Министерство сельского хозяйства Российской Федерации
Трубчевский аграрный колледж -
филиал федерального государственного бюджетного образовательного
учреждения высшего образования
«Брянский государственный аграрный университет»

Живодеров А. Н.

ЭКСПЛУАТАЦИЯ И МОДИФИКАЦИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

Методические рекомендации
по выполнению практических работ
по профессиональному модулю ПМ.01 для обучающихся
специальности 09.02.04 Информационные системы
(по отраслям)

Брянская область
2020 г.

УДК 004.65 (076)
ББК 32.973.202
Ж 67

Живодеров, А. Н. **Эксплуатация и модификация информационных систем:** методические рекомендации по выполнению практических работ по профессиональному модулю ПМ.01 для обучающихся специальности 09.02.04 Информационные системы (по отраслям) / А. Н. Живодеров. – Брянск: Изд-во Брянский ГАУ, 2020. – 115 с.

Методические рекомендации по выполнению практических работ по профессиональному модулю ПМ.01 Эксплуатация и модификация информационных систем для специальности 09.02.04 Информационные системы (по отраслям) разработаны на основе Федерального государственного образовательного стандарта по специальности среднего профессионального образования 09.02.04 Информационные системы (по отраслям).

Настоящие методические рекомендации по выполнению практических работ определяют цели и задачи, порядок выполнения, содержат требования к выполнению практических работ.

Рецензент:
Лопаткин В. В. - преподаватель высшей категории Трубчевского филиала ФГБОУ ВО Брянский ГАУ

Рекомендации одобрены к печати методическим советом филиала, протокол № 2 от 29 ноября 2019 г.

СОДЕРЖАНИЕ

Введение

CASE-средства – это программные средства, автоматизирующие процессы создания и сопровождения информационных систем (ИС), включая анализ и формулировку требований, проектирование прикладного программного обеспечения (приложений) и баз данных, генерацию кода, тестирование, документирование, обеспечение качества, конфигурационное управление и управление проектом.

В методических рекомендациях поставлена цель дать основные сведения об архитектуре и основных компонентах CASE-средства, научить работать с интерфейсом пользователя, создавать функциональные модели, построение которых осуществляется с использованием иерархии функций и диаграмм потоков данных.

CASE-средства является мощным средством моделирования и документирования бизнес-процессов и предназначен для облегчения труда и увеличения производительности системного аналитика на первом этапе разработки системы.

BPwin поддерживает сразу три методологии: IDEF0, DFD и IDEF3, позволяющие проводить анализ предметной области с трех ключевых точек зрения.

ERwin – средство концептуального моделирования БД, использующее методологию IDEF1X. ERwin реализует проектирование схемы БД, генерацию ее описания на языке целевой СУБД

Результатом применения BPwin является модель предметной области, которая состоит из диаграмм, фрагментов текстов и глоссария, имеющих ссылки друг на друга.

BPwin обеспечивает логическую четкость в определении и описании элементов диаграмм, а также проверку целостности связей между диаграммами. Одним из важнейших средств BPwin является генератор отчетов.

1. Стадия проектирования информационных систем

На стадии проектирования выполняется ряд обязательных этапов.

1. Обследование деятельности предприятия. На этом этапе осуществляется:

- определение организационно-штатной и топологической структур предприятия;
- определение основных задач деятельности предприятия;
- проведение опросов сотрудников с целью построения функциональной модели деятельности «как есть» и, в случае эксплуатации какой-либо ИС, модели логической организации данных.

Результатом являются модели функциональной деятельности каждого из подразделений, способы взаимодействия между этими подразделениями, информационные потоки (как электронные, так и на традиционных носителях) между ними и внутри них.

Длительность этапа зависит как от размеров предприятия, так и от количества системных аналитиков, участвующих в проекте, и на практике составляет 1-2 недели. В некоторых случаях обследование может длиться и несколько месяцев, это приемлемо для организаций, деятельность которых достаточно консервативна. Для динамичных организаций такие сроки чреваты тем, что к концу обследования аналитики будут обладать устаревшей информацией.

2. Разработка системного проекта. Предварительным этапом здесь является построение моделей деятельности «как должно быть». Существует несколько видов работ, рекомендуемых при построении моделей деятельности:

- разработка структурной функциональной модели деятельности (методологии IDEF0, DFD – средства BPwin, Design/IDEF и др.);
- разработка информационной модели предприятия (методологии IDEF1X, ERD – средства ERwin, DatabaseDesigner, Design/IDEF, S-Designer);

- разработка событийной модели предприятия (метод динамического функционального анализа на основе сетей Петри различного вида).

Построение модели «как должно быть» является изменением технологий и переосмыслением бизнес-процессов (BPR).

Создание системного проекта (модели требований к будущей системе) является первой фазой разработки собственно системы автоматизации и строится на основе модели «как должно быть» и результатов обследования предприятия в части выявления требований к будущей системе. Системный проект должен включать:

- полную функциональную модель требований к будущей системе;
- комментарии к функциональной модели (спецификации процессов нижнего уровня в текстовом виде);
- пакет отчетов и документов по функциональной модели, включающий характеристику объекта моделирования, перечень подсистем, требования к способам и средствам связи для информационного обмена между компонентами, требования к характеристикам взаимосвязей системы со смежными системами, требования к функциям системы;
- концептуальную модель интегрированной базы данных (пакет диаграмм);
- архитектуру системы с привязкой к концептуальной модели;
- предложения по штатной структуре для поддержки системы.

Системный проект позволяет:

- увидеть и скорректировать будущую систему до того, как она будет реализована физически;
- уменьшить затраты на разработку и внедрение системы;
- оценить разработку по времени и результатам;
- достичь взаимопонимания между всеми участниками работы (заказчиками, пользователями, разработчиками);
- улучшить качество разрабатываемой системы.

Системный проект полностью независим и отделяем от конкретных разработчиков.

3. Разработка предложений по автоматизации предприятия. На основании системного проекта осуществляется:

- составление перечня автоматизированных рабочих мест предприятия и способов взаимодействия между ними;
- анализ применимости существующих систем управления предприятиями для решения требуемых задач и формирование рекомендаций по выбору такой системы;
- совместное с заказчиком принятие решения о выборе конкретной системы управления предприятием или разработке собственной системы;
- разработка требований к техническим средствам;
- разработка требований к программным средствам;
- разработка предложений по этапам и срокам реализации.

4. Разработка технического проекта. На данном этапе на основе системного проекта и принятых решений по автоматизации осуществляется проектирование системы. Фактически здесь дается ответ на вопрос: «Как мы будем строить систему, чтобы она удовлетворяла предъявленным к ней требованиям?». Этот этап разделяется на:

- проектирование архитектуры системы, включающее разработку структуры и интерфейсов ее компонент (автоматизированных рабочих мест), согласование функций и технических требований к компонентам, определение информационных потоков между основными компонентами, связей между ними и внешними объектами;
- детальное проектирование, включающее разработку спецификаций каждой компоненты, разработку требований к тестам и плана интеграции компонент, а также построение моделей иерархии программных модулей и межмодульных взаимодействий и проектирование внутренней структуры модулей.

При этом происходит расширение системного проекта:

- за счет его уточнения;
- за счет построения моделей автоматизированных рабочих мест, включающих подсхемы информационной модели и функциональные модели, ориентированные на эти подсхемы вплоть до идентификации конкретных сущностей информационной модели;

- за счет построения моделей межмодульных и внутри модульных взаимодействий с использованием техники структурных карт.

2. CASE-технологии

CASE-технология представляет собой совокупность методологий анализа, проектирования, разработки и сопровождения сложных систем и поддерживается комплексом взаимоувязанных средств автоматизации. CASE-технология – это инструментарий для системных аналитиков, разработчиков и программистов, заменяющий бумагу и карандаш компьютером, автоматизируя процесс проектирования и разработки ПО.

При использовании методологий структурного анализа появился ряд ограничений (сложность понимания, большая трудоемкость и стоимость использования, неудобство внесения изменений в проектные спецификации и т.д.) С самого начала CASE-технологии и развивались с целью преодоления этих ограничений путем автоматизации процессов анализа и интеграции поддерживающих средств. Они обладают следующими достоинствами и возможностями.

Единый графический язык. CASE-технологии обеспечивают всех участников проекта, включая заказчиков, единым строгим, наглядным и интуитивно понятным графическим языком, позволяющим получать обозримые компоненты с простой и ясной структурой. При этом программы представляются двумерными схемами (которые проще в использовании, чем многостраничные описания), позволяющими заказчику участвовать в процессе разработки, а разработчикам – общаться с экспертами предметной области, разделять деятельность системных аналитиков, проектировщиков и программистов, облегчая им защиту проекта перед руководством, а также обеспечивая легкость сопровождения и внесения изменений в систему.

Единая БД проекта. Основа CASE-технологии – использование базы данных проекта (репозитория) для хранения всей информации о проекте, которая может разделяться между разработчиками в соответствии с их правами доступа. Содержимое репозитория включает не только информационные объекты раз-

личных типов, но и отношения между их компонентами, а также правила использования или обработки этих компонентов. Репозиторий может хранить свыше 100 типов объектов: структурные диаграммы, определения экранов и меню, проекты отчетов, описания данных, логика обработки, модели данных, их организации и обработки, исходные коды, элементы данных и т. п.

Интеграция средств. На основе репозитория осуществляется интеграция CASE-средств и разделение системной информации между разработчиками. При этом возможности репозитория обеспечивают несколько уровней интеграции: общий пользовательский интерфейс по всем средствам, передачу данных между средствами, интеграцию этапов разработки через единую систему представления фаз жизненного цикла, передачу данных и средств между различными платформами.

Поддержка коллективной разработки и управления проектом. CASE-технология поддерживает групповую работу над проектом, обеспечивая возможность работы в сети, экспорт-импорт любых фрагментов проекта для их развития и/или модификации, а также планирование, контроль, руководство и взаимодействие, т.е. функции, необходимые в процессе разработки и сопровождения проектов. Эти функции также реализуются на основе репозитория. В частности, через репозиторий может осуществляться контроль безопасности (ограничения и привилегии доступа), контроль версий и изменений и др.

Макетирование. CASE-технология дает возможность быстро строить макеты (прототипы) будущей системы, что позволяет заказчику на ранних этапах разработки оценить, насколько она приемлема для будущих пользователей и устраивает его.

Генерация документации. Вся документация по проекту генерируется автоматически на базе репозитория (как правило, в соответствии с требованиями действующих стандартов). Несомненное достоинство CASE-технологии заключается в том, что документация всегда отвечает текущему состоянию дел, поскольку любые изменения в проекте автоматически отражаются в репозитории (известно, что при традиционных подходах к разработке ПО документация в лучшем случае запаздывает, а ряд модификаций вообще не находит в ней отражения).

Верификация проекта. CASE-технология обеспечивает автоматическую верификацию и контроль проекта на полноту и состоятельность на ранних этапах разработки, что влияет на успех разработки в целом – по статистическим данным анализа пяти крупных проектов фирмы TRW (США) ошибки проектирования и кодирования составляют соответственно 64% и 32% от общего числа ошибок, а ошибки проектирования в 100 раз труднее обнаружить на этапе сопровождения ПО, чем на этапе анализа требований.

Автоматическая генерация объектного кода. Генерация программ в машинном коде осуществляется на основе репозитория и позволяет автоматически построить до 85—90% объектного кода или текстов на языках высокого уровня.

Сопровождение и реинжиниринг. Сопровождение системы в рамках CASE-технологии характеризуется сопровождением проекта, а не программных кодов. Средства реинжиниринга и обратного инжиниринга позволяют создавать модель системы из ее кодов и интегрировать полученные модели в проект, автоматически обновлять документацию при изменении кодов и т. п.

Таблица 1

Традиционная технология разработки	Разработка с помощью CASE-технологий
Основные усилия – на кодирование и тестирование	Основные усилия – на анализ и проектирование
«Бумажные» спецификации	Быстрое итеративное макетирование
Ручное кодирование	Автоматическая генерация машинного кода
Тестирование ПО	Автоматический контроль проекта
Сопровождение программного кода	Сопровождение проекта

При использовании CASE-технологий изменяются все фазы жизненного цикла ИС, причем наибольшие изменения касаются фаз анализа и проектирования. В табл. 1 приведены основные изменения жизненного цикла ИС при использовании CASE-технологий по сравнению с традиционной технологией разработки.

Таблица 2

Анализ	Проектирование	Программирование	Тестирование
20%	15%	20%	45%
30%	30%	15%	25%
40%	40%	5%	15%

В табл. 2 приведены оценки трудозатрат по фазам жизненного цикла программного обеспечения (ПО). Первая строка таблицы соответствует традиционной технологии разработки, вторая – разработке с использованием структурных методологий вручную, третья – разработке с использованием CASE-технологий.

3. Характеристика современных CASE-систем

Современные CASE-средства охватывают обширную область поддержки многочисленных технологий проектирования ИС: от простых средств анализа и документирования до полномасштабных средств автоматизации, покрывающих весь жизненный цикл ПО.

В разряд CASE-средств попадают как относительно дешевые системы для персональных компьютеров с весьма ограниченными возможностями, так и дорогостоящие системы для неоднородных вычислительных платформ и операционных сред. Современный рынок программных средств насчитывает около 300 различных CASE-средств, наиболее мощные из которых, так или иначе, используются практически всеми ведущими западными фирмами.

Полный комплекс CASE-средств, обеспечивающий поддержку жизненного цикла ПО, содержит следующие компоненты:

- репозиторий, являющийся основой CASE-средства. Он должен обеспечивать хранение версий проекта и его отдельных компонентов, синхронизацию поступления информации от различных разработчиков при групповой разработке, контроль метаданных на полноту и непротиворечивость;
- графические средства анализа и проектирования, обеспечивающие создание и редактирование иерархически свя-

занных диаграмм (потоков данных, «сущность-связь» и др.), образующих модели ИС;

- средства разработки приложений, включая языки 4GL и генераторы кодов;
- средства конфигурационного управления;
- средства документирования;
- средства тестирования;
- средства управления проектом;
- средства реинжиниринга.

Все современные CASE-средства могут быть классифицированы, прежде всего, по типам. Классификация по типам отражает функциональную ориентацию CASE-средств на те или иные процессы ЖЦ. Помимо этого, CASE-средства можно классифицировать по следующим признакам:

- применяемым методологиям и моделям систем и баз данных (БД);
- степени интегрированности с системами управления базами данных (СУБД);
- доступным платформам.

Классификация по типам в основном совпадает с компонентным составом CASE-средств и включает следующие основные типы:

- средства анализа (Upper CASE), предназначенные для построения и анализа моделей предметной области (Design/IDEF, BPwin);
- средства анализа и проектирования (Middle CASE), поддерживающие наиболее распространенные методологии проектирования и использующиеся для создания проектных спецификаций (VantageTeamBuilder, Designer, Silverrun, PRO-IV, CASE.Аналитик). Выходом таких средств являются спецификации компонентов и интерфейсов системы, архитектуры системы, алгоритмов и структур данных;
- средства проектирования БД, обеспечивающие моделирование данных и генерацию схем баз данных (как правило, на языке SQL) для наиболее распространенных СУБД. К ним относятся ERwin, S-Designoги DataBase Designer (ORACLE). Средства проектирования баз данных имеются также в составе CASE-

средств VantageTeamBuilder, Designer, Silverrun и PRO-IV;

- средства разработки приложений. К ним относятся средства 4GL (Uniface, JAM, PowerBuilder, Developer, NewEra, SQLWindows, Delphi и др.) и генераторы кодов, входящие в состав VantageTeamBuilder, PRO-IV и частично – в Silverrun;
- средства реинжиниринга, обеспечивающие анализ программных кодов и схем баз данных и формирование на их основе различных моделей и проектных спецификаций. Средства анализа схем БД и формирования ERD входят в состав VantageTeamBuilder, PRO-IV, Silverrun, Designer, ERwin и S-Designor. В области анализа программных кодов наибольшее распространение получают объектно-ориентированные CASE-средства, обеспечивающие реинжиниринг программ на языке C++ (RationalRose, ObjectTeam).

Российский рынок программного обеспечения располагает следующими наиболее развитыми CASE-средствами:

- Vantage Team Builder (Westmount I-CASE);
- Designer;
- Silverrun;
- ERwin+BPwin;
- S-Designor;
- CASE.Аналитик;
- RationalRose.

Кроме того, на рынке постоянно появляются как новые для отечественных пользователей системы, так и новые версии и модификации перечисленных систем.

CASE-средство Silverrun американской фирмы Computer Systems Advisers, Inc. (CSA) используется для анализа и проектирования ИС бизнес-класса и ориентировано в большей степени на спиральную модель ЖЦ. Оно применимо для поддержки любой методологии, основанной на раздельном построении функциональной и информационной моделей (диаграмм потоков данных и диаграмм «сущность-связь»).

Система Silverrun реализована на трех платформах – MS Windows, Macintosh и OS/2 PresentationManager – с возможностью обмена проектными данными между ними.

VantageTeamBuilder представляет собой интегрированный

программный продукт, ориентированный на реализацию каскадной модели ЖЦ ПО и поддержку полного ЖЦ ПО.

VantageTeamBuilder обеспечивает выполнение следующих функций:

- проектирование диаграмм потоков данных, «сущность-связь», структур данных, структурных схем программ и последовательностей экранных форм;
- проектирование диаграмм архитектуры системы – SAD (проектирование состава и связи вычислительных средств, распределения задач системы между вычислительными средствами, моделирование отношений типа «клиент-сервер», анализ использования менеджеров транзакций и особенностей функционирования систем в реальном времени);
- генерация кода программ на языке 4GL целевой СУБД с полным обеспечением программной среды и генерация SQL-кода для создания таблиц БД, индексов, ограничений целостности и хранимых процедур;
- программирование на языке C со встроенным SQL;
- управление версиями и конфигурацией проекта;
- многопользовательский доступ к репозиторию проекта;
- генерация проектной документации по стандартным и индивидуальным шаблонам;
- экспорт и импорт данных проекта в формате CDIF (CASE DataInterchangeFormat).

VantageTeamBuilder поставляется в различных конфигурациях в зависимости от используемых СУБД (ORACLE, Informix, Sybase или Ingres) или средств разработки приложений (Uniface). Конфигурация VantageTeamBuilderforUniface отличается от остальных некоторой степенью ориентации на спиральную модель ЖЦ ПО за счет возможностей быстрого прототипирования, предоставляемых Uniface.

VantageTeamBuilder функционирует на всех основных UNIX-платформах (Solaris, SCO UNIX, AIX, HP-UX) и VMS.

CASE-средство Designer фирмы ORACLE является интегрированным CASE-средством, обеспечивающим в совокупности со средствами разработки приложений Developer поддержку полного ЖЦ ПО для систем, использующих СУБД ORACLE.

Designer представляет собой семейство методологий и поддерживающих их программных продуктов. Базовая методология Designer (CASE*Method) – структурная методология проектирования систем, охватывающая полностью все этапы жизненного цикла ИС.

Генерация приложений, помимо продуктов ORACLE, выполняется также для VisualBasic.

Designer можно интегрировать с другими средствами, используя открытый интерфейс приложений API (ApplicationProgrammingInterface). Кроме того, можно использовать средство ORACLE CASE Exchange для экспорта/импорта объектов репозитория с целью обмена информацией с другими CASE-средствами.

BPwin, ERwin – средства функционального и концептуального моделирования, реализующие методологии IDEF0 и IDEF1X соответственно. Об этих системах более подробная информация представлена ниже.

S-Designor представляет собой CASE-средство для проектирования реляционных баз данных. По своим функциональным возможностям и стоимости он близок к CASE-средству Erwin, отличаясь внешне используемой на диаграммах нотацией. S-Designor реализует стандартную методологию моделирования данных и генерирует описание БД для таких СУБД, как ORACLE, Informix, Ingres, Sybase, DB/2, Microsoft SQL Server и др. Для существующих систем выполняется реинжиниринг БД.

S-Designor совместим с рядом средств разработки приложений (PowerBuilder, Uniface, TeamWindows и др.) и позволяет экспортировать описание БД в репозитории данных средств. Для PowerBuilder выполняется прямая генерация шаблонов приложений.

CASE.Аналитик 1.1 является практически единственным в настоящее время конкурентоспособным отечественным CASE-средством функционального моделирования . Его основные функции:

- построение и редактирование диаграмм потоков данных DFD;
- анализ диаграмм и проектных спецификаций на пол-

ноту и непротиворечивость;

- получение разнообразных отчетов по проекту;
- генерация макетов документов в соответствии с требованиями ГОСТ 19.XXX и 34.XXX.

С помощью отдельного программного продукта (Catherine) выполняется обмен данными с CASE-средством ERwin. При этом из проекта, выполненного в CASE. Аналитике, экспортируется описание структур данных и накопителей данных, которое по определенным правилам формирует описание сущностей и их атрибутов.

RationalRose – CASE-средство фирмы RationalSoftwareCorporation (США) – предназначено для автоматизации этапов анализа и проектирования ПО, а также длягенерации кодов на различных языках и выпуска проектной документации. RationalRose использует синтез-методологию объектно-ориентированного анализа и проектирования, основанную на подходах трех ведущих специалистов в данной области: Буча, Рамбо и Джекобсона. Разработанная ими универсальная нотация для моделирования объектов (UML – UnifiedModelingLanguage) претендует на роль стандарта в области объектно-ориентированного анализа и проектирования. Конкретный вариант RationalRose определяется языком, на котором генерируются коды программ (C++, Smalltalk, PowerBuilder, Ada, SQL Windows и ObjectPro). Основной вариант – RationalRose/C++ - позволяет разрабатывать проектную документацию в виде диаграмм и спецификаций, а также генерировать программные коды на C++. Кроме того, RationalRose содержит средства реинжиниринга программ, обеспечивающие повторное использование программных компонент в новых проектах.

В основе работы RationalRose лежит построение различного рода диаграмм и спецификаций, определяющих логическую и физическую структуры модели, ее статические и динамические аспекты. В их число входят диаграммы классов, состояний, сценариев, модулей, процессов.

