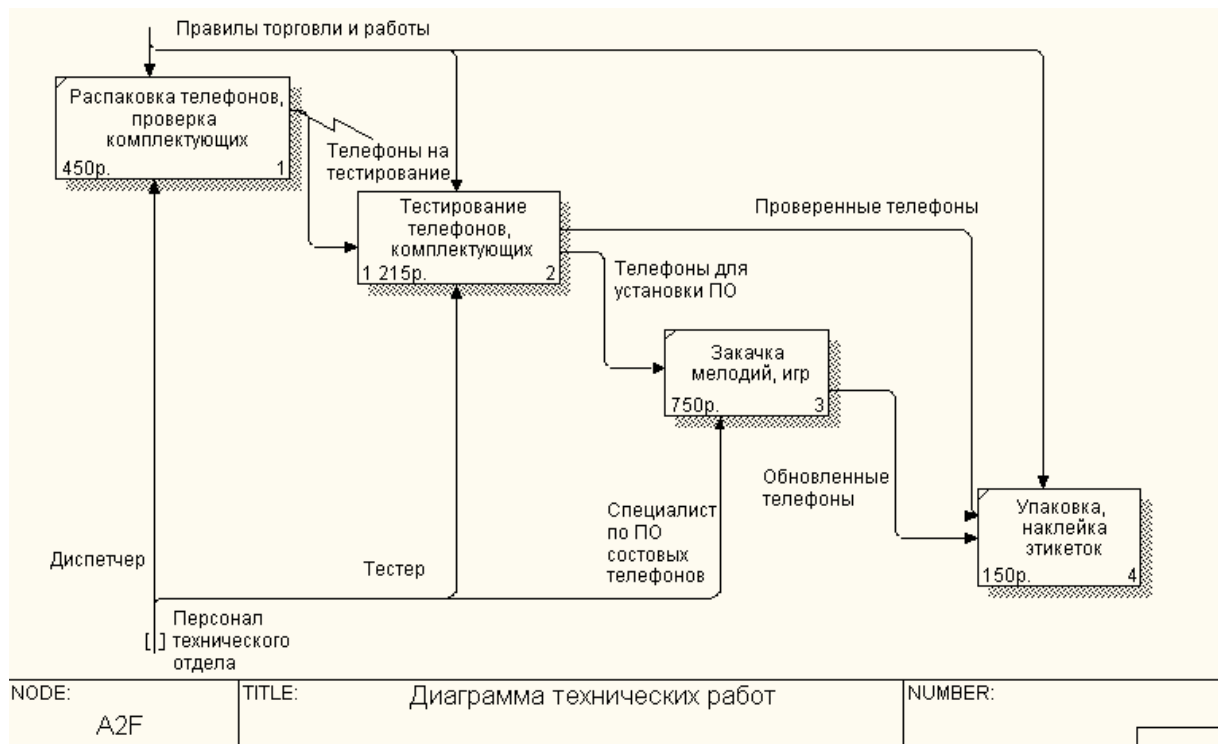


Рис. А.10. FEO-диаграмма "Диаграмма технических работ"

