

Документ подписан простой электронной подписью

Информация о владельце:

ФИО: Шебзухова Татьяна Александровна

Должность: Директор Пятигорского института (филиал) Северо-Кавказского
федерального университета

Дата подписания: 12.09.2023 10:21:00

Уникальный программный ключ: «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

d74ce93cd40e39275c3ba2f58486412a1c8ef96f

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ

Пятигорский институт (филиал) СКФУ

Методические указания

по выполнению лабораторных работ
по дисциплине «**ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ**»
для студентов направления подготовки /специальности

09.03.02 Информационные системы и технологии

(ЭЛЕКТРОННЫЙ ДОКУМЕНТ)

СОДЕРЖАНИЕ

1. Цель и задачи освоения дисциплины	3
2. Место дисциплины в структуре образовательной программы	3
3. Связь с предшествующими дисциплинами	3
4. Связь с последующими дисциплинами	3
5. СОДЕРЖАНИЕ ЛАБОРАТОРНЫХ РАБОТ	3
6. КРИТЕРИИ ОЦЕНИВАНИЯ КОМПЕТЕНЦИЙ.....	108
7. МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ, ОПРЕДЕЛЯЮЩИЕ ПРОЦЕДУРЫ ОЦЕНИВАНИЯ ЗНАНИЙ, УМЕНИЙ, НАВЫКОВ И (ИЛИ) ОПЫТА ДЕЯТЕЛЬНОСТИ, ХАРАКТЕРИЗУЮЩИХ.....	109
7.1 ЭТАПЫ ФОРМИРОВАНИЯ КОМПЕТЕНЦИЙ	109
8. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ.....	109

1. Цель и задачи освоения дисциплины

Целью изучения дисциплины «Информационные технологии» является получение студентами теоретических знаний в области создания и использования информационных технологий, а также приобретение практических навыков в использовании отдельных информационных технологий.

Задачами изучения дисциплины являются:

сформировать у студентов компетентности в области современных информационных технологий;

ознакомить студентов с историей, классификацией и перспективами развития информационных технологий;

ознакомить студентов с методологиями, методами и технологиями, применяемыми на разных этапах жизненного цикла информационных систем;

сформировать у студентов навыки практического применения ряда перспективных информационных технологий.

2. Место дисциплины в структуре образовательной программы

Дисциплина «Информационные технологии» относится к обязательной части блока Б1 учебного плана подготовки бакалавров направления 09.03.02 Информационные системы и технологии. Ее освоение происходит в 3 семестре.

3. Связь с предшествующими дисциплинами

При изучении данной дисциплины необходимы знания, полученные в результате освоения дисциплины «Цифровая грамотность и обработка больших данных», «Информатика», «Корректирующий курс по информатике», «Введение в специальность», «Операционные системы».

4. Связь с последующими дисциплинами

Знания, полученные при изучении данной дисциплины, необходимы для успешного освоения таких дисциплин, как «Предметно-ориентированные информационные системы», «Управление данными», «Теория информационных процессов и систем», «Мультимедиа технологии», «Инструментальные средства мультимедиа технологии», «Научно-исследовательская работа (получение первичных навыков научно-исследовательской работы)», «Подготовка к сдаче государственного экзамена», «Сдача государственного экзамена», «Защита выпускной квалификационной работы».

5. СОДЕРЖАНИЕ ЛАБОРАТОРНЫХ РАБОТ

Лабораторная работа № 1

«Предпроектное обследование прикладной предметной области.»

Форма проведения: лабораторная работа.

Цель работы:

- Ознакомиться с целями и задачами предпроектного обследования
- Понять и усвоить методологию функционального моделирования
- Научиться определять границы моделирования

- Научиться определять точку зрения при построении модели.
- Изучить принцип декомпозиции.

Порядок выполнения лабораторной работы.

При создании информационной системы первым этапом работ является проведение предпроектного обследования предприятия. На этом этапе производится сбор информации о предприятии, его целях и задачах, структуре и финансово-хозяйственной деятельности. Также изучается внешнее окружение, взаимодействующее с предприятием, и в котором предприятие осуществляет свою деятельность. В результате предпроектного обследования должно быть сформировано описание организационной структуры предприятия, описание функционального взаимодействия подразделений предприятия и информационная схема, содержащая информационные потоки предприятия, места

получения и накопления информации и прочие информационные аспекты деятельности предприятия. В целом в результате предпроектного обследования должно быть получено комплексное описание предприятия и его бизнеса.

Методология функционального моделирования IDEF0 – это технология описания системы в целом как множества взаимозависимых действий, или функций. Важно отметить функциональную направленность IDEF0 – функции системы исследуются независимо от объектов, которые обеспечивают их выполнение. «Функциональная» точка зрения позволяет четко разделить аспекты назначения системы от аспектов ее физической реализации.

IDEF0 методология представляет собой формализованный подход к созданию функциональных моделей – структурных схем изучаемого процесса или системы. Схемы строятся по иерархическому принципу с необходимой степенью подробности и помогают разобраться в том, что происходит в изучаемой системе или процессе, какие функции выполняются и в какие отношения вступают между собой и с окружающей средой ее функциональные блоки. IDEF0-модель принципиально не может ответить на вопросы о том, как протекают процессы в системе во времени и в пространстве.

В этом случае рекомендуется переходить к другим моделям – математическим, имитационным, описывающим процессы в функциональных блоках IDEF0-модели. По терминологии, принятой в исследовании процессов, IDEF0-модели относятся к классу концептуальных. Именно концептуальные модели являются основой построения имитационных и математических моделей.

Наиболее часто IDEF0 применяется как технология исследования и проектирования систем на логическом уровне. По этой причине она, как правило, используется на ранних этапах разработки проекта, для сбора данных и моделирования процесса «как есть». Результаты IDEF0-анализа могут применяться при проведении проектирования с использованием моделей для описания потоков работ IDEF3 и потоков данных DFD.

IDEF0 сочетает в себе небольшую по объему графическую нотацию (она содержит только два обозначения: блоки и стрелки) со строгими и четко определенными рекомендациями, в совокупности предназначенными для построения качественной и понятной модели системы.

Первый шаг при построении модели IDEF0 заключается в определении назначения модели – набора вопросов, на которые должна отвечать модель. **Границы моделирования** предназначены для обозначения ширины охвата предметной области и глубины детализации и являются логическим продолжением уже определенного назначения модели. Следующим шагом указывается предполагаемая целевая аудитория, для нужд которой создается модель. Зачастую от выбора целевой аудитории зависит уровень детализации, с которым должна создаваться модель. Перед построением модели необходимо иметь представление о том, какие сведения о предмете моделирования уже известны, какие дополнительные материалы и (или) техническая документация для понимания модели могут быть необходимы целевой аудитории, какие язык и стиль изложения являются наиболее подходящими.

Под **точкой зрения** понимается перспектива, с которой наблюдалась система при построении модели. Точка зрения выбирается таким образом, чтобы учесть уже обозначенные границы моделирования и назначение модели. Однажды выбранная точка зрения остается неизменной для всех элементов модели. При необходимости могут быть созданы другие модели, отображающие систему с других точек зрения. Вот несколько примеров точек зрения при построении моделей: клиент, поставщик, владелец, редактор.

Действие, обычно в IDEF0 называемое функцией, обрабатывает или переводит входные параметры (сырье, информацию и т.п.) в выходные. Поскольку в модели IDEF0 представляют иерархических функций, в систему как множество



Рисунок 1.1 – Функциональный блок IDEF0

первую очередь должна быть определена функция, описывающая систему в целом – **контекстная функция**. Функции изображаются на диаграммах как поименованные прямоугольники, или функциональные блоки. Имена функций в IDEF0 подбираются по сходным правилам с именами действий – с использованием глаголов или отглагольных существительных. Важно подбирать имена таким образом, чтобы они отражали систему так, как если бы она обозревалась с точки зрения, выбранной для моделирования. Пример функционального блока приведен на рисунке 1.

Каждый блок, не имеющей декомпозиции, помечается небольшой диагональной чертой, расположенной в левом верхнем углу блока.

Любой блок может быть декомпозирован на составляющие его блоки. Функция декомпозиции позволяет разбить сложные процессы на составляющие его операции. При этом уровень детализации процесса определяется непосредственно разработчиком модели (рисунок 2).

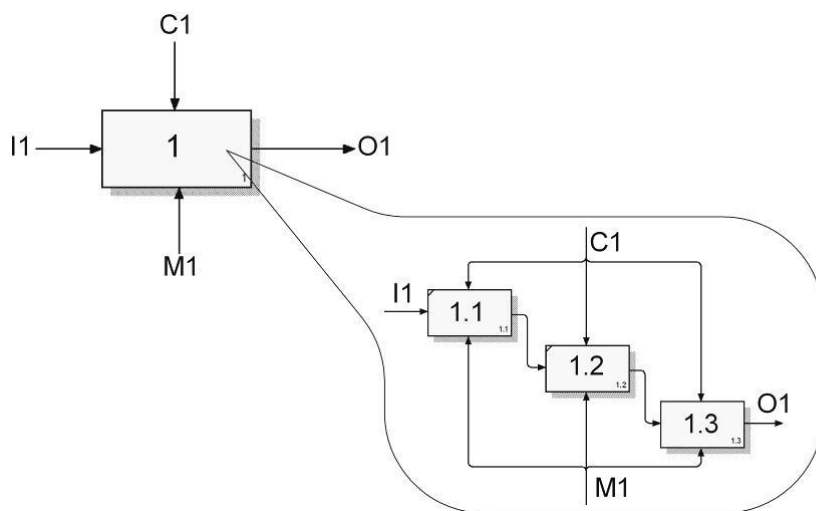


Рисунок 1.2 – Принцип декомпозиции

Декомпозиция позволяет постепенно и структурировано представлять модель системы в виде иерархической структуры отдельных диаграмм, что делает ее менее перегруженной и легко читаемой.

1.2.3 Границы и связи

Чтобы быть полезным, описание любого блока должно, как минимум, включать в себя описание объектов, которые блок создает в результате своей работы («выхода»), и объектов, которые блок потребляет или преобразует («вход»).

В IDEF0 также моделируются управление и механизмы исполнения. Под управлением понимаются объекты, воздействующие на способ, которым блок преобразует вход в выход. Механизм исполнения – объекты, которые непосредственно выполняют преобразование входа в выход, но не потребляются при этом сами по себе.

Для отображения категорий информации, присутствующих на диаграммах IDEF0, существует аббревиатура ICOM, отображающая четыре возможных типа стрелок:

I (Input) – вход – нечто, что потребляется в ходе выполнения процесса;

C (Control) – управление – ограничения и инструкции, влияющие на ход выполнения процесса;

O (Output) – выход – нечто, являющееся результатом выполнения процесса;

M (Mechanism) – исполняющий механизм – нечто, что используется для выполнения процесса, но не потребляется само по себе (рисунок 3).

Рисунок 3 показывает четыре возможных типа стрелок в IDEF0, каждый из типов соединяется со своей стороной функционального блока.

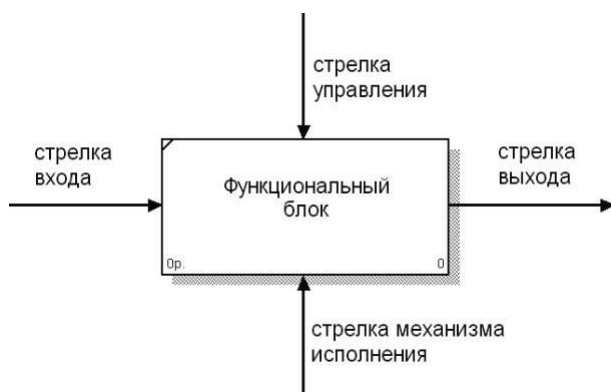


Рисунок 1.3 – Типы стрелок функционального блока

Для названия стрелок, как правило, употребляются имена существительные. Стрелки могут представлять собой людей, места, вещи, идеи или события. Как и в случае с функциональными блоками, присвоение имен всем стрелкам на диаграмме является только необходимым условием для понимания читателем сути изображенного. Отдельное описание каждой стрелки в текстовом виде может оказаться критическим фактором для построения точной и полезной модели.

Стрелки входа. Вход представляет собой сырье, или информацию, потребляемую или преобразуемую функциональным блоком для производства выхода. Стрелки входа всегда направлены в левую сторону прямоугольника, обозначающего в IDEF0 функциональный блок. Наличие входных стрелок на диаграмме не является обязательным, так как возможно, что некоторые блоки ничего не преобразуют и не изменяют. Примером блока, не имеющего входа, может служить «принятие решения руководством», где для принятия решения анализируется несколько факторов, но ни один из них непосредственно не преобразуется и не потребляется в результате принятия какого-либо решения.

Стрелки управления. Стрелки управления отвечают за регулирование того, как и когда выполняется функциональный блок, и, если он выполняется, какой выход получается в результате его выполнения. Так как управление контролирует поведение функционального блока для обеспечения создания желаемого выхода, каждый функциональный блок должен иметь, как минимум, одну стрелку управления. Стрелки управления всегда входят в функциональный блок сверху.

Управление часто существует в виде правил, инструкций, законов, политики, набора необходимых процедур или стандартов. Влияя на работу блока, оно непосредственно не потребляется и не трансформируется в результате. Может оказаться, что целью функционального блока является как раз изменение того или иного правила, инструкции, стандарта и т.п. В этом случае стрелка, содержащая соответствующую информацию, должна рассматриваться не как управление, а как вход функционального блока.

Управление можно рассматривать как специфический вид входа. В случаях, когда неясно, относить ли стрелку к входу или к управлению, предпочтительно относить ее к управлению до момента, пока неясность не будет разрешена.

Стрелки выхода. Выход – это продукция или информация, получаемая в результате работы функционального блока. Каждый блок должен иметь, как минимум, один выход. Действие, которое не производит никакого четко определяемого выхода, не должно моделироваться вообще (по меньшей мере, должно рассматриваться в качестве одного из первых кандидатов на исключение из модели).

При моделировании непроектируемых предметных областей выходами, как правило, являются данные, в каком-либо виде обрабатываемые функциональным блоком.

В этом случае важно, чтобы названия стрелок входа и выхода были достаточно различимы по своему смыслу. Например, блок «Прием пациентов» может иметь стрелку «Данные о пациенте» как на входе, так и на выходе. В такой ситуации входящую стрелку можно назвать «Предварительные данные о пациенте», а исходящую – «Подтвержденные данные о пациенте».

Стрелки механизма исполнения. Механизмы являются ресурсом, который непосредственно исполняет моделируемое действие. С помощью механизмов исполнения могут моделироваться: ключевой персонал, техника и (или) оборудование. Стрелки механизма исполнения могут отсутствовать в случае, если оказывается, что они не являются необходимыми для достижения поставленной цели моделирования.

Комбинированные стрелки. В IDEF0 существует пять основных видов комбинированных стрелок: выход – вход, выход – управление, выход – механизм исполнения, выход – обратная связь на управление и выход – обратная связь на вход.

Стрелка *выход – вход* применяется, когда один из блоков должен полностью завершить работу перед началом работы другого блока (рисунок 4).

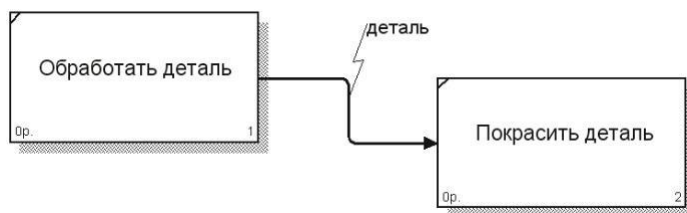


Рисунок 1.4 – Комбинированная стрелка *выход – вход*

Стрелка *выход – управление* отражает ситуацию преобладания одного блока над другим, когда один блок управляет работой другого. На рисунке 5 принципы формирования инвестиционного портфеля управляют поведением брокеров на бирже.



Рисунок 1.5 – Комбинированная стрелка *выход – управление*

Стрелки *выход – механизм исполнения* встречаются реже и отражают ситуацию, когда выход одного функционального блока применяется в качестве оборудования для работы другого блока. На рисунке 6 зажим – устройство, используемое для закрепления детали во время ее сборки, должен быть собран для того, чтобы выполнить сборку детали.

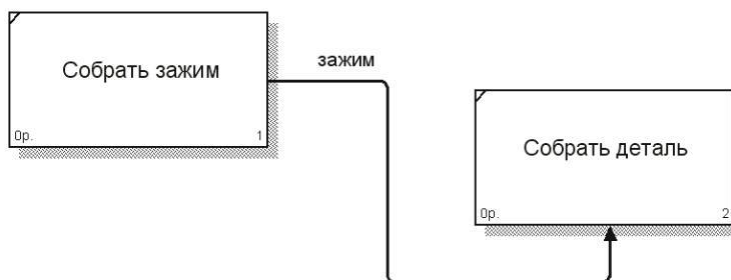


Рисунок 1.6 – Комбинированная стрелка *выход – механизм исполнения*

Обратные связи на вход и на управление применяются в случаях, когда зависимые блоки формируют обратные связи для управляющих ими блоков. На рисунке 7 получаемая от брокеров информация о текущих биржевых курсах применяется для корректировки стратегии игры на бирже.



Рисунок 1.7 – Комбинированная стрелка *выход – обратная связь на управление*

Стрелка *выход – обратная связь на вход* обычно применяется для описания циклов повторной обработки чего-либо. Рисунок 8 может служить примером применения стрелки такого типа. Кроме того, связи *выход – обратная связь на вход* могут применяться в случае, если бракованная продукция может заново использоваться в качестве сырья, как это происходит, например, при производстве оконного стекла, когда разбитое в процессе производства стекло перемалывается и переплавляется заново вместе с обыкновенным сырьем.

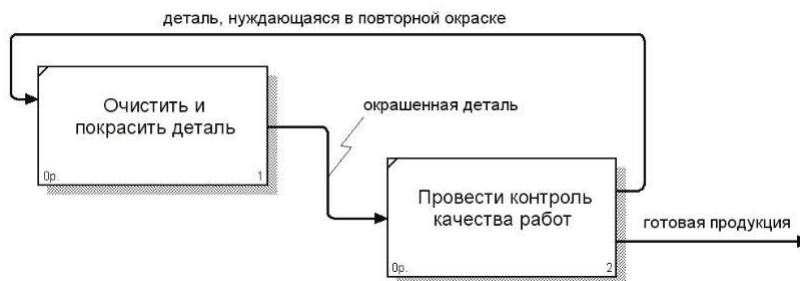


Рисунок 1.8 – Комбинированная стрелка *выход – обратная связь на вход*

Ветвление и слияние стрелок. Выход функционального блока может использоваться в нескольких других блоках. Фактически чуть ли не главная ценность IDEF0 заключается в том, что эта методология помогает выявить взаимозависимости между блоками системы. Соответственно IDEF0 предусматривает как ветвление, так и слияние стрелок на диаграмме. Разбитые на несколько частей стрелки могут иметь наименования, отличающиеся от наименования исходной стрелки. Исходная и разбитые (или объединенные) стрелки в совокупности называются связанными. Такая техника обычно применяется для того, чтобы отразить использование в процессе только части сырья или информации, обозначаемых исходной стрелкой.

Задание на лабораторную работу

- 1 Запустите **BPwin**. (Кнопка Start  /BPwin ).

2 Если появляется диалог **ModelMart Connection Manager**, нажмите на кнопку **Cancel** (Отмена).

3 Щелкните по кнопке . Появляется диалоговое окно **I would like to** (рисунок 1.9). Внесите в текстовое поле **Name** имя модели "Деятельность компании" и выберите **Type – Business Process (IDEF0)**. Нажмите кнопку **OK**.

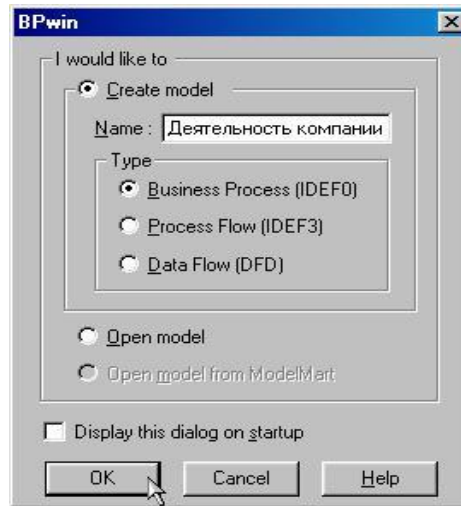


Рисунок 1.9

4 Откроется диалоговое окно **Properties for New Models** (Свойства новой модели) (рисунок 1.10).