RationalRose функционирует на различных платформах: IBM PC (в среде Windows), Sun SPARC stations (UNIX, Solaris, SunOS), Hewlett-Packard (HP UX), IBM RS/6000 (AIX).

Практическая работа №1
“Изучение основных функций пакета BPwin”

BPwin позволяет аналитику создавать сложные модели бизнес-процессов при минимальных усилиях. BPwin поддерживает три методологии – IDEF0, IDEF3 и DFD. Каждая из них призвана решать свои специфические задачи. Также можно строить смешанные модели.

Модель в BPwin рассматривается как совокупность работ, каждая из которых оперирует с некоторым набором данных. Работы изображаются в виде прямоугольников (блоков), данные – в виде стрелок (дуг).

Основу методологии IDEF0 составляет графический язык описания бизнес-процессов. Модель в IDEF0 представлена совокупностью иерархически упорядоченных и логически связанных диаграмм. Каждая диаграмма располагается на отдельном листе. Можно выделить четыре типа диаграмм:

- контекстную диаграмму A-0 (в каждой модели может быть только одна контекстная диаграмма);
- диаграммы декомпозиции (в том числе диаграмма первого уровня декомпозиции A0, раскрывающая контекстную);
- 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 (как правило, здесь описывается основное назначение моделируемого объекта). 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 (получения диаграммы A0) проводится декомпозиция каждого блока диаграммы A0 на более мелкие фрагменты и так далее, до достижения нужного уровня подробности описания. После каждого сеанса декомпозиции проводятся сеансы экспертизы – эксперты предметной области (обычно это интервьюиру-

емые аналитиками сотрудники предприятий) указывают на соответствие реальных бизнес-процессов созданным диаграммам. Найденные несоответствия исправляются и только после прохождения экспертизы без замечаний можно приступать к следующему сеансу декомпозиции. Так достигается соответствие модели реальным бизнес-процессам на любом и каждом уровне модели. 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 сколько угодно, 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.

Каркас диаграммы. На рис.1 показан пример контекстной диаграммы с граничными рамками, которые называются каркасом диаграммы. Каркас содержит заголовок (верхняя часть рамки, табл.3) и подвал (нижняя часть, табл.4). Заголовок каркаса используется для отслеживания диаграммы в процессе моделирования. Нижняя часть используется для идентификации и позиционирования в иерархии диаграмм. Значения полей каркаса задаются в диалоге DiagramProperties (в меню Edit/DiagramProperties).

Рис. 1. Контекстная диаграмма

Поля заголовка каркаса (слева направо)

Таблица 3

Поле	Смысл
Used At	Используется для указания на родительскую работу в случае, если на текущую диаграмму ссылались посредством стрелки вызова.
Author, Date, Rev, Project	Имя создателя диаграммы, дата создания и имя проекта, в рамках которого была создана диаграмма. REV – дата последнего редактирования диаграммы.
Notes 1 2 3 4 5 6 7 8 9 10	Используется при проведении сеанса экспертизы. Эксперт должен (на бумажной копии диаграммы) указать число замечаний, вычеркивая цифру из списка каждый раз при внесении нового замечания.
Status	Статус отображает стадию создания диаграммы, отображая все этапы публикации.
Working	Новая диаграмма, кардинально обновленная диаграмма или новый автор диаграммы.
Draft	Диаграмма прошла первичную экспертизу и готова к дальнейшему обсуждению.
Recommended	Диаграмма и все ее сопровождающие документы прошли экспертизу. Новых изменений не ожидается.
Publication	Диаграмма готова к окончательной печати и публикации.
Reader	Имя читателя (эксперта).
Date	Дата прочтения (экспертизы).
Context	Схема расположения работ в диаграмме верхнего уровня. Работа, являющаяся родительской, показана темным прямоугольником, остальные – светлым. На контекстной диаграмме (A-0) показывается надпись TOP. В левом нижнем углу показывается номер по узлу родительской диаграммы.

Поля подвала каркаса (слева направо)

Таблица 4

Поле	Смысл
Node	Номер узла диаграммы (номер родительской работы)
Title	Имя диаграммы. По умолчанию – имя родительской работы
Number	C-Number, уникальный номер версии диаграммы
Page	Номер страницы, может использоваться как номер страницы при формировании папки

Задание. На основе резюме, описывающих функционирование конкретного отдела РГУ нефти и газа им. И.М. Губкина, создать контекстную диаграмму А-0. Выделить основные его функции и создать диаграмму А0. Разбить каждую функцию на подфункции и диаграммы третьего уровня. Предоставить иерархию диаграмм.

Вопросы.

1. Каковы стадии жизненного цикла информационных систем, их основное содержание?

2. Что такое реинжиниринг бизнес-процессов?

3. Какие виды работ рекомендуется выполнить при построении моделей деятельности, какие средства и методологии при этом используются?

4. Каковы основные функции CASE-средства BPwin?

5. Как представляется функциональная модель деятельности в методологии IDEF0?

Практическая работа №2
“Изучение объектов диаграмм функциональной модели”

Работы (Activity). Работы обозначают поименованные процессы, функции или задачи, которые происходят в течение определенного времени и имеют распознаваемые результаты. Работы изображаются в виде прямоугольников (блоков). Все работы должны быть названы и определены. Имя работы должно быть в глагольной или отглагольной форме (например, «Принять заказ», «Изготовление детали» и т.д.). Работу можно добавить, щелкнув по кнопке  на палитре инструментов, а затем по свободному месту на диаграмме. Работы на диаграммах декомпозиции располагаются по диагонали от левого верхнего угла к правому нижнему (рис. 2). В левом верхнем углу располагается самая важная работа или работа, выполняемая по времени первой. Далее вправо вниз располагаются менее важные или выполняемые позже работы.

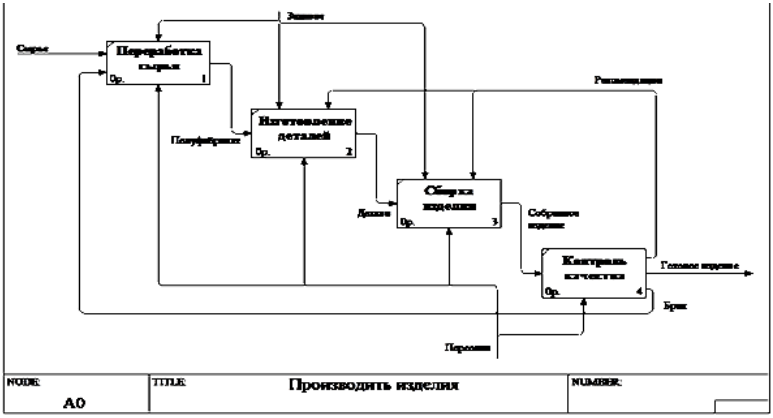

Рис. 2. Диаграмма декомпозиции

Для внесения имени работы следует щелкнуть по работе правой кнопкой мыши, выбрать в меню пункт NameEditor и в появившемся диалоге внести имя работы (рис. 3).

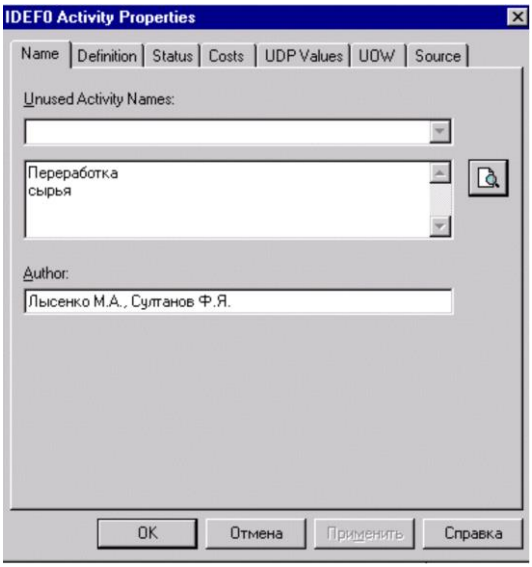

Рис. 3. Внесение имени работы

Для создания диаграммы декомпозиции следует щелкнуть по кнопке  и выбрать на диаграмме работу, которую необходимо декомпозировать.

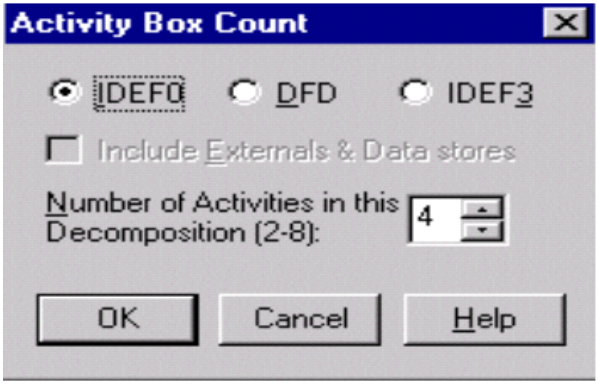

Рис. 4. Выбор нотации диаграммы

Возникает диалог ActivityBoxCount (рис. 4), в котором следует указать нотацию новой диаграммы. Надо выбрать IDEF0 и надавить ОК.

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 (Arrows). Взаимодействие работ с внешним миром 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, которая используется или преобразуется работой для получения результата (выхода). Допускается, что работа может не иметь ни одной стрелки входа. Каждый тип стрелок подходит к определенной стороне блока или выходит из нее. Очень часто сложно определить, являются ли данные входом или управлением. В этом случае подсказкой может служить то, перерабатываются,/изменяются ли данные в работе или нет. Если изменяются, то, скорее всего это вход, если нет – управление.
- Управление (Control) – правила, стратегии, процедуры или стандарты, которыми руководствуется работа. Каждая работа должна иметь хотя бы одну стрелку управления. Управление влияет на работу, но не преобразуется ей. Если цель работы – изменить процедуру или стратегию, то такая процедура или стратегия будет для работы входом.
- Выход (Output) – материал или информация, которые производятся работой. Каждая работа должна иметь хотя бы одну стрелку выхода.
- Механизм (Mechanism) – ресурсы, которые выполняют работу, например персонал предприятия, станки, устройства и т.д.
- Вызов (Call) – специальная стрелка, указывающая на другую модель работы. Рисуется как исходящая из нижней грани работы. Стрелка вызова используется для указания того, что некоторая работа выполняется за пределами моделируемой систе-

мы. Используются в механизме слияния и разделения моделей.

Каждый тип стрелок подходит к определенной стороне блока или выходит из нее. Стрелка входа рисуется как входящая в левую грань работы. Стрелка управления рисуется как входящая в верхнюю грань. Выход рисуется как исходящая стрелка из правой грани. Механизм – входит в нижнюю.

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 надо:

- щелкнуть по кнопке с символом стрелки  в палитре инструментов. Дальше перенести курсор к левой стороне экрана, пока не появится начальная штриховая полоска;
- щелкнуть один раз по полоске (откуда выходит стрелка) и еще раз в левой части работы со стороны входа (где заканчивается стрелка);
- вернуться в палитру инструментов и выбрать редактирование стрелки 
- щелкнуть правой кнопкой мыши на линии стрелки, во всплывающем меню выбрать пункт NameEditor и добавить имя стрелки в закладке Name диалога IDEF0 ArrowProperties.

Стрелки управления, входа, 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 штриховая полоска, и щелкнуть один раз по ней.

Имена вновь внесенных стрелок автоматически заносятся в словарь.

Словарь стрелок (ArrowDictionary) редактируется при помощи специального редактора ArrowDictionaryEditor (рис.5), в котором определяется стрелка и вносится относящийся к ней комментарий.

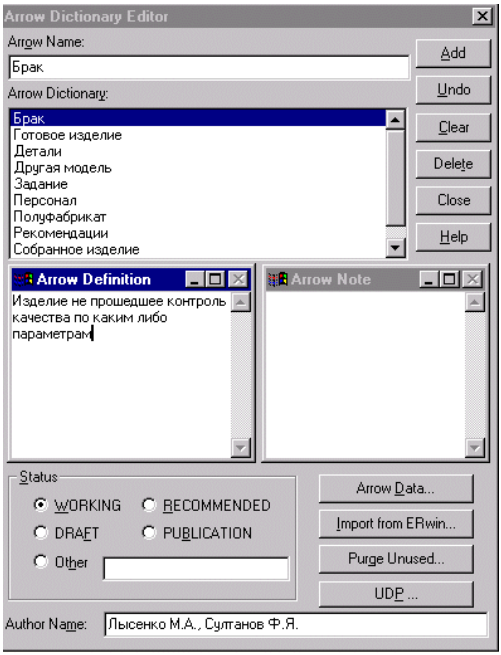

Рис. 5. Редактор словаря стрелок

Словарь стрелок решает очень важную задачу. Диаграммы создаются аналитиком для того, чтобы провести сеанс экспертизы, т.е. обсудить диаграмму со специалистом предметной области. В любой предметной области формируется профессиональный жаргон, причем очень часто жаргонные выражения имеют нечеткий смысл и воспринимаются разными специалистами поразному. В то же время аналитик – автор диаграмм должен употреблять те выражения, которые наиболее понятны экспертам.

Поскольку формальные определения часто сложны для восприятия, аналитик вынужден употреблять профессиональный жаргон, а чтобы не возникало неоднозначных трактовок, в словаре стрелок каждому понятию можно дать расширенное и, если это необходимо, формальное определение.

Внутренние стрелки. Для связи работ между собой используются внутренние стрелки, т.е. стрелки, которые не каса-

ются 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 входу показывает доминирование вышестоящей работы. Данные или объекты выхода вышестоящей работы не меняются в нижестоящей;
- обратная связь по входу (output-inputfeedback), когда выход нижестоящей работы направляется на вход вышестоящей. Такая связь, как правило, используется для описания циклов;
- обратная связь по управлению (output-controlfeedback), 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 IDEF0 используются разветвляющиеся и сливающиеся стрелки. Для разветв-

ления 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.

Тоннелирование стрелок. Вновь внесенные граничные стрелки на диаграмме декомпозиции нижнего уровня изображаются в квадратных скобках и автоматически не появляются на диаграмме верхнего уровня. Для их «перетаскивания» наверх нужно сначала выбрать кнопку  на палитре инструментов и щелкнуть по квадратным скобкам граничной стрелки. Появится диалог BorderArrowEditor (рис. 6).

Рис. 6. Диалог для тоннелирования стрелок

Если щелкнуть по кнопке ResolveBorderArrow, стрелка мигрирует на диаграмму верхнего уровня, если по кнопке ChangeToTunnel – 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. Если эти данные не используются на родительской диаграмме, их нужно направить еще выше и т.д. В ре-

зультате 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 тоннелирования 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. В этом случае стрелка механизма на нижнем уровне может быть удалена, после чего на родительской диаграмме она может быть затоннелирована («Не-в-дочерней-работе»).

Задание. Исходя из результатов предыдущей практической работы, создать все диаграммы в программе, расположить на них все блоки и дуги, описывающие заданный отдел. Получить законченную модель функционирования отдела.

Вопросы

1. Что такое CASE-технологии, их достоинства и преимущества?
2. Проведите сравнительный анализ традиционной технологии разработки и разработки с помощью CASE-технологии.
3. Каковы основные объекты диаграмм функциональной модели по методологии IDEF0?
4. Что обозначают работы в диаграммах функциональной модели, как они отображаются по методологии IDEF0?
5. Для чего предназначены стрелки в диаграммах функциональной модели, каковы их типы и виды?
6. Для чего предназначен словарь стрелок?
7. Каковы типы связей работ по методологии IDEF0?
8. Что такое тоннелирование стрелок, для чего оно нужно, каковы виды тоннелирования?

Практическая работа №3
“Составление отчетов в пакете BPwin”

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

1. ModelReport. Этот отчет включает информацию о контексте модели – имя модели, точку зрения, область, цель, имя автора, дату создания и др.
2. DiagramReport. Отчет по конкретной диаграмме. Включает список объектов (работ, стрелок, хранилищ данных, внешних ссылок и т.д.).
3. DiagramObjectReport. Наиболее полный отчет по модели. Может включать полный список объектов модели (работ, стрелок с указанием их типа и др.) и свойства, определяемые пользователем.
4. ActivityCostReport. Отчет о результатах стоимостного анализа.
5. ArrowReport. Отчет по стрелкам. Может содержать информацию из словаря стрелок, информацию о работе-источнике, работе-назначении стрелки и информацию о разветвлении и слиянии стрелок.
6. DataUsageReport. Отчет о результатах связывания модели процессов и модели данных.
7. ModelConsistencyReport. Отчет, содержащий список синтаксических ошибок модели.

Синтаксические ошибки IDEF0 с точки зрения BPwin разделяются на три типа:

- во-первых, это ошибки, которые BPwin выявить не в состоянии. 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;
- 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 отмечает их. Полный их список можно получить в отчете ModelConsistencyReport. Это единственный неопциональный отчет в BPwin. Список ошибок может содержать, например, неименованные работы и стрелки (unnamedarrow, unnamedactivity), несвязанные стрелки (unconnectedborderarrow), неразрешенные стрелки (unresolved (squaretunneled) arrowconnections), работы, не имеющие, по крайней мере, одной стрелки выхода и одной стрелки управления, 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. Рассмотрим типичный диалог ArrowReport (рис. 7).

Раскрывающийся список StandardReports 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 (UserDefinedProperties). Они сохраняются в виде указателя и поэтому доступны только из родной модели. Стандартный отчет можно изменить или удалить.

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;
- FixedColumn – каждое поле печатается в собственной колонке;

Tab-CommaDelimited – каждое поле печатается в собственной колонке. Колонки

- разделяются знаком табуляции или запятыми;
- DDETable – данные передаются по DDE приложению,

например, MSWord или Excel;

- RPTwin – отчет создается в формате PlatinumRPTwin – специализированного генератора отчетов, который входит в поставку BPwin.

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

Опция Multi-ValuedFormat регулирует вывод полей в отчете при группировке данных:

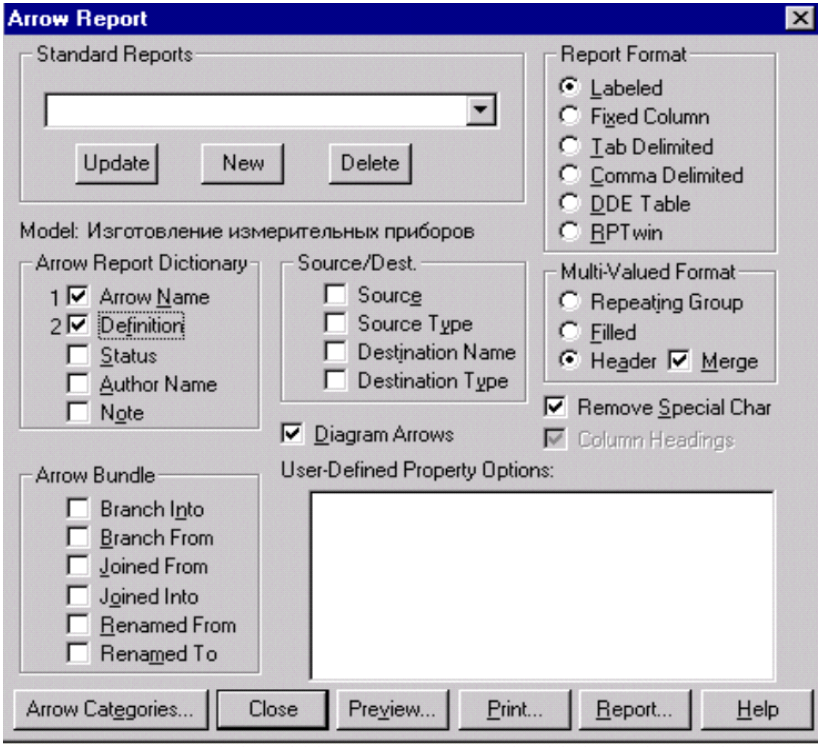

Рис. 7. Диалог настройки отчета

- RepeatingGroup – 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.

Задание. По полученной модели получить основные отчеты: по дугам и блокам модели. Проанализировать полученные отчеты.

Вопросы

1. Какие компоненты должны входить в полный комплекс CASE-средств, обеспечивающий поддержку жизненного цикла ПО?
2. По каким признакам можно классифицировать CASE-средства?
3. По каким основным типам классифицируются CASE-средства, какие конкретные системы им соответствуют?
4. Какие существуют типы отчетов в пакете BPwin, для чего каждый из них предназначен?
5. Какого рода синтаксические ошибки выявляет пакет BPwin?

Практическая работа №4
“Изучение объектов DFD-диаграмм”

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

DFD описывает:

- функции обработки информации (работы, activities);
- документы (стрелки, arrows), объекты, сотрудников или отделы, которые участвуют в обработке информации;
- внешние ссылки (externalreferences), которые обеспечивают интерфейс с внешними объектами, находящимися за границами моделируемой системы;
- таблицы для хранения документов (хранилище данных, datastore).

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 ActivityBoxCount надавить на радио-кнопку DFD. В палитре инструментов на новой диаграмме появляются кнопки:

-  добавить в диаграмму внешнюю ссылку. Внешняя ссылка является источником или приемником данных извне модели;
-  добавить в диаграмму хранилище данных. Хранилище данных позволяет описать данные, которые необходимо сохранить в памяти прежде, чем использовать в работах;
- ссылка на другую страницу. В отличие от IDEF0 инструмент off-pagereference 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 (dataflow), хранение объектов (datastores), поставка и распространение объектов.

В отличие от IDEF0, где система рассматривается как взаимосвязанные работы, 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.

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.

Внешние сущности. Изображают входы в систему и/или выходы из нее. 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, как в IDEF0, стрелки могут подходить 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 яв-

ляются 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 является подход, популярный при создании программного обеспечения, называемый событийным разделением, в котором различные диаграммы DFD выстраивают модель системы.

Логическая модель строится как совокупность работ и документирования того, что эти работы должны делать.