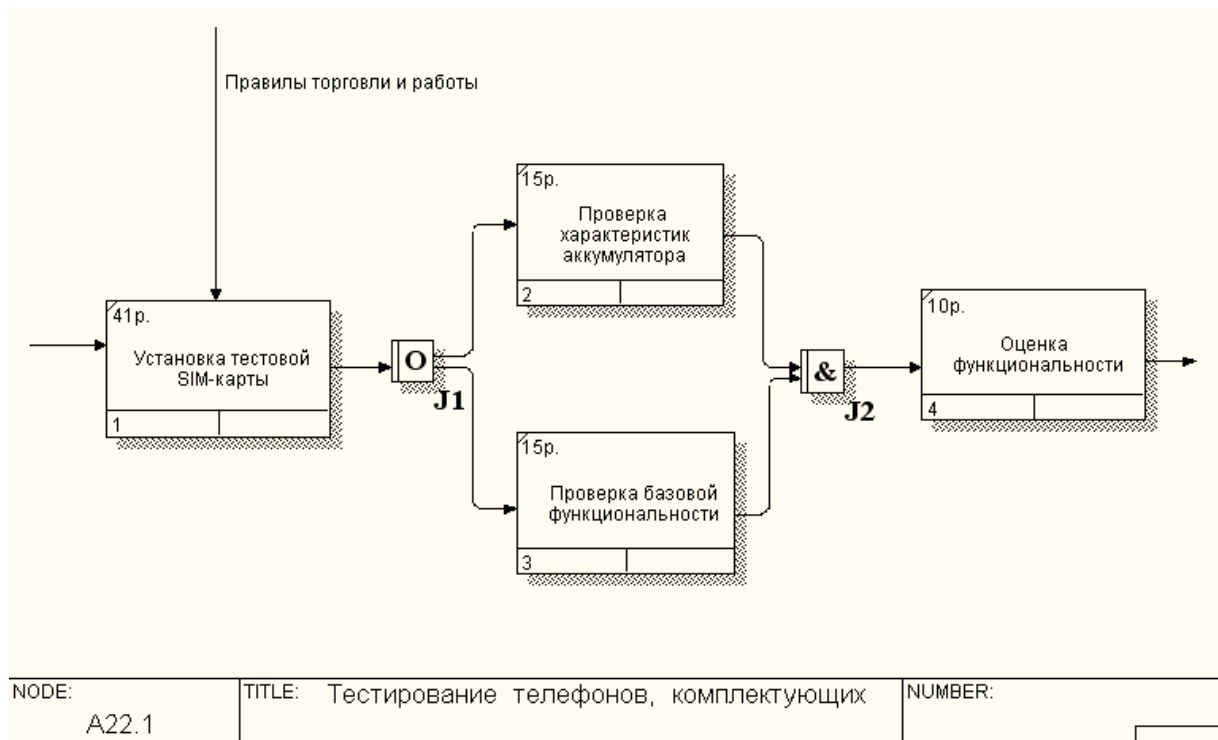


Рис. А.11. IDEF3-диаграмма

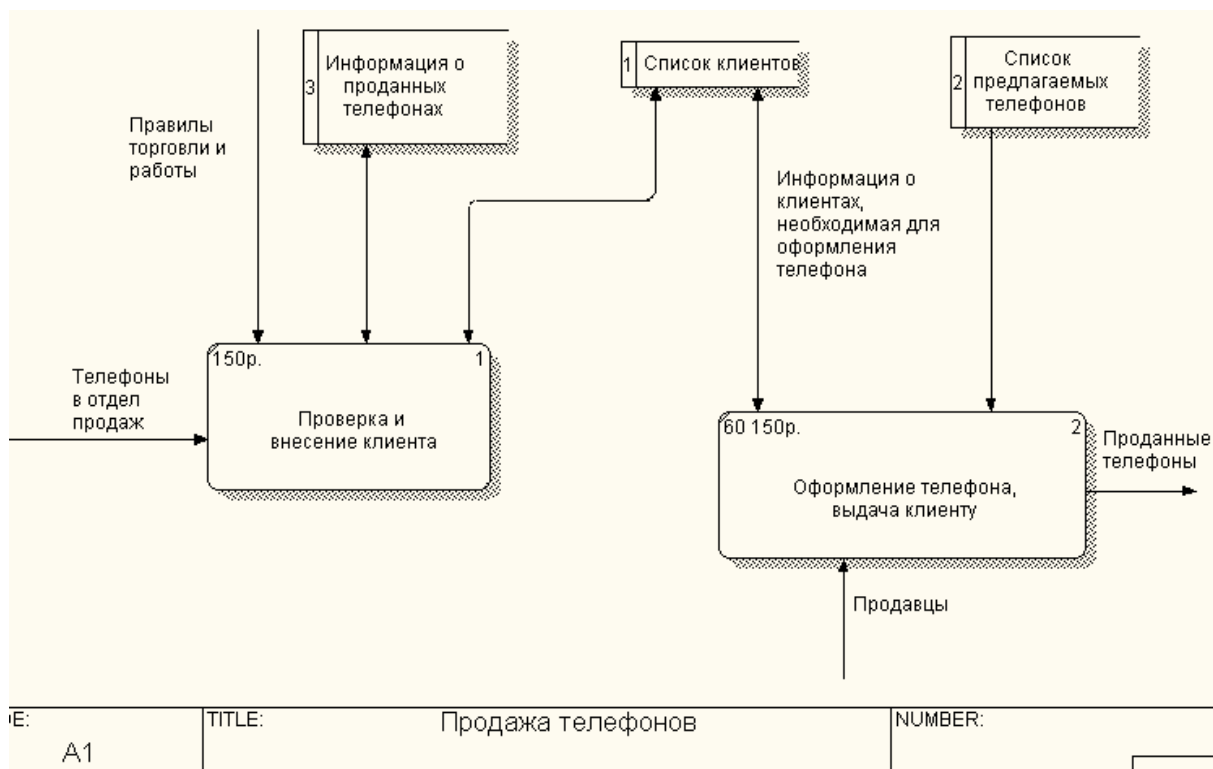


Рис. А.12. DFD-диаграмма

Потоки данных в компании распределены следующим образом:

5. Прайс-лист(2) необходим менеджеру как во время консультаций клиентов, так и во время оформления телефона.

6. Список клиентов (1) необходим для отчетности пред оператором, а также для хранения информации о проданных телефонах для обмена или гарантийного ремонта.

Описание информационной модели

1. Логический уровень.

Сущности в модели.

Для информационной модели магазина по продаже сотовых телефонов потребуются следующие понятия.

- Справочник фирм-производителей сотовых телефонов;
- Справочник моделей телефонов. Полное описание модели телефонов;
- Справочник операторов сотовой связи;
- Справочник пакетов SIM-подключений;
- Справочник поставщиков;
- Таблица, где хранятся данные о поставках;
- Таблица, где хранятся данные о покупателях;
- Таблица, где хранятся данные о телефонах, имеющих в наличии;
- Таблица, хранящая данные о продажах.

Создадим следующие сущности.

Таблица А.12

Сводная таблица сущностей

| Имя сущности | Описание |
|----------------------|--|
| <i>Buyer</i> | Покупатели телефонов |
| <i>Oreators</i> | Справочник операторов сотовой связи |
| <i>Postavchik</i> | Справочник поставщиков |
| <i>Postavka</i> | Данные о поставках |
| <i>Prodagy</i> | Продажи магазина |
| <i>SIM_pakets</i> | Справочник пакетов SIM-предложений |
| <i>Spr_firm</i> | Справочник фирм-производителей телефонов |
| <i>Spr_telephons</i> | Справочник моделей телефонов |