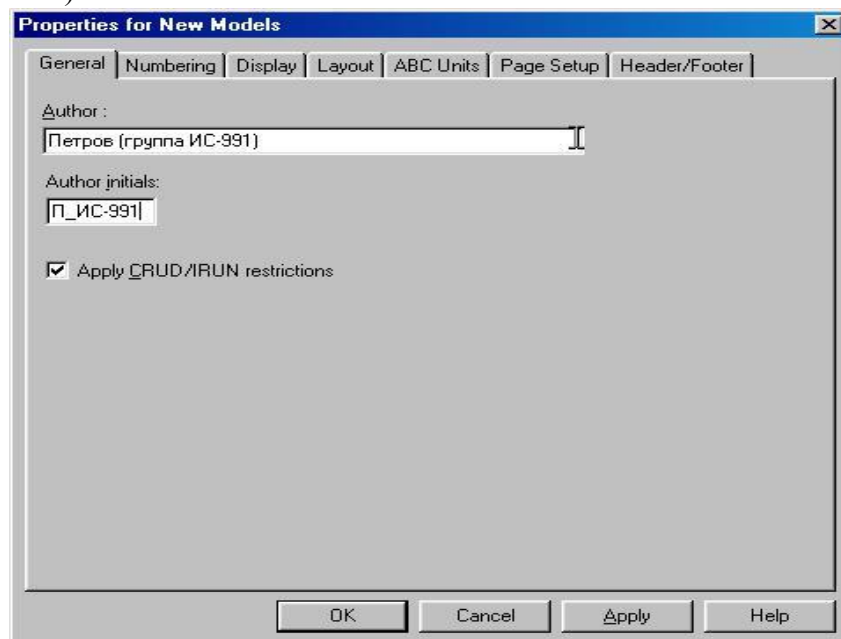


Рисунок 1.10

Введите в текстовое поле **Author** (Автор) имя автора модели и в текстовое поле **Author initials** его инициалы. Нажмите последовательно кнопки **Apply** и **OK**.

5 Автоматически создается незаполненная контекстная диаграмма (рисунок 1.11).

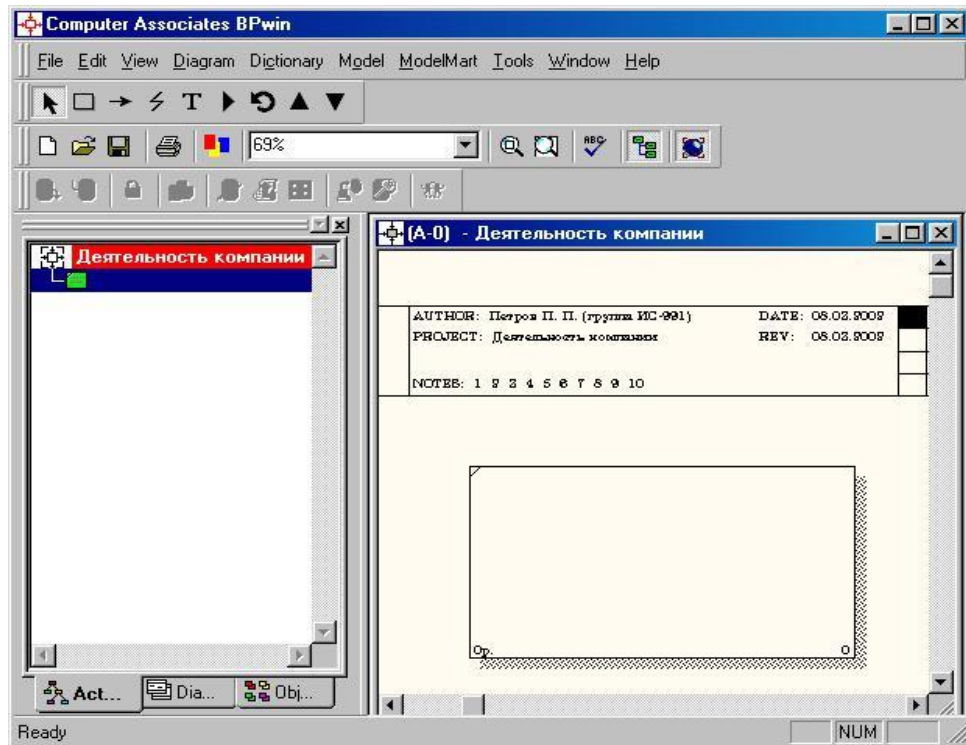


Рисунок 1.11.

6 Обратите внимание на кнопку  на панели инструментов. Эта кнопка включает и выключает инструмент просмотра и навигации – **Model Explorer** (Браузер модели). **Model Explorer** 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 1.12).

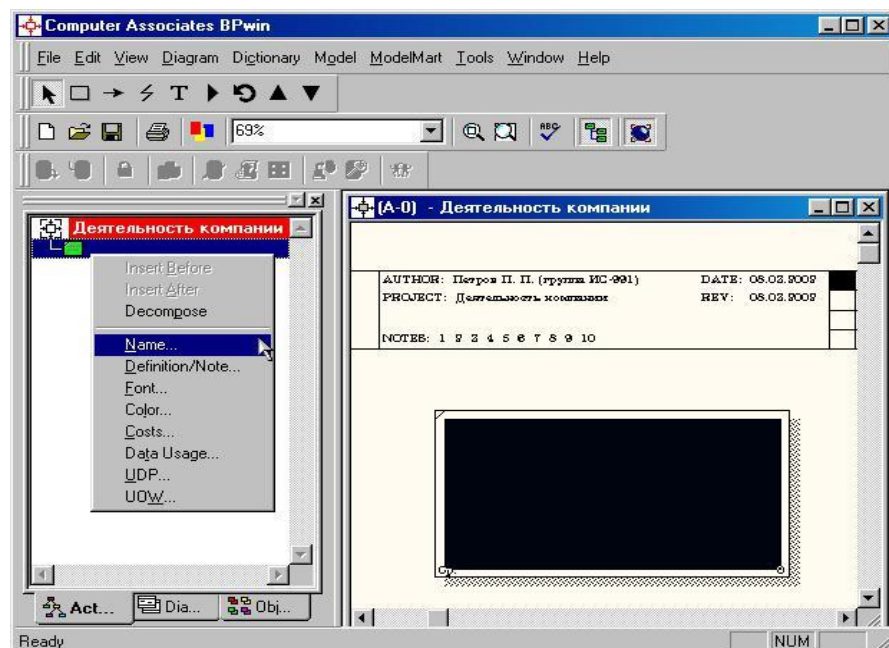


Рисунок 1.12.

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

8 Перейдите в меню **Model/Model Properties**. Во вкладке **General** диалогового окна **Model Properties** в текстовое поле **Model name** следует внести имя модели

"Деятельность компании", а в текстовое поле **Project** имя проекта "Модель деятельности компании", и, наконец, в текстовое **Time Frame** (Временной охват) – **AS-IS** (Как есть) (рисунок 1.13).

9 Во вкладке **Purpose** диалогового окна **Model Properties** 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".

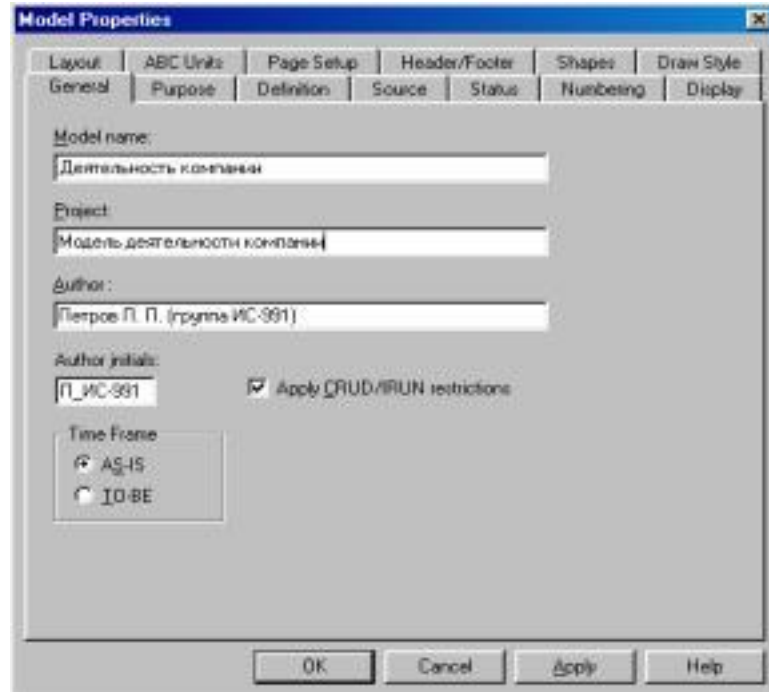


Рисунок 1.13.

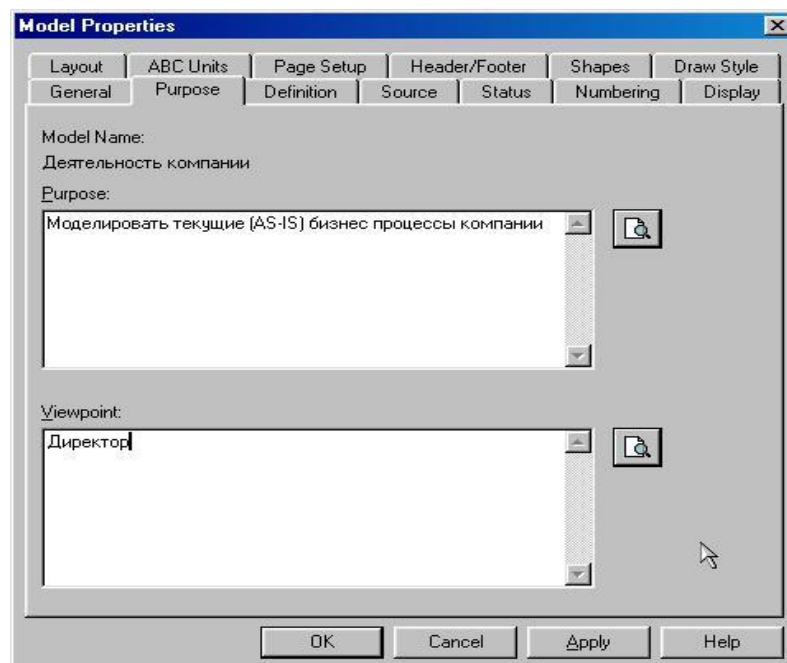


Рисунок 1.14.

10 Во вкладке **Definition** диалогового окна **Model Properties** 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".

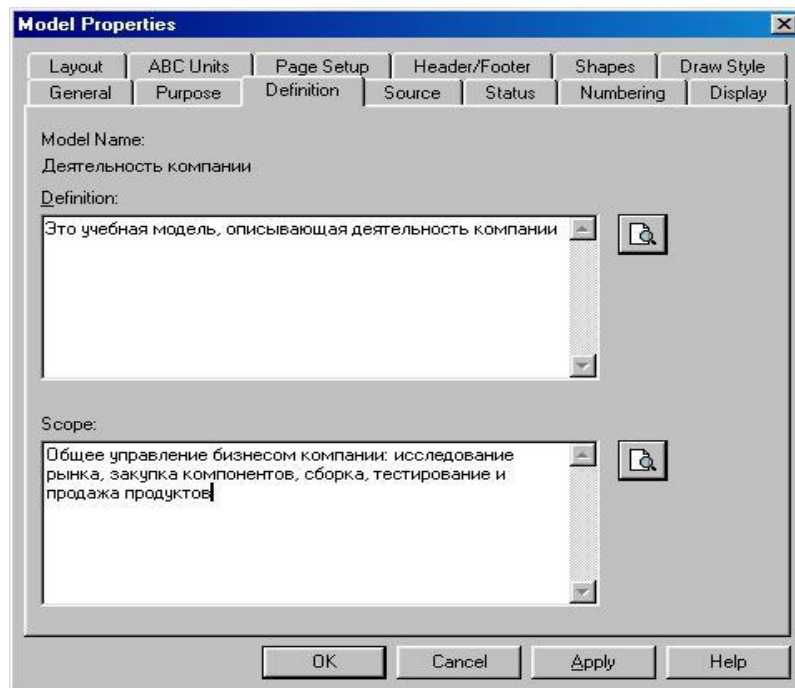


Рисунок 1.15

11 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 1.16). Во вкладке **Name** внесите имя "Деятельность компании" (рисунок 1.17).

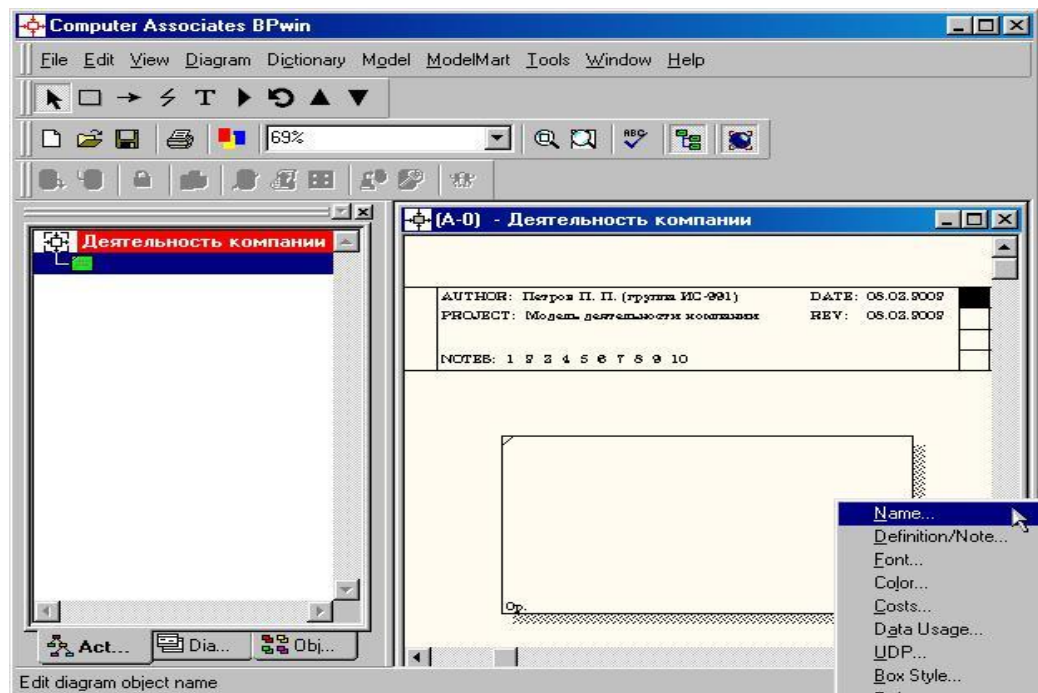


Рисунок 1.16.

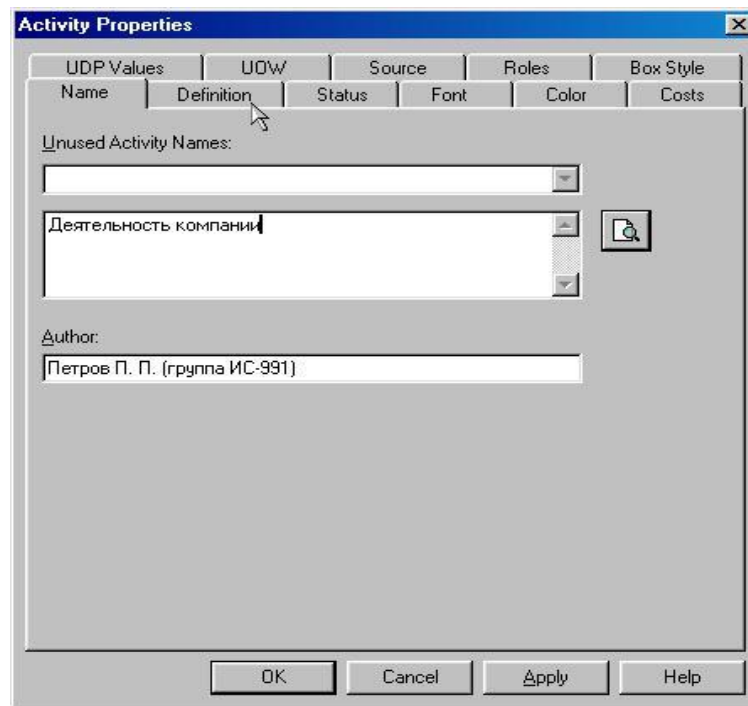


Рисунок 1.17.

Во вкладке **Definition** диалогового окна **Activity Properties** в текстовое поле **Definition** (Определение) внесите "Текущие бизнес-процессы компании" (рисунок 1.18). Текстовое поле **Note** (Примечания) оставьте незаполненным.

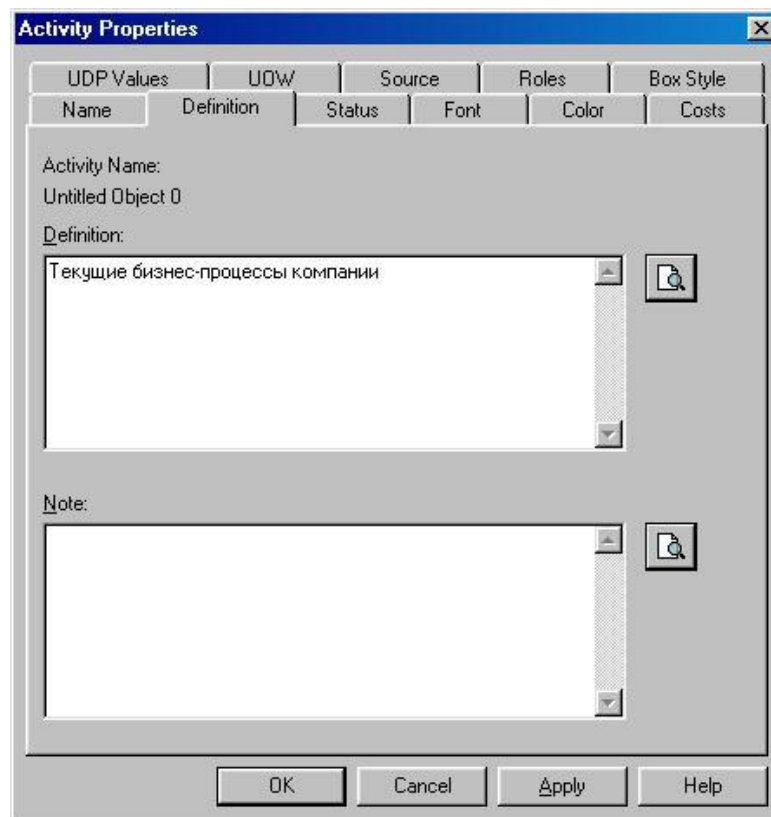


Рисунок 1.18

12 Создайте **ICOM**–стрелки на контекстной диаграмме (таблица 1).

Таблица 1. Стрелки контекстной диаграммы

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

13. С помощью кнопки **Т** внесите текст в поле диаграммы - точку зрения и цель (рисунок 1.19). Результат показан на рисунке 1.20.

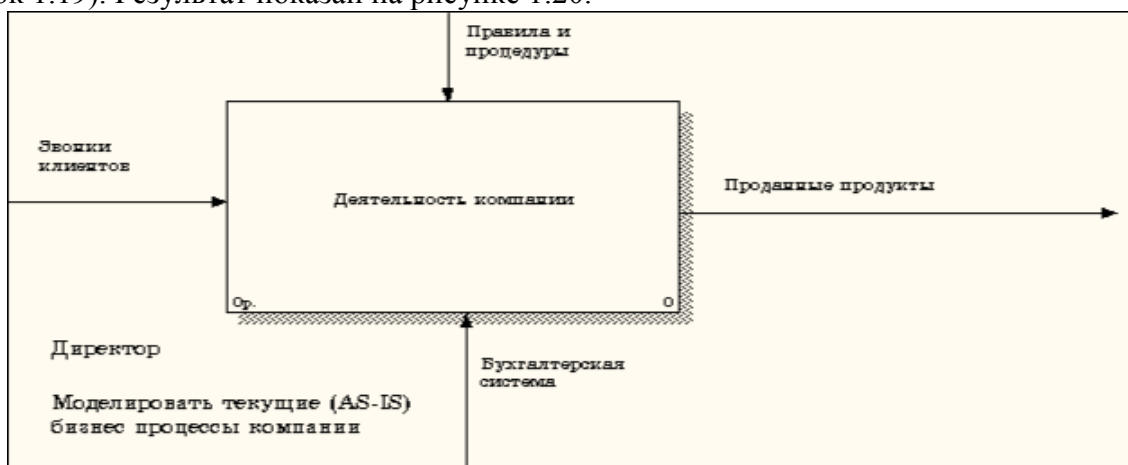


Рисунок 1.19.