Модель окружения 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 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 (А) и номер объекта. Номер объекта – это уникальный номер работы на диаграмме. Например, работа может иметь номер А.12.4. Уникальный номер имеют хранилища данных и внешние сущности независимо от их расположения на диаграмме. Каждое хранилище данных имеет префикс D и уникальный номер, например D5. Каждая внешняя сущность имеет префикс Е и уникальный номер, например Е4.

Задание. По заданному отделу построить диаграмму верхнего уровня взаимодействия отдела с внешними данными.

Вопросы

1. Какие CASE-средства наиболее известны на российском рынке программного обеспечения?
2. Каковы основные функции наиболее известного российского CASE-средства функционального моделирования?
3. В чем особенности CASE-средства RationalRose?
4. В чем особенности DFD-диаграмм, что в них описывается?
5. В чем особенности объектов DFD-диаграмм?
6. В чем различия функциональной, логической, физической моделей, а также моделей окружения и поведения?

Практическая работа №5
“Изучение основных функций пакета ERwin. Создание логической модели”

ERwin – средство концептуального моделирования БД, использующее методологию IDEF1X. ERwin реализует проектирование схемы БД, генерацию ее описания на языке целевой СУБД (ORACLE, Informix, Ingres, Sybase, DB/2, Microsoft SQL Server, Progress и др.) и реинжиниринг существующей БД. ERwin выпускается в нескольких различных конфигурациях, ориентированных на наиболее распространенные средства разработки приложений 4GL. Версия ERwin/OPEN полностью совместима со средствами разработки приложений PowerBuilder и SQLWindows и позволяет экспортировать описание спроектированной БД непосредственно в репозитории данных средств.

Для ряда средств разработки приложений (PowerBuilder, SQLWindows, Delphi, VisualBasic) выполняется генерация форм и прототипов приложений.

Сетевая версия ERwin ModelMart обеспечивает согласованное проектирование БД и приложений в рабочей группе.

Основные получаемые преимущества:

- существенное повышение скорости разработки за счет мощного редактора диаграмм, автоматической генерации базы данных, автоматической подготовки документации;
- нет необходимости ручной подготовки SQL-предложений для создания базы данных;
- возможность легко вносить изменения в модель при разработке и расширении системы;
- возможность автоматической подготовки отчетов по базе данных; важно, что эти отчеты всегда в точности соответствуют реальной структуре БД;
- разработчики прикладного программного обеспечения снабжены удобными в работе диаграммами;
- тесная интеграция со средствами 4GL позволяет уже на стадии информационного моделирования задавать отображение данных в приложениях;
- обратное проектирование позволяет документировать и

вносить изменения в существующие информационные системы;

- поддержка однопользовательских СУБД позволяет использовать для персональных систем современные технологии, что значительно упрощает переход от настольных систем к системам в технологии клиент-сервер (upsizing).

Построение моделей в ERwin.

Возможны две точки зрения на информационную модель и, соответственно, два уровня модели. Первый – логический уровень (точка зрения пользователя)означает прямое отображение фактов из реальной жизни. Например, люди, столы, отделы, собаки и компьютеры являются реальными объектами. Они именуются на естественном языке, с любыми разделителями слов (пробелы, запятые и т.д.). На физическом уровне модели рассматривается использование конкретной СУБД, определяются типы данных (например, целое или вещественное число), индексы для таблиц.

ERwin предоставляет возможности создавать и управлять этими двумя различными уровнями представления одной диаграммы (модели), равно как и иметь много вариантов отображения на каждом уровне. Термин “логический уровень” в ERwin соответствует концептуальной модели.

Этапы построения информационной модели:

- определение сущностей;
- определение зависимостей между сущностями;
- задание первичных и альтернативных ключей;
- определение атрибутов сущностей;
- приведение модели к требуемому уровню нормальной формы;
- переход к физическому описанию модели: назначение соответствий имя сущности – имя таблицы, атрибут сущности – атрибут таблицы;
- задание триггеров, процедур и ограничений;
- генерация базы данных.

Erwin создает визуальное представление (модель данных) для решаемой задачи. Это представление может использоваться для детального анализа, уточнения и распространения документации, необходимой в цикле разработки. Однако ERwin далеко

не только инструмент для рисования. ERwin автоматически создает базу данных (таблицы, индексы, хранимые процедуры, триггеры для обеспечения ссылочной целостности и другие объекты, необходимые для управления данными).

Создание сущности

Для внесения сущности в модель необходимо щелкнуть по кнопке сущности на панели инструментов (ErwinToolbox)  , затем – по тому месту на диаграмме, где необходимо расположить новую сущность. Щелкнув правой кнопкой мыши по сущности и выбрав из всплывающего меню пункт EntityEditor, можно вызвать диалог EntityEditor, в котором определяются имя, описание и комментарии сущности.

Каждая сущность должна быть полностью определена с помощью текстового описания в закладке Definition. Эти определения полезны как на логическом уровне, поскольку позволяют понять, что это за объект, так и на физическом уровне, поскольку их можно экспортировать как часть схемы и использовать в реальной БД (CREATECOMMENTonentity_name). Закладки Note, Note2, Note3, UDP (UserDefinedProperties – Свойства, определенные пользователем) служат для внесения дополнительных комментариев и определений к сущности.

В закладке Icon каждой сущности можно поставить в соответствие изображение, которое будет отображаться в режиме просмотра модели на уровне иконок и изображение, которое будет отображаться на всех других уровнях.

Закладка UDP диалога EntityEditor служит для определения свойств, определяемых пользователем (User – DefinedProperties). При нажатии на кнопку  этой закладки вызывается диалог User – DefinedPropertyEditor (также вызывается из меню Edit/UDPs). В нем необходимо указать вид объекта, для которого заводится UDP (диаграмма в целом, сущность, атрибут и т.д.) и тип данных. Для внесения нового свойства следует щелкнуть в таблице по кнопке  и внести имя, тип данных, значение по умолчанию и определение.

Создание атрибутов

Для описания атрибутов следует, щелкнув правой кнопкой по сущности, выбрать в появившемся меню пункт

AttributeEditor. Появится диалог AttributeEditor.

Если щелкнуть по кнопке New, то в появившемся диалоге NewAttribute можно указать имя атрибута, имя соответствующей ему в физической модели колонки и домен. Домен атрибута будет использоваться при определении типа колонки на уровне физической модели.

Для атрибутов первичного ключа в закладке General диалога AttributeEditor необходимо сделать пометку в окне выбора PrimaryKey.

Закладки Definition, Note и UDP несут те же функции, что и при определении сущности, но на уровне атрибутов.

Для большей наглядности диаграммы каждый атрибут можно связать с иконкой. Это можно сделать при помощи списка выбора Icon в закладке General.

Очень важно дать атрибуту правильное имя. Атрибуты должны именоваться в единственном числе и иметь четкое смысловое значение.

Согласно синтаксису IDEF1X, имя атрибута должно быть уникальным в рамках модели (а не только в рамках сущности!). По умолчанию при попытке внесения уже существующего имени атрибута ERwin переименовывает его. Например, если атрибут Комментарий уже существует в модели, другой атрибут (в другой сущности) будет назван Комментарий/2, затем Комментарий/3 и т.д.

При переносе атрибутов внутри и между сущностями можно воспользоваться техникой drag&drop, выбрав кнопку  в палитре инструментов.

Создание связи

Для создания новой связи следует выбрать идентифицирующую или неидентифицирующую связь в палитре инструментов (ERwinToolbox), щелкнуть сначала по родительской, а затем по дочерней сущности.

В палитре инструментов кнопка  соответствует идентифицирующей связи, кнопка  связи многие-ко-многим и кнопка  соответствует неидентифицирующей связи.

Для редактирования свойств связи следует щелкнуть пра-

вой кнопкой мыши по связи и выбрать на контекстном меню пункт RelationshipEditor.

В закладке General появившегося диалога можно задать мощность, имя и тип связи.

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

Различают четыре типа мощности:

общий случай, когда одному экземпляру родительской сущности соответствуют 0, 1 или много экземпляров дочерней сущности, не помечается каким-либо символом;

символом P помечается случай, когда одному экземпляру родительской сущности соответствуют 1 или много экземпляров дочерней сущности (исключено нулевое значение);

символом Z помечается случай, когда одному экземпляру родительской сущности соответствуют 0 или 1 экземпляр дочерней сущности (исключены множественные значения);

цифрой помечается случай, когда одному экземпляру родительской сущности соответствует заранее заданное число экземпляров дочерней сущности.

По умолчанию символ, обозначающий мощность связи, не показывается на диаграмме. Для отображения имени следует в контекстном меню, которое появляется, если щелкнуть правой кнопкой мыши по любому месту диаграммы, не занятому объектами модели, выбрать пункт DisplayOptions/Relationship и затем включить опцию Cardinality.

Тип связи (идентифицирующая/неидентифицирующая).

В IDEF1X различают зависимые и независимые сущности. Тип сущности определяется ее связью с другими сущностями. Идентифицирующая связь устанавливается между независимой (родительский конец связи) и зависимой (дочерний конец связи) сущностями. Когда рисуется идентифицирующая связь, ERwin автоматически преобразует дочернюю связь в зависимую. Зависимая сущность изображается прямоугольником со скругленными углами.

Экземпляр зависимой сущности определяется только через отношение к родительской сущности. При установлении идентифицирующей связи атрибуты первичного ключа роди-

тельской сущности автоматически переносятся в состав первичного ключа дочерней сущности. Эта операция дополнения атрибутов дочерней сущности при создании связи называется миграцией атрибутов. В дочерней сущности новые атрибуты помечаются как внешние ключи – (FK).

При установлении не идентифицирующей связи дочерняя сущность остается независимой, а атрибуты первичного ключа родительской сущности мигрируют в состав не ключевых компонентов дочерней. Не идентифицирующая связь служит для связи независимых сущностей.

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

Для не идентифицирующей связи можно указать обязательность (Nulls в закладке General диалога RelationshipEditor). В случае обязательной связи (NoNulls) при генерации схемы БД атрибут внешнего ключа получит признак NOTNULL, несмотря на то, что внешний ключ не войдет в состав первичного ключа дочерней сущности. В случае необязательной связи (NullsAllowed) внешний ключ может принимать значение NULL. Необязательная не идентифицирующая связь помечается прозрачным ромбом со стороны родительской сущности

Имя связи (VerbPhrase) – фраза, характеризующая отношение между родительской и дочерней сущностями. Для связи один-ко-многим идентифицирующей или не идентифицирующей достаточно указать имя, характеризующей отношение от родительской к дочерней сущности (Parent-to-Child). Для связи многие-ко-многим следует указывать имена как Parent-to-Child, так и Child-to-Parent. Для отображения имени следует в контекстном меню, которое появляется, если щелкнуть правой кнопкой мыши по любому месту диаграммы, не занятому объектами модели, выбрать пункт DisplayOptions/Relationship и затем включить опцию VerbPhrase.

Имя роли или функциональное имя (Rolename) – это синоним атрибута внешнего ключа, который показывает, какую роль играет атрибут в дочерней сущности. Задать имя роли можно в закладке Rolename/RIActions диалога RelationshipEditor.

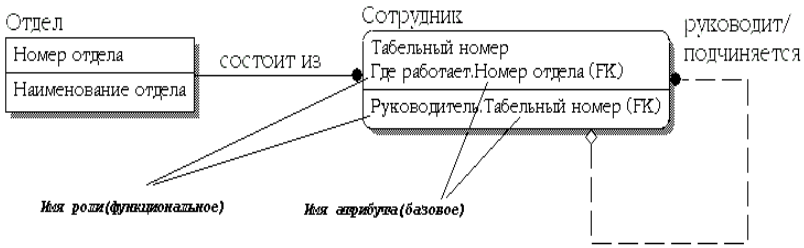

Рис. 8. Имена ролей внешних ключей

В примере, приведенном на рис.8, в сущности Сотрудник внешний ключ Номер отдела имеет имя роли «Где работает», которое показывает, какую роль играет этот атрибут в сущности. По умолчанию в списке атрибутов показывается только имя роли. Для отображения полного имени атрибута (как функционального имени, так и имени роли) следует в контекстном меню, которое появляется, если щелкнуть правой кнопкой мыши по любому месту диаграммы, не занятому объектами модели, выбрать пункт DisplayOptions/Entities и затем включить опцию Rolename/Attribute. Полное имя показывается как функциональное имя и базовое имя, разделенные точкой (рис. 8).

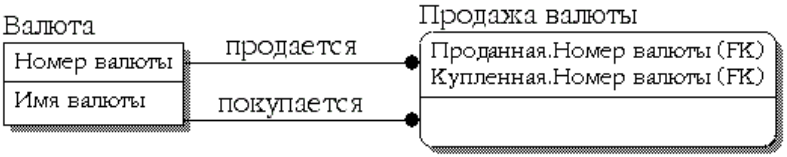

Рис. 9. Случай обязательности имен ролей

Обязательным является применение имен ролей в том случае, когда два или более атрибутов одной сущности определены по одной и той же области, т.е. они имеют одну и ту же область значений, но разный смысл.

На рис. 9 сущность Продажа валюты содержит информацию об акте обмена валюты, в котором участвуют две валюты –

проданная и купленная. Информация о валютах содержится в сущности Валюта. Следовательно, сущности Продажа валюты и Валюта должны быть связаны дважды, и первичный ключ – Номер валюты должен дважды мигрировать в сущность Валюта в качестве внешнего ключа. Необходимо различать эти атрибуты, которые содержат информацию о номере проданной и купленной валюты (имеют разный смысл), но ссылаются на одну и ту же сущность Валюта (имеют общую область значений). В примере на рис. 9 атрибуты получили имена ролей Проданная и Купленная.

Другим примером обязательного применения имен ролей являются рекурсивные связи, когда одна и та же сущность является и родительской и дочерней одновременно.

Правила ссылочной целостности (ReferentialIntegrity (RI)) – логические конструкции, которые выражают бизнес–правила использования данных и представляют собой правила вставки, замены и удаления. Задать правила ссылочной целостности можно в закладке Rolename/RIActions диалога RelationshipEditor.

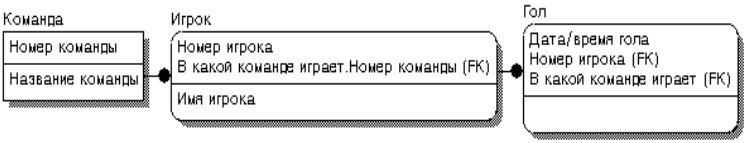

Рис. 10. Миграция имен ролей

При генерации схемы БД на основе опций логической модели будут сгенерированы правила декларативной ссылочной целостности, которые должны быть предписаны для каждой связи, и триггеры, обеспечивающие ссылочную целостность.

На рис. 10 существует идентифицирующая связь между сущностями Команда и Игрок. Что будет, если удалить команду? Экземпляр сущности Игрок не может существовать без команды (атрибут первичного ключа В какой команде играет. Номер команды не может принимать значение NULL), следовательно, нужно либо запретить удаление команды, пока в ней числится хотя бы один игрок, либо удалять вместе с командой и

всех ее игроков. Такие правила удаления (ParentDelete) называются ParentRestrict (ограничение) и ParentCascade (каскад). Сущности Игрок и Гол, в свою очередь, тоже связаны идентифицирующей связью и, если на удаление игрока наложено правило каскадного удаления всех записей о его голах, то при удалении команды будут удалены все игроки команды и все голы, забитые этими игроками.

Связь многие-ко-многим возможна только на уровне логической модели данных. Такая связь обозначается сплошной линией с двумя точками на концах. Для внесения связи следует сначала нажать на кнопку  в палитре инструментов (ERwinToolbox), а затем по очереди щелкнуть по обеим связанным сущностям.

Связь многие-ко-многим должна именоваться (VerbPhrase) двумя фразами – в обе стороны. Это облегчает чтение диаграммы.

Создание ключей

Каждый экземпляр сущности должен быть уникален.

Первичный ключ (primarykey) – это атрибут или группа атрибутов, однозначно идентифицирующие экземпляр сущности. Атрибуты первичного ключа на диаграмме не требуют специального обозначения – это те атрибуты, которые находятся в списке атрибутов выше горизонтальной линии. При внесении нового атрибута в диалоге AttributeEditor для того, чтобы сделать его атрибутом первичного ключа, нужно включить флажок PrimaryKey в нижней части закладки General. На диаграмме ключевой атрибут можно внести в состав первичного ключа, воспользовавшись режимом переноса атрибутов (кнопка  в палитре инструментов).

В одной сущности может оказаться несколько атрибутов или наборов атрибутов, претендующих на роль первичного ключа. Такие претенденты называются потенциальными ключами (candidatekey).

Ключи могут быть сложными, т.е. содержащими несколько атрибутов. Сложные первичные ключи не требуют специального обозначения – это список атрибутов выше горизонтальной линии.

При выборе первичного ключа предпочтение должно отдаваться более простым ключам, т.е. ключам, содержащим меньшее количество атрибутов.

Многие сущности имеют только один потенциальный ключ. Такой ключ становится первичным. Некоторые сущности могут иметь более одного возможного ключа. Тогда один из них становится первичным, а остальные – альтернативными ключами. Альтернативный ключ (AlternativeKey) – это потенциальный ключ, не ставший первичным.

Каждому ключу соответствует индекс, имя которого также присваивается автоматически. Имена ключа и индекса при желании можно изменить вручную.

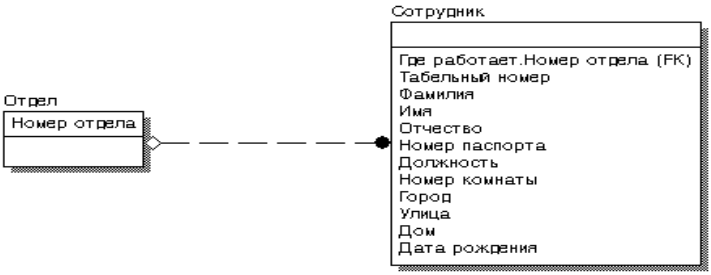

Рис. 11. Сущность «Сотрудник» с отображением ключей

На диаграмме атрибуты альтернативных ключей обозначаются как (Akn.m.), где n – порядковый номер ключа, m – порядковый номер атрибута в ключе.

Внешние ключи (ForeignKey) создаются автоматически, когда связь соединяет сущности: связи образуют ссылку на атрибуты первичного ключа в дочерней сущности и эти атрибуты образуют внешний ключ в дочерней сущности (миграция ключа). Атрибуты внешнего ключа обозначаются символом (FK) после своего имени (рис.11). Атрибут внешнего ключа Где работает.Номер отдела (“Где работает” – имя роли) сущности Сотрудник является атрибутом первичного ключа (PK) в сущности Отдел.

Зависимая сущность может иметь один и тот же ключ из нескольких родительских сущностей. Сущность может также по-

лучить 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, он распознает, что два атрибута одинаковы, и помещает атрибуты внешнего ключа в зависимой сущности только один раз. Это называется унификацией.

Есть случаи, когда унификация нежелательна. Например, когда два атрибута имеют одинаковые имена, но на самом деле они отличаются по смыслу. В этом случае необходимо использовать имена ролей внешнего ключа (рис. 9).

Домены

Домен можно определить как совокупность значений, из которых берутся значения атрибутов. Каждый атрибут может быть определен только на одном домене, но на каждом домене может быть определено множество атрибутов. В понятие домена входит не только тип данных, но и область значений данных. Например, домен “Возраст” можно определить как положительное целое число и определить атрибут Возраст сотрудника как принадлежащий этому домену.

В ERwin домен может быть определен только один раз и использоваться как в логической, так и в физической модели.

На логическом уровне домены можно описать без конкретных физических свойств. На физическом уровне они получают специфические свойства, которые можно изменить вручную. Так, домен “Возраст” может иметь на логическом уровне тип Number, на физическом уровне домену будет присвоен тип INTEGER.

Для создания домена в логической модели служит диалог DomainDictionaryEditor. Его можно вызвать из меню Edit/DomainDictionary. Для создания нового домена в диалоге DomainDictionaryEditor следует:

- щелкнуть по кнопке New. Появляется диалог NewDomain;
- выбрать родительский домен из списка DomainParent.

Новый домен можно создать на основе уже созданного пользователем домена, либо на основе изначально существующего. По умолчанию ERwin имеет четыре предопределенных доменов (String, Number, Blob, Datetime). Новый домен наследует все

свойства родительского домена. Эти свойства в дальнейшем можно переопределить;

- набрать имя домена в поле LogicalName. Можно также указать имя домена на физическом уровне в поле PhysicalName. Если физическое имя не указано, по умолчанию оно принимает значение логического имени;
- щелкнуть по кнопке OK;

В диалоге DomainDictionaryEditor можно связать домен с иконкой, с которой он будет отображаться в списке доменов (DomainIcon), иконкой, с которой атрибут, определенный на домене будет отображаться в модели (IconInheritedbyAttribute).

Каждый домен может быть описан в закладке Definition, снабжен комментарием в закладке Note или свойством определенным пользователем в закладке UDP.

ERwin имеет специальный инструмент, который значительно облегчает создание новых атрибутов в модели, используя описание доменов, - IndependentAttributeBrowser. Этот диалог вызывается (и скрывается) по горячему ключу CTRL+B. С его помощью можно выбрать в списке домен и по методу drag&drop перенести его в какую-либо сущность. В ней будет создан новый атрибут с именем, которое следует задать в окне NameInheritedbyAttribute диалога DomainDictionaryEditor. Если значение поля не задано, по умолчанию принимается имя домена.

На физическом уровне диалог DomainDictionaryEditor позволяет редактировать физические свойства домена. Имя этой закладки зависит от выбранного сервера БД. На ней можно задать конкретный тип данных, соответствующих домену, правила присвоения NULL – значений, правила валидации (правила проверки допустимых значений) и задания значения по умолчанию. Правила валидации и значения по умолчанию должны быть предварительно описаны и именованы. Для вызова диалогов редактирования правил валидации и значений по умолчанию служат кнопки  справа от соответствующего списка выбора (Valid и Default).

Функции других закладок диалога DomainDictionaryEditor:

General. Задание родительского домена (DomainParent) и имени, присваиваемого колонке при ее создании с помощью

IndependentColumnBrowser. С помощью опции PhisicalOnly домен можно определить только на уровне физической модели.