| <i>telephons</i> | Телефоны в наличии |

Для каждой сущности определим набор атрибутов.
Сущность "Покупатели" (*Buyer*).

Таблица А.13

Атрибуты сущности "Покупатели"

| Имя атрибута | Описание | <i>Is PK</i> (первичный ключ) | <i>Is FK</i> (внешний ключ) |
|------------------|----------------|-------------------------------|-----------------------------|
| <i>cod_buyer</i> | Код покупателя | <i>Yes</i> | <i>No</i> |
| <i>FIO</i> | ФИО покупателя | <i>No</i> | <i>No</i> |
| <i>Adres</i> | Адрес | <i>No</i> | <i>No</i> |
| <i>Pasport</i> | Паспорт | <i>No</i> | <i>No</i> |

Сущность "Операторы" (*Oreators*).

Таблица А.14

Атрибуты сущности "Операторы"

| Имя атрибута | Описание | <i>Is PK</i> (первичный ключ) | <i>Is FK</i> (внешний ключ) |
|---------------------|------------------------------|-------------------------------|-----------------------------|
| <i>Cod_operator</i> | Код оператора связи | <i>Yes</i> | <i>No</i> |
| <i>Operator</i> | Наименование оператора связи | <i>No</i> | <i>No</i> |

Сущность "Поставщики" (*Postavchik*)

Таблица 15

Атрибуты сущности "Поставщики"

| Имя атрибута | Описание | Is PK (первичный ключ) | Is FK(внешний ключ) |
|--------------------|---------------------------|------------------------|---------------------|
| <i>code_postav</i> | Код поставщика | <i>Yes</i> | <i>No</i> |
| <i>Postav</i> | Название фирмы поставщика | <i>No</i> | <i>No</i> |
| <i>Comment</i> | Комментарий | <i>No</i> | <i>No</i> |
| <i>Bank</i> | Банковские реквизиты | <i>No</i> | <i>No</i> |

Сущность "Поставка" (*Postavka*).