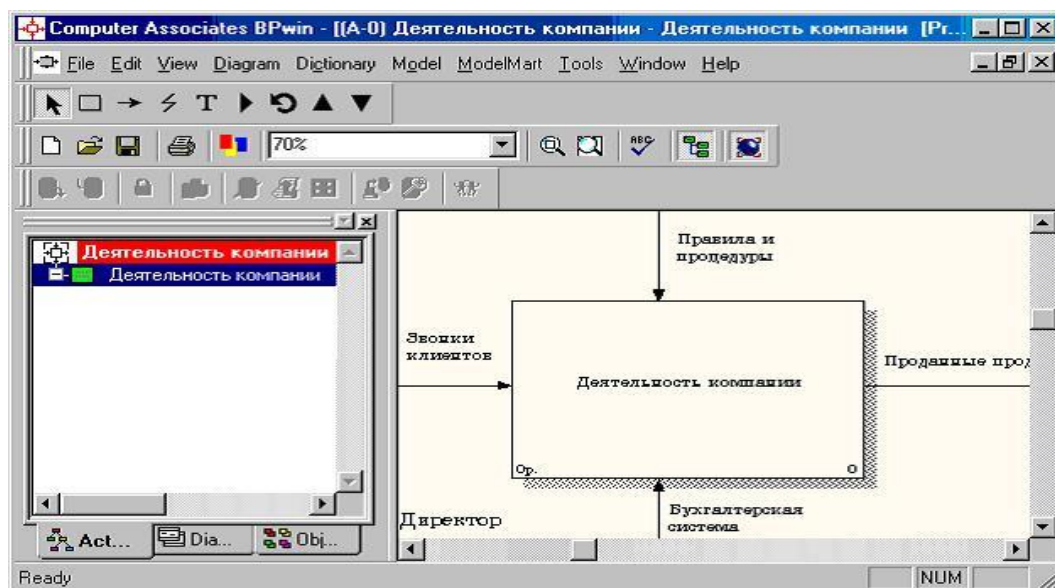


Рисунок 1.20.

Создайте отчет по модели. В меню **Tools/Reports/Model Report** (рисунок 1.21) задайте опции генерирования отчета (установите галочки) и нажмите кнопку **Preview** (Предварительный просмотр) (рисунок 1.22).

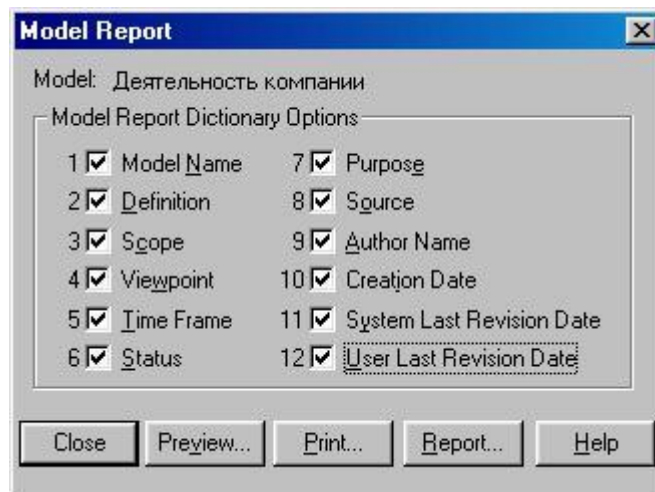


Рисунок 1.21.

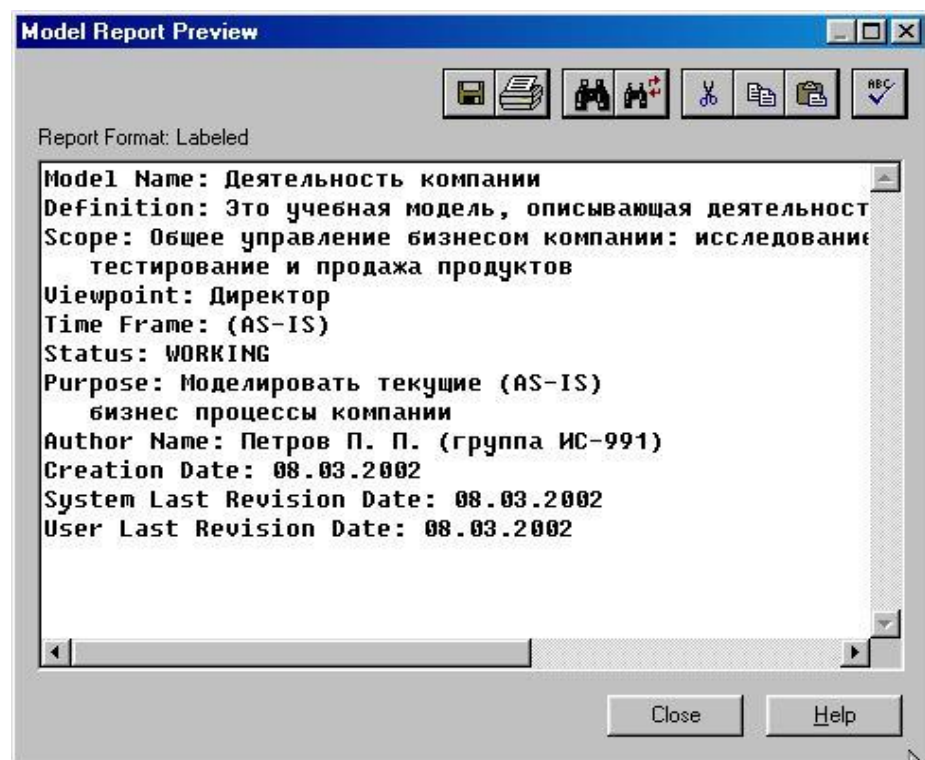


Рисунок 1.22.

Создание диаграммы декомпозиции

1 Выберите кнопку  перехода на нижний уровень в палитре инструментов и в диалоговом окне **Activity Box Count** (рисунок 1.23) установите число работ на диаграмме нижнего уровня – 3 и нажмите кнопку **OK**.

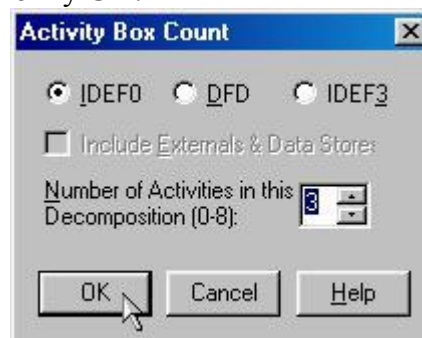


Рисунок 1.23.

2 Автоматически будет создана диаграмма декомпозиции (рисунок 1.24).

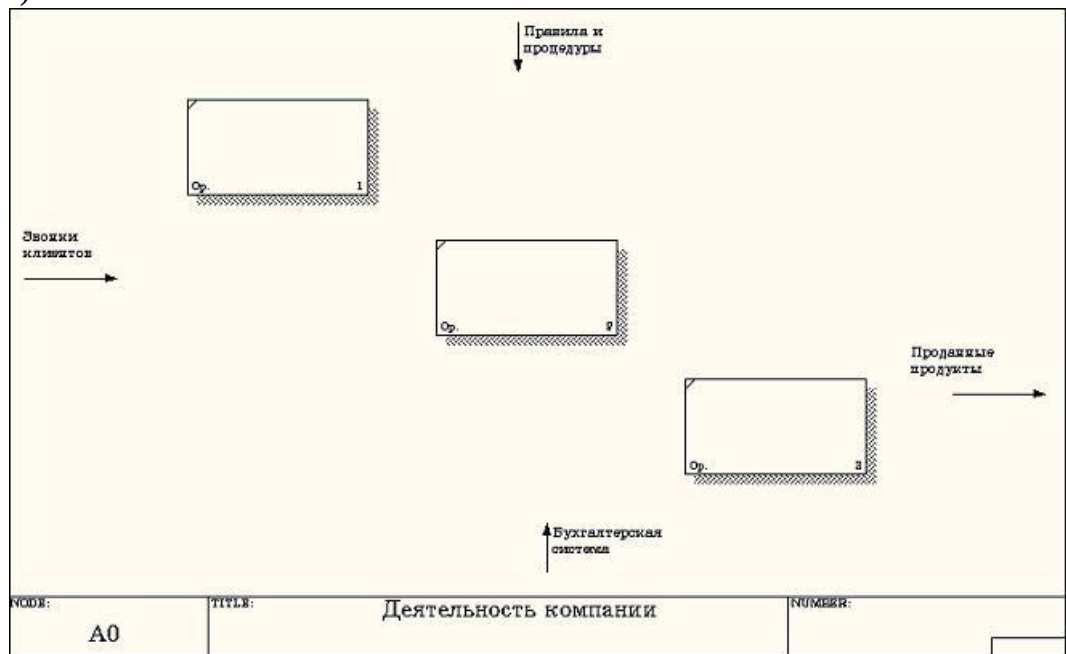


Рисунок 1.24.

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 для каждой работы согласно данным таблицы 2.

Таблица 2

Название работы (Activity Name)	Определение работы (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 1.25.

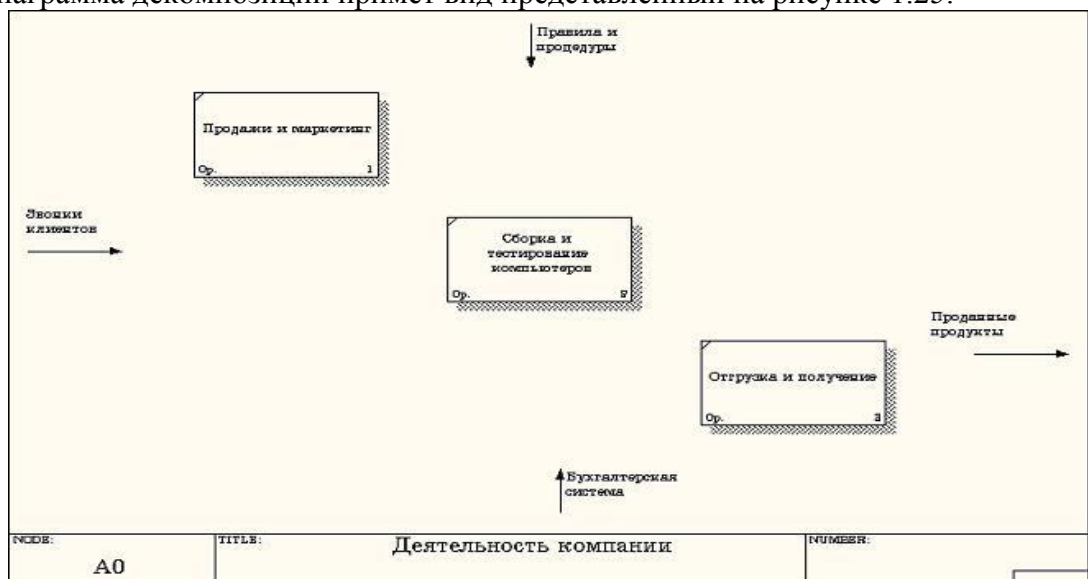


Рисунок 1.25.

3 Для изменения свойств работ после их внесения в диаграмму можно воспользоваться словарем работ (рисунок 1.26). Вызов словаря производится при помощи пункта главного меню **Dictionary /Activity**.

Activity Dictionary		
Name	Definition	Author
Деятельность	Текущие бизнес-процессы компании	Петров П. П. (грчп)
Отгрузка и пол	Отгрузка заказов клиентам и получение компонентов от поставщиков	Петров П. П. (грчп)
Продажи и мар	Телемаркетинг и презентации, выставки	Петров П. П. (грчп)
Сборка и тестирование компьютеров	Сборка и тестирование настольных и портативных компьютеров	Петров П. П. (группа ИС-991)

Рисунок 1.26.

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

4 Перейдите в режим рисования стрелок и свяжите граничные стрелки, воспользовавшись кнопкой  на палитре инструментов так, как это показано на рисунке 1.27.

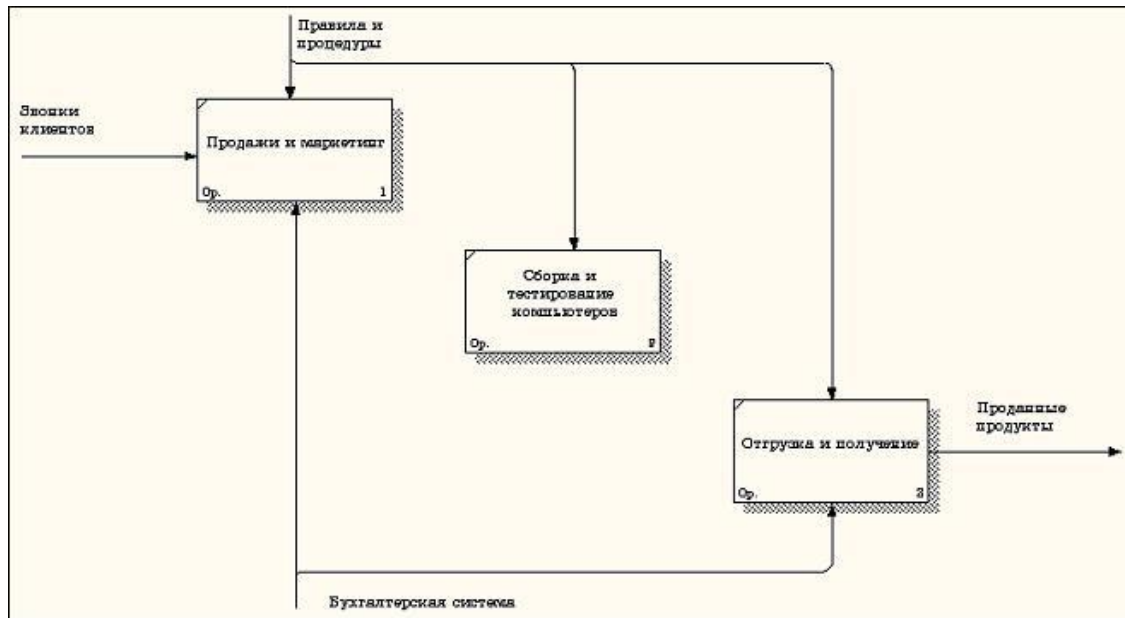


Рисунок 1.27.

5 Правой кнопкой мыши щелкните по ветви стрелки управления работы "Сборка и тестирование компьютеров" и переименуйте ее в "Правила сборки и тестирования" (рисунок 1.28).

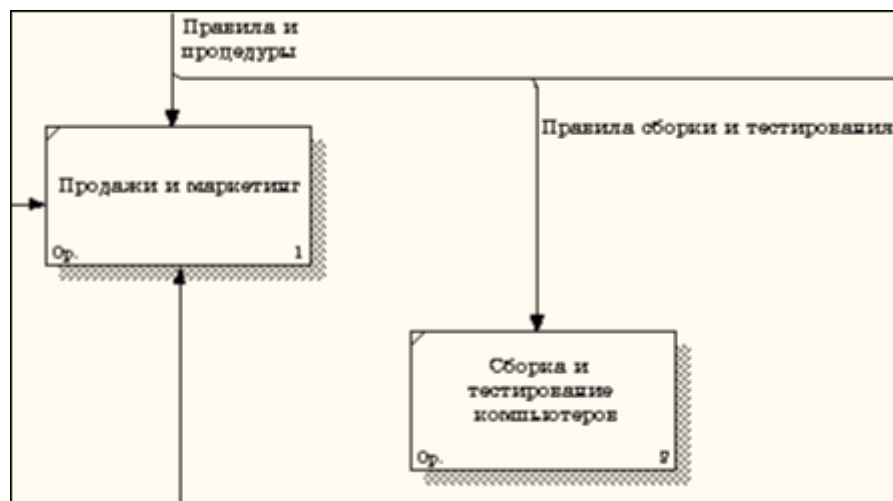


Рисунок 1.28.

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 1.29).



Рисунок 1.29

Альтернативный метод внесения имен и свойств стрелок – использование словаря стрелок (вызов словаря - меню **Dictionary/ Arrow**). Если внести имя и свойства стрелки в словарь (рисунок 1.30), ее можно будет внести в диаграмму позже.

Arrow Dictionary			
Name	Definition	Author	Status
Бухгалтерская с		Петров П. П. (группа)	WORKING
Звонки клиентов		Петров П. П. (группа)	WORKING
Маркетинговые		Петров П. П. (группа)	WORKING
Правила и проце		Петров П. П. (группа)	WORKING
Правила сборки	Инструкции по сборке, процедуры тестирования, критерии	Петров П. П. (группа)	WORKING
Прданные продук	Настольные и портативные компьютеры	Петров П. П. (группа)	WORKING
Проданные продч		Петров П. П. (группа)	WORKING
Система оформл		Петров П. П. (группа)	WORKING

Рисунок 1.30

Стрелку нельзя удалить из словаря, если она используется на какой-либо диаграмме. Если удалить стрелку из диаграммы, из словаря она не удаляется. Имя и описание такой стрелки может быть использовано в дальнейшем. Для добавления стрелки необходимо перейти в конец списка и щелкнуть правой кнопкой по последней строке. Возникает новая строка, в которой нужно внести имя и свойства стрелки.

6 Создайте новые внутренние стрелки так, как показано на рисунке 1.31.



Рисунок 1.31.

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 **Extra Arrowhead** (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** (Загогулину). Результат возможных изменений показан на рисунке 1.32.

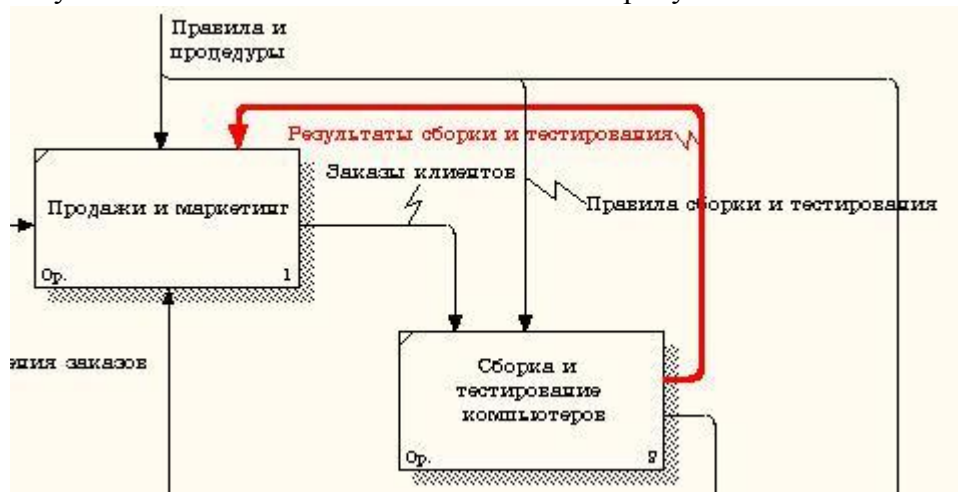


Рисунок 1.32.

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 1.33).