Comment. Внесение комментария к атрибуту.

UDP. Свойства, определяемые пользователем.

VisualBasic – PowerBuilder. Задание специальных свойств домена для кодогенерации клиентского приложения.

Задание. На основе ранее созданной функциональной модели и описания заданного отдела создать логическую модель с использованием пакета ERwin.

Вопросы

1. Каково назначение пакета ERwin и его основные функции?
2. В чем состоят главные преимущества пакета ERwin?
3. Опишите этапы построения информационной модели.
4. Из каких элементов состоит диаграмма "сущность-связь"?
5. Опишите характеристики связей в методологии IDEF1X.
6. Какие типы ключей используются в пакете ERwin, каково их назначение?
7. Каково предназначение доменов, приведите примеры доменов различного вида.

Практическая работа №6
“Создание физической модели в ERwin”

Сервер

Физический уровень представления модели зависит от выбранного сервера. Для выбора СУБД служит редактор TargetServer (меню Server/TargetServer... доступно только на физическом уровне).

ERwin поддерживает практически все распространенные СУБД, всего более 20 реляционных и нереляционных БД. Диалог TargetServer позволяет задать тип данных и опцию NULL для новых колонок, а также правила ссылочной целостности, принимаемые по умолчанию. Группа кнопок DefaultNonKeyNullOption позволяет разрешить или запретить значения NULL для неключевых колонок.

По умолчанию ERwin генерирует имена таблиц и индексов по шаблону на основе имен соответствующих сущностей и ключей логической модели. Окна TableNameMacro и IndexNameMacro позволяют изменить шаблон генерации имен.

Кнопка ResetNames вызывает диалог GloballyResetDBMSProperty, который позволяет заменить все имена таблиц, связей, индексов, колонок и соответствующих свойств, заданных вручную, на значения по умолчанию.

Имена таблиц и колонок по умолчанию будут сгенерированы на основе имен сущностей и атрибутов логической модели. Если в имени сущности или атрибута встречается пробел, он будет заменен на символ "_".

При смене СУБД ERwin предлагает автоматически преобразовать тип данных, связанный с каждым атрибутом, на ближайший, доступный для новой СУБД.

Таблицы

Для внесения новой таблицы в модель на физическом уровне служит кнопка на палитре инструментов. Связи между таблицами создаются так же, как на логическом уровне. Щелкнув правой клавишей мыши по таблице и выбрав во всплывающем меню пункты TableEditor или ColumnEditor, можно вызвать редакторы для задания свойств таблиц и колонок.

ERwin автоматически создает имена таблиц и колонок на основе имен соответствующих сущностей и атрибутов, учитывая максимальную длину имени и другие синтаксические ограничения, накладываемые СУБД. При генерации имени таблицы или колонки по умолчанию все пробелы автоматически преобразуются в символы подчеркивания, а длина имени обрезается до максимальной длины, допустимой для выбранной СУБД. Все изменения, сделанные в TableEditor или ColumnEditor, не отражаются на именах сущностей и атрибутов, поскольку информация на логическом и физическом уровнях в ERwin хранится отдельно.

Редактор TableEditor позволяет задать свойства любой таблицы модели, отличные от значения по умолчанию, в том числе имя таблицы, синонимы, правила валидации, процедуры и т. д. Переключиться на другую таблицу можно при помощи раскрывающегося списка выбора в верхней части диалога.

Окно Name служит для задания имени текущей таблицы. Окно Owner позволяет внести имя владельца таблицы, отличное от имени пользователя, производящего генерацию схемы БД. Окно выбора PhysicalOnly служит для создания объектов только на физическом уровне. Если выбрана опция Generate, при генерации схемы БД будет выполняться команда CREATETABLE. Кнопка DBSync служит для немедленной синхронизации модели с системным каталогом БД.

Колонки

Для задания свойств колонок, отличных от значения по умолчанию, служит редактор ColumnEditor. Чтобы вызвать его, нужно щелкнуть правой клавишей мыши по таблице и выбрать во всплывающем меню пункт ColumnEditor.

По умолчанию ERwin присваивает режимы нулевых значений всем неключевым колонкам, исходя из значений по умолчанию, устанавливаемых в редакторе TargetServer. Для колонок первичного ключа и альтернативных ключей устанавливается режим NOTNULL.

При создании связи колонки первичного ключа родительской таблицы мигрируют в состав колонок дочерней таблицы в качестве внешнего ключа. Кнопка Migrate вызывает диалог

MigrateColumnProperty, который позволяет определять, какие характеристики мигрировавшей колонки будут сохранены в дочерней таблице.

Для переноса каких-либо характеристик колонки необходимо включить соответствующую опцию в диалоге MigrateColumnProperty, для отказа от переноса - выключить. Опциями диалога следует пользоваться осторожно, во-первых, потому, что новые свойства колонки перезаписывают старые, а во-вторых, поскольку установленные опции действуют в рамках всей диаграммы, а не только текущей таблицы.

Представления

Представления (view), или, как их иногда называют, временные или производные таблицы, представляют собой объекты БД, данные в которых не хранятся постоянно, как в таблице, а формируются динамически при обращении к представлению. Представление не может существовать само по себе, а определяется только в терминах одной или нескольких таблиц. Применение представлений позволяет разработчику БД обеспечить каждому пользователю или группе пользователей свой взгляд на данные, что решает проблемы простоты использования и безопасности данных.

ERwin имеет специальные инструменты для создания и редактирования представлений. Палитра инструментов на физическом уровне содержит кнопки внесения представлений и установления связей между таблицами и представлениями. Для внесения представления нужно щелкнуть по кнопке  в палитре инструментов, затем по свободному месту диаграммы. По умолчанию представление получает номер V_n, где n - уникальный порядковый номер представления. Для установления связи нужно щелкнуть по кнопке  , затем по родительской таблице и, наконец, по представлению. Связи с представлениями и прямоугольники представлений показываются на диаграмме пунктирными линиями.

Для редактирования представления служит диалог ViewEditor. Для его вызова следует щелкнуть правой кнопкой мыши по представлению и выбрать в меню пункт ViewEditor.

Правила валидации и значения по умолчанию

ERwin поддерживает правила валидации для колонок, а также значение, присваиваемое колонкам по умолчанию. Правило валидации задает список допустимых значений для конкретной колонки и/или правила проверки допустимых значений. Значение по умолчанию - значение, которое нужно ввести в колонку, если никакое другое значение не задано явным образом во время ввода данных. С каждой колонкой или доменом можно связать значение по умолчанию (если выбранная СУБД поддерживает домены).

Если щелкнуть по кнопке  , расположенной справа от раскрывающегося списка Valid, появляется диалог ValidationRuleEditor, который служит для задания правил валидации. В нем можно задать максимальное и минимальное значение и тип валидации (где проверять - на сервере или в клиентском приложении).

Например, значение, вводимое в колонку Age, должно быть больше 18, но меньше 180. Для описания этого правила можно создать правило валидации с именем "Проверка_возраста", которое содержит выражение: AgeBETWEEN 18AND 180. Использование этого правила валидации гарантирует, что диапазон вводимых значений будет от 18 до 180. СУБД выдаст сообщение об ошибке, если вводимый возраст находится вне границ заданного диапазона.

После создания правила валидации и значения по умолчанию можно присвоить одной или нескольким колонкам или доменам.

Индексы

Чтобы решить проблему поиска данных, СУБД использует особый объект, называемый индексом. Он подобен содержанию книги, которое указывает на все номера страниц, посвященных конкретной теме. Индекс содержит отсортированную по колонке или нескольким колонкам информацию и указывает на строки, в которых хранится конкретное значение колонки.

Например, если необходимо найти клиента по имени (рис. 12), можно создать индекс по колонке CustomerName таблицы CUSTOMER. В индексе имена клиентов будут отсортированы в

алфавитном порядке. Для имени индекс будет содержать ссылку, указывающую, в каком месте таблицы хранится эта строка.

Для поиска клиента серверу направляется запрос с критерием поиска (CustomerName ="Иванов"). При выполнении запроса СУБД просматривает индекс, вместо того чтобы просматривать по порядку все строки таблицы CUSTOMER. Поскольку значения в индексе хранятся в определенном порядке, просматривать нужно гораздо меньший объем данных, что значительно уменьшает время выполнения запроса. Индекс можно создать для всех колонок таблицы, по которым часто производится поиск.

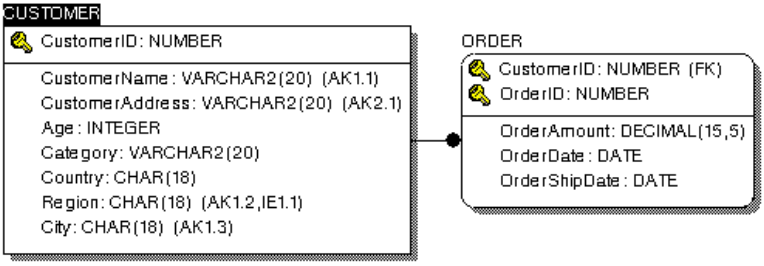

Рис. 12. Поиск с помощью индекса

При генерации схемы физической БД ERwin автоматически создает отдельный индекс на основе первичного ключа каждой таблицы, а также на основе всех альтернативных ключей, внешних ключей и инверсионных входов, поскольку эти колонки наиболее часто используются для поиска данных. Можно отказаться от генерации индексов по умолчанию и для повышения производительности создать собственные индексы. Администратор СУБД должен анализировать наиболее часто выполняемые запросы и создавать индексы с различными колонками и порядком сортировки для увеличения эффективности поиска при работе конкретных приложений.

Изменить характеристики существующего индекса или создать новый можно в редакторе IndexEditor. Для его вызова следует щелкнуть правой кнопкой мыши по таблице и выбрать во всплывающем меню пункт Index.

Прямое и обратное проектирование

Процесс генерации физической схемы БД из логической модели данных называется прямым проектированием (ForwardEngineering). При генерации физической схемы ERwin включает триггеры ссылочной целостности, хранимые процедуры, индексы, ограничения и другие возможности, доступные при определении таблиц в выбранной СУБД.

Процесс генерации логической модели из физической БД называется обратным проектированием (ReverseEngineering). ERwin позволяет создать модель данных путем обратного проектирования имеющейся БД. После того как модель создана, можно переключиться на другой сервер (модель будет конвертирована) и произвести прямое проектирование структуры БД для другой СУБД. Кроме режима прямого и обратного проектирования ERwin поддерживает синхронизацию между логической моделью и системным каталогом СУБД на протяжении всего жизненного цикла создания ИС.

Для генерации системного каталога БД следует выбрать пункт меню Tasks/ForwardEngineer/SchemaGeneration или нажать кнопку  на панели инструментов. Появляется диалог SchemaGeneration.

Диалог SchemaGeneration имеет три закладки:

Options. Служит для задания опций генерации объектов БД - триггеров, таблиц, представлений, колонок, индексов и т. д. Для задания опций генерации какого-либо объекта следует выбрать объект в левом списке закладки. после чего включить соответствующую опцию в правом списке.

В закладке Summary отображаются все опции, заданные в закладке Options. Список опций в Summary можно редактировать так же, как и в Options.

Comment. Позволяет внести комментарий для каждого набора опций. Каждый набор опций может быть именован (окно OptionSet, кнопки New, Rename и Delete) и использован многократно.

Кнопка Preview вызывает диалог SchemaGenerationPreview, в котором отображается SQL-скрипт, создаваемый ERwin для генерации системного каталога СУБД.

Нажатие на кнопку Generate приведет к запуску процесса генерации схемы.

Кнопка Print предназначена для вывода на печать создаваемого ERwin SQL-скрипта.

Кнопка Report сохраняет тот же скрипт в ERS или SQL текстовом файле. Эти команды можно в дальнейшем редактировать любым текстовым редактором и выполнять при помощи соответствующей утилиты сервера.

Кнопка Generate запускает процесс генерации схемы. Возникает диалог связи с БД, устанавливается сеанс связи с сервером и начинает выполняться SQL-скрипт. При этом возникает диалог GenerateDatabaseSchema.

Для выполнения обратного проектирования следует выбрать пункт меню Tasks/ReverseEngineer

При этом возникает диалог ERwinTemplateSelection, в котором нужно выбрать шаблон диаграммы, затем диалог выбора СУБД и, наконец, диалог задания опций обратного проектирования ReverseEngineer - SetOptions.

В диалоге ReverseEngineer -SetOptions можно задать следующие опции:

Группа ReverseEngineerFrom позволяет задать источник обратного проектирования - БД или SQL(DDL)-скрипт. При помощи кнопки Browse можно выбрать текстовый файл, содержащий SQL-скрипт.

Группа ItemstoReverseEngineer позволяет задать объекты БД, на основе которых будет создана модель. При помощи списка выбора OptionSet, а также кнопок New, Update и Delete можно создавать и редактировать именованные конфигурации объектов БД, которые могут быть использованы многократно при других сеансах обратного проектирования.

Группа ReverseEngineer (доступна только при обратном проектировании из БД) позволяет включить в модель системные объекты (окно выбора SystemObjects) и установить фильтр на извлекаемые таблицы по их владельцу.

В процессе работы модель может изменяться и дополняться. С другой стороны, системный каталог БД может редактироваться другими проектировщиками. В результате спустя некоторое время после последнего сеанса обратного проектиро-

вания могут возникнуть расхождения между реальным состоянием системного каталога и моделью данных.

Для синхронизации системного каталога БД и текущей модели следует выбрать пункт меню Tasks/CompleteCompare или нажать кнопку  на панели инструментов. Возникает диалог CompleteCompare - SetOptions, который во многом похож на описанный выше диалог ReverseEngineer-SetOptions. Разница заключается в том, что в отличие от обратного проектирования сравнивать текущую модель можно не только с БД или SQL-скриптом, но и с другой моделью ERwin, хранящейся в файле или репозиторииModelMart.

Задание. На основе логической модели автоматически получить в пакете ERwin физическую модель. Модифицировать модель по различным параметрам – серверам, таблицам, представлениям и т.п. Представить отчет в виде исходной и модифицированной моделей.

Вопросы

1. Когда возникает необходимость в редактировании физической модели?
2. Для чего предназначены представления, как их можно создать?
3. Что такое правила валидации, каким образом они задаются?
4. Каким образом в СУБД предусмотрено ускорение поиска информации?
5. Какой смысл в обратном проектировании базы данных? Что создается в результате этого процесса?

Практическая работа №7
“Создание отчетов в пакете ERwin“

Для генерации отчетов в ERwin имеется эффективный и простой в использовании инструмент - ReportBrowser. Он позволяет выполнять предопределенные отчеты (объединенные по типам), сохранять результаты их выполнения, создавать собственные отчеты, печатать и экспортировать их в распространенные форматы. Каждый отчет может быть настроен индивидуально, данные в нем могут быть отсортированы и отфильтрованы.

Диалог ReportBrowser вызывается кнопкой  в панели инструментов ERwin.

Диалог ReportBrowser имеет собственное меню и панель инструментов. Назначение кнопок панели инструментов показано в табл. 5.

Таблица 5 - Кнопки панели инструментов ReportBrowser

Кнопки	Назначение кнопки
	Создание нового отчета или папки
	Печать отчета
	Просмотр результата выполнения отчета
	Выполнение отчета
	Фиксация изменений (для редактируемого отчета)
	Поиск элементов отчета: задание условий поиска, поиск следующей строки и поиск другого отчета, соответствующего строке
	Включение и выключение дерева отчетов
	Показать список выполненных отчетов в хронологическом порядке
	Перейти к предыдущему отчету (при создании нового отчета на основе строки существующего)
	Выбор колонок и сортировка выполненного отчета
	Ассоциирование строки отчета с иконкой
	Сохранение выполненного отчета в виде представления

В верхней левой части диалога расположено окно, отображающее дерево отчетов. Отчеты могут быть сгруппированы в папки. Каждый отчет может включать несколько результирующих наборов данных, каждое из которых генерируется при очередном выполнении отчета. Каждый элемент дерева помечен иконкой:

 - папка;

 - отчет;

 - редактируемый отчет;

 - результирующий набор данных;

 - представление.

По умолчанию ReportBrowser содержит предварительно определенные отчеты, позволяющие наглядно представить информацию об основных объектах модели данных - как логической, так и физической. Для выполнения отчета достаточно дважды щелкнуть по нему в дереве отчетов или щелкнуть по соответствующей кнопке на панели инструментов. Результат выполнения отчета будет отображен в правом окне диалога ReportBrowser. Иконка результирующего набора будет также добавлена в дерево отчетов.

В нижней части диалога содержится дополнительная панель инструментов для управления деревом отчетов (табл.6).

Таблица 6 - Кнопки нижней панели инструментов ReportBrowser

Кнопка	Назначение кнопки
	Редактировать выделенный отчет
	Удалить отчет
	Показать только верхний уровень дерева
	Сделать выбранную папку корнем дерева (показать только выбранную ветку дерева)
	Сделать корнем дерева родительскую папку (по отношению к выбранной)

Создание нового отчета

Для создания нового отчета следует выбрать пункт меню File/NewERwinReport или щелкнуть по кнопке  на панели инструментов. Появляется диалог ERwinReportEditor.

В поле Name следует внести имя отчета. Категория отчета (Category) указывает на тип объектов модели, по которым будет создаваться отчет (атрибуты, сущности, домены, связи и т. д.).

Закладки Definition и Note служат соответственно для внесения определения и комментария к отчету.

Закладка Options отображает информацию, которая будет включена в отчет. В левой части закладки находится иерархический список категорий (Category). Папки в этом списке могут раскрываться и сворачиваться. Окно выбора ☑ позволяет включить соответствующий пункт списка в отчет. Иконка  показывает, что соответствующую колонку в полученном отчете можно будет редактировать. Папка с символом  позволяет выбрать условия фильтрации данных отчета, а с символом  - условия сортировки.

Кроме списка, закладка содержит следующие элементы управления:

• группу Options - позволяет выбрать режим отображения элементов в списке - показывать все возможные или только выбранные;

•CollapseAll - сворачивает все папки списка;

•ClearAll - отменяет все предварительно выбранные опции;

•ShowSelected - раскрывает папки с выбранными опциями.

После щелчка по кнопке OK отчет будет добавлен в список отчетов диалога ReportBrowser. Для выполнения отчета нужно либо дважды щелкнуть по его имени в списке, либо щелкнуть по кнопке  в палитре инструментов.

Существующий отчет, в том числе предопределенный, тоже можно изменить с помощью редактора, если в списке щелкнуть правой кнопкой мыши по имени отчета и выбрать во всплывающем меню пункт EditERwinReport.

Полученный после выполнения отчета результирующий набор данных можно отформатировать, распечатать, экспортировать или сохранить в виде представления.

Для форматирования результирующего набора данных следует в списке щелкнуть правой кнопкой мыши по имени набора и выбрать во всплывающем меню пункт Editreportformat. В появляющемся диалоге ReportFormat можно изменить сортировку данных, очередность колонок, сделать колонку невидимой, задать ее стиль.

Для редактирования результирующего набора данных следует в списке щелкнуть правой кнопкой мыши по имени набора и выбрать во всплывающем меню пункт Exportresultset. Допустимые форматы экспорта:

•CSV - текстовый файл;

•HTLM;

•DDE - экспорт в MSWord или MSExcel;

•RPTwin - экспорт в специализированный генератор отчетов.

После форматирования и настройки результирующего набора данных его можно сохранить в качестве именованного представления. Использование представлений облегчает использование отчетов, поскольку все настройки достаточно сделать один раз. Каждый отчет может иметь несколько представлений. Для создания представления следует установить фокус в списке на нужный набор и щелкнуть по кнопке  на панели инструментов. В диалоге SaveView следует указать имя и определение представления. После щелчка по кнопке OK представление добавится в список отчетов.

Задание

Создать отчеты по следующим типам: атрибуты, сущности, домены, связи. Внести определения и комментарии к отчетам. Полученный после выполнения отчета результирующий набор данных отформатировать, распечатать, сохранить в виде представления.

Вопросы

1. Каковы основные возможности генератора отчетов ReportBrowser?
2. Какие недостатки в работе данного пакета Вы видите?

Практическая работа №8
Методология IDEF0

Цель работы: выполнить построение диаграмм по методологии IDEF0.

Задачи работы: освоить приемы построения диаграмм по методологии IDEF0 с применением CASE-средства BPwin.

Содержание работы:

- построение диаграммы A0;
- построение диаграмм декомпозиции A0;
- построение диаграммы узлов;
- построение диаграммы FEO.

1.1 Теоретическая часть

В ходе реализации программы интегрированной компьютеризации производства (ICAM), предложенной в начале 80-х годов ВВС для аэрокосмической промышленности США, была выявлена потребность в разработке методов анализа взаимодействия процессов в производственных системах. Для удовлетворения этой потребности была разработана методология IDEF0 (IntegratedDefinitionFunctionModeling), которая в настоящее время принята в качестве федерального стандарта США.

Методология IDEF0 представляет собой совокупность методов, правил и процедур, предназначенных для построения функциональной модели объекта какой-либо предметной области. Функциональная модель IDEF0 отображает функциональную структуру объекта, т.е. производимые им действия и связи между этими действиями.

IDEF0 может быть использована для моделирования широкого класса систем. Для новых систем применение IDEF0 имеет своей целью определение требований и указание функций для последующей разработки системы, отвечающей поставленным требованиям и реализующей выделенные функции. Применительно к уже существующим системам IDEF0 может быть использована для анализа функций, выполняемых системой, и отображения механизмов, посредством которых эти функции выполняются.

Модель в IDEF0 представлена совокупностью иерархически упорядоченных и логически связанных диаграмм, а также текста документации и словарей, связанных друг с другом с помощью перекрестных ссылок.

Основу методологии IDEF0 составляет графический язык описания бизнес-процессов (графика блоков и дуг) (рисунок 1).