Таблица А.16

Атрибуты сущности "Поставка"

| Имя атрибута | Описание | Is PK (первичный ключ) | Is FK(внешний ключ) |
|----------------------|-------------------------|------------------------|---------------------|
| <i>code_postav</i> | Код поставщика | <i>Yes</i> | <i>Yes</i> |
| <i>cod_telephons</i> | Код модели телефона | <i>Yes</i> | <i>Yes</i> |
| <i>cod_firm</i> | Код фирмы производителя | <i>Yes</i> | <i>Yes</i> |
| <i>Kol</i> | Количество телефонов | <i>No</i> | <i>No</i> |
| <i>Data_postavka</i> | Дата поставки | <i>No</i> | <i>No</i> |
| <i>Zena</i> | цена | <i>No</i> | <i>No</i> |

Сущность "Продажи" (*Prodagy*).

В этой сущности только три своих атрибута. Она является ключевой в модели, т.к. по ее связям можно получить любую информацию.

Таблица А.17

Атрибуты сущности "Продажи"

| Имя атрибута | Описание | Is PK (первичный ключ) | Is FK(внешний ключ) |
|----------------------|----------------------------|------------------------|---------------------|
| <i>Data_prodagy</i> | Дата продажи | <i>Yes</i> | <i>No</i> |
| <i>Cod_paket</i> | Код тарифного плана | <i>Yes</i> | <i>Yes</i> |
| <i>Cod_operator</i> | Код оператора связи | <i>Yes</i> | <i>Yes</i> |
| <i>cod_telephons</i> | Код модели телефона | <i>Yes</i> | <i>Yes</i> |
| <i>cod_firm</i> | Код фирмы производителя | <i>Yes</i> | <i>Yes</i> |
| <i>cod_buyer</i> | Код покупателя | <i>Yes</i> | <i>Yes</i> |
| <i>code_postav</i> | Код поставщика | <i>Yes</i> | <i>Yes</i> |
| <i>Comment</i> | Комментарий | <i>No</i> | <i>No</i> |
| <i>Phon_number</i> | Телефонный номер SIM карты | <i>No</i> | <i>No</i> |

Сущность "SIM-пакеты" (*SIM_paket*).

Таблица А.18

Атрибуты сущности "SIM-пакеты"

| Имя атрибута | Описание | Is PK (первичный ключ) | Is FK(внешний ключ) |
|---------------------|---------------------|------------------------|---------------------|
| <i>Cod_paket</i> | Код тарифного плана | <i>Yes</i> | <i>No</i> |
| <i>Cod_operator</i> | Код оператора связи | <i>Yes</i> | <i>Yes</i> |
| <i>Paket</i> | Тарифный план | <i>No</i> | <i>No</i> |

Сущность "Фирмы-производители" (*Spr_firm*).

Таблица А.19

Атрибуты сущности "Фирмы-производители"

| Имя атрибута | Описание | Is PK (первичный ключ) | Is FK(внешний ключ) |
|-----------------|---------------------------------|------------------------|---------------------|
| <i>cod_firm</i> | Код фирмы производителя | <i>Yes</i> | <i>No</i> |
| <i>firm</i> | Наименование фирмы изготовителя | <i>No</i> | <i>No</i> |
| <i>Country</i> | Страна | <i>No</i> | <i>No</i> |

Сущность "Модели телефонов" (*Spr_telephons*).

Таблица А.20

Атрибуты сущности "Модели телефонов"

| Имя атрибута | Описание | Is PK (первичный ключ) | Is FK(внешний ключ) |
|------------------|--------------------------------|------------------------|---------------------|
| <i>Cod_model</i> | Код модели телефона | <i>Yes</i> | <i>No</i> |
| <i>cod_firm</i> | Код фирмы производителя | <i>Yes</i> | <i>Yes</i> |
| <i>model</i> | Модель телефона | <i>No</i> | <i>No</i> |
| <i>GPRS</i> | Поддержка интерфейса GPRS | <i>No</i> | <i>No</i> |
| <i>camera</i> | Наличие встроенной камеры | <i>No</i> | <i>No</i> |
| <i>display</i> | Характеристики дисплея | <i>No</i> | <i>No</i> |
| <i>comment</i> | Комментарий | <i>No</i> | <i>No</i> |
| <i>Bluetooth</i> | Поддержка интерфейса Bluetooth | <i>No</i> | <i>No</i> |
| <i>poliphon</i> | Полифония | <i>No</i> | <i>No</i> |

Сущность "Телефоны" (*telephons*).