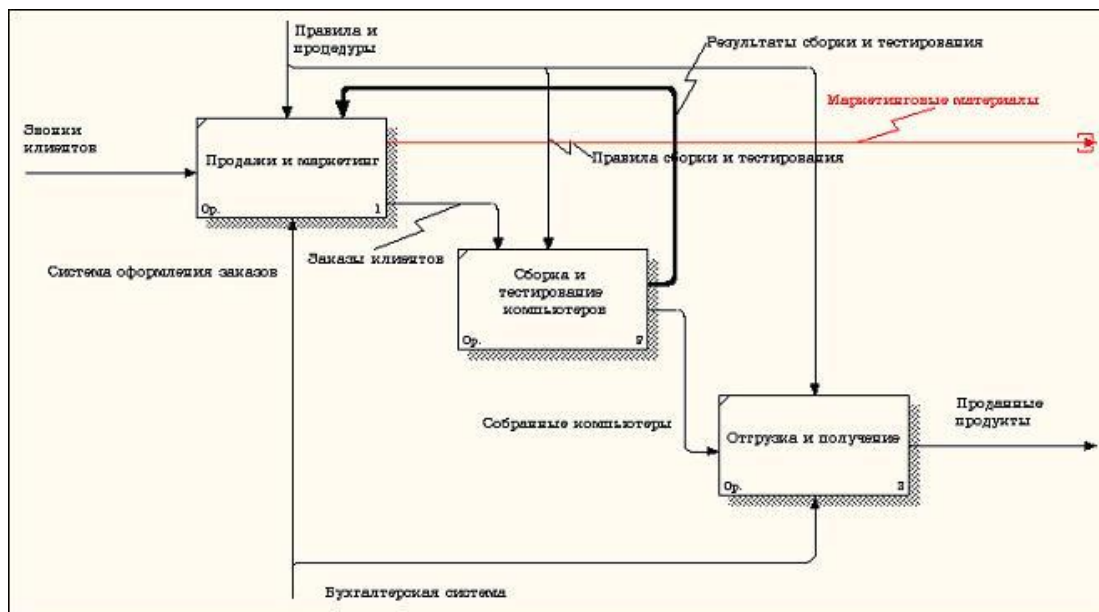


Рисунок 1.33.

Щелкните правой кнопкой мыши по квадратным скобкам и выберите пункт меню **Arrow Tunnel** (рисунок 1.34).

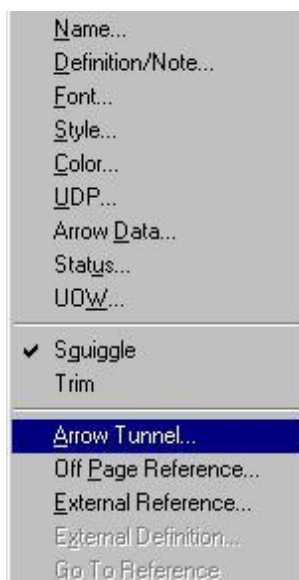


Рисунок 1.34.

В диалоговом окне **Border Arrow Editor** (Редактор Граничных Стрелок) выберите опцию **Resolve it to Border Arrow** (Разрешить как Граничную Стрелку) (рисунок 1.35).

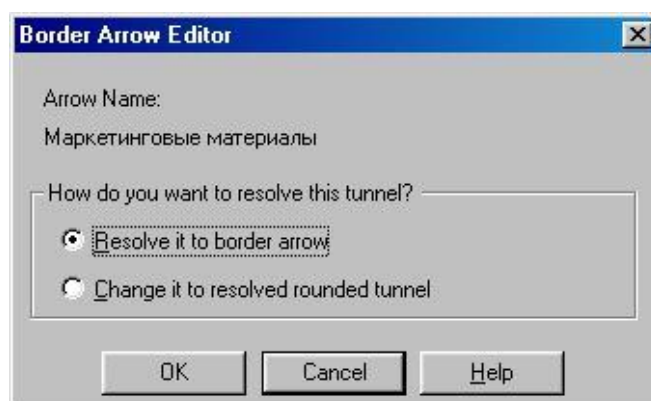


Рисунок 1.35.

Для стрелки "Маркетинговые материалы" выберите опцию **Trim** (Упорядочить) из контекстного меню. Результат показан на рис. 1.36.

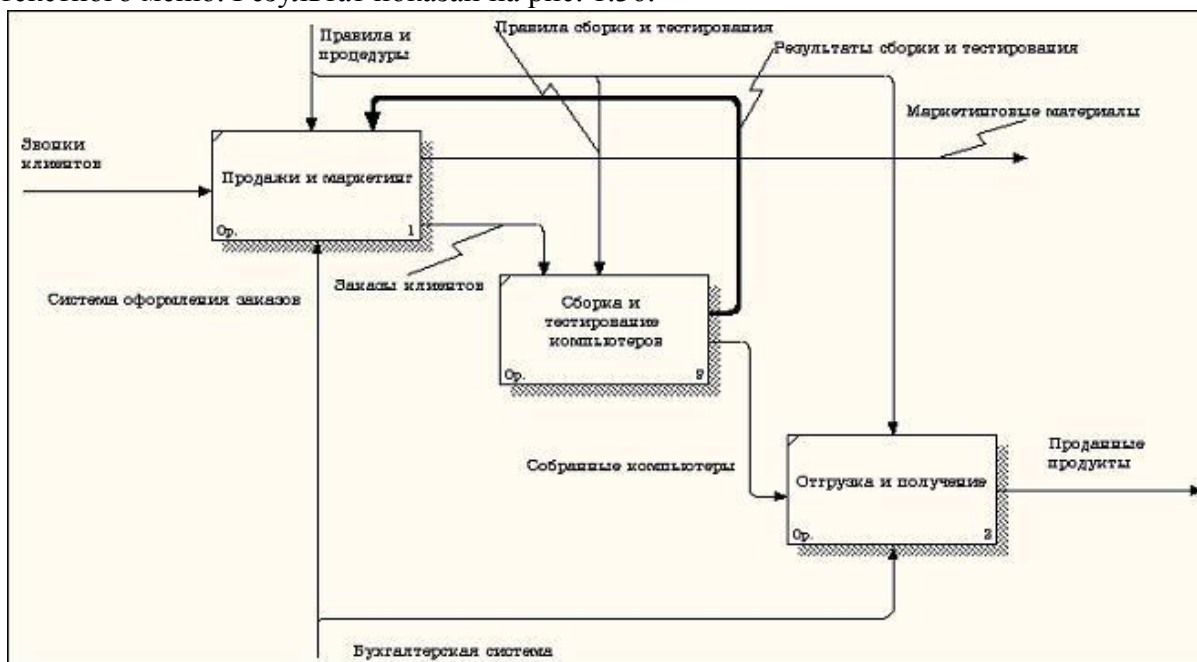


Рисунок 1.36.

Контрольные вопросы

При защите лабораторной работы студент должен ориентироваться в проделанной работе, знать:

- Цели и задачи предпроектного обследования.
- Методологию функционального моделирования.
- Как определять границы моделирования
- Как определять точку зрения при построении модели.
- Принцип декомпозиции.

Студент должен уметь изменить модель (добавить новые или удалить элементы) по требованию преподавателя.

Работа с литературой:

Рекомендуемые источники информации (№ источника)			
Основная	Дополнительная	Методическая	Интернет-ресурсы
1-2	1-3	1-3	1-4

Оценочные средства: отчет к лабораторной работе (См.: Фонд оценочных средств)

Лабораторная работа №2

«Создание диаграммы декомпозиции.»

Форма проведения: лабораторная работа

Цель работы:

Освоение принципов создания диаграмм декомпозиции. Приобретение практических навыков в создании диаграмм декомпозиции.

1. Запустите BPwin 4.0.
2. С помощью команды *File – Open* откройте ранее сохраненный проект.

Необходимо декомпозировать работу "**Сборка и тестирование компьютеров**". В результате проведения экспертизы получена следующая информация.

Производственный отдел получает заказы клиентов от отдела продаж по мере их поступления. Диспетчер координирует работу сборщиков, сортирует заказы, группирует их и дает указание на отгрузку компьютеров, когда они готовы.

Каждые 2 часа диспетчер группирует заказы - отдельно для настольных компьютеров и ноутбуков - и направляет на участок сборки.

Сотрудники участка сборки собирают компьютеры согласно спецификациям заказа и инструкциям по сборке. Когда группа компьютеров, соответствующая группе заказов, собрана, она направляется на тестирование. Тестировщики тестируют каждый компьютер и в случае необходимости заменяют неисправные компоненты.

Тестировщики направляют результаты тестирования диспетчеру, который на основании этой информации принимает решение о передаче компьютеров, соответствующих группе заказов, на отгрузку.

3. Необходимо создать диаграмму декомпозиции. Для этого следует выбрать на диаграмме работу, которую необходимо декомпозировать и щелкнуть по кнопке . В открывшемся диалоговом окне Activity Box Count следует выбрать *IDEF0* и установить число работ на диаграмме нижнего уровня – 4 (рис. 3.1).

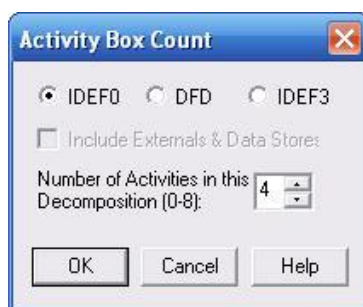


Рис. 3.1. Диалоговое окно Activity Box Count

1. Нажмите кнопку ОК. При этом автоматически создается диаграмма декомпозиции.
2. Правой кнопкой мыши щелкните по работе, расположенной в левом верхнем углу области редактирования модели, выберите в контекстном меню опцию Name и внесите имя работы «Отслеживание расписания и управление сборкой и тестированием».
3. Перейдите во вкладку **Definition** и внесите определение работы – «Просмотр заказов, установка расписания выполнения заказов, просмотр результатов тестирования, формирование групп заказов на сборку и отгрузку».
4. Повторите операцию для оставшихся трех работ согласно данным таблицы 3.1.

Таблица 3.1

Работы диаграммы декомпозиции А2

Название работы (Activity Name)	Определение работы (Activity Definition)
Отслеживание расписания и управление сборкой и тестированием	Просмотр заказов, установка расписания выполнения заказов, просмотр результатов тестирования, формирование групп заказов на сборку и отгрузку
Сборка настольных компьютеров	Сборка настольных компьютеров в соответствии с инструкциями и указаниями диспетчера
Сборка ноутбуков	Сборка ноутбуков в соответствии с инструкциями и указаниями диспетчера
Тестирование компьютеров	Тестирование компьютеров и компонентов. Замена неработающих компонентов

5. Внесите новые стрелки, согласно таблице 3.2.

Таблица 3.2

Стрелки диаграммы декомпозиции A2

№	Наименование стрелки (Arrow Name)	Источник стрелки (Arrow Source)	Тип стрелки источника (Arrow Source Type)	Приемник стрелки (Arrow Dest.)	Тип стрелки приемника (Arrow Dest. Type)
1	Персонал производственного отдела	"Tunnel"		Сборка настольных компьютеров	Mechanism
2				Сборка ноутбуков	Mechanism
3.	Диспетчер	Персонал производственного отдела		Отслеживание расписания и управление сборкой и тестированием	Mechanism
4.	Заказы клиентов	Граница диаграммы		Отслеживание расписания и управление сборкой и тестированием	Input
5.	Заказы на настольные компьютеры	Отслеживание расписания и управление сбор-	Output	Сборка настольных компьютеров	Control

		кой и тестированием			
6.	Заказы на ноут- буки	Отслеживание расписания и управление сбор- кой и тестированием	Output	Сборка ноутбуков	Control
7.	Компоненты	«Tunnel»		Сборка настольных компьютеров	Input
8.				Сборка ноутбуков	Input
9.				Тестирование компьютеров	Input
10.	Настольные ком- пьютеры	Сборка настольных компьютеров	Output	Тестирование компьютеров	Input
11.	Ноутбуки	Сборка ноутбуков	Output	Тестирование компьютеров	Input
12	Правила сборки и тестирования	Граница диаграммы		Сборка настольных компьютеров	Control
13				Сборка ноутбуков	Control
14				Тестирование компьютеров	Control
15	Результаты сборки и тести- рования	Сборка настольных компьютеров	Output	Граница диаграммы	Output
16		Сборка ноутбуков	Output		
17		Тестирование компьютеров	Output		

18	Результаты тестирования	Тестирование компьютеров	Output	Отслеживание расписания и управление сборкой и тестированием	Input
19	Собранные компьютеры	Тестирование компьютеров	Output	Граница диаграммы	Output
20	Тестирующий	Персонал производственного отдела		Тестирование компьютеров	Mechanism
21	Указание передать компьютеры на отгрузку	Отслеживание расписания и управление сборкой и тестированием	Output	Тестирование компьютеров	Control

9. Тоннелируйте и свяжите на верхнем уровне граничные стрелки, если это необходимо. Возможный результат выполнения лабораторной работы показан на рисунке 3.1.

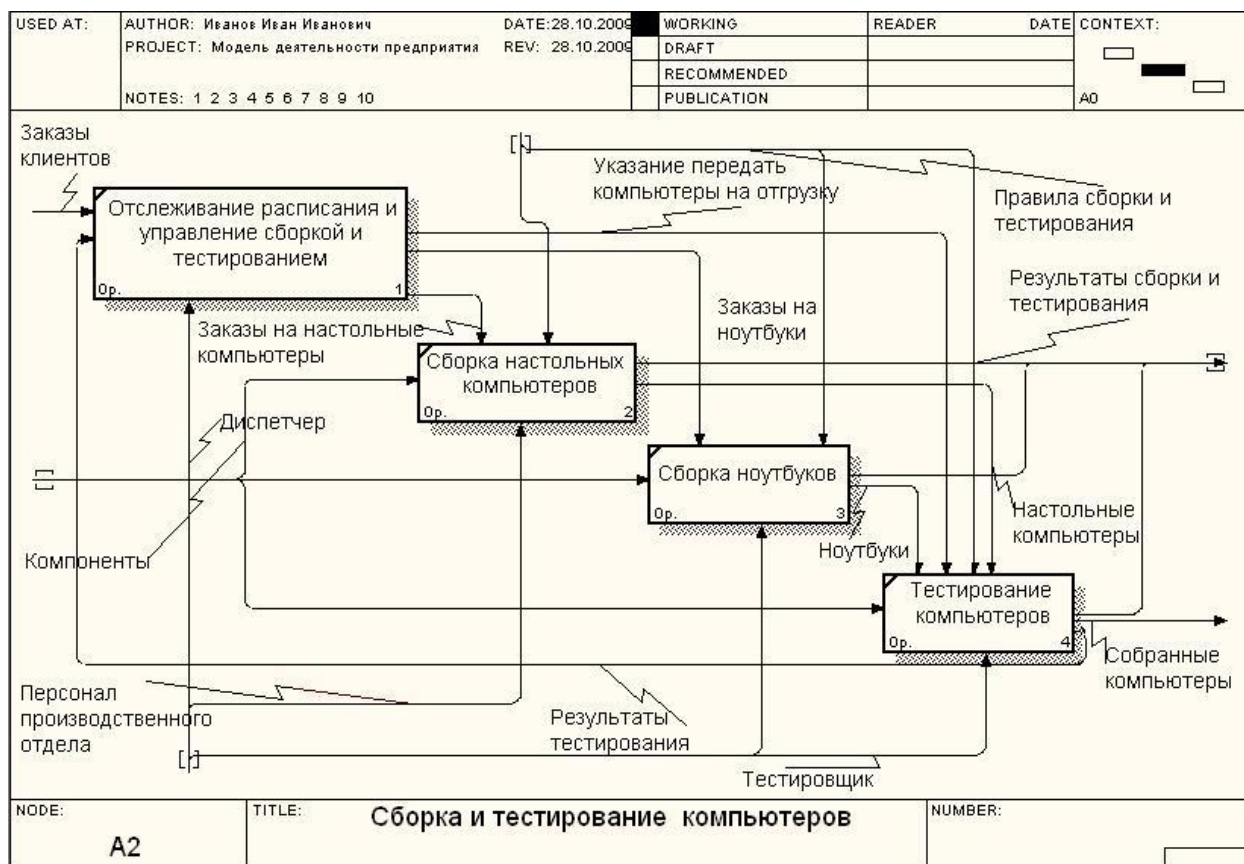


Рис. 3.1. Результат выполнения лабораторной работы

10. Закройте BPwin 4.0, сохранив все изменения.

Контрольные вопросы

При защите лабораторной работы студент должен ориентироваться в проделанной работе, знать:

- Цели и задачи проведения декомпозиции.
- Методологию формирования списка работ.
- Правила тоннелирования.

Студент должен уметь изменить модель (добавить новые или удалить элементы) по требованию преподавателя.

Работа с литературой:

Рекомендуемые источники информации (№ источника)			
Основная	Дополнительная	Методическая	Интернет-ресурсы
1-2	1-3	1-3	1-4

Оценочные средства: отчет к лабораторной работе (См.: Фонд оценочных средств)

Лабораторная работа №3

«Создание диаграммы узлов.»

Форма проведения: лабораторная работа

Цель работы:

Освоение принципов создания диаграммы узлов. Приобретение практических навыков в создании диаграммы узлов.

1. Запустите BPwin 4.0.
2. С помощью команды *File – Open* откройте ранее сохраненный проект.
3. Выберите пункт главного меню **Diagram/Add Node Tree**.
4. В первом диалоговом окне гида **Node Tree Wizard** внесите имя диаграммы, укажите диаграмму корня дерева и количество уровней (рис.4.1).

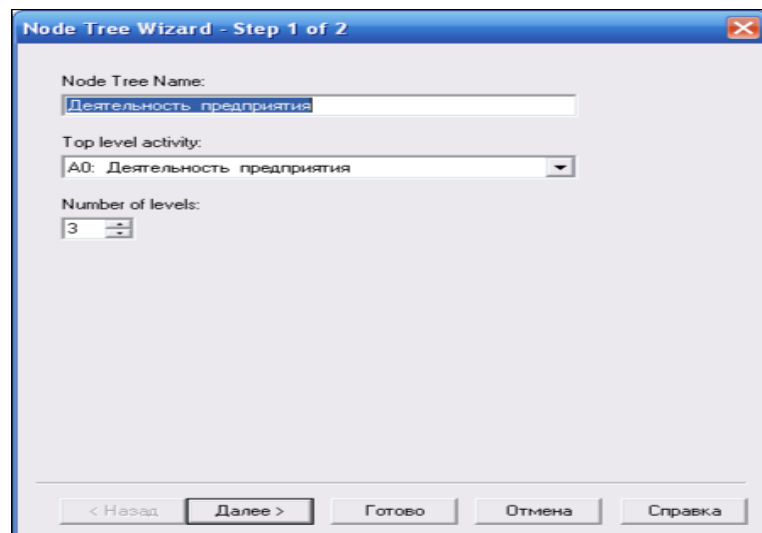


Рис. 4.1. Диалоговое окно Node Tree Wizard – Step 1

5. Во втором диалоговом окне гида **Node Tree Wizard** установите опции, как показано на рисунке 4.2.

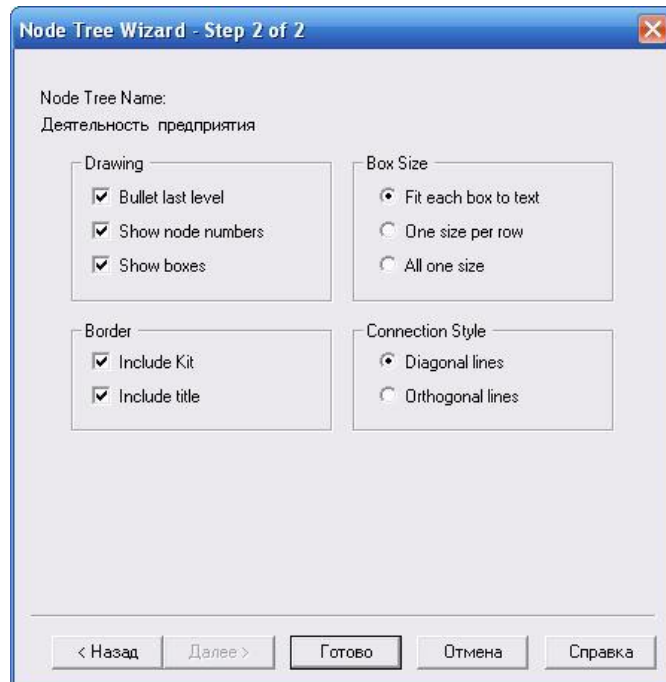


Рис. 4.2. Диалоговое окно Node Tree Wizard – Step 2

6. Щелкните по кнопке **Готово**. В результате будет создана диаграмма дерева узлов (**Node tree Diagram**) (рис. 4.3).