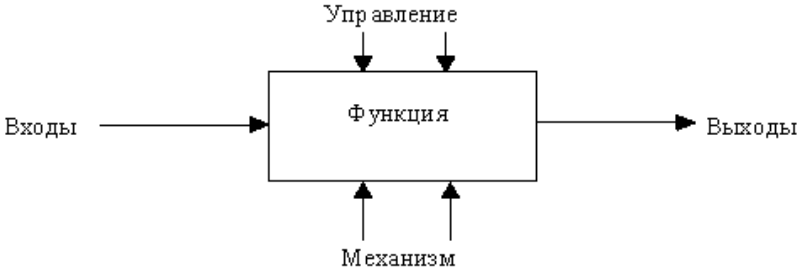

Рисунок 1 – Функциональный блок и дуги

Можно выделить четыре типа диаграмм:

- контекстную диаграмму Аз0 (в каждой модели может быть только одна контекстная диаграмма);
- диаграммы декомпозиции (в том числе диаграмма первого уровня декомпозиции А0, раскрывающая контекстную);
- диаграммы дерева узлов;
- диаграммы только для экспозиции (FEO).

Контекстная диаграмма представляет собой самое общее описание системы и ее взаимодействия с внешней средой. После описания системы в целом проводится разбиение ее на подсистемы. Этот процесс называется функциональной декомпозицией, а диаграммы, которые описывают каждый фрагмент, называются диаграммами декомпозиции. Каждый компонент модели может быть декомпозирован на другой диаграмме. Каждая диаграмма иллюстрирует "внутреннее строение" блока на родительской диаграмме. После декомпозиции контекстной диаграммы (т.е. получения диаграммы А0) проводится декомпозиция каждого блока диаграммы А0 на более мелкие фрагменты и так далее, до достижения нужного уровня подробности описа-

ния. После каждого сеанса декомпозиции проводятся сеансы экспертизы: эксперты предметной области (обычно это интервьюируемые аналитиками сотрудники предприятий) указывают на соответствие реальных бизнес-процессов созданным диаграммам. Найденные несоответствия исправляются, и только после прохождения экспертизы без замечаний можно приступать к следующему сеансу декомпозиции. Так достигается соответствие модели реальным бизнес-процессам на любом и каждом уровне модели. 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 сколько угодно, 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.

Правила IDEF0 включают:

- ограничение количества блоков на каждом уровне декомпозиции (правило 3-6 блоков);
- связность диаграмм (номера блоков);
- уникальность меток и наименований (отсутствие повторяющихся имен);
- синтаксические правила для графики (блоков и дуг);
- разделение входов и управлений (правило определения роли данных);
- отделение организации от функции, т.е. исключение влияния организационной структуры на функциональную модель.

1.2 Выполнение практической работы

Для выполнения последующего упражнения необходимо иметь результат выполнения предыдущего, поэтому рекомендуется сохранять модель, полученную в конце каждой лабораторной работы.

В качестве примера рассматривается деятельность вымышленной компании «Компьютер +». Компания занимается

сборкой и продажей настольных компьютеров и ноутбуков.

Основные виды работ в компании таковы:

- продавцы принимают заказы клиентов;
- операторы группируют заказы по типам компьютеров;
- операторы собирают и тестируют компьютеры;
- операторы упаковывают компьютеры согласно заказам;
- кладовщик отгружает клиентам заказы.

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

1.2.1 Для выполнения задания, необходимо запустить CASE-средство BPwin. Последовательность действий следующая.

1.2.1.1 Щелкните по кнопке  , появится диалоговое окно Iwouldliketo (рисунок 2). Внесите в текстовое поле Name имя модели "Деятельность компании" и выберите Type – BusinessProcess (IDEF0). Нажмите кнопку OK.

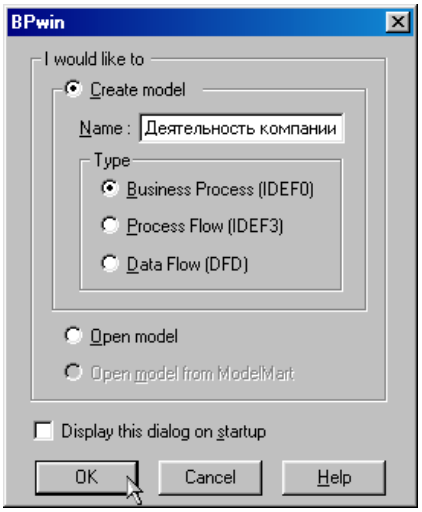

Рисунок 2 – Присвоение модели имени и выбор типа модели

1.2.1.2 В открывшемся диалоговом окне PropertiesforNewModels (Свойства новой модели) (рисунок 3) введите в текстовое поле Author (Автор) имя автора модели и в текстовое поле Authorinitials его инициалы; нажмите последовательно кнопки Apply и OK.

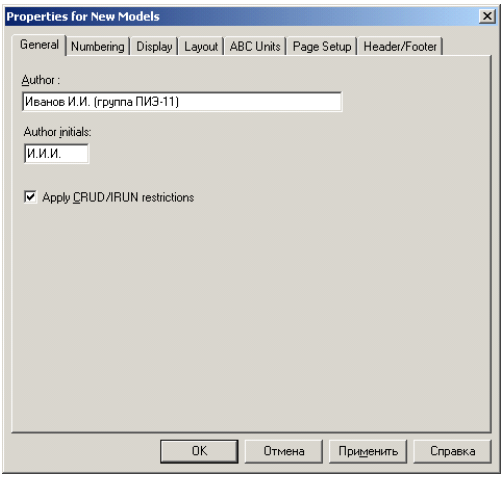

Рисунок 3 Ввод имени автора модели и его инициалов

1.2.1.3 Автоматически создается незаполненная контекстная диаграмма (рисунок 4).

1.2.1.4 Обратите внимание на кнопку  на панели инструментов. Эта кнопка включает и выключает инструмент просмотра и навигации ModelExplorer (Браузер модели). ModelExplorer имеет три вкладки – Activities (

), Diagrams (

) и Objects (

). Во вкладке Activities щелчок правой кнопкой по объекту в браузере модели позволяет выбрать опции редактирования его свойств (рисунок 5).

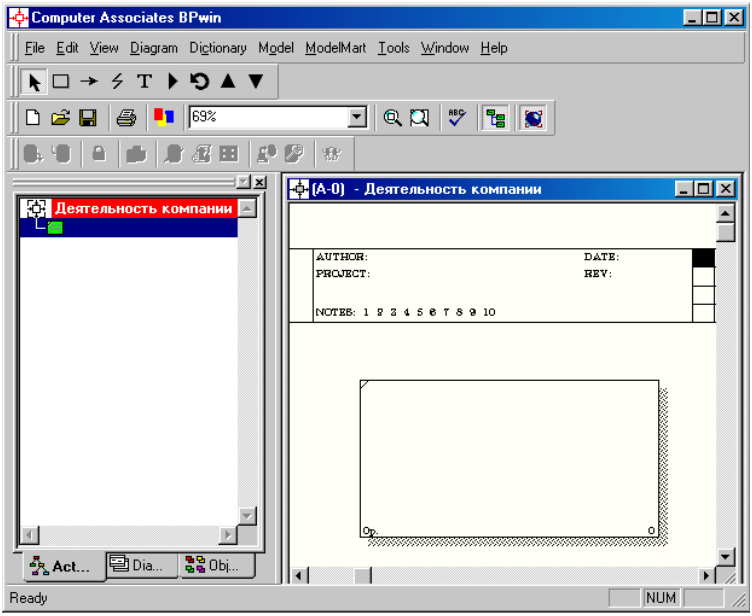

Рисунок 4 – Незаполненная контекстная диаграмма

1.2.1.5 Если вам непонятно, как выполнить то или иное действие, вы можете вызвать контекстную помощь клавиша F1 или воспользоваться меню Help.

1.2.1.6 Перейдите в меню Model/ModelProperties. Во вкладке General диалогового окна ModelProperties в текстовое поле Modelname следует внести имя модели "Деятельность компании", а в текстовое поле Project имя проекта "Модель деятельности компании", и, наконец, в текстовое TimeFrame (Временной охват) AS-IS (Как есть) (рисунок 6).

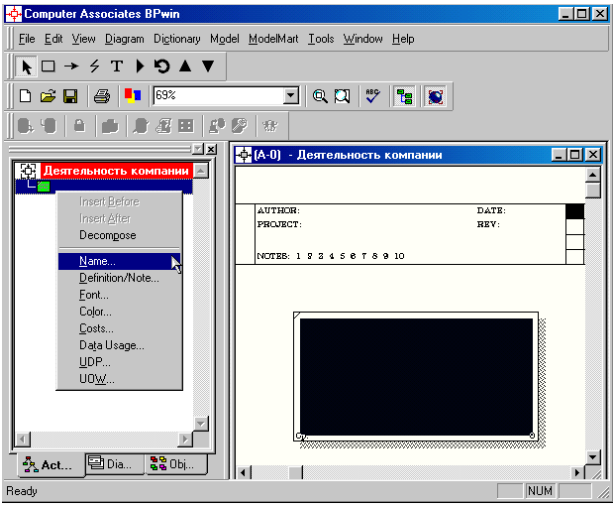

Рисунок 5 –Контекстное меню для редактирования свойств

1.2.1.7 Во вкладке Purpose диалогового окна ModelProperties в текстовое поле Purpose (Цель) внесите данные о цели разработки модели " Моделировать текущие (AS-IS) бизнес-процессы компании", а в текстовое поле Viewpoint (Точка зрения) "Директор" (рисунок 7).

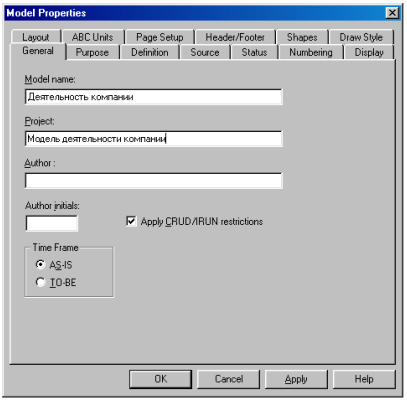

Рисунок 6 – Окно задания свойств модели

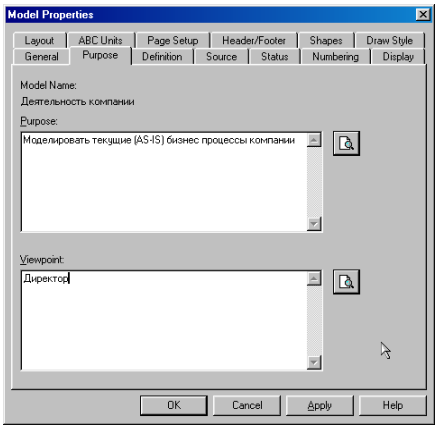

Рисунок 7 – Внесение данных о цели моделирования и точке зрения на модель

1.2.1.8 Во вкладке Definition диалогового окна ModelProperties в текстовое поле Definition (Определение) внесите "Это учебная модель, описывающая деятельность компании" и в текстовое поле Scope (Охват) "Общее управление бизнесом компании: исследование рынка, закупка компонентов, сборка, тестирование и продажа продуктов" (рисунок 8).

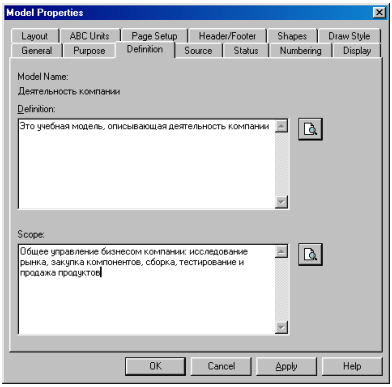

Рисунок 8 – Внесение дополнительных данных, определяющих модель

1.2.1.9 Перейдите на контекстную диаграмму и правой кнопкой мыши щелкните по прямоугольнику, представляющему в нотации IDEF0 условное графическое обозначение работы. В контекстном меню выберите опцию Name (рисунок 9). Во вкладке Name внесите имя "Деятельность компании" (рисунок 10).

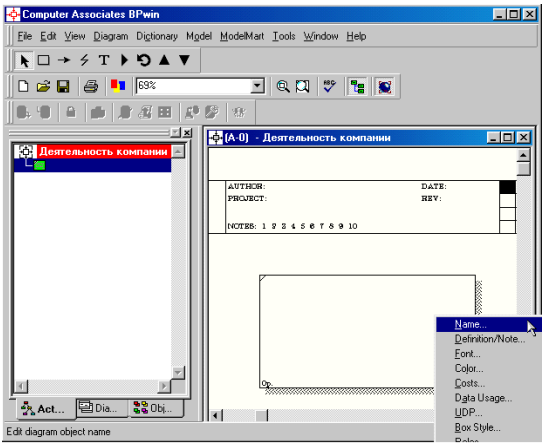

Рисунок 9 – Контекстное меню для работы с выбранной опцией Name

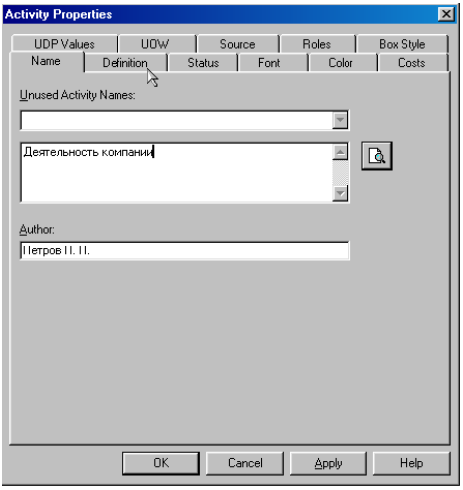

Рисунок 10 – Присвоение работе названия

1.2.1.10 Во вкладке Definition диалогового окна ActivityProperties в текстовое поле Definition (Определение) внесите "Текущие бизнес-процессы компании" (рисунок 11).

Текстовое поле Note (Примечания) оставьте незаполненным.

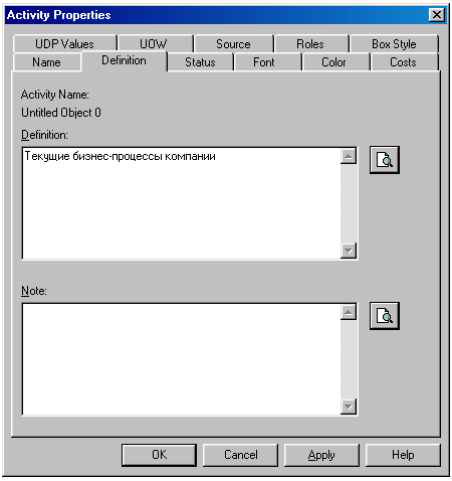

Рисунок 11 – Внесение дополнительных данных о работе

1.2.1.11 Создайте ICOM-стрелки на контекстной диаграмме (таблица 1).

Таблица 1 - Стрелки контекстной диаграммы

Название стрелки (ArrowName)	Определение стрелки (ArrowDefinition)	Тип стрелки (ArrowType)
Звонки клиентов	Запросы информации, заказы, техподдержка и т.д.	Input
Правила и процедуры	Правила продаж, инструкции по сборке, процедуры тестирования, критерии производительности	Control
Проданные продукты	Настольные и портативные компьютеры	Output
Бухгалтерская система	Оформление счетов, оплата счетов, работа с заказами	Mechanism

1.2.1.12 С помощью кнопки **T** внесите текст в поле диаграммы точку зрения и цель (рисунок 12).

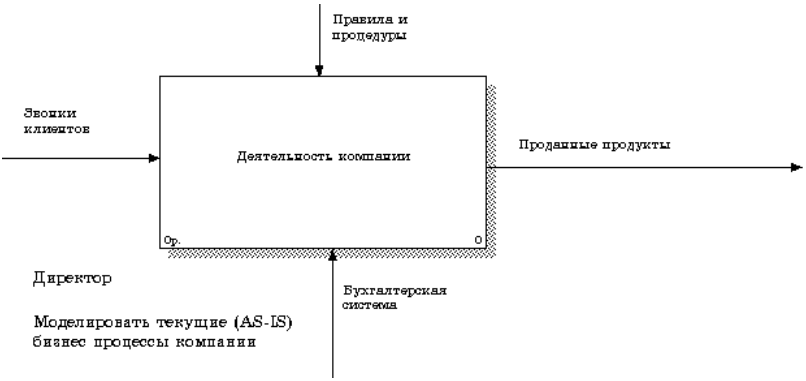

Рисунок 12 - Внесение текста в поле диаграммы с помощью редактора TextBlockEditor

Результат выполнения показан на рисунке 13.

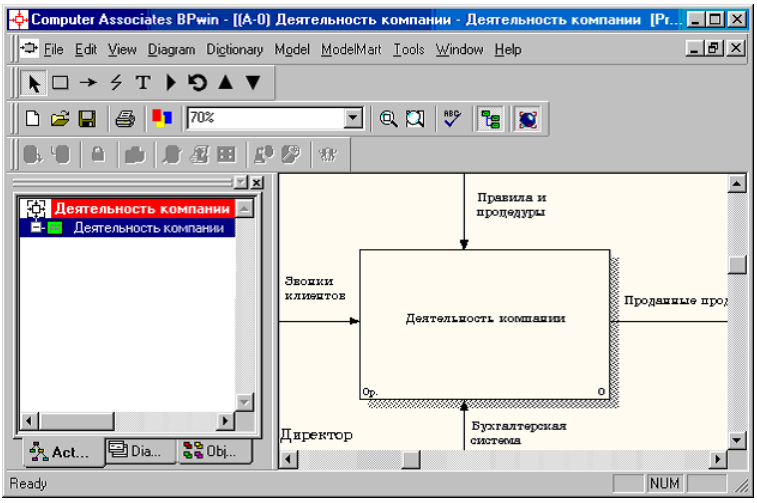

Рисунок 13 – Построенная контекстная диаграмма

1.2.1.13 Создайте отчет по модели. В меню Tools/Reports/ModelReport (рисунок 14) задайте опции генерирования отчета (установите галочки) и нажмите кнопку Preview (Предварительный просмотр) (рисунок 15).

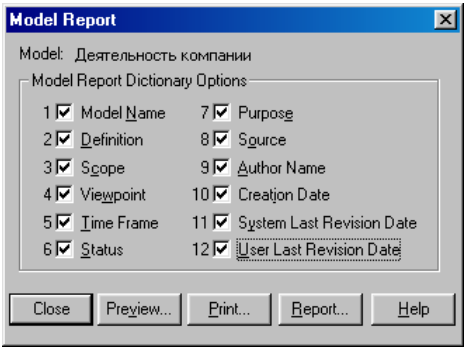

Рисунок 14 – Задание опций генерирования отчета ModelReport

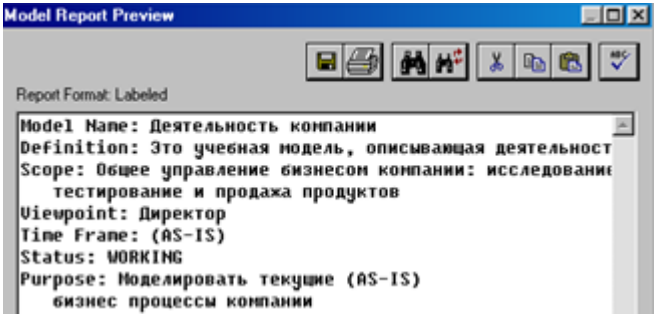

Рисунок 15 – Предварительный просмотр отчета ModelReport

1.2.2 Для создания диаграммы декомпозиции A0 необходимо выполнить следующее:

1.2.2.1 Выбрать кнопку ▼ перехода на нижний уровень в палитре инструментов, в диалоговом окне ActivityBoxCount (рисунок 16) установить число работ на диаграмме нижнего уровня 3 и нажать кнопку OK.

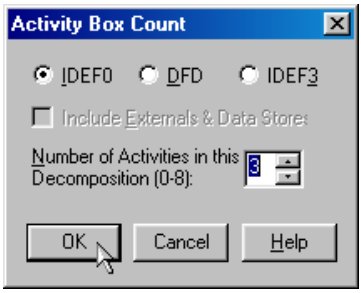

Рисунок 16 – Диалоговое окно ActivityBoxCount

1.2.2.2 Автоматически будет создана диаграмма декомпозиции (рисунок 17).

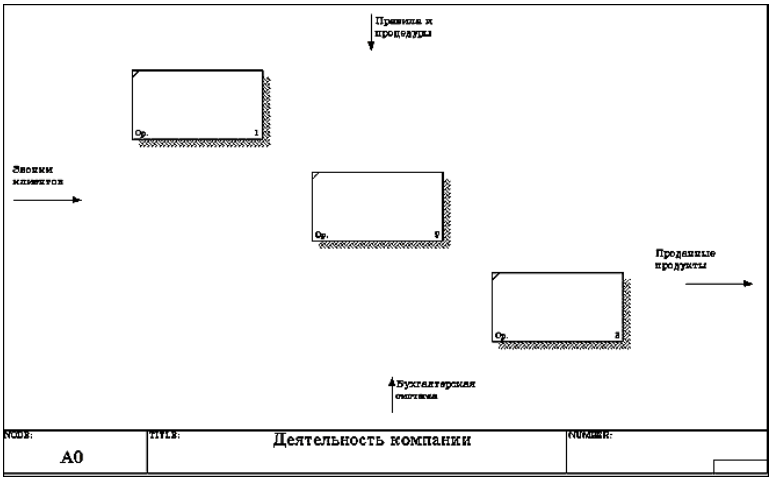

Рисунок 17 – Диаграмма декомпозиции

Правой кнопкой мыши щелкните по работе, расположенной в левом верхнем углу области редактирования модели, выберите в контекстном меню опцию Name и внесите имя работы.

Повторите операцию для оставшихся двух работ. Затем внесите определение, статус и источник для каждой работы согласно данным таблицы 2.

Таблица 2 - Работы диаграммы декомпозиции A0

Название (Activity Name)	Определение (Activity Definition)
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

Диаграмма декомпозиции примет вид, представленный на рисунке 18.