Атрибуты сущности "Телефоны"

| Имя атрибута | Описание | Is PK (первичный ключ) | Is FK(внешний ключ) |
|----------------------|---|------------------------|---------------------|
| <i>cod_telephons</i> | Код модели телефона | <i>Yes</i> | <i>Yes</i> |
| <i>cod_firm</i> | Код фирмы производителя | <i>Yes</i> | <i>Yes</i> |
| <i>code_postav</i> | Код поставщика | <i>Yes</i> | <i>Yes</i> |
| <i>ser_nomer</i> | Серийный номер | <i>No</i> | <i>No</i> |
| <i>zena</i> | Цена телефона | <i>No</i> | <i>No</i> |
| <i>data_izgot</i> | Дата изготовления | <i>No</i> | <i>No</i> |
| <i>comment</i> | Комментарий относительно данного экземпляра | <i>No</i> | <i>No</i> |

Отношения между сущностями

Сущности в системе не могут существовать сами по себе. Они должны быть взаимосвязаны. Связи между сущностями называются отношениями.

Диаграмма отношений сущностей (на уровне сущностей) приведена на рис. А.13, а на уровне атрибутов на рис. А.14.

2. Физический уровень.

В качестве базовой СУБД выберем *SQL Server 2000*.

На основе логического уровня физический уровень создается автоматически.

Сущности превращаются в таблицы данных.

Ключи превращаются в индексы.

Ниже приведена диаграмма физического уровня информационной модели (рис. А.15).

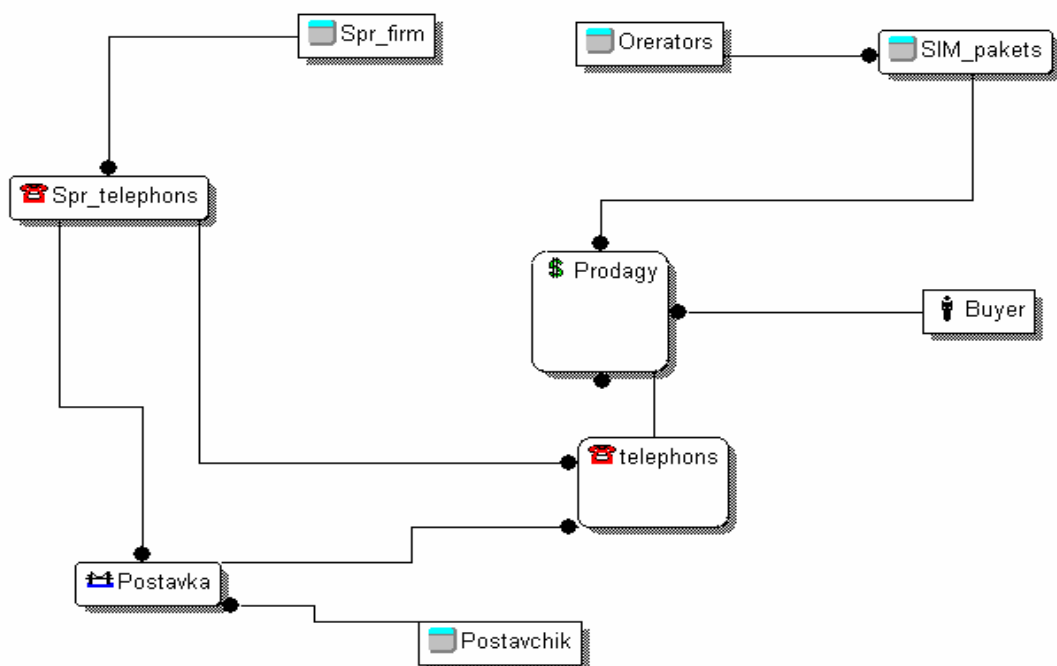


Рис. А.13. Отношения между сущностями (*Entyty Level*)