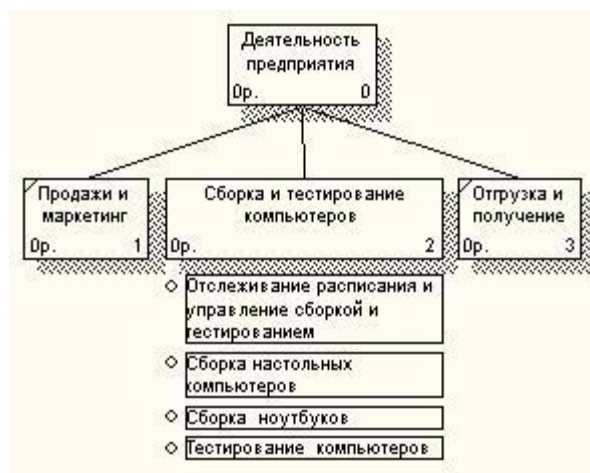


Рис. 4.3. Диаграмма дерева узлов

7. Диаграмму дерева узлов можно модифицировать. Нижний уровень может быть отображен не в виде списка, а в виде прямоугольников, так же как и верхние уровни. Для модификации диаграммы правой кнопкой мыши щелкните по свободному месту, не занятому объектами, выберите меню **Node tree Diagram Properties** и во вкладке **Style** диалога **Node tree Properties** отключите опцию **Bullet last level** (рис. 4.4).

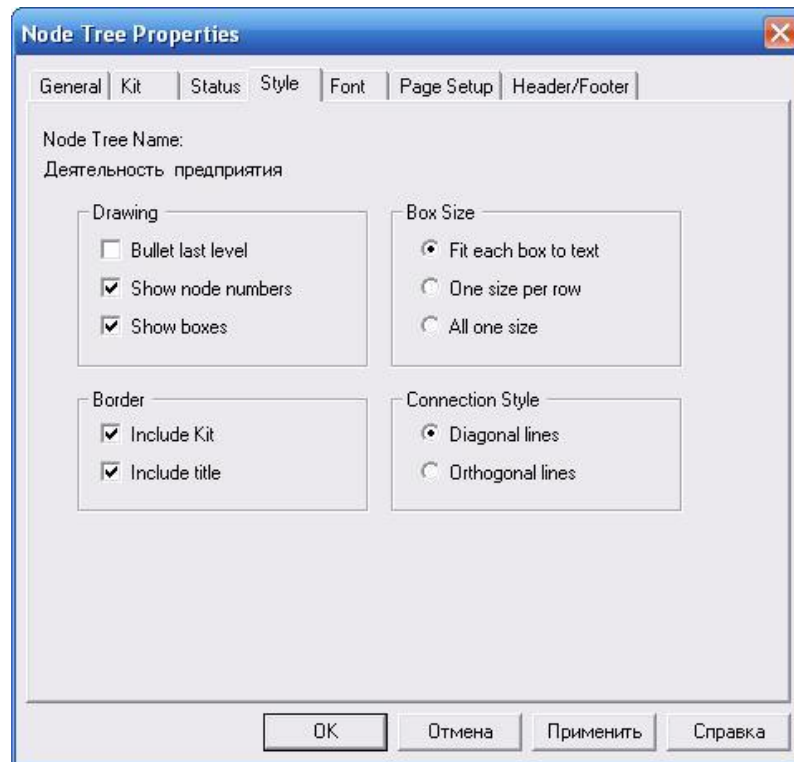


Рис. 4.4. Диалоговое окно Node tree Properties

8. Щелкните по **ОК**. Результат модификации диаграммы дерева узлов показан на рисунке 4.5.

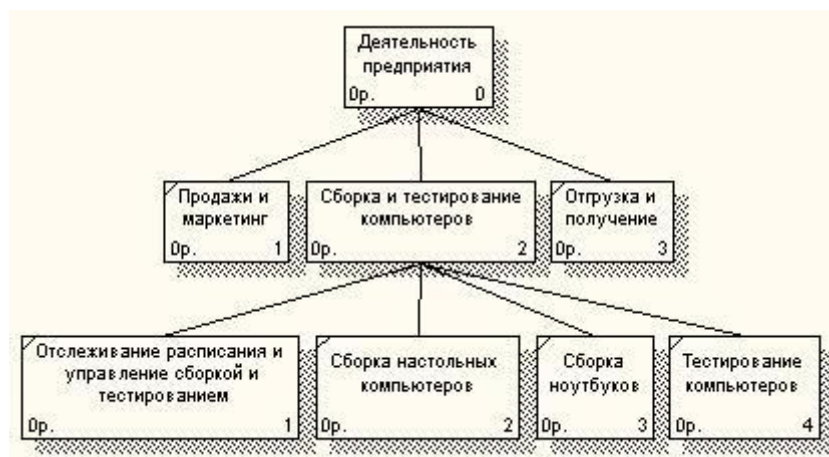


Рис. 4.5. Диаграмма дерева узлов

9. Закройте BPwin 4.0, сохранив все изменения.

Контрольные вопросы

При защите лабораторной работы студент должен ориентироваться в проделанной работе, зная:

- Цели и задачи создания дерева узлов.

- Методологию формирования дерева узлов.
- Правила модификации диаграммы дерева узлов.

Студент должен уметь изменить модель (добавить новые или удалить элементы) по требованию преподавателя.

Работа с литературой:

Рекомендуемые источники информации (№ источника)			
Основная	Дополнительная	Методическая	Интернет-ресурсы
1-2	1-3	1-3	1-4

Оценочные средства: отчет к лабораторной работе (См.: Фонд оценочных средств)

Лабораторная работа №4

«Создание FEO диаграммы.»

Форма проведения: лабораторная работа

Цель работы:

Освоение принципов создания FEO диаграммы. Приобретение практических навыков в создании FEO диаграммы.

1. Запустите BPwin 4.0.

2. С помощью команды *File – Open* откройте ранее сохраненный проект.

Можно предположить, что при обсуждении бизнес-процессов возникла необходимость детально рассмотреть взаимодействие работы "**Сборка и тестирование компьютеров**" с другими работами. Чтобы не портить диаграмму декомпозиции, создайте **FEO**-диаграмму (**FEO** - расшифровывается как «только для экспозиции»), на которой будут только стрелки работы "**Сборка и тестирование компьютеров**".

3. Нажмите на кнопку  (если она уже не нажата) на панели инструментов. В браузере модели (**Model Explorer**) во вкладке **Activities** активизируйте самую верхнюю работу **«Деятельность предприятия»**.

4. Выберите пункт главного меню **Diagram/Add FEO Diagram**.

5. В диалоговом окне **Add New FEO Diagram** выберите тип и внесите имя диаграммы **FEO** как показано на рис. 5.1. Щелкните по кнопке **OK**.

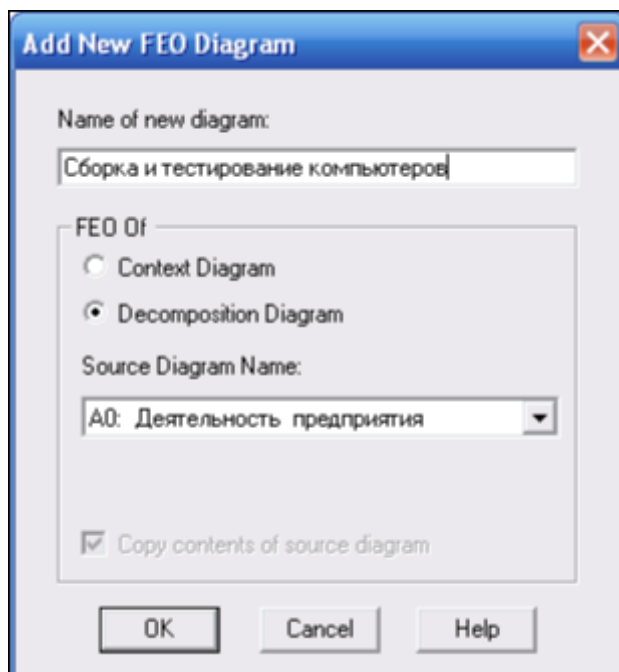


Рис. 5.1. Диалоговое окно Add New FEO Diagram

6. Для определения содержания диаграммы перейдите в пункт меню **Diagram/Diagram Properties** и во вкладке **Diagram Text** внесите определение – FEO-диаграмма, показывающая стрелки работы "Сборка и тестирование компьютеров" (рис. 5.2) и щелкните OK.

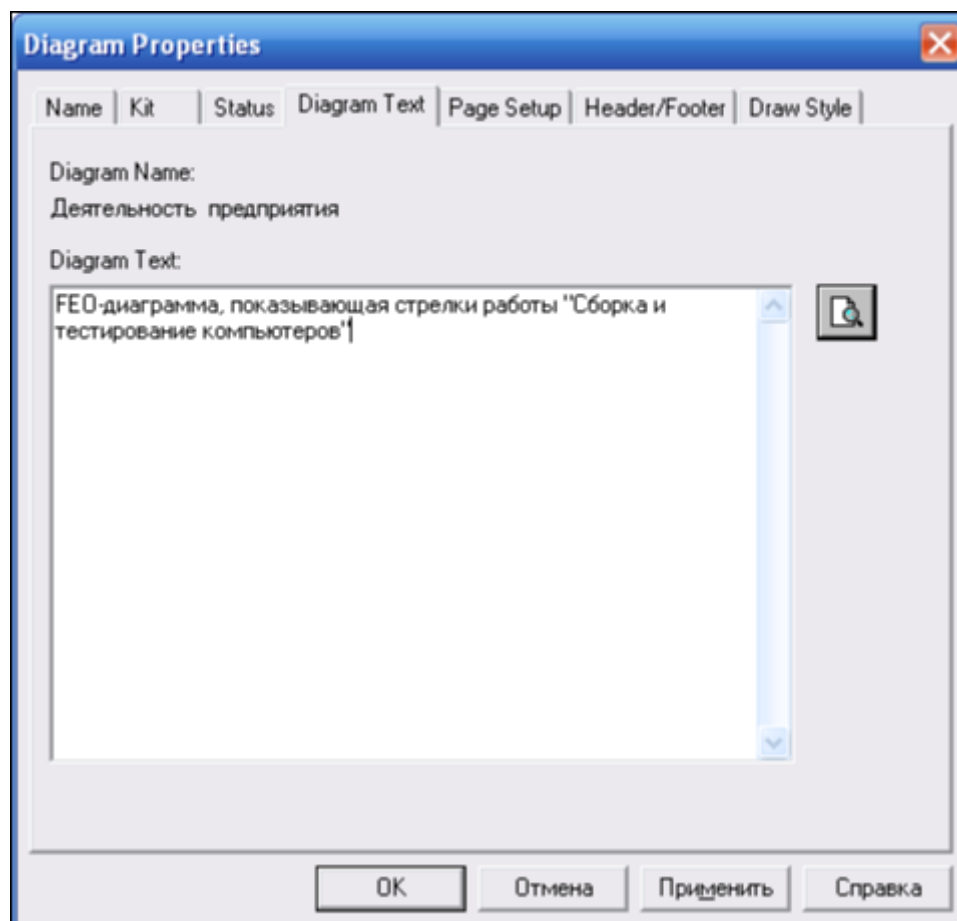


Рис. 5.2. Окно Diagram Properties

7. Удалите лишние стрелки на диаграмме **FEO** как показано на рисунке 5.3.

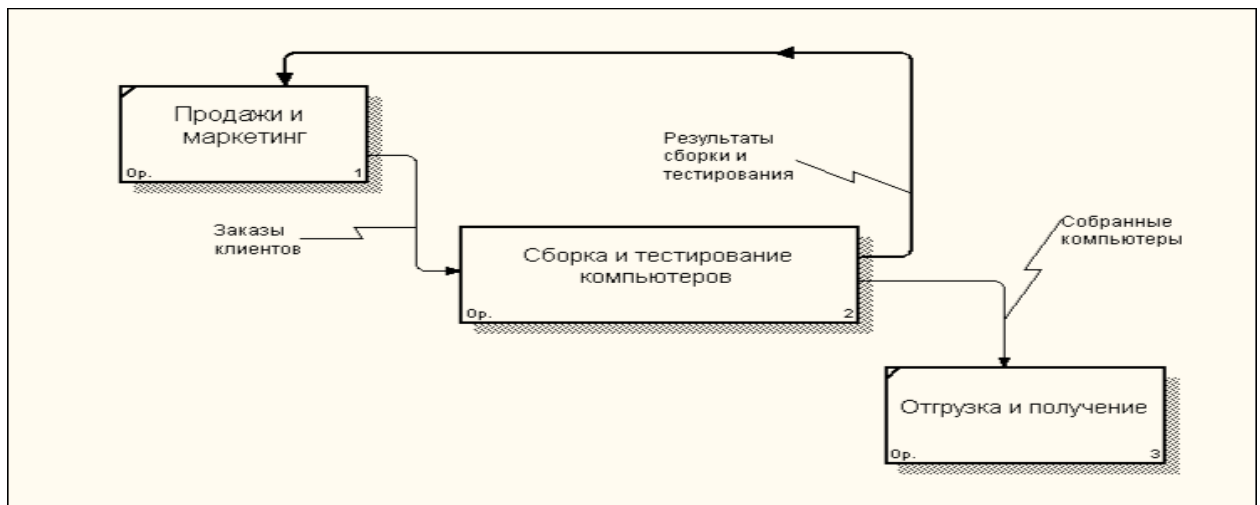


Рис. 5.3. Диаграмма FEO

8. Нажмите на кнопку  на палитре инструментов для перехода между стандартной диаграммой, деревом узлов и FEO.
9. В браузере модели (**Model Explorer**) во вкладке **Activities** активизируйте работу «**Сборка настольных компьютеров**» и нажмите правой кнопкой мыши на квадратные скобки в начале стрелки «**Компоненты**». Из контекстного меню выберите команду «**Arrow Tunnel...**».
10. Просмотрите как изменилась FEO-диаграмма (рис. 5.4).

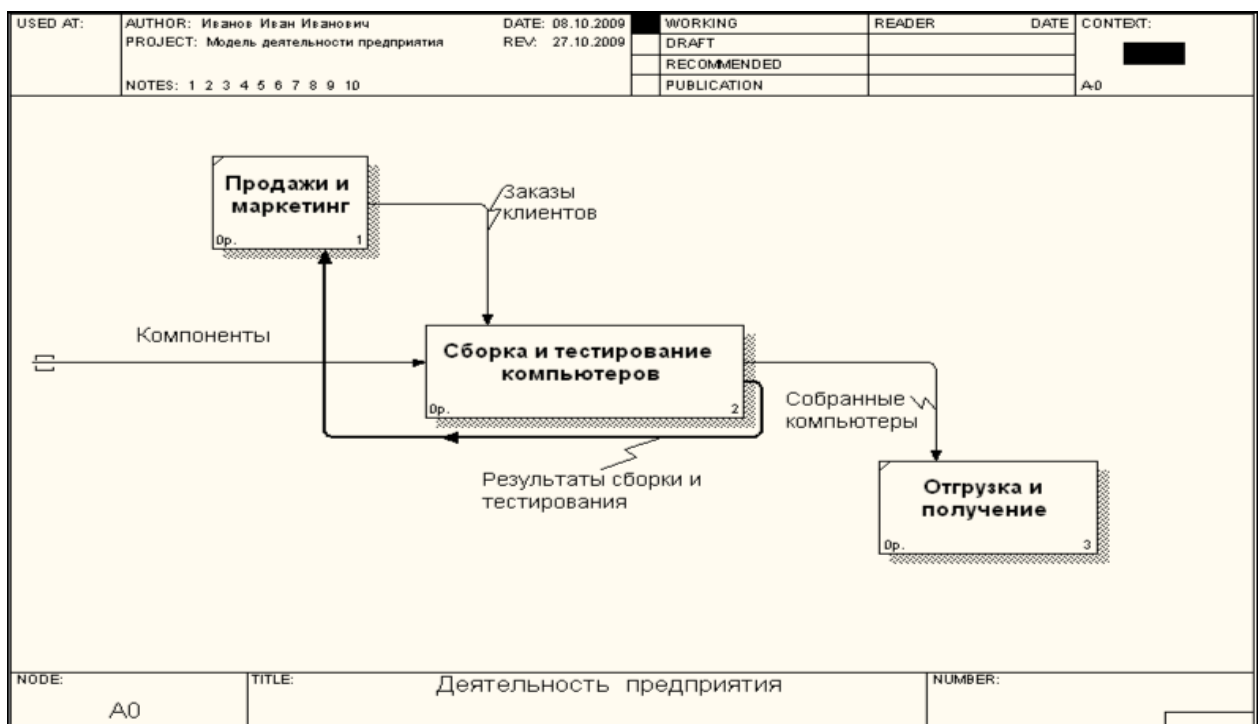


Рис. 5.4. Результат выполнения работы

11. Закройте BPwin 4.0, сохранив все изменения.

Контрольные вопросы

При защите лабораторной работы студент должен ориентироваться в проделанной работе, знать:

- Цели и задачи создания FEO диаграмм.
- Методологию формирования FEO диаграмм.
- Правила модификации FEO диаграмм.

Студент должен уметь изменить модель (добавить новые или удалить элементы) по требованию преподавателя.

Работа с литературой:

Рекомендуемые источники информации (№ источника)			
Основная	Дополнительная	Методическая	Интернет-ресурсы
1-2	1-3	1-3	1-4

Оценочные средства: отчет к лабораторной работе (См.: Фонд оценочных средств)

Лабораторная работа №5

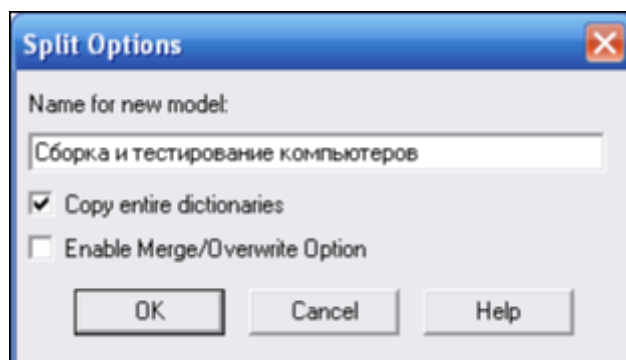
«Расщепление и слияние моделей.»

Форма проведения: лабораторная работа

Цель работы:

Освоение принципов расщепления и слияния моделей. Приобретение практических навыков в расщеплении и слиянии моделей.

1. Запустите BPwin 4.0.
2. С помощью команды *File – Open* откройте ранее сохраненный проект.
3. Перейдите на диаграмму A2. Правой кнопкой мыши щелкните по работе «**Сборка и тестирование компьютеров**» и выберете **Split model** (Разделить модель).
4. В диалоге **Split Option** (Опции разделения) внесите имя новой модели «**Сборка и тестирование компьютеров**», установите опции, как на рисунке 6.1, и щелкните по кнопке **ОК**.

**Рис. 6.1.** Диалоговое окно Split Option

5. В **Model Explorer** появилась новая модель (рис. 6.2), а на диаграмме АО модели «Деятельность предприятия» появилась стрелка вызова «Сборка и тестирование компьютеров» (рис. 6.3).