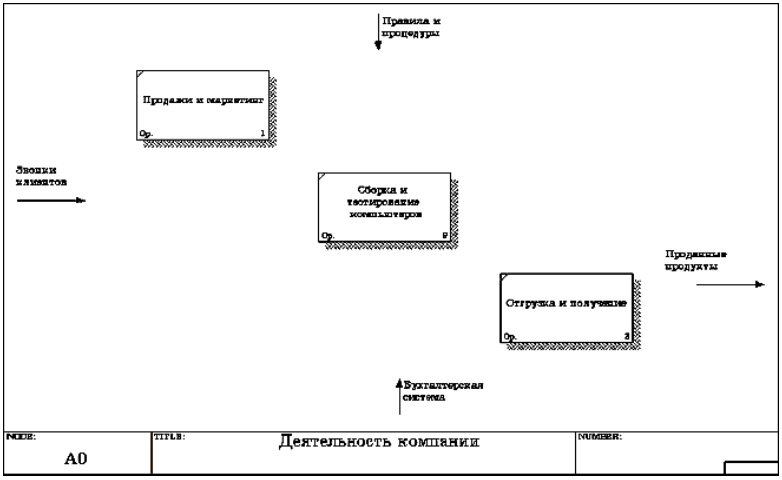

Рисунок 18 – Диаграмма декомпозиции после присвоения работам наименований

1.2.2.3 Для изменения свойств работ после их внесения в диаграмму можно воспользоваться словарем работ (рисунок 19).

Вызов словаря производится при помощи пункта главного меню Dictionary/Activity.

Рисунок 19 Словарь ActivityDictionary

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 которую нужно внести имя и свойства работы.

Для удаления всех имен работ, не использующихся в модели, щелкните по кнопке  (Purge (Чистить)).

1.2.2.4 Перейдите в режим рисования стрелок и свяжите граничные стрелки, воспользовавшись кнопкой  на палитре инструментов так, как это показано на рисунке 20.

1.2.2.5 Правой кнопкой мыши щелкните по ветви стрелки управления работы "Сборка и тестирование компьютеров" и переименуйте ее в "Правила сборки и тестирования" (рисунок 21).

Внесите определение для новой ветви: "Инструкции по сборке, процедуры тестирования, критерии производительности и т.д."

Правой кнопкой мыши щелкните по ветви стрелки механизма работы "Продажи и маркетинг" и переименуйте ее как "Система оформления заказов" (рисунок 22).

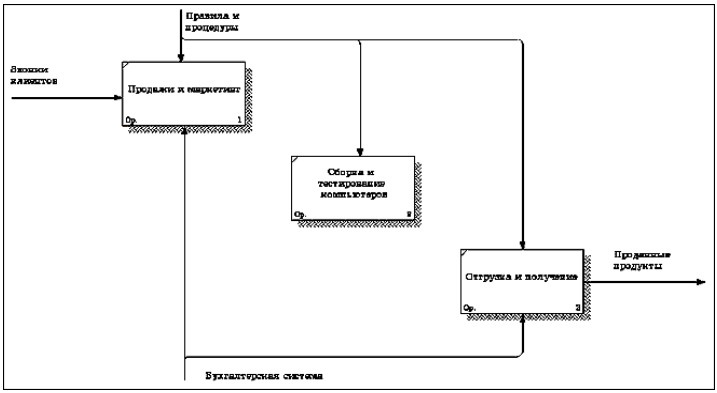

Рисунок 20 - Связанные граничные стрелки на диаграмме A0

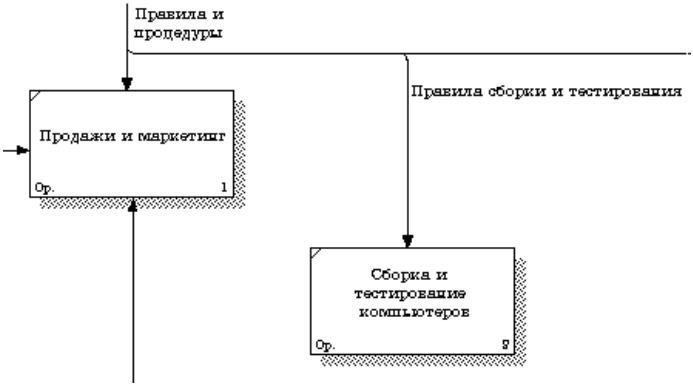

Рисунок 21 - Стрелка "Правила сборки и тестирования"

Рисунок 22 - Стрелка " Система оформления заказов"

1.2.2.6 Альтернативный метод внесения имен и свойств стрелок использование словаря стрелок (вызов словаря меню Dictionary/Arrow). Если внести имя и свойства стрелки в словарь (рисунок 23), ее можно будет внести в диаграмму позже.

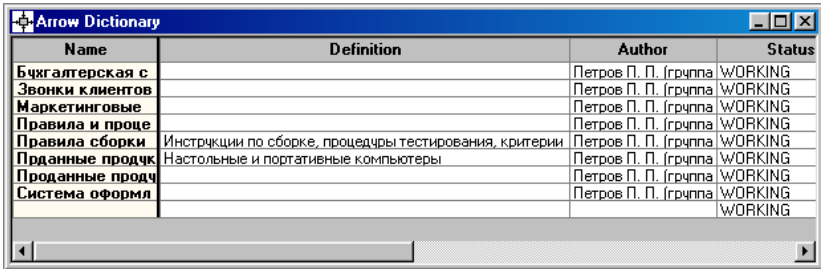

Рисунок 23 – Словарь стрелок

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 которую нужно внести имя и свойства стрелки.

1.2.2.7 Создайте новые внутренние стрелки так, как показано на рисунке 24.

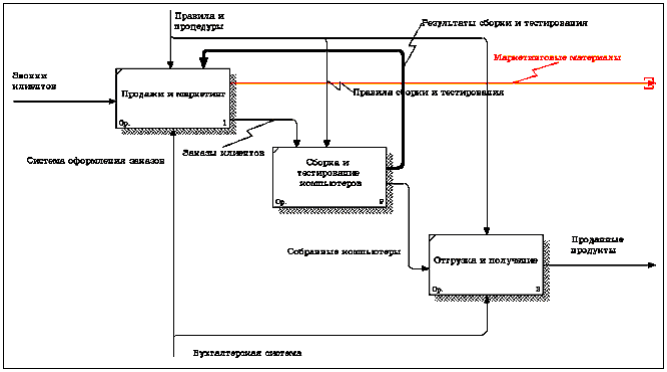

Рисунок 24 - Внутренние стрелки диаграммы A0

1.2.2.8 Создайте стрелку обратной связи (по управлению) "Результаты сборки и тестирования", идущую от работы "Сборка и тестирование компьютеров" к работе "Продажи и маркетинг". Измените, при необходимости, стиль стрелки (толщина линий) и установите опцию ExtraArrowhead (Дополнительный наконечник стрелы (из контекстного меню)). Методом drag&drop перенесите имена стрелок так, чтобы их было удобнее читать. Если необходимо, установите из контекстного меню Squiggle (Загогулина). Результат возможных изменений показан на рисунке 25.

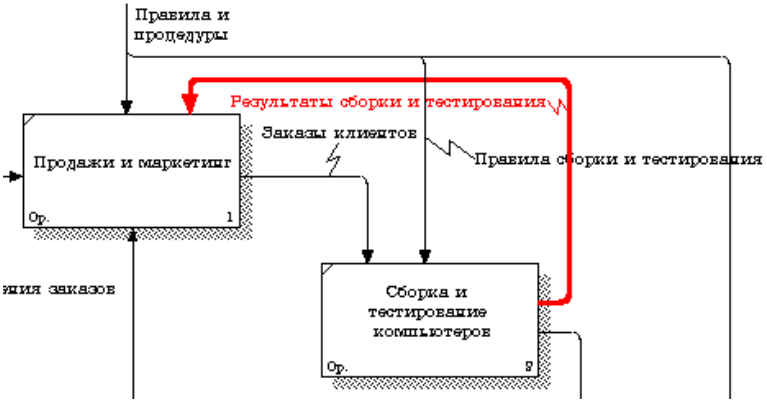

Рисунок 25 - Результат редактирования стрелок на диаграмме A0

1.2.2.9 Создайте новую граничную стрелку выхода "Маркетинговые материалы", выходящую из работы "Продажи и маркетинг".

Эта стрелка автоматически не попадает на диаграмму верхнего уровня и имеет квадратные скобки на наконечнике  (рисунок 26).

1.2.2.10 Щелкните правой кнопкой мыши по квадратным скобкам и выберите пункт меню ArrowTunnel (рисунок 27).

В диалоговом окне BorderArrowEditor (Редактор граничных стрелок) выберите опцию ResolveittoBorderArrow (Разрешить как граничную стрелку) (рисунок 28).

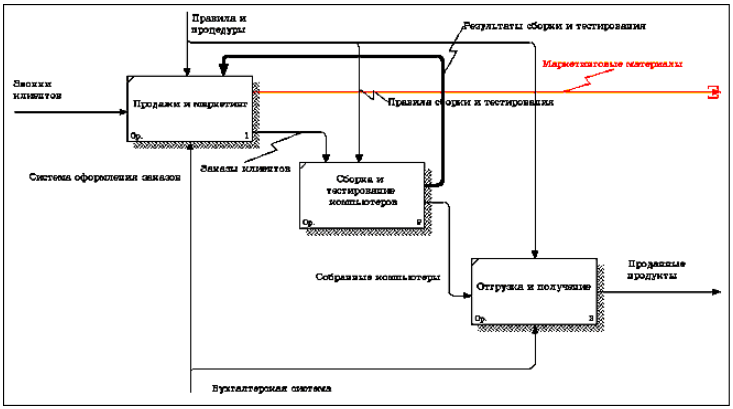

Рисунок 26 – Стрелка «Маркетинговые материалы»

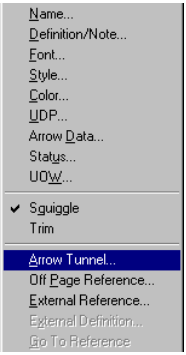

Рисунок 27 - Пунктменю Arrow Tunnel

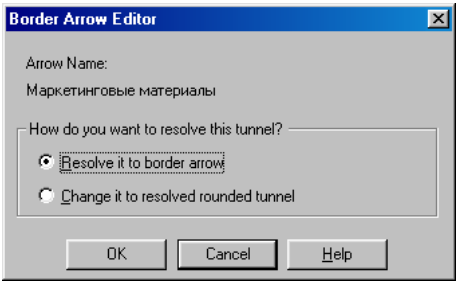

Рисунок 28 – Диалоговоеокно Border Arrow Editor

Для стрелки "Маркетинговые материалы" выберите опцию Trim (Упорядочить) из контекстного меню.

Результат выполнения показан на рисунке 29.

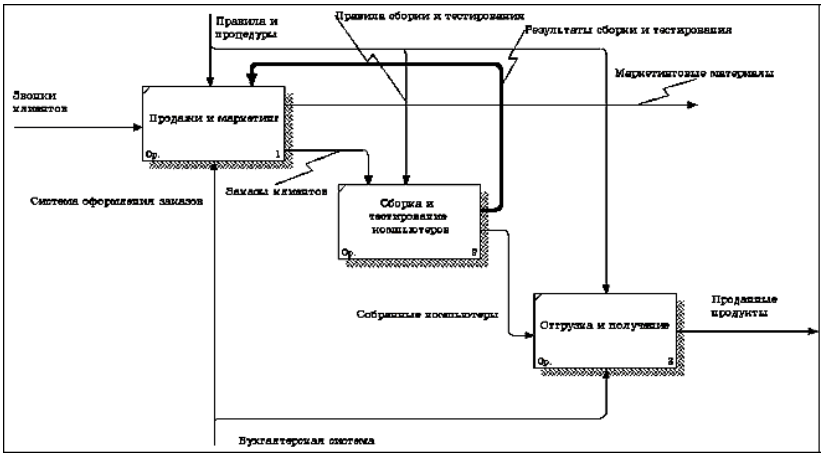

Рисунок 29 - Результат выполнения диаграмма A0

1.2.2.11 Аналогично проведите декомпозицию функционального блока «Сборка и тестирование» на следующие блоки:

- отслеживание расписания и управление сборкой и тестированием;
- сборка настольных компьютеров;
- сборка ноутбуков;
- тестирование компьютеров.

1.2.3 Диаграммы узлов создаются в результате последовательности следующих действий.

1.2.3.1 Выберите пункт главного меню Diagram/AddNodeTree (рисунок 30).

1.2.3.2 В первом диалоговом окне гида NodeTreeWizard внесите имя диаграммы, укажите диаграмму корня дерева и количество уровней (рисунок 31).

1.2.3.3 Во втором диалоговом окне гида NodeTreeWizard установите опции, как показано на рисунке 32.

1.2.3.4 Щелкните по кнопке Finish. В результате будет создана диаграмма дерева узлов NodetreeDiagram (рисунок 33).

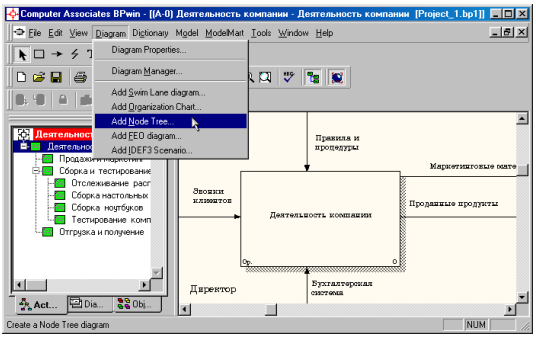

Рисунок 30 - Пункт главного меню Diagram/AddNodeTree

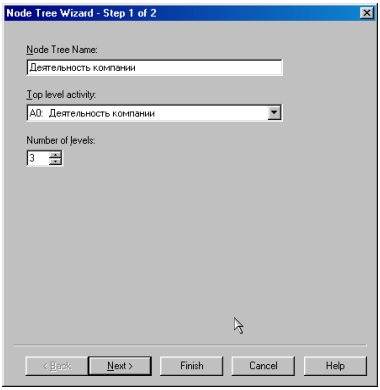

Рисунок 31 – Первое диалоговое окно гида NodeTreeWizard

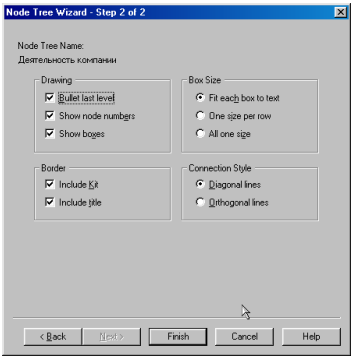

Рисунок 32 – Второе диалоговое окно гида NodeTreeWizard

1.2.3.5 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 ивовкладке Style диалога Node Tree Properties отключитеопцию BulletLastLevel (рисунок 34).

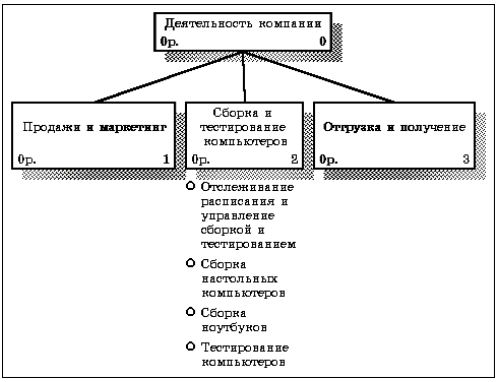

Рисунок 33 - Диаграмма дерева узлов

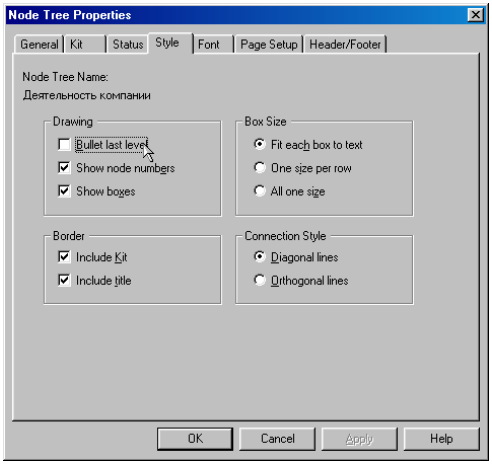

Рисунок 34 – Отключение опции BulletLastLevel

1.2.3.6 Щелкните по OK. Результат модификации диаграммы дерева узлов показан на рисунке 35.

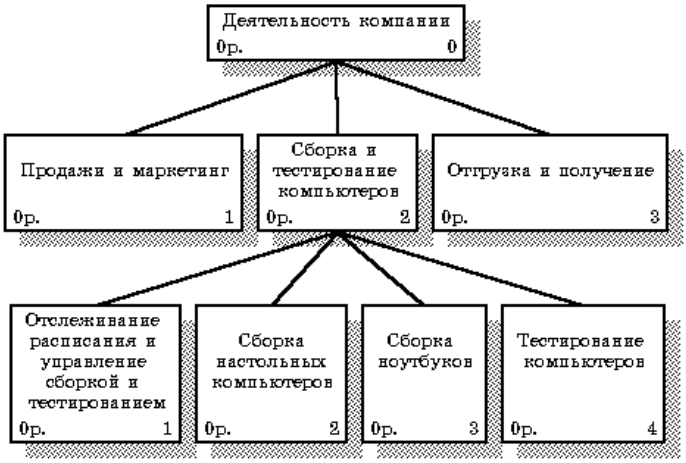

Рисунок 35 - Результат выполнения

1.2.4 Предположим, что при обсуждении бизнес-процессов возникла необходимость детально рассмотреть взаимодействие работы "Сборка и тестирование компьютеров" с другими работами. Чтобы не портить диаграмму декомпозиции, создается FEO-диаграмма (FEO расшифровывается как «только для экспозиции»), на которой будут только стрелки работы "Сборка и тестирование компьютеров ". Диаграмма создается следующими действиями.

1.2.4.1 Выберите пункт главного меню Diagram/AddFEODiagram (рисунок 36).

1.2.4.2 В диалоговом окне AddNewFEODiagram выберите тип и внесите имя диаграммы FEO как показано на рисунке 37. Щелкните по кнопке OK.

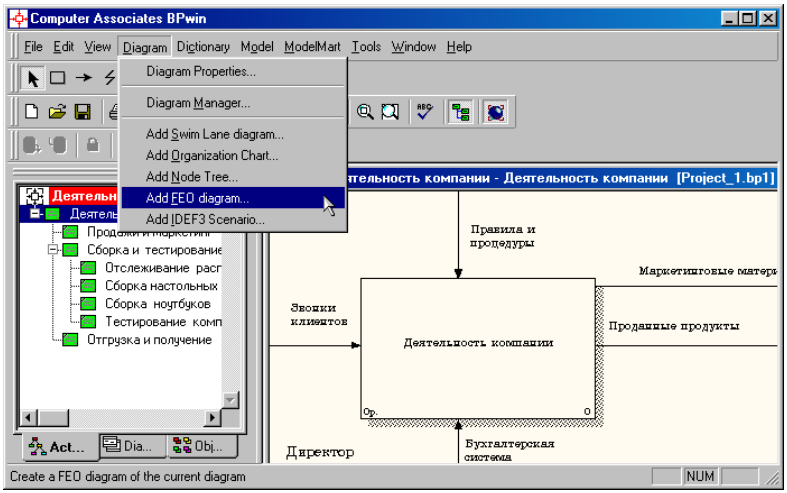

Рисунок 36 - Пункт главного меню Diagram/AddFEODiagram

1.2.4.3 Для определения содержания диаграммы перейдите в пункт меню Diagram/DiagramProperties и во вкладке DiagramText внесите определение (рисунок 38).

1.2.4.4 Удалите лишние стрелки на диаграмме FEO. Результат показан на рисунке 39.

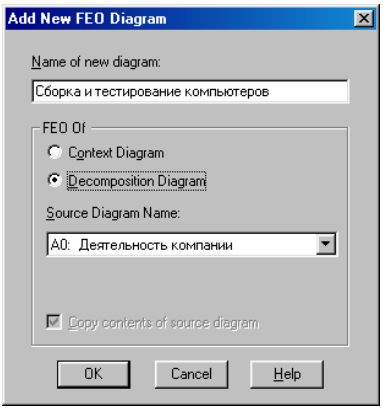

Рисунок 37 - Диалоговое окно AddNewFEODiagram

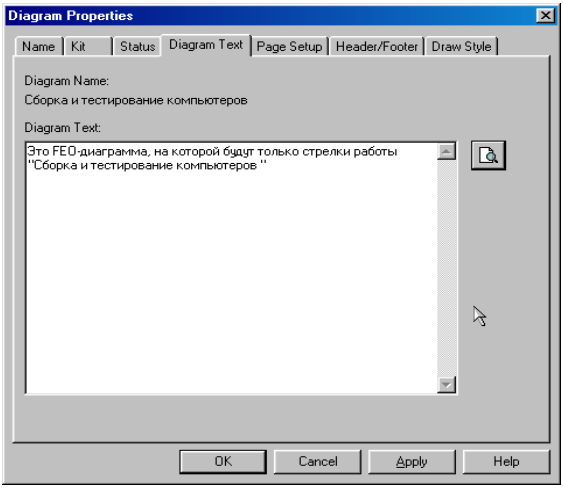

Рисунок 38 – Вкладка DiagramText диалогового окна DiagramProperties

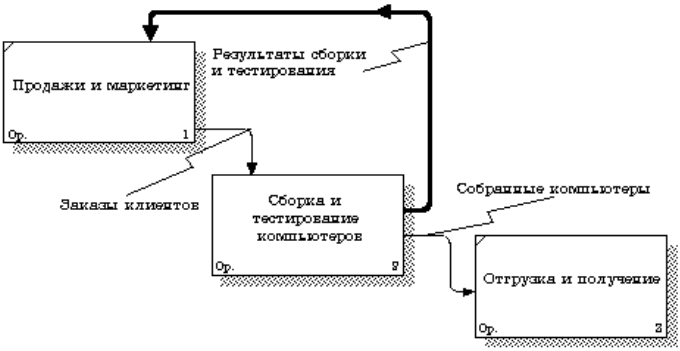

Рисунок 39 - Диаграмма FEO

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

Практическая работа №9
Диаграммы DFD

Цель работы: выполнить построение диаграмм по методологии DFD.