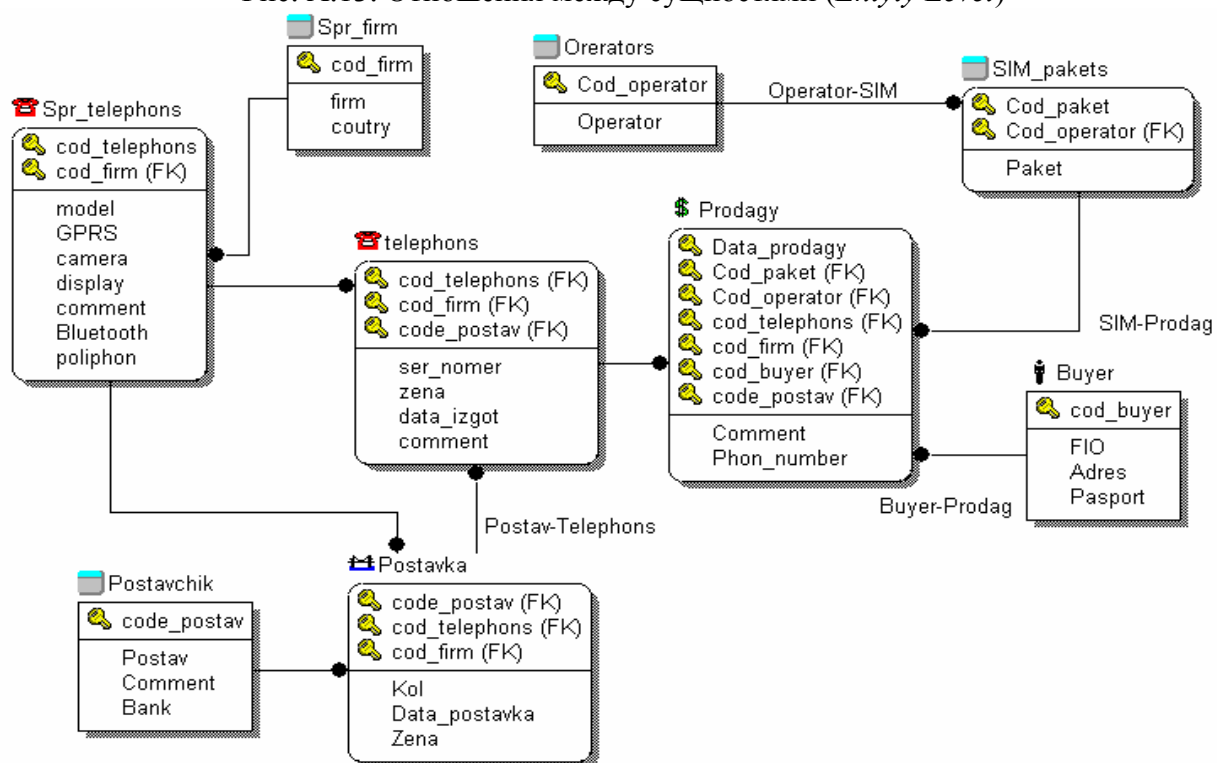


Рис. А.14. Отношения между сущностями (*Attribute Level*)