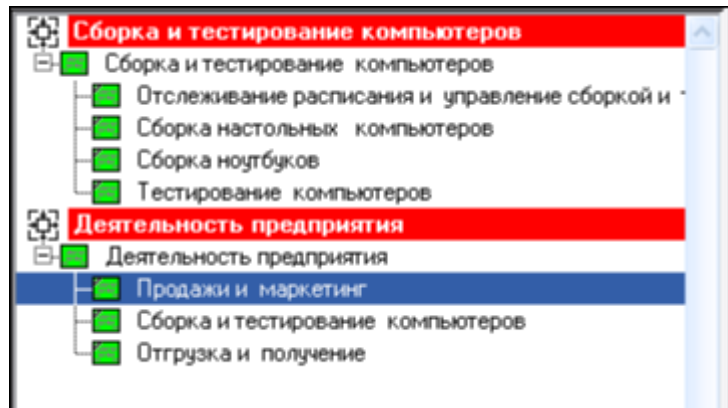


Рис. 6.2. Новая модель «Сборка и тестирование компьютеров» в Model Explorer

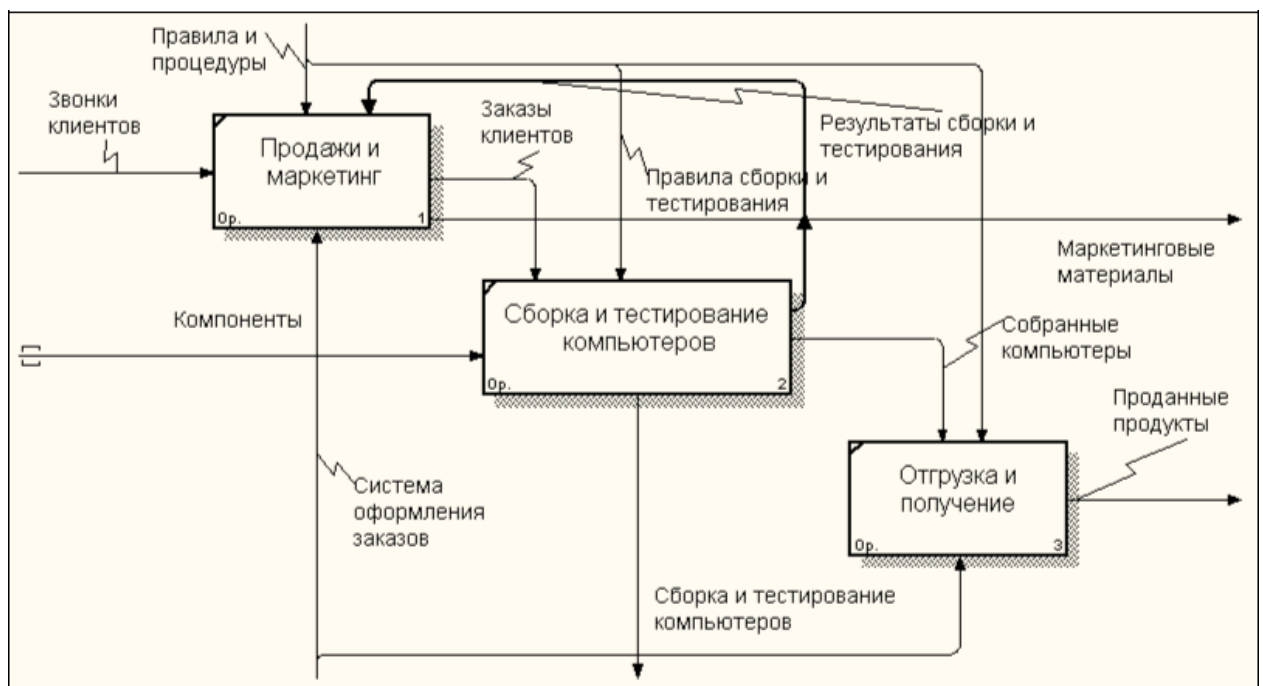


Рис. 6.3. Стрелка вызова "Сборка и тестирование компьютеров" на диаграмме АО модели "Деятельность предприятия"

6. Создайте в модели «Сборка и тестирование компьютеров» новую стрелку «Неисправные компоненты». На диаграмме АО это будет граничная стрелка выхода, на диаграмме А-0 - граничная стрелка выхода от работ «Сборка настольных компьютеров», «Тестирование компьютеров» и «Сборка ноутбуков» (рис. 6. 4).

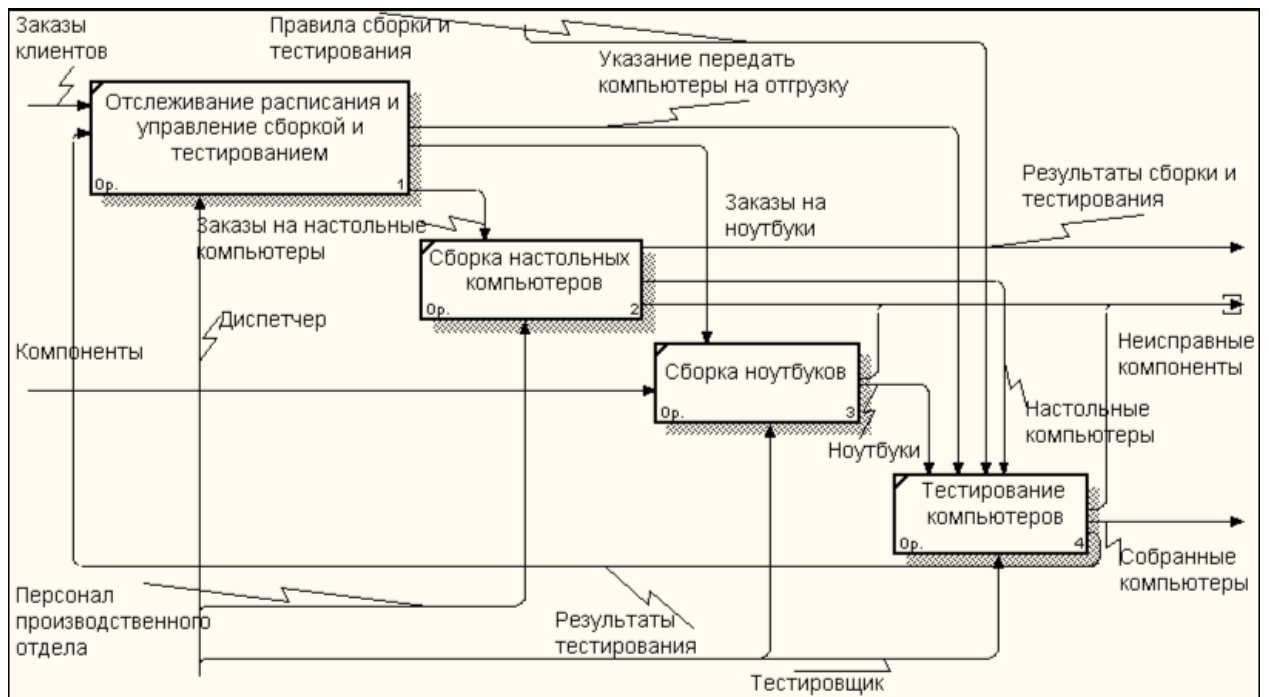


Рис. 6.4. Граничная стрелка выхода от работ "Сборка настольных компьютеров", "Тестирование компьютеров" и "Сборка ноутбуков"

7. Перейдите на диаграмму АО модели «**Деятельность предприятия**».
8. Правой кнопкой мыши щелкните по работе «**Сборка и тестирование компьютеров**» и выберите в контекстном меню опцию **Merge model**.
9. В диалоговом окне **Merge Model** включите опцию **Cut/Paste entire dictionaries** и щелкните по кнопке **OK** (рис. 6.5).

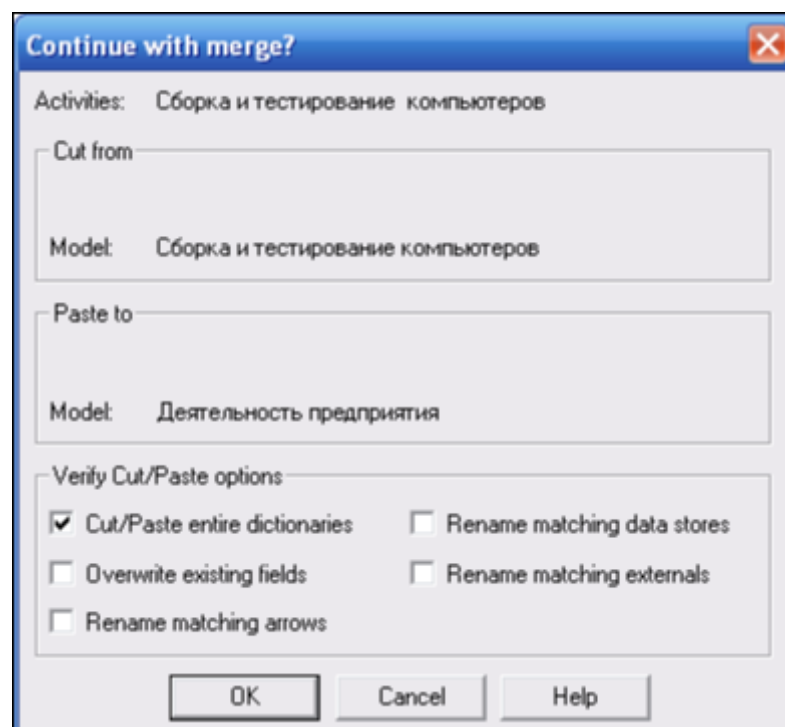


Рис. 6.5. Диалоговое окно «Continue with merge?»

10. Посмотрите на результат. В Model Explorer видно, что две модели слились. Модель **"Сборка и тестирование компьютеров"** осталась и может быть сохранена в отдельном файле. В результате, на диаграмме АО модели **«Деятельность предприятия»** исчезла стрелка вызова **«Сборка и тестирование компьютеров»**.

11. Появилась неразрешенная граничная стрелка **«Неисправные компоненты»**. Направьте эту стрелку к входу работы **«Отгрузка и получение»** (рис. 6.6).

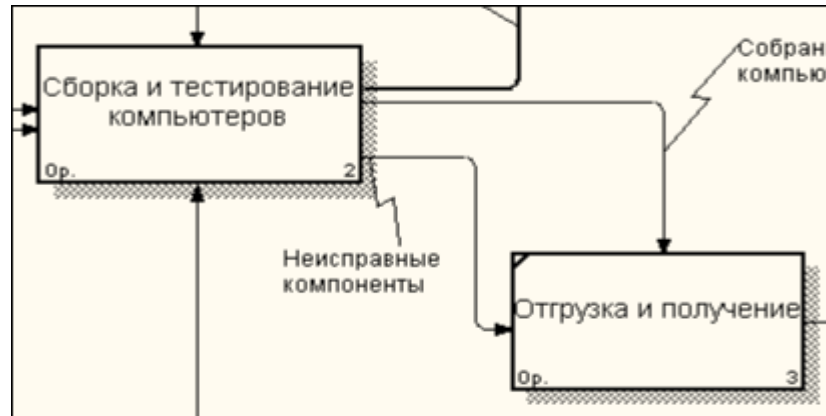


Рис. 6.6. Изменение направления стрелки «Неисправные компоненты»

12. Закройте BPwin 4.0, сохранив все изменения.

Контрольные вопросы

При защите лабораторной работы студент должен ориентироваться в проделанной работе, знать:

Цели и задачи расщепления и слияния моделей.

Методологию расщепления и слияния моделей.

Правила модификации диаграммы путем расщепления и слияния моделей.

Студент должен уметь изменить модель (добавить новые или удалить элементы) по требованию преподавателя.

Работа с литературой:

Рекомендуемые источники информации (№ источника)			
Основная	Дополнительная	Методическая	Интернет-ресурсы
1-2	1-3	1-3	1-4

Оценочные средства: отчет к лабораторной работе (См.: Фонд оценочных средств)

Лабораторная работа №6

«Дополнение моделей процессов диаграммой

DFD.» Форма проведения: лабораторная работа

Цель работы: построить диаграмму декомпозиции в нотации DFD одной из работ диаграмм IDEF0, построенных в предыдущих лабораторных работах

Диаграммы потоков данных (Data flow diagram, DFD) используются для описания документооборота и обработки информации. Подобно IDEF0, DFD представляет

моделируемую 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. Главная цель DFD - показать, как каждая работа преобразует свои входные данные в выходные, а также выявить отношения между этими работами.

Любая DFD-диаграмма может содержать работы, внешние сущности, стрелки (потoki данных) и хранилища данных.

Работы. Работы изображаются прямоугольниками с закругленными углами (рис. 1), смысл их совпадает со смыслом работ IDEF0 и IDEF3. Так же как работы IDEF3, они имеют входы и выходы, но не поддерживают управления и механизмы, как IDEF0. Все стороны работы равнозначны. В каждую работу может входить и выходить по несколько стрелок.



Рисунок 1. Работа в DFD

Внешние сущности. Внешние сущности изображают входы в систему и/или выходы из нее. Одна внешняя сущность может одновременно предоставлять входы (функционируя как поставщик) и принимать выходы (функционируя как получатель). Внешняя сущность представляет собой материальный объект, 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. Внешние сущности изображаются в виде прямоугольника с тенью и обычно располагаются по краям диаграммы (рис. 2).



Рисунок 2. Внешняя сущность в DFD

Стрелки (потoki данных). Стрелки описывают движение объектов из одной части системы в другую (отсюда следует, что диаграмма DFD не может иметь граничных стрелок). Поскольку все стороны работы в DFD равнозначны, стрелки могут начинаться и заканчиваться на любой стороне прямоугольника. Стрелки могут быть двунаправлены.

Хранилище данных. В отличие от стрелок, описывающих объекты в движении, хранилища данных изображают объекты в покое (рис. 3). Хранилище данных - это 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. Оно в общем случае является прообразом будущей базы данных, и описание хранящихся в нем данных должно соответствовать информационной модели (Entity-Relationship Diagram).



Рисунок 3. Хранилище данных в DFD

Декомпозиция работы IDEF0 в диаграмму DFD. При декомпозиции работы IDEF0 в DFD необходимо выполнить следующие действия:

- удалить все граничные стрелки на диаграмме DFD;
- создать соответствующие внешние сущности и хранилища данных;
- создать внутренние стрелки, начинающиеся с внешних сущностей вместо граничных стрелок;
- стрелки на диаграмме IDEF0 затоннелировать

Строго придерживаться правил нотации DFD не всегда удобно, поэтому BPWin позволяет создавать в DFD диаграммах граничные стрелки.

Построение диаграммы декомпозиции. Проведем декомпозицию работы *Отгрузка и снабжение* диаграммы A0 "Деятельность предприятия по сборке и продаже компьютеров и ноутбуков". В этой работе мы выделили следующие дочерние работы:

- снабжение необходимыми комплектующими - занимается действиями, связанными с поиском подходящих поставщиков и заказом у них необходимых комплектующих
- хранение комплектующих и собранных компьютеров
- отгрузка готовой продукции - все действия, связанные с упаковкой, оформлением документации и собственно отгрузкой готовой продукции

Выделим работу *Отгрузка и снабжение* диаграммы A0 "Деятельность предприятия по сборке и продаже компьютеров и ноутбуков", нажмем на кнопку "Go to Child Diagram" панели инструментов и выберем нотацию DFD. При создании дочерней диаграммы BPWin переносит граничные стрелки родительской работы, их необходимо удалить и заменить на внешние сущности. Стрелки механизмов, стрелки управления "Правила и процедуры", "Управляющая информация" и стрелку выхода "Отчеты" на дочерней диаграмме задействованы не будут, чтоб не загромождать диаграмму менее существенными деталями. Остальные стрелки заменим на внешние сущности - кнопка "External Reference Tool" на панели инструментов, в появившемся окне выбрать переключатель "Arrow" и выбрать из списка нужное название (рис. 4):

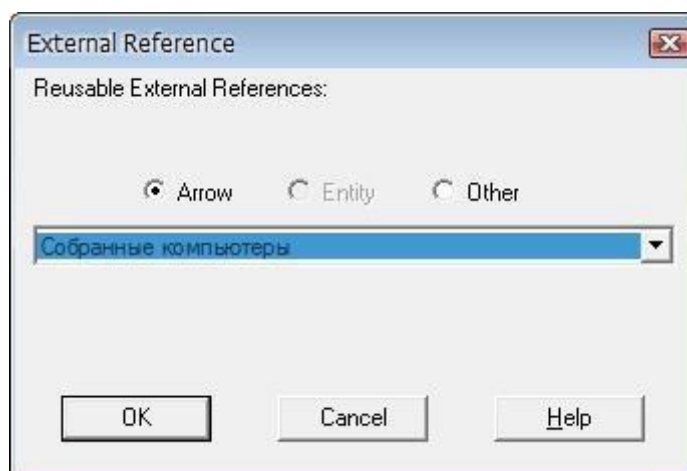


Рисунок 4. Добавление внешней сущности

Далее разместим дочерние работы, свяжем их со внешними сущностями и между собой (рис. 5):

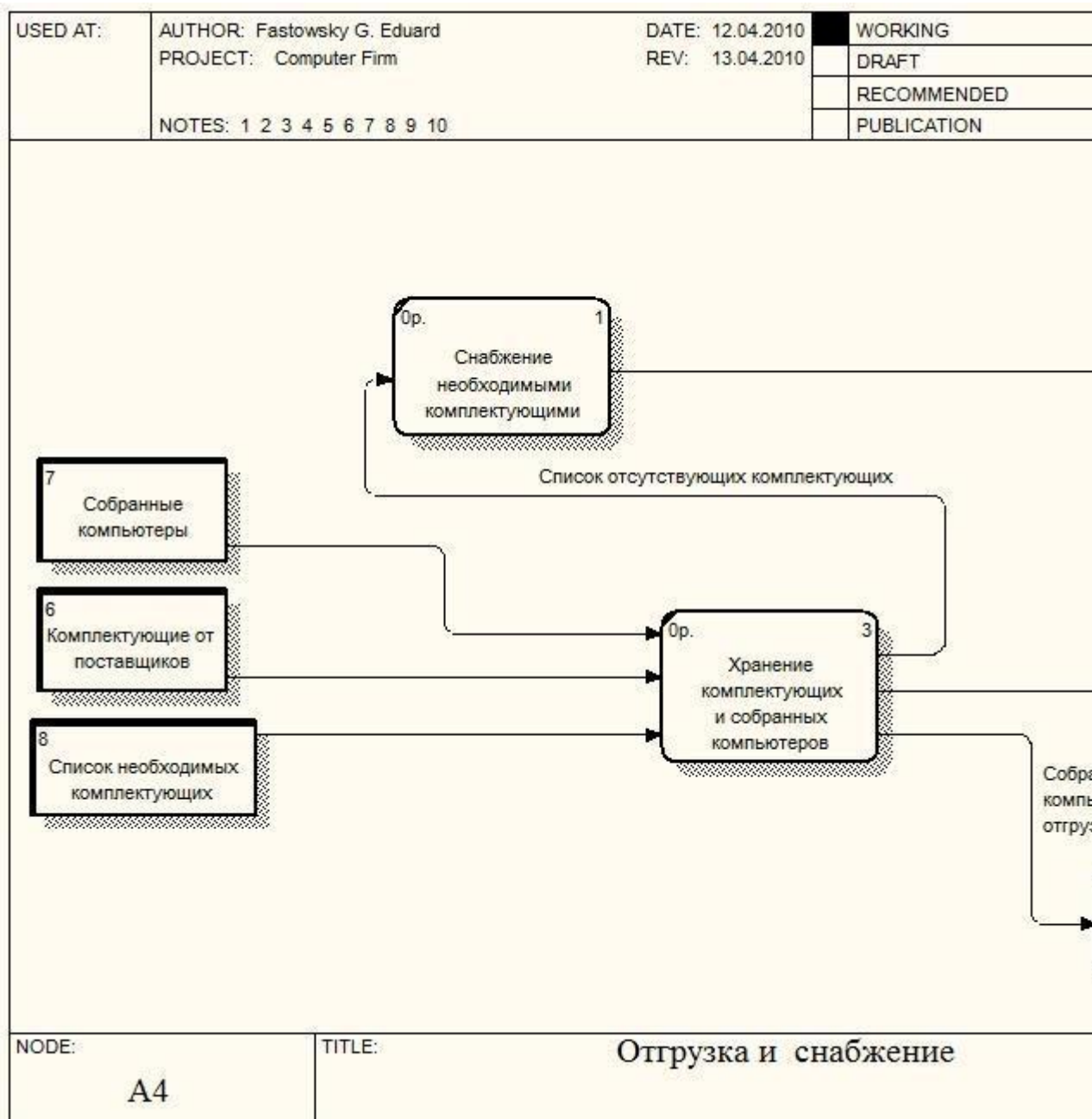


Рисунок 5. Работы и внешние сущности

Центральной здесь является работа "Хранение комплектующих и собранных компьютеров". На ее вход поступают собранные компьютеры и полученные от поставщиков комплектующие, а также список необходимых для сборки компьютеров комплектующих. Выходом этой работы будут необходимые комплектующие (если они есть в наличии), список отсутствующих комплектующих, передаваемый на вход работы "Снабжение необходимыми комплектующими" и собранные компьютеры, передаваемые на отгрузку. Выходами работ "Снабжение необходимыми комплектующими" и "Отгрузка готовой продукции" будут, соответственно, заказы поставщикам и готовая продукция.

Следующим шагом необходимо определить, какая информация необходима для каждой работы, т.е. необходимо разместить на диаграмме хранилища данных (рис. 6).