Задачи работы: освоить приемы построения диаграмм по методологии DFD с применением CASE-средства BPwin.

Содержание работы:

- построение диаграммы A0;
- построение диаграмм декомпозиции A0.

2.1 Теоретическая часть

Диаграммы потоков данных (Dataflowdiagramm, DFD) являются средством моделирования функциональных требований к проектируемой системе и используются для описания документооборота и обработки информации. С их помощью эти требования представляются в виде иерархии функциональных компонентов (процессов), связанных потоками данных. Главная цель такого представления продемонстрировать, как каждый процесс преобразует свои входные данные в выходные, а также выявить отношения между этими процессами.

В соответствии с данными методами 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 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 уровень декомпозиции, на котором процессы становятся элементарными, и детализировать их далее невозможно.

Источники информации (внешние сущности) порождают информационные потоки (потоки данных), переносящие информацию к подсистемам или процессам. Те, в свою очередь,

преобразуют информацию и порождают новые потоки, которые переносят информацию к другим процессам или подсистемам, накопителям данных или внешним сущностям потребителям информации.

DFD описывает:

- функции обработки информации (работы, процессы);
- документы (стрелки, arrow), объекты, сотрудников или отделы, которые участвуют в обработке информации;
- внешние ссылки (externalreferences), которые обеспечивают интерфейс с внешними объектами, находящимися за границами моделируемой системы;
- таблицы для хранения документов (хранилище данных, datastore).

В BPwin для построения диаграмм потоков данных используется нотация Гейна-Сарсона. DFD рассматривает систему как совокупность предметов (таблица 3). Контекстная диаграмма часто включает работы и внешние ссылки. Работы обычно именуются по названию системы, например, «Система обработки информации».

Таблица 3 – Объекты диаграммы DFD

Наименование	Назначение
Работы (процессы)	Представляют собой преобразование входных потоков данных в выходные в соответствии с определенным алгоритмом. Физически процессы могут быть реализованы различными способами: это может быть подразделение организации (отдел), выполняющее обработку входных документов и выпуск отчетов; программа; аппаратно реализованное логическое устройство и т. д. Изображаются прямоугольниками со скругленными углами
Внешние сущности	Представляют собой материальные объекты или физические лица, представляющие собой источник или приемник информации, например, заказчики, персонал, поставщики, клиенты, склад. Определение некоторого объекта или системы в качестве внешней сущности указывает на то, что они находятся за пределами границ анализируемой системы. Изображаются в виде прямоугольника с тенью и обычно располагаются по краям диаграммы

Продолжение таблицы 3

Стрелки (потоки данных)	Поток данных определяет информацию, передаваемую через некоторое соединение от источника к приемнику. Реальный поток данных может быть информацией, передаваемой по кабелю между двумя устройствами; пересылаемыми по почте письмами; магнитными лентами или дискетами, переносимыми с одного компьютера на другой и т. д. Поток данных на диаграмме изображается линией, оканчивающейся стрелкой, которая показывает направление потока. Каждый поток данных имеет имя, отражающее его содержание
Хранилище данных	Абстрактное устройство для хранения информации, которую можно в любой момент поместить в накопитель и через некоторое время извлечь, причем способы помещения и извлечения могут быть любыми. Накопитель данных может быть реализован физически в виде микрофиши, ящика в картотеке, таблицы в оперативной памяти, файла на магнитном носителе и т.д. Накопитель данных в общем случае является прообразом будущей базы данных, и описание хранящихся в нем данных должно быть увязано с информационной моделью (ERD)

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

В DFD номер каждой работы может включать префикс, номер родительской работы A и номер объекта. Номер объекта – это уникальный номер работы на диаграмме. Уникальный номер имеют хранилища данных и внешние сущности независимо от их расположения на диаграмме. Каждое хранилище данных имеет префикс D и уникальный номер, например, D5. Каждая внешняя сущность имеет префикс E и уникальный номер.

2.2 Выполнение практической работы

Для того чтобы дополнить модель IDEF0 диаграммой DFD, нужно в процессе декомпозиции в диалоге

ActivityBoxCount «кликнуть» по радиокнопке DFD. В палитре инструментов на новой диаграмме DFD появляются новые кнопки (рисунок 40).

Рисунок 40 – Кнопки

Назначение кнопок:

1 добавить в диаграмму внешнюю ссылку (ExternalReference): внешняя ссылка является источником или приемником данных извне модели;

2 добавить в диаграмму хранилище данных (Datastore): хранилище данных позволяет описать данные, которые необходимо сохранить в памяти прежде, чем использовать в работах;

3 ссылка на другую страницу: в отличие от IDEF0 инструмент offpagereference позволяет направить стрелку на любую диаграмму.

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 (dataflow), хранение объектов (datastores), поставка и распространение объектов (externalentities).

2.2.1 Декомпозируйте функциональный блок «Продажи и маркетинг» на контекстной диаграмме A0 на диаграмму DFD, используя инструмент ▼ на панели инструментов. В диалоговом окне из предложенных вариантов диаграмм выберите DFD, а количество блоков установите 4 (рисунок 41).

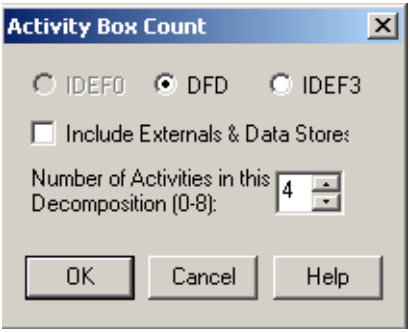

Рисунок 41 - Диалоговое окно параметров декомпозиции ActivityBoxCount

2.2.2. Внесите следующие имена процессов:

- проверка данных о клиенте;
- оформление заказа;
- разработка прогнозов продаж;
- привлечение новых клиентов.

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

- список клиентов;
- список продуктов;
- список заказов.

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

Удалите граничные стрелки. Используя кнопку  на палитре инструментов, добавьте внешние ссылки:

- клиент;
- маркетинговые материалы;
- прогноз продаж;

- система оформления.

2.2.5. Свяжите объекты диаграммы DFD стрелками (потоками данных) как показано на рисунке 42.

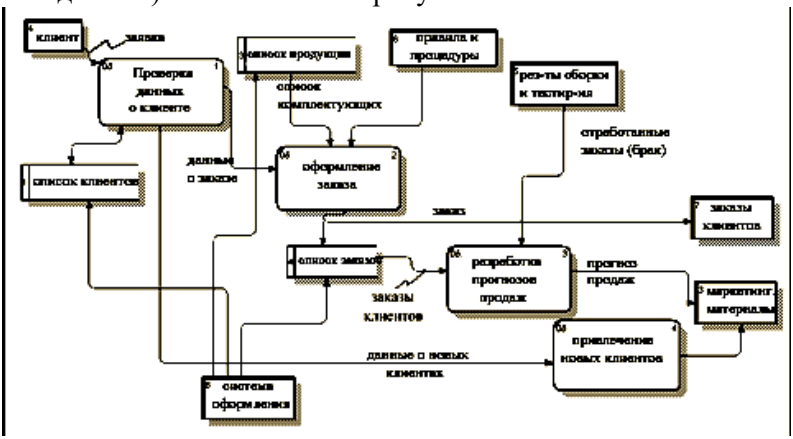

Рисунок 42 - Декомпозиция блока «Продажи и маркетинг» на диаграмме DFD

2.2.6. Аналогично декомпозируйте функциональный блок A24 «Тестирование компьютеров» на диаграмме A2 на диаграмму DFD и оформите ее в соответствии с рисунком 43.

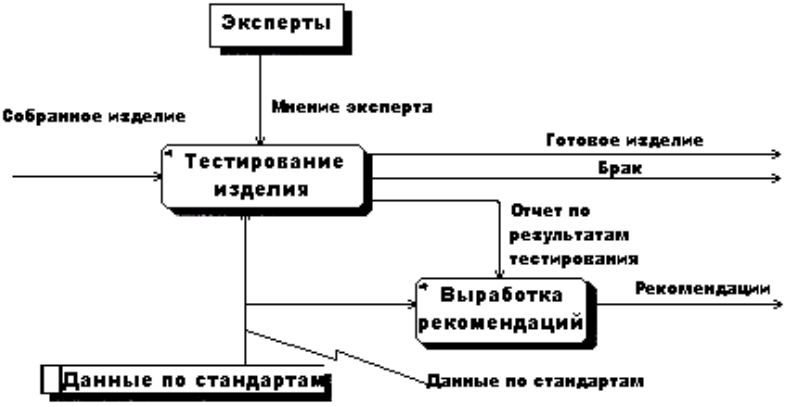

Рисунок 43 – Пример диаграммы DFD

Практическая работа №10
Методология IDEF3

Цель работы: выполнить построение диаграмм по методологии IDEF3.

Задачи работы: освоить приемы построения диаграмм по методологии IDEF3.

Содержание работы:

- построение диаграммы A0;
- построение диаграмм декомпозиции A0;
- построение диаграммы узлов;
- построение диаграммы FEO.

3.1 Теоретическая часть

IDEF3 – используется для описания логики взаимодействия информационных потоков. Эта методология моделирования использует графическое описание информационных потоков, взаимоотношений между процессами обработки информации и объектами, являющимися частью этих процессов. Диаграммы Workflow могут быть использованы 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.

Основные элементы диаграммы показаны в таблице 4.

В IDEF3 различают три типа стрелок (рисунок 44), изображающих связи, стиль которых (таблица 5) устанавливается через меню Edit/Arrow Style.

Рисунок 44 – Кнопки: старшая связь, отношения, потоки объектов

Старшая связь показывает, что работа-источник заканчивается ранее, чем начинается работа-цель. Часто результатом

работы-источника становится объект, необходимый для запуска работы-цели.
В этом случае стрелку, обозначающую объект, изображают с двойным наконечником.

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 и разветвления стрелок. Для внесения перекрестка служит кнопка со значком “&” в палитре инструментов. В диалоге JunctionTypeEditor необходимо указать тип перекрестка (таблица 6).

Таблица 4 – Элементы диаграммы

Диаграммы	Основная единица описания в IDEF3
Единица работы (UOW)	Центральный компонент модели. Изображается прямоугольником с прямыми углами и имеет имя, выраженное отглагольным существительным, обозначающим процесс действия, одиночным или в составе фразы, и номер; другое имя существительное в составе той же фразы обычно отображает основной выход (результат работы)
Связи	Показывают взаимоотношение 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

Таблица 5 – Стили стрелок

Наименование	Описание
Старшая	Сплошная линия, связывающая единицы работ. Рисуется слева направо или сверху вниз. Показывает, что работа-источник должна закончиться прежде, чем работа-цель начнется
Отношения	Пунктирная линия, использующаяся для изображения связей между единицами работ, а также между единицами работ и объектами ссылок
Потоки объектов	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

Таблица 6 – Тип перекрестка

Наименование	Смысл в случае слияния стрелок	Смысл в случае разветвления стрелок
Asynchronous AND	Все предшествующие процессы должны быть завершены	Все следующие процессы должны быть запущены
Synchronous AND	Все предшествующие процессы завершены одновременно	Все следующие процессы запускаются одновременно
Asynchronous OR	Один или несколько предшествующих процессов должны быть завершены	Один или несколько следующих процессов должны быть запущены
Synchronous OR	Один или несколько предшествующих процессов завершены одновременно	Один или несколько следующих процессов запускаются одновременно
XOR (Exclusive OR)	Только один предшествующий процесс завершен	Только один следующий процесс запускается

Все перекрестки на диаграмме нумеруются, каждый номер имеет префикс J. Можно редактировать свойства перекрестка при помощи диалога DefinitionEditor. В IDEF3 стрелки могут сливаться и разветвляться только через перекрестки.

Объект ссылки выражает некую идею, концепцию или данные, которые нельзя связать со стрелкой, перекрестком или работой.

Для внесения объекта ссылки служит кнопка со значком “R” в палитре инструментов. Объект ссылки изображается в виде прямоугольника, похожего на прямоугольник работы. Имя объекта ссылки задается в диалоге Referent (пункт всплывающего меню NameEditor), 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, синхронные и асинхронные.

При внесении объектов ссылок помимо имени следует указывать тип объекта ссылки (таблица 7).

Таблица 7 – Типы объектов ссылок

Тип	Цель описания
OBJECT	Описывает участие важного объекта в работе
GOTO	Инструмент циклического перехода (в повторяющейся последовательности работ)
UOB	Применяется, когда необходимо подчеркнуть множественное использование какой-либо работы, но без цикла
NOTE	Используется для документирования важной информации
ELAB	Используется для усовершенствования графиков или их более детального описания. Обычно употребляется для детального описания разветвления и слияния стрелок на перекрестках

В IDEF3 декомпозиция используется для детализации работ. Можно многократно декомпозировать работу, т.е. работа может иметь множество дочерних работ. При этом номер работы состоит из номера родительской работы, версии декомпозиции и собственного номера работы на текущей диаграмме.

3.2 Выполнение практической работы

Проведем построение диаграмм IDEF3 производственного процесса, модель которого создана в лабораторной работе №1.

3.2.1 Перейдите на диаграмму A2 и декомпозируйте работу "Сборка настольных компьютеров" (рисунок 45).

3.2.2 В диалоге ActivityBoxCount (рисунок 46) установите число работ 4 и нотацию IDEF3.

Возникает диаграмма IDEF3 (рисунок 47), содержащая работы UnitofWork (UOW), также называемые единицами работы или работами (activity). Правой кнопкой мыши щелкните по работе с номером 1, выберите в контекстном меню Name и внесите имя работы "Подготовка компонентов" (рисунок 48).

Затем во вкладку Definition внесите определение работы с номером 1 "Подготавливаются все компоненты компьютера согласно спецификации заказа" (рисунок 49).

3.2.3 Во вкладке UOW диалогового окна ActivityProperties (рисунок 50) внесите свойства работы 1 в соответствии с данными таблицы 8.

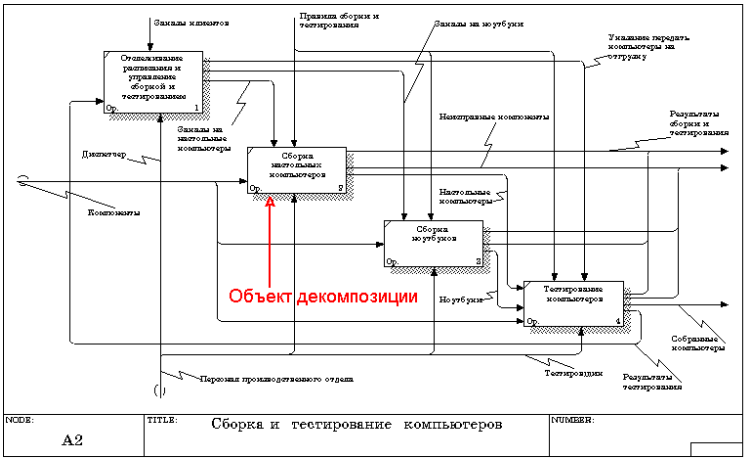

Рисунок 45 – Диаграмма A2 с объектом декомпозиции

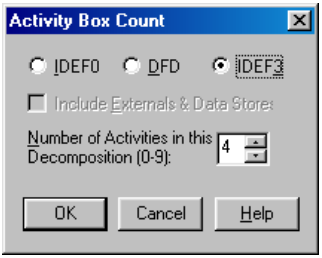

Рисунок 46 - Выбор нотации IDEF3 в диалоге ActivityBoxCount

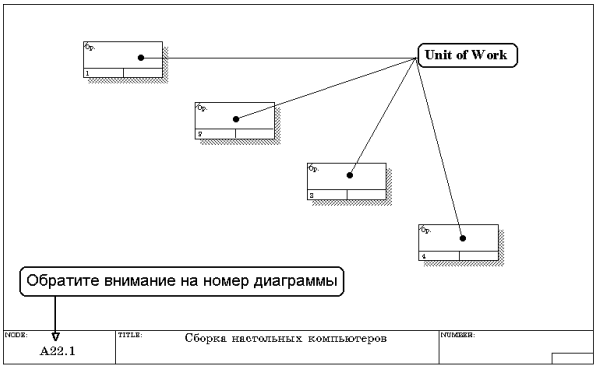

Рисунок 47 - Диаграмма IDEF3, содержащая четыре работы UnitofWork

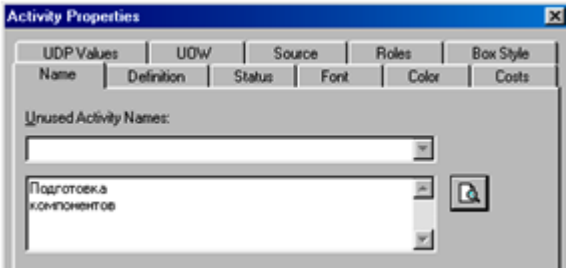

Рисунок 48 – Диалоговое окно ActivityProperties (Свойства работ)

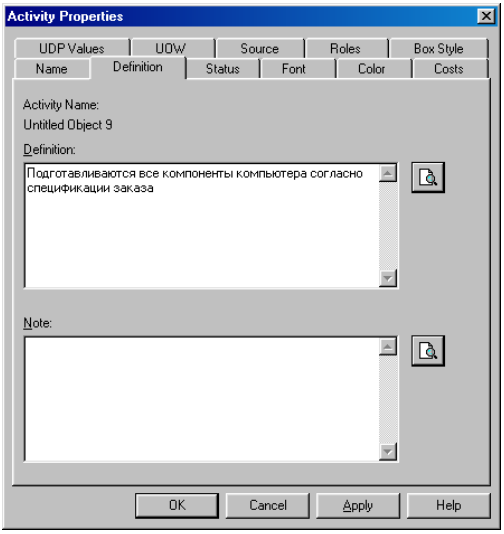

Рисунок 49 – Диалоговоеокно Activity Properties вкладка Definition

Таблица 8 - Свойства UOWдиалогового окна ActivityProperties

Objects	Компоненты: винчестеры, корпуса, материнские платы, видеокарты, звуковые карты, дисководы CD-ROM и флоппи, модемы, программное обеспечение
Facts	Доступные операционные системы: Windows
Con-strains	Установка модема требует установки дополнительного программного обеспечения

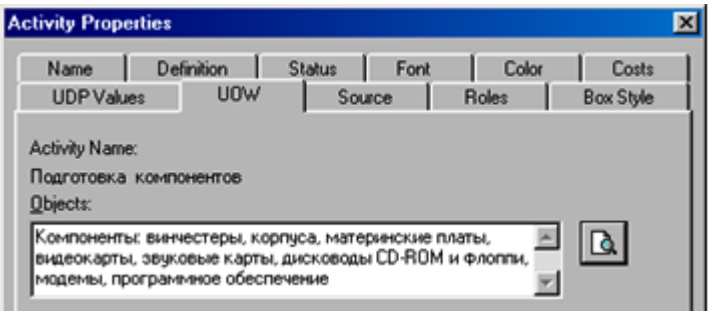

Рисунок 50 – Вкладка UOW диалогового окна ActivityProperties

3.2.4 Внесите в диаграмму еще три работы (кнопка) и присвойте имена работам с номерами 2…7 в соответствии с данными таблицы 17.

Таблица 9 – Названия работ

Номер работы	Название работы
2	Установка материнской платы и винчестера
3	Установка модема
4	Установка дисковода CD-ROM
5	Установка флоппи- дисковода
6	Инсталляция операционной системы
7	Инсталляция дополнительного программного обеспечения

Диаграмма IDEF3 должна выглядеть так, как показано на рисунке 51.

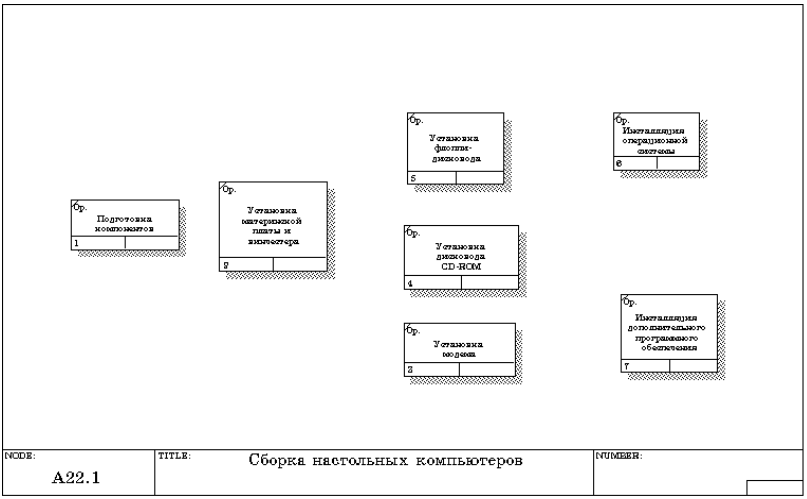

Рисунок 51 – Диаграмма IDEF3 после присвоения работам названий

3.2.5 С помощью кнопки  палитры инструментов создайте объект ссылки. Внесите имя объекта внешней ссылки " Компоненты" (рисунок 52).

Свяжите стрелкой объект ссылки и работу "Подготовка компонентов" (рисунок 53).

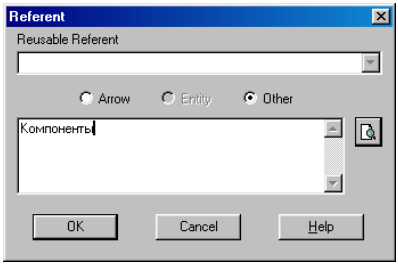

Рисунок 52 – Создание объекта ссылки

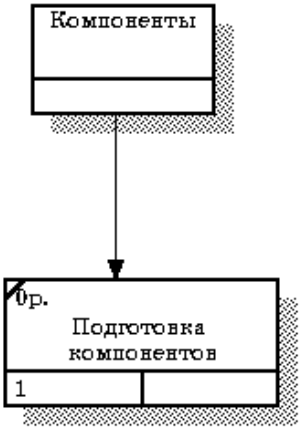

Рисунок 53 - Объект ссылки и работа "Подготовка компонентов" связаны стрелкой

Измените стиль стрелки, связывающей объект ссылки и работу "Подготовка компонентов", воспользовавшись диалоговым окном ArrowProperties, как показано на рисунке 54.