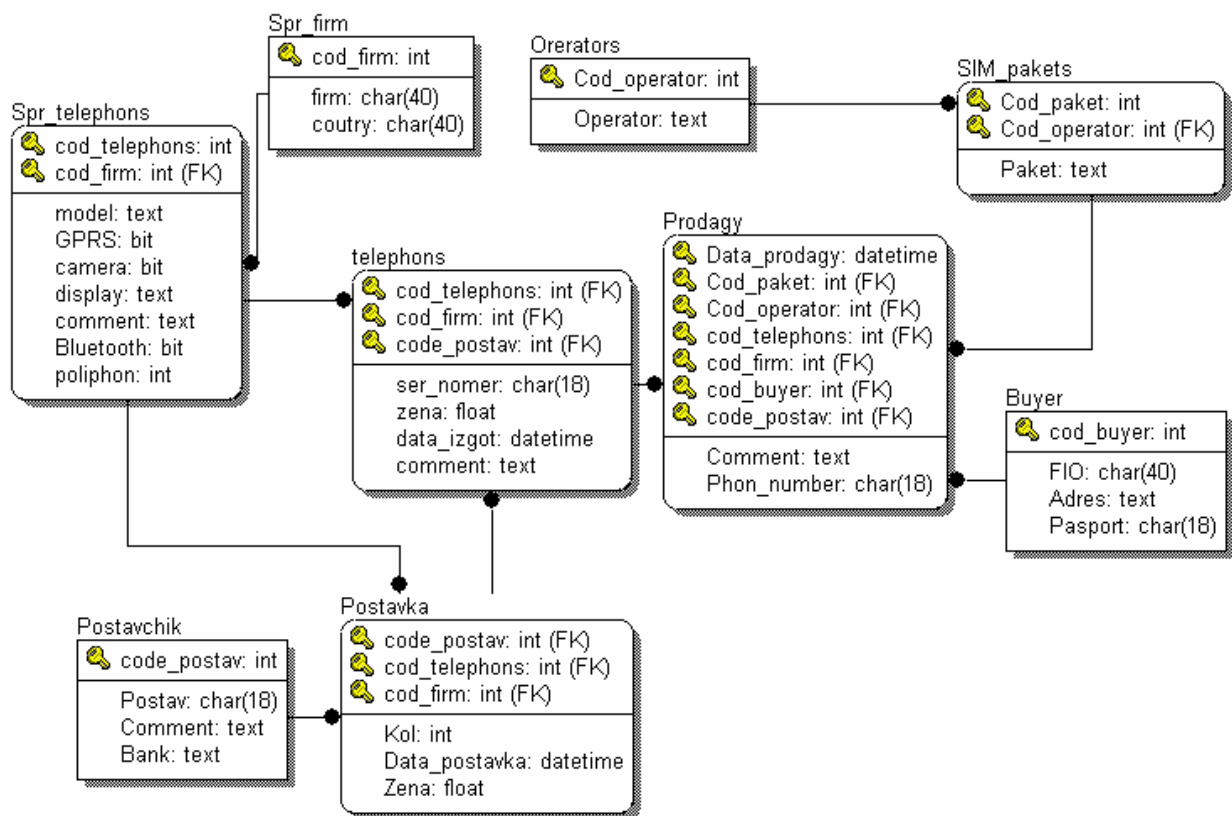


Рис. А.15. Физический уровень диаграммы.

ВЫВОД

Итоги обращения компании в консалтинговую фирму:

1. Выработка новой концепции деятельности. Концепция: "идти на встречу клиенту". Она заключается в расширении спектра услуг, выработки новых правил поведения.
2. Определения целей развития, критериев достижения этой цели. Цель: вырасти до специализированного магазина. Критерий: увеличение производственных площадей, торгового оборота до 4 раз.
3. Реинжиниринг процессов. Построение готовой модели "как должно быть".
4. Проведение стоимостного анализа. В результате стоимостного анализа не выявлено узких мест в организации работы компании.
5. Составление информационной модели (базы данных) предприятия

СПИСОК ЛИТЕРАТУРЫ

1. Маклаков С.В. *BPwin* и *ERwin*: CASE-средства для разработки информационных систем. – М.: Диалог-МИФИ, 1999.
2. Маклаков С.В. Моделирование бизнес-процессов с *BPwin* 4.0. –М.: ДИАЛОГ-МИФИ, 2002.
3. Вендров А.М. Проектирование программного обеспечения экономических информационных систем: Учебник. –М.: Финансы и статистика, 2000.
4. Вендров А.М. CASE-технологии. Современные методы и средства проектирования информационных систем, www.citforum.perm.ru
5. Федотова Д.Э., Семенов Ю.Д., Чижик К.Н. CASE-технологии: Практикум. – М.: Горячая линия – Телеком, 2003.
6. Калянов Г.Н. CASE-технологии. Консалтинг в автоматизации бизнес процессов. – 3-е изд. – М.: Горячая линия Телеком, 2002. – 320 с.: ил.