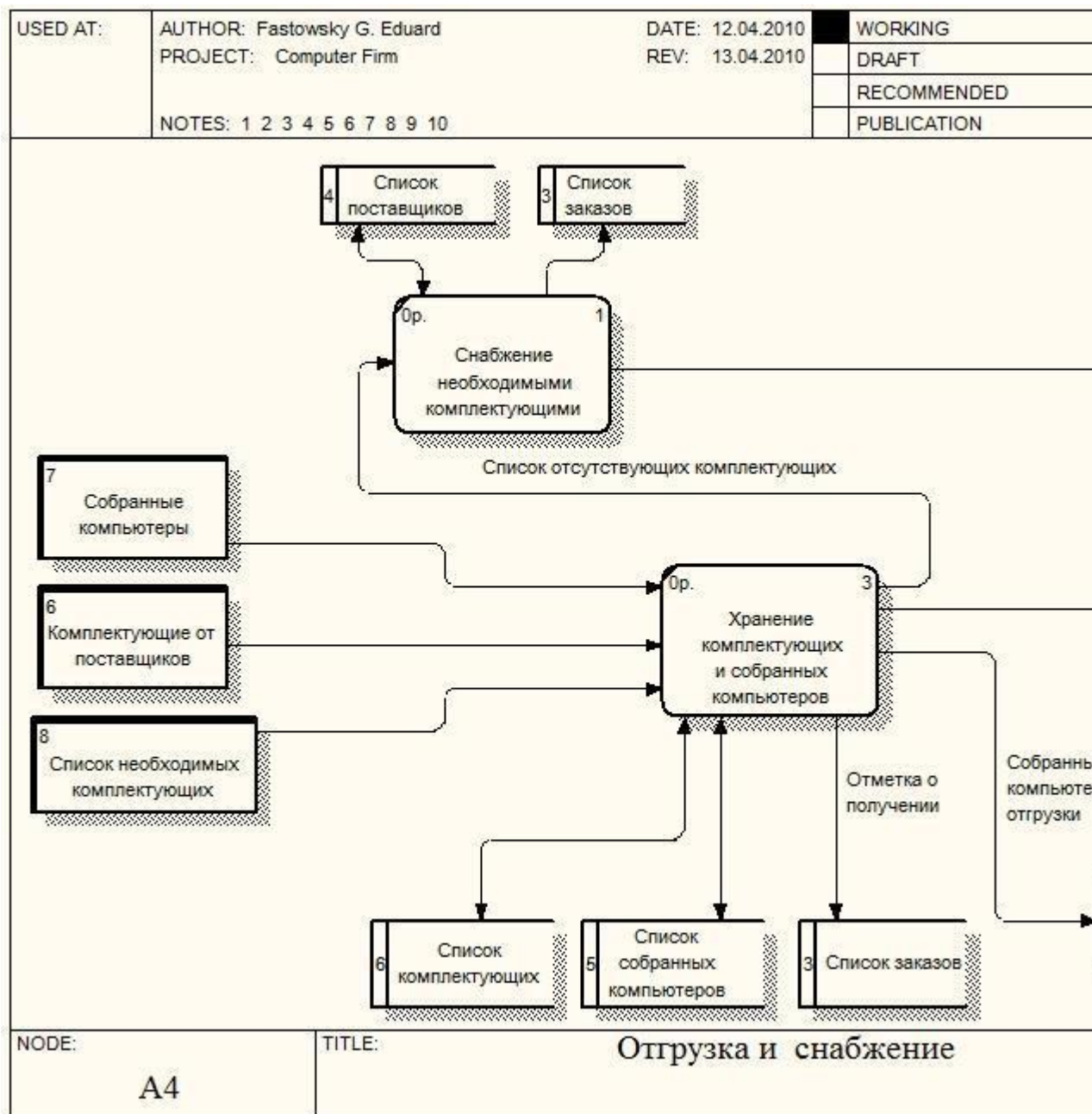


Рисунок 6. Итоговая диаграмма декомпозиции

Работа "Снабжение необходимыми комплектующими" работает с информацией о поставщиках и с информацией о заказах, сделанных у этих поставщиков. Стрелка, соединяющая работу и хранилище данных "Список поставщиков" двунаправленная, т.к. работа может как получать информацию о имеющихся поставщиках, так и вносить данные о новых поставщиках. Стрелка, соединяющая работу с хранилищем данных "Список заказов" однонаправленная, т.к. работа только вносит информацию о сделанных заказах.

Работа "Хранение комплектующих и собранных компьютеров" работает с информацией о получаемых и выдаваемых комплектующих и собранных компьютерах, поэтому стрелки, соединяющая работу с хранилищами данных "Список комплектующих" и "Список собранных компьютеров" двунаправленные. Также эта работа при получении комплектующих должна делать отметку о том, что заказ поставщикам выполнен. Для

этого она связана с хранилищем данных "Список заказов" однонаправленной стрелкой. Обратите внимание, что на DFD диаграммах одно и тоже хранилище данных может дублироваться.

Наконец, работа "Отгрузка готовой продукции" должна хранить информацию по выполненным отгрузкам. Для этого вводится соответствующее хранилище данных - "Данные по отгрузке".

Последним действием необходимо стрелки родительской работы затуннелировать (рис. 7):

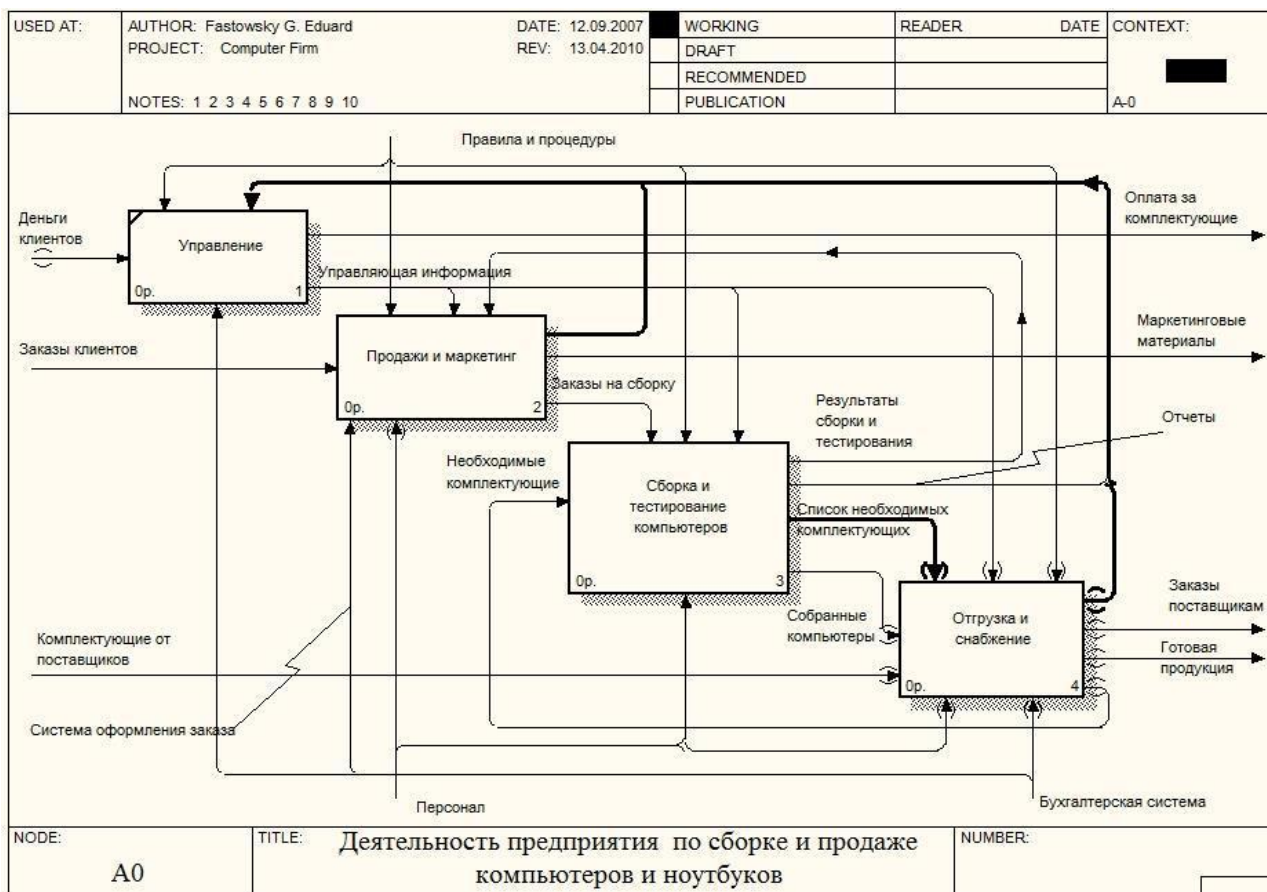


Рисунок 7. Диаграмма IDEF0 с затуннелированными стрелками работы "Отгрузка и снабжение"

Содержание отчета:

- краткое описание декомпозируемой работы
- диаграмма декомпозиции

Контрольные вопросы

При защите лабораторной работы студент должен ориентироваться в проделанной работе, знать:

Цели и задачи декомпозиции работы IDEF0 в диаграмму DFD.

Методологию декомпозиции работы IDEF0 в диаграмму DFD.

Правила модификации диаграммы DFD.

Студент должен уметь изменить модель (добавить новые или удалить элементы) по требованию преподавателя.

Работа с литературой:

Рекомендуемые источники информации (№ источника)			
Основная	Дополнительная	Методическая	Интернет-ресурсы
1-2	1-3	1-3	1-4

Оценочные средства: отчет к лабораторной работе (См.: Фонд оценочных средств)

Лабораторная работа №7**«Создание диаграммы IDEF3. Создание**

сценария.» Форма проведения: лабораторная работа

Цель работы:

Освоение принципов создания диаграммы IDEF3. Приобретение практических навыков в создании диаграммы IDEF3. Освоение принципов создания сценария в IDEF3. Приобретение практических навыков в создании сценария в IDEF3.

1) Перейдите на диаграмму A2 и декомпозируйте работу «Сборка настольных компьютеров». В диалоге Activity Box Count (рис. 1) установите число работ 4 и нотацию IDEF3.

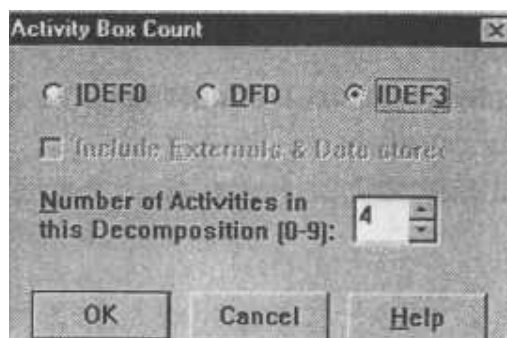


Рис. 1. Выбор нотации IDEF3 в диалоге Activity Box Count

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

2) Во вкладке UOW внесите свойства работы (табл.

1). Таблица 1. Свойства UOW

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

3) Внесите в диаграмму еще 3 работы (кнопка ). Внесите имена работ:

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

4) С помощью кнопки  – палитры инструментов создайте объект ссылки. Внесите имя объекта внешней ссылки "Компоненты". Свяжите стрелкой объект ссылки и работу "Подготовка компонентов".

5) Свяжите стрелкой работы "Подготовка компонентов" (выход) и "Установка материнской платы и винчестера". Измените стиль стрелки на Object Flow.

В IDEF3 имя стрелки может отсутствовать, хотя BPwin показывает отсутствие имени как ошибку. Результат показан на рис. 2.



Рис. 2. Результат создания UOW и объекта ссылки

6) С помощью кнопки  на палитре инструментов внесите два перекрестка типа "асинхронное или" и свяжите работы с перекрестками, как показано на рис. 3.



Рис. 3. Диаграмма IDEF3 после создания перекрестков

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

Создайте два перекрестка типа исключающего "ИЛИ" и свяжите работы, как показано на рис. 4.

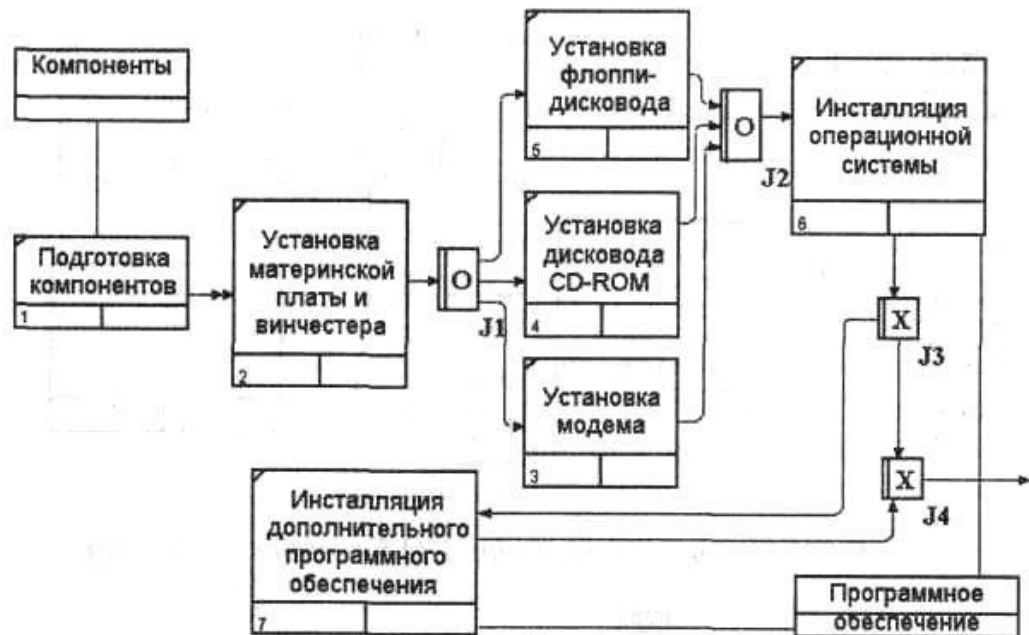


Рис. 4. Результат создания диаграммы IDEF3

Создание сценария

- 1) Выберите пункт меню Diagram/Add IDEF3 Scenario. Создайте диаграмму сценария на основе диаграммы IDEF3 "Сборка настольных компьютеров" (A22.1).
- 2) Удалите элементы, не входящие в сценарий (рис. 5).



Рис. 5. Результат создания сценария

Контрольные вопросы

При защите лабораторной работы студент должен ориентироваться в проделанной работе, зная:

- Цели и задачи создания диаграмм IDEF3.
- Методологию создания диаграмм IDEF3.
- Правила создания перекрестков.

Цели и задачи создания сценария в IDEF3.

Методологию создания сценария в IDEF3.

Правила создания сценария в IDEF3.

Студент должен уметь изменить модель (добавить новые или удалить элементы) по требованию преподавателя.

Работа с литературой:

Рекомендуемые источники информации (№ источника)			
Основная	Дополнительная	Методическая	Интернет-ресурсы
1-2	1-3	1-3	1-4

Оценочные средства: отчет к лабораторной работе (См.: Фонд оценочных средств)

Лабораторная работа №8

«Создание модели "Питание семьи"»

Форма проведения: лабораторная работа

Цель работы:

Изучить и освоить основные принципы структурного анализа и моделирования реальных систем SADT. Составить описание того, как реализовать питание семьи, состоящей из двоих взрослых, двоих маленьких детей и двоих подростков, так чтобы подростки могли понять это описание и следовать ему без посторонней помощи.

Этап 1. Очерчивание границ объекта

Цель: Создаём очерченный контекст модели "Питание семьи".

Действия:

Начинаем составлять список всех основных предметов, которые, являются частью системы. На этом этапе мы не беспокоимся о точности.

Теперь оценим исходный список критически. Вычеркнем названия, не относящихся к системе объектов. Объединяем названия в группы. Добавляем новые названия по необходимости. Групп должно быть не более четырех: входные данные; ресурсы; механизмы управления; выходные данные.

Продельваем то же самое для функций системы. Для перечисления функций пользуемся списком данных, затем оцениваем новый список. Вычеркиваем те названия, которые не входят в систему. Группируем сходные функции, соединяя их названия линиями. Меняем список данных по мере постижения работы системы.

Остановились, так как перечислено достаточное для создания диаграммы число объектов и функций.

ИСПОЛЬЗУЕТСЯ В:	АВТОР: Марса	ДАТА: 01/16/93	☒ РАБОЧАЯ ВЕРСИЯ	ЧИТАТЕЛЬ	ДАТА	КОНТЕКСТ:
	ПРОЕКТ: Питание семьи	ПЕРЕСМОТР:	☐ ЭСКИЗ			
	ЗАМЕЧАНИЯ: 1 2 3 4 5 6 7 8 9 10		☐ РЕКОМЕНДОВАНО			
			☐ ПУБЛИКАЦИЯ			

<u>Список данных</u> Расположение ресторанов Расположение магазинов Меню Рецепты Список покупок Бюджет Деньги Критерии съедобности Вкусы Диетические требования Режим дня Правила предосторожности Безопасность Приготовленная пища Наличие продуктов Грязная посуда Чистая посуда Время года		<u>Список функций</u> Спланировать меню Купить продукты Приготовить Стряпать Пробовать Съесть Убрать со стола Убрать в кухне Составить режим дня Вычислить энергетический баланс Обслужить за столом Сохранить остатки Накрыть на стол Убрать покупки Распределить запасы Есть	<u>Подросток</u> • компетентный • терпеливый • зрелый • заинтересованный • желающий помочь • дисциплинированный
--	--	---	---

УЗЕЛ:	НАЗВАНИЕ: Организовать питание семьи (Контекст)	НОМЕР: DAM001
-------	---	---------------

Рис. 1. Организовать питание семьи (контекст)

Этап 2. Определение цели и точки зрения модели

Цель: Сформулировать цель модели "Питание семьи" и определить, с чьей точки зрения будет описан этот процесс.

Действия:

Составляем множество вопросов, на которые должна отвечать модель. Уточняем это множество, определив, кто задает вопросы. Записываем 5-10 вопросов. Затем задаем степень точности ответа на каждый из них.

С помощью этого набора вопросов определяем, как будет использоваться модель. В одном предложении формулируем, как будет использоваться модель. Это и есть цель модели (см. рис.2: Цель:).

Теперь решаем, кто рассказывает о "Питании семьи". Составляем список кандидатов, чтобы убедиться, что мы выбрали самую подходящую точку зрения. Выбираем из всех того, кто сможет ответить на большинство вопросов. Эта точка зрения становится точкой зрения модели.

ИСПОЛЬЗУЕТСЯ В:	АВТОР: Марса ПРОЕКТ: Питание семьи ЗАМЕЧАНИЯ: ☒ 2 3 4 5 6 7 8 9 10	ДАТА: 01/16/93 ПЕРЕСМОТР:	☒ РАБОЧАЯ ВЕРСИЯ ЭСКИЗ РЕКОМЕНДОВАНО ПУБЛИКАЦИЯ	ЧИТАТЕЛЬ	ДАТА	КОНТЕКСТ:
-----------------	--	------------------------------	--	----------	------	-----------

Вопросы:

- Как я узнаю, какие продукты покупать?
- Что я делаю сначала?
- Как я готовлю кухню к стряпне?
- Как обычно накрывают на стол?
- Что я делаю с остатками?

~ ~ ~

Цель: Понять действия, необходимые для организации питания семьи, когда родители в отъезде.

Позиции:

Дети
 Подростки

Родители

 Соседи
 Продавцы

~ ~ ~

Точка зрения: Родителей

1

 Нужен авторитетный специалист!

УЗЕЛ:	НАЗВАНИЕ: Организовать питание семьи. (Цель и точка зрения)	НОМЕР: DAM002
-------	---	---------------

Рис. 2. Организовать питание семьи (цель и точка зрения)

Этап 3. Построение контекстной диаграммы

Цель: Нарисовать единственный блок с его входами, управлениями и выходами, который обобщает всю модель.

Действия:

Рисуем единственный большой блок и помечаем его названием основной функции системы, например: «Организовать питание семьи». Таким образом, мы обобщаем всю систему.

Рисуем и помечаем все входные стрелки, стрелки управления, ресурсов и выходные стрелки.

Пишем под большим блоком цель и точку зрения модели. Это сразу же определяет смысл и направленность модели.