3.2.6 Свяжите стрелкой работы "Подготовка компонентов" (выход) и "Установка материнской платы и винчестера" (вход). Измените стиль стрелки на ObjectFlow.

На диаграммах IDEF3 имя стрелки может отсутствовать, хотя BPwin показывает отсутствие имени как ошибку. Результат выполнения пункта 6 показан на рисунке 55.

3.2.7 С помощью кнопки  на палитре инструментов внесите два перекрестка типа "асинхронное ИЛИ" (рисунок 56).

Свяжите работы с перекрестками, как показано на рисунке 57.

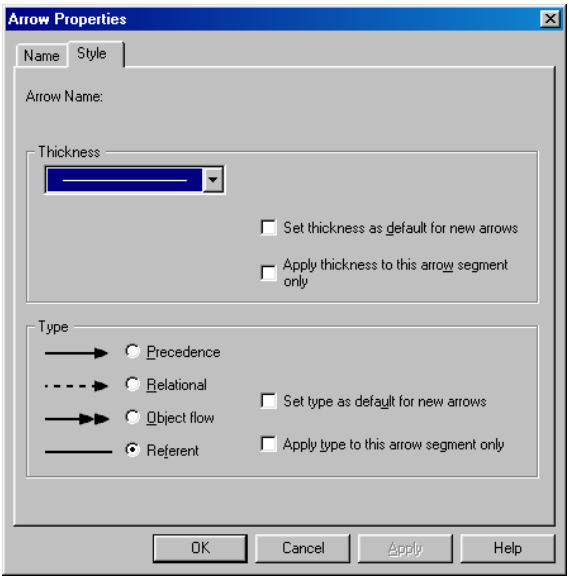

Рисунок 54 – Изменение стиля стрелки

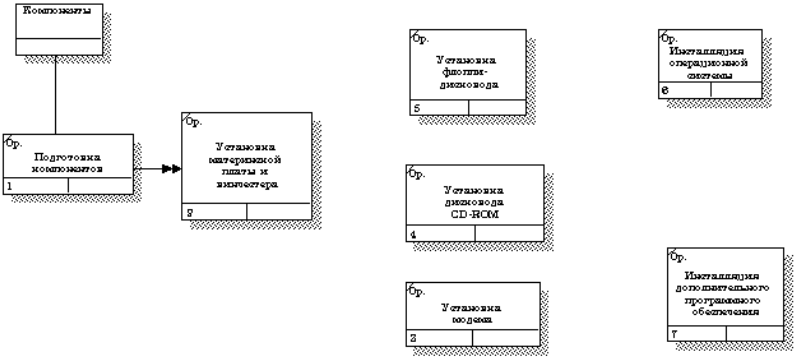

Рисунок 55 - Результат создания UOW и объекта ссылки

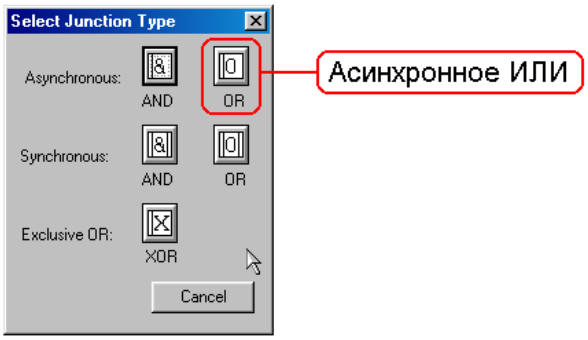

Рисунок 56 - Перекресток типа "асинхронное ИЛИ"

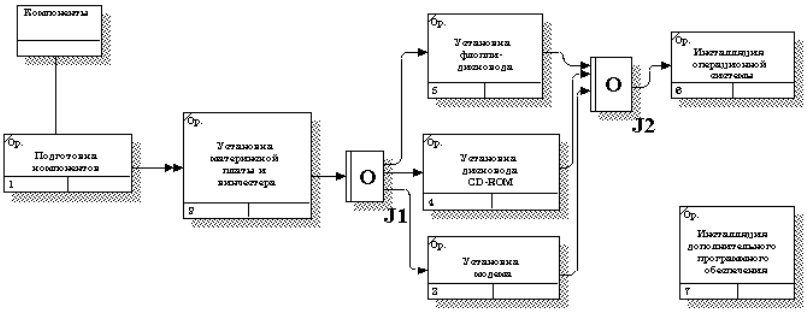

Рисунок 57 - Диаграмма IDEF3 после создания перекрестков

3.2.8 Правой кнопкой щелкните по перекрестку для разветвления J1 (fan-out), выберите Name и внесите имя "Компоненты, требуемые в спецификации заказа" (рисунок 58).

3.2.9 С помощью кнопки  палитры инструментов введите в диаграмму еще один объект ссылки и присвойте ему имя "Программное обеспечение".

3.2.10 Создайте два перекрестка типа "исключающее ИЛИ". Свяжите работы и соответствующие ссылки, как это показано на рисунке 59.

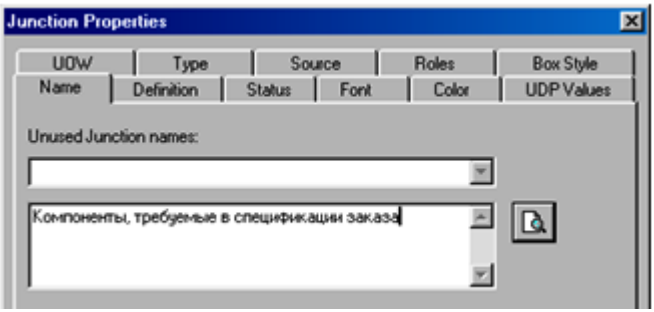

Рисунок 58 – Присвоение имени перекрестку J1

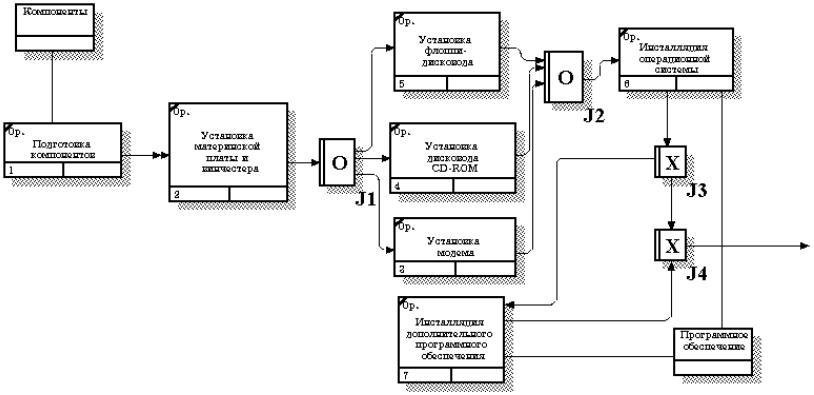

Рисунок 59 - Результат выполнения

Словарь терминов

Атрибут

Атрибут представляет собой тип характеристики, связанной с множеством реальных или абстрактных предметов (людей, мест, событий и т.д.).

Атрибут не ключевой

Любой атрибут, не являющийся частью первичного ключа сущности. Не ключевые атрибуты могут входить в инверсионный вход и (или) альтернативный ключ, а также могут быть внешними ключами.

Атрибут собственный

Атрибут, не являющийся внешним ключом. Собственный атрибут является представителем первичной связи с единичным доменом внутри информационной модели.

Атрибуты расширенные

Они называются также "расширенная информация, относящаяся к определениям колонок". Представляют собой информацию, которую определяют с целью контроля за изображением на экране и валидацией данных, хранящихся в колонке.

База данных

Зарезервированный объем памяти на одном или более устройствах хранения информации, используемый для хранения данных и определений объектов, например, таблиц и индексов.

Базовое имя

Исходное имя внешнего ключа, которому присвоено имя роли.

Бинарная связь

Связь, в которой ровно один экземпляр родительской сущности соответствует 0,1 или более экземплярам дочерней. В IDEF1X идентифицирующие, не идентифицирующие связи и связи подтипа являются бинарными связями.

Валидации правила

Правила проверки допустимых значений.

Вход инверсионный

Атрибут (атрибуты), который(е) не определяют уникальным образом экземпляр сущности, но часто используются для обращения к экземплярам сущностей. ERwin генерирует неуникальные индексы для всех инверсионных входов.

Дискриминатор

Значение атрибута в экземпляре общего родителя определяет, к какому из возможных подтипов принадлежит этот экземпляр. Этот атрибут принято называть дискриминатором. Например, значение атрибута "пол" в экземпляре сущности "служащий" определяет, к какому из возможных подтипов (мужчина-служащий или женщина-служащий) принадлежит этот экземпляр.

Домен

Совокупность значений, из которых берутся значения атрибутов. Каждый атрибут может быть определен только на одном домене, но на каждом домене может быть определено множество атрибутов. В понятие домена входит не только тип данных, но и область значений данных. Например, можно определить домен “Возраст” как положительное целое число и определить атрибут Возраст сотрудника как принадлежащий этому домену. В ERwin домен может быть определен только один раз и использоваться как в логической, так и в физической модели.

Имя роли

Новое имя, присваиваемое внешнему ключу. Имя роли используется для указания, что домен внешнего ключа является подмножеством домена атрибута родительской сущности и выполняет определенную функцию (или роль) в сущности.

Индекс

Объект СУБД, предназначенный для поиска данных. Он подобен содержанию книги, которое указывает на все номера страниц, посвященных конкретной теме. Индекс содержит отсортированную по колонке или нескольким колонкам информацию и указывает на строки, в которых хранится конкретное значение колонки.

Кардинальность

Называется также "Мощность связи". Отношение числа экземпляров родительской сущности к числу экземпляров дочерней. В IDEF1X кардинальность бинарных связей равна 1:n, где n может равняться:

- 0,1 или более - обозначается пробелами
- 1 или более - обозначается буквой "P"

- 0 или 1 - обозначается буквой "z"
- ровно n - где n - некоторое число

Кластер подтипа неполный

Если кластер подтипа не включает в себя все возможные подтипы (каждый экземпляр общего родителя не связан с одним подтипом), тогда кластер подтипа называется неполным. Например, если часть служащих работает на договорной основе, то кластер подтипа, состоящий из служащих-на окладе и служащих-с частичной занятостью, будет неполным.

Кластер подтипа полный

Кластер подтипа, включающий в себя все возможные подтипы. Например, любой служащий относится к мужскому или женскому полу. Кластер подтипа, состоящий из мужчины-служащего и женщины-служащего, является полным кластером подтипа.

Ключ альтернативный

1) Атрибут, который уникальным образом идентифицирует экземпляр сущности.

2) Если правилу 1 удовлетворяет более чем один атрибут (группа атрибутов), то альтернативным ключом называются те атрибуты или группы атрибутов, которые не были выбраны в качестве первичного ключа.

ERwin генерирует уникальный индекс для каждого альтернативного ключа.

Ключ внешний

Атрибут, мигрировавший от родительской сущности к дочерней через связь. Представляет собой вторичную ссылку на единичный домен, где первичной ссылкой является собственный атрибут.

Ключ первичный

1) Атрибут (атрибуты), который(е) уникальным образом идентифицирует(ют) экземпляр сущности.

2) Если более чем один атрибут (группа атрибутов) удовлетворяют правилу 1, то первичный ключ выбирается из этого списка кандидатов, исходя из того, каким представляется его значение для бизнеса в качестве идентификатора. В идеале первичные ключи не должны меняться со временем и должны быть как можно меньшего размера. ERwin генерирует уникальный

индекс для каждого первичного ключа.

Ключа внешнего миграция

Ситуация, при которой ключ родительской сущности автоматически появляется в ключе дочерней сущности со значком (FK), обозначающим внешний ключ.

Метамодель

Определяет структуры данных, необходимые для хранения всей информации о диаграмме, включающей в себя определения, адреса, шрифты, цвета и т.д.

Представление

Объект БД, данные в котором не хранятся постоянно, как в таблице, а формируются динамически при обращении к нему. Представление не может существовать само по себе, а определяется только в терминах одной или нескольких таблиц. Применение представлений позволяет разработчику БД обеспечить каждому пользователю или группе пользователей свой взгляд на данные, что решает проблемы простоты использования и безопасности данных.

Проектирование обратное

Процесс генерации логической модели из физической базы данных.

Проектирование прямое

Процесс генерации физической модели (схемы базы данных) из логической модели данных.

Репозиторий

База данных проекта. Может хранить свыше 100 типов объектов: структурные диаграммы, определения экранов и меню, проекты отчетов, описания данных, логика обработки, модели данных, их организации и обработки, исходные коды, элементы данных и т.п. На основе репозитория осуществляется интеграция CASE-средств и разделение системной информации между разработчиками в соответствии с их правами доступа.

Связь

Служит для описания связей или отношений между сущностями.

Связь идентифицирующая

Связь, в которой экземпляр дочерней сущности идентифицируется с помощью своего отношения к родительской сущ-

ности. Атрибуты первичного ключа родительской сущности становятся атрибутами первичного ключа дочерней.

Связь не идентифицирующая

Связь, в которой экземпляр дочерней сущности не идентифицируется с помощью ее отношения к родительской сущности. Атрибуты первичного ключа родительской сущности становятся не ключевыми атрибутами дочерней.

Связь неопределенная

Связи "родительская-дочерняя сущность" и связи подтипа считаются определенными связями, поскольку они точно определяют, каким образом экземпляры одной сущности связаны с экземплярами другой. Однако на начальных этапах разработки модели часто бывает полезно задание "неопределенных" связей между двумя сущностями. Неопределенная связь, которую называют также связью "многие-ко-многим", - отношение между двумя сущностями, при котором каждый экземпляр первой сущности связан с 0,1 или более экземплярами второй сущности и каждый экземпляр второй сущности связан с 0,1 или более экземплярами первой сущности.

Связь определенная

Отношение между сущностями, в котором каждый экземпляр родительской сущности связан с 0,1 или более экземплярами дочерней сущности и каждый экземпляр дочерней сущности связан с 0 или 1 экземплярами родительской сущности.

Связь подтипа

Связью подтипа (другое название - категоризационная связь) называют связь между сущностью подтипа и ее групповым родителем. Связь подтипа всегда связывает один экземпляр группового родителя с 0 или одним экземпляром подтипа.

Сегмент

Именованное множество из одного или более устройств, зарезервированное для использования какой-то определенной базой данных SQL Server. После того как создан сегмент, можно использовать его для хранения объектов базы данных, например, таблиц и индексов.

Сегмент отката

Зарезервированный объем памяти внутри табличного пространства, используемое для хранения "снимка" данных в том

виде, в котором они находились до выполнения транзакции. Если транзакция не завершится вследствие сбоя, все изменения данных откатываются и восстанавливается тот образ данных, который хранится в сегменте отката.

Словарь ERwin

База данных, которая генерируется из метамодели ERwin и в которой хранится информация о структурах данных, используемых в моделях, в отличие от бизнес-информации, хранящейся в других базах данных.

Ссылочная целостность

Утверждение, что для значений внешнего ключа в экземпляре родительской сущности существуют соответствующие значения родительской сущности.

Сущность

Набор реальных или абстрактных предметов (людей, мест, событий и т.д.), имеющих общие атрибуты или характеристики.

Сущность зависимая

Сущность, экземпляры которой не могут быть уникальным образом идентифицированы, если не определена ее связь с другой сущностью или сущностями.

Сущность независимая

Сущность, экземпляры которой могут быть уникальным образом идентифицированы без определения ее связи с другой сущностью.

Сущность подтипа

Сущность, которая является типом другой сущности. Например, служащий, работающий на окладе - это определенный тип служащего. Они полезны при формулировании таких связей, которые допустимы только для данного подтипа, например, того факта, что служащий на окладе имеет право на определенную пенсию, а служащий, работающий на условиях частичной занятости, не имеет такого права. В IDEF1X подтипы внутри кластера подтипа являются взаимно исключающими.

Схема

Структура базы данных. Как правило, строится на основе файла скрипта, написанного на DDL (языке определения данных). DDL состоит из операторов CREATETABLE, CREATEINDEX и других.

Табличное пространство

Именованный сегмент базы данных, состоящий из одного или более файлов данных. После того как создано табличное пространство, можно использовать его для хранения таблиц, индексов или сегментов отката.

Триггер

Процедура (именованный блок кода SQL), которая выполняется автоматически при свершении определенного события.

Унификация

Слияние двух или более атрибутов внешнего ключа в один атрибут внешнего ключа на основе утверждения, что значения исходных атрибутов внешнего ключа должны быть идентичны.

Уровень логический

Представление и моделирование предметов непосредственно из реального мира.

Уровень физический

Информация, относящаяся к модели, которая определяется в зависимости от базы данных и СУБД; например, таблицы, колонки, типы данных и т.д.

Формат ERX

Собственный текстовый формат ERwin, позволяющий сохранять информацию, содержащуюся в графической модели данных, в виде текстового описания.

Формат MPDModelPro

Тип файла, в котором хранится информация о модели данных в текстовом формате.

Формат SML

Специальный тип файла, предназначенный для хранения информации, относящейся к модели "Сущность-связь", в текстовом формате.

CASE-технология

Совокупность методологий анализа, проектирования, разработки и сопровождения сложных систем, поддерживаемая комплексом средств автоматизации.

Литература

1. Модели информационных систем [Электронный ресурс]: учебное пособие / В.П. Бубнов [и др.]. Электрон. текстовые данные. М.: Учебно-методический центр по образованию на железнодорожном транспорте, 2015. 188 с. — Режим доступа: http://www.iprbookshop.ru/45279.html

2. Бурков А.В. Проектирование информационных систем в Microsoft SQL Server 2008 и Visual Studio 2008 [Электронный ресурс]. Электрон. текстовые данные. М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. 310 с. — Режим доступа: http://www.iprbookshop.ru/52166.html

3. Вичугова А.А. Инструментальные средства разработки компьютерных систем и комплексов [Электронный ресурс]: учебное пособие для СПО. Электрон. текстовые данные. Саратов: Профобразование, 2017. 135 с. — Режим доступа: http://www.iprbookshop.ru/66387.html

4. Васильев Р.Б., Калянов Г.Н., Лёвочкина Г.А. Управление развитием информационных систем [Электронный ресурс]. Электрон. текстовые данные. М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. 507 с. — Режим доступа: http://www.iprbookshop.ru/62828.html

5. Гвоздева В.А. Основы построения автоматизированных информационных систем: учеб._М.: ИД «Форум»: ИНФРА-М, 2017.

6. Гладких Т.В., Воронова Е.В. Информационные системы и сети [Электронный ресурс]: учебное пособие. Электрон. текстовые данные. Воронеж: Воронежский государственный университет инженерных технологий, 2016. 87 с. — Режим доступа: http://www.iprbookshop.ru/64403.html

7. Грекул В.И., Денищенко Г.Н., Коровкина Н.Л. Управление внедрением информационных систем [Электронный ресурс]: учебник. Электрон. текстовые данные. М., Саратов: Интернет-Университет Информационных Технологий (ИНТУИТ), Вузовское образование, 2017. 224 с. — Режим доступа: http://www.iprbookshop.ru/72342.html

8. Долженко А.И. Технологии командной разработки программного обеспечения информационных систем [Элек-

тронный ресурс]. Электрон. текстовые данные. М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. 300 с. — Режим доступа: http://www.iprbookshop.ru/39569.html

9. Извозчикова В.В. Эксплуатация и диагностирование технических и программных средств информационных систем [Электронный ресурс]: учебное пособие. Электрон. текстовые данные. Оренбург: Оренбургский государственный университет, ЭБС АСВ, 2017. 137 с. — Режим доступа: http://www.iprbookshop.ru/71353.html

10. Коцюба И.Ю., Чунаев А.В., Шиков А.Н. Основы проектирования информационных систем [Электронный ресурс]: учебное пособие. Электрон. текстовые данные. СПб.: Университет ИТМО, 2015. 205 с. — Режим доступа: http://www.iprbookshop.ru/67498.html

11. Лазебная Е.А. Методы и средства проектирования информационных систем и технологий [Электронный ресурс]: учебное пособие. Электрон. текстовые данные. Белгород: Белгородский государственный технологический университет им. В.Г. Шухова, ЭБС АСВ, 2015. 127 с. — Режим доступа: http://www.iprbookshop.ru/66663.html

12. Мочалов В.П., Братченко Н.Ю. Модели массового обслуживания в информационных системах [Электронный ресурс]: учебное пособие. Электрон. текстовые данные. Ставрополь: Северо-Кавказский федеральный университет, 2016. 126 с. — Режим доступа: http://www.iprbookshop.ru/66031.html

13. Трофимова М.В. Менеджмент в сфере информационных технологий [Электронный ресурс]: учебное пособие. Электрон. текстовые данные. Ставрополь: Северо-Кавказский федеральный университет, 2015. 195 с. — Режим доступа: http://www.iprbookshop.ru/62956.html

14. Шаньгин В.Ф. Защита компьютерной информации. Эффективные методы и средства [Электронный ресурс]. Электрон. текстовые данные. Саратов: Профобразование, 2017. 544 с. — Режим доступа: http://www.iprbookshop.ru/63592.html

Учебное издание

Живодеров А. Н.

ЭКСПЛУАТАЦИЯ И МОДИФИКАЦИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

Методические рекомендации
по выполнению практических работ
по профессиональному модулю ПМ.01 для обучающихся
специальности 09.02.04 Информационные системы
(по отраслям)

Редактор Лебедева Е.М.

Подписано к печати 07.02.2020 г. Формат 60x84 $^{1}/_{16}$.
Бумага офсетная. Усл. п. л. 6,68. Тираж 25 экз. Изд. № 6623.

Издательство Брянского государственного аграрного университета
243365 Брянская обл., Выгоничский район, с. Кокино, Брянский ГАУ