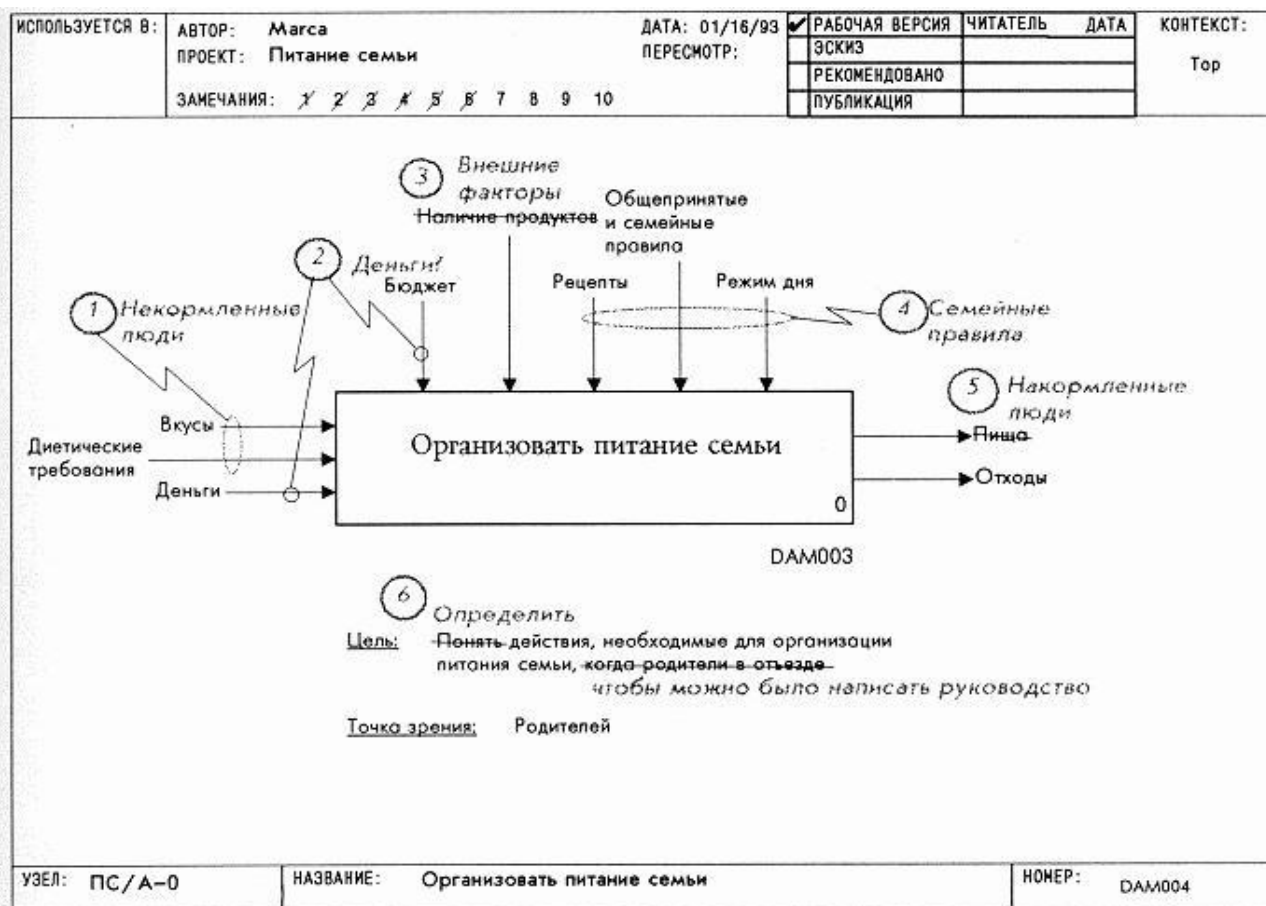


Рис. 3. Контекстная диаграмма

Этап 4. Построение диаграммы А0

Цель: Нарисовать блоки, соответствующие перечисленным функциям системы.

Действия:

Производим декомпозицию контекстной диаграммы. Число создаваемых блоков должно совпадать с числом функций, описывающих систему.

В иерархическом порядке подписываем блоки теми функциями, которым они соответствуют.

Соединяем все блоки между собой. Необходимо создать хотя бы одну стрелку, ведущую от блока верхнего уровня к блоку нижнего уровня. Возможно соединение блоков верхнего уровня с несколькими блоками нижнего. При необходимости создать стрелки обратной связи по тем же правилам.

Все созданные на контекстной диаграмме входные и выходные стрелки переносятся на данную диаграмму. Необходимо связать все стрелки контекстной диаграммы с некоторыми блоками. При этом стрелки могут разделяться и соединяться с несколькими блоками диаграммы верхнего уровня.

Каждая изображенная стрелка на диаграмме соответствует некоторому элементу данных (предмету), описывающему систему.

После всех действий производим обобщение диаграммы. Для этого подписываем все вышедшие из контекстной диаграммы стрелки и убеждаемся в том, что их количество достаточно для описания системы. Если число стрелок больше или меньше, чем приходит с контекстной диаграммы, значит, были созданы или удалены некоторые элементы системы. Необходимо проверить наличие туннельных стрелок на обеих диаграммах.

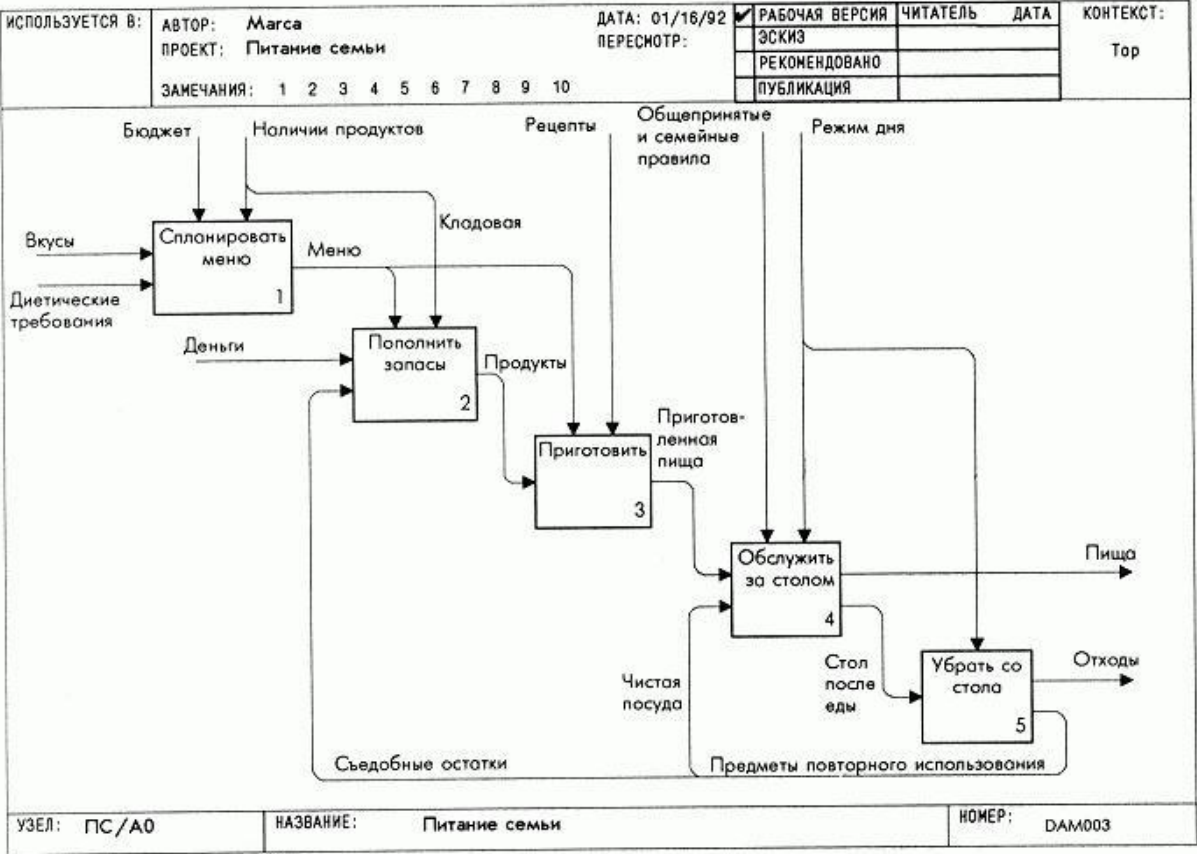


Рис. 4. Диаграмма А0

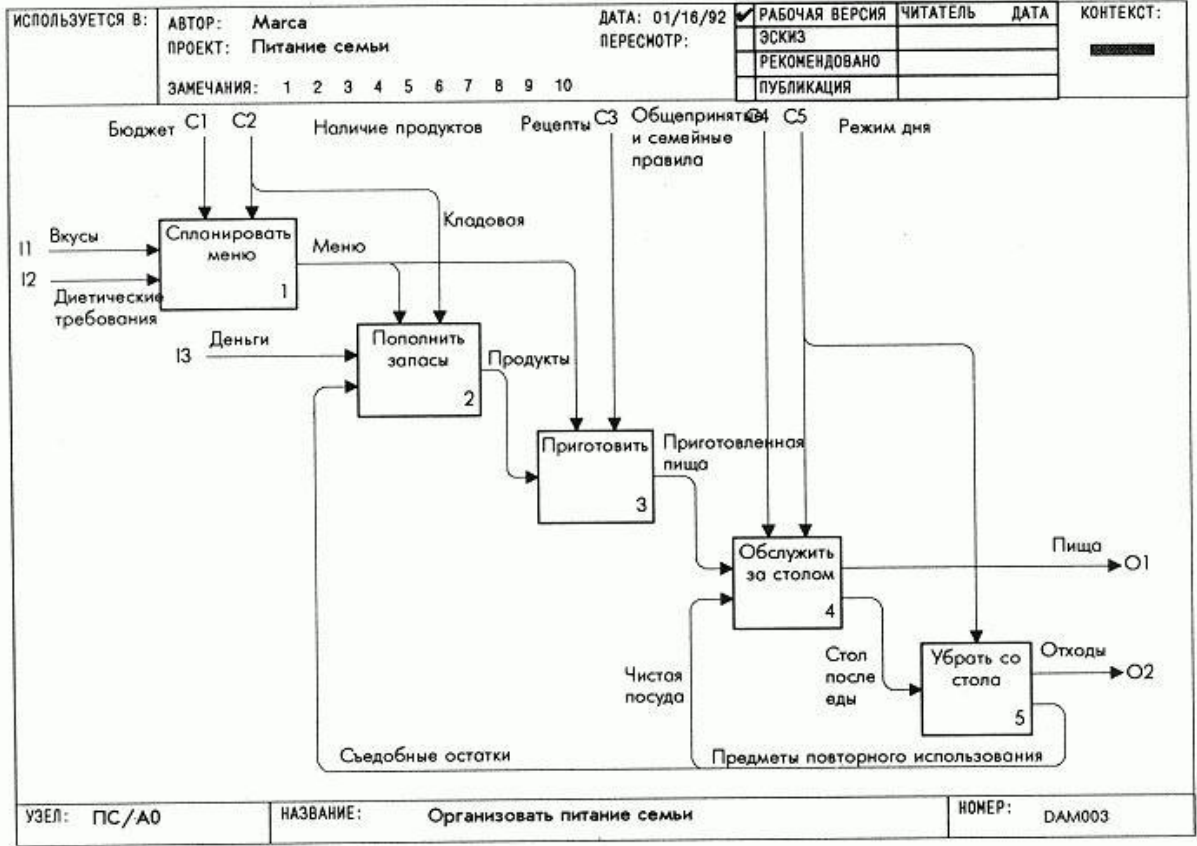


Рис. 5. Обобщение диаграммы А0

Этап 5. Критическая оценка контекстной диаграммы

Цель: Документировать все вопросы, возникшие при составлении контекстной диаграммы и исправить ее.

Действия:

Рассматриваем контекстную диаграмму. Производим обобщение всех предметов диаграммы. Определяем неувязки, записываем, в чем они заключаются, и проводим соответствующую переделку (меняем метки и объединяем стрелки).

Оценив чертеж, оцениваем также цель и точку зрения. Записываем неувязки и пересматриваем цель и точку зрения.

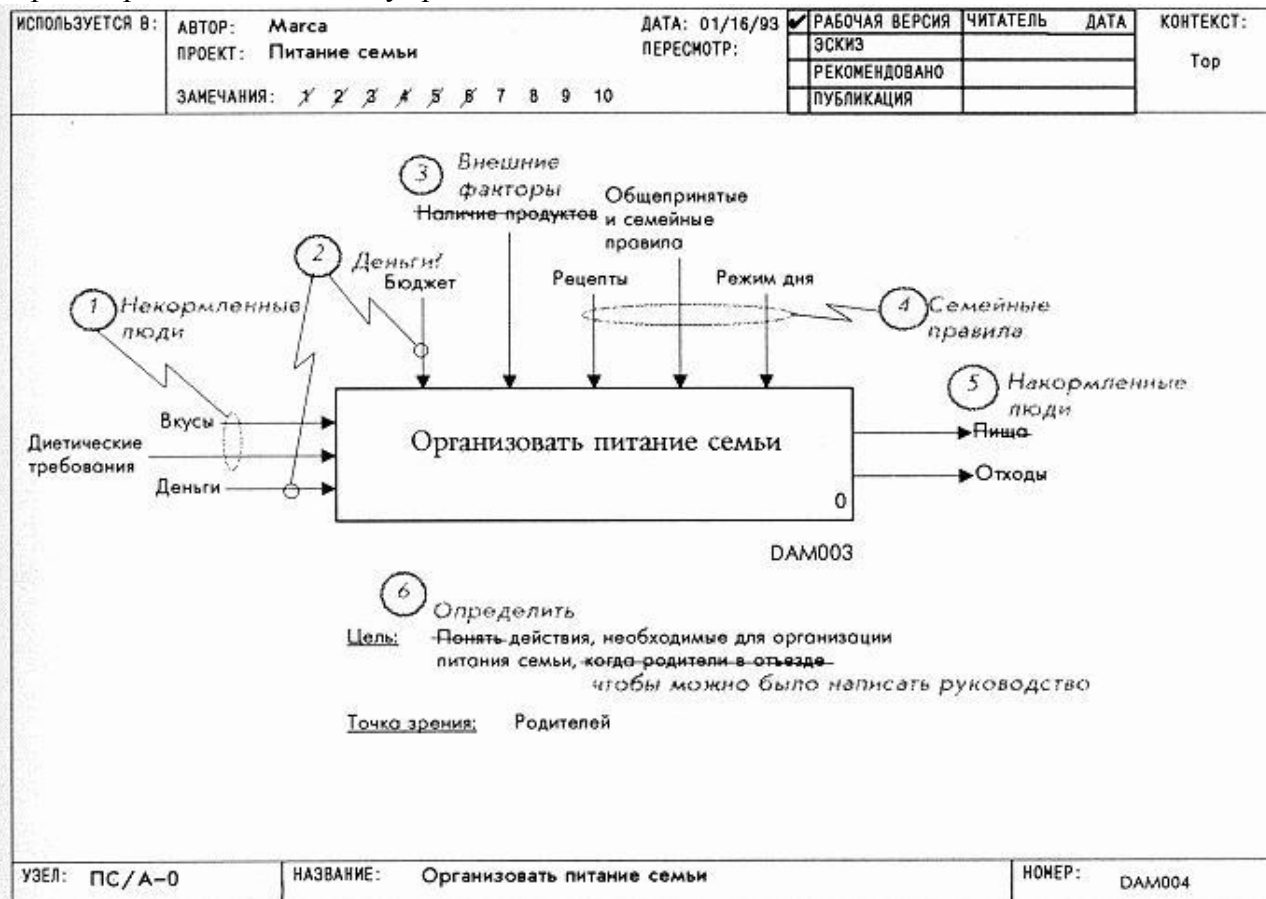


Рис. 6. Критическая оценка контекстной диаграммы

Этап 6. Критическая оценка диаграммы А0

Цель: Документировать все вопросы, возникшие с диаграммой А0 и исправить ее.

Действия:

Вносим в эту диаграмму все исправления, соответствующие исправлениям на контекстной диаграмме А-0. Если на стрелке управления диаграммы А-0 изменилась метка, то меняем соответствующую внешнюю стрелку на данной диаграмме.

Читаем диаграмму А0 вслух, пользуясь для изложения шаблоном типа: "функция (имя блока) преобразует (имена входных стрелок) в (имена выходных стрелок) в соответствии с (имена стрелок управления) при наличии (имена стрелок ресурсов).

Определяем смысл данной диаграммы после исправления всех связанных с диаграммой А-0 недостатков. Оцениваем его адекватность. Определяем недостатки нового варианта, записываем их и вносим соответствующие изменения (изменяем метки, объединяем стрелки).

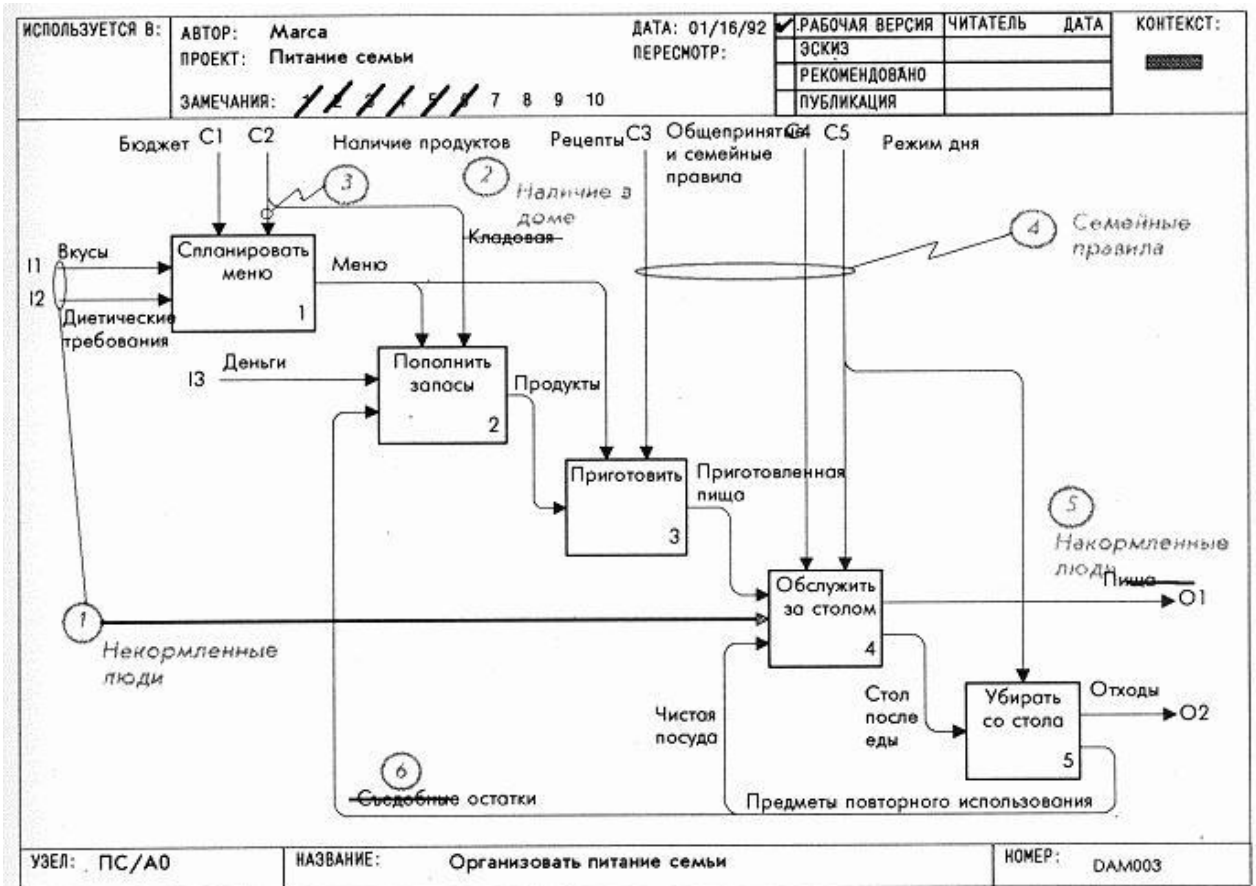


Рис. 7. Критическая оценка диаграммы А0

Этап 7. Переделка контекстной диаграммы и диаграммы А0

Цель: Переделать в соответствии с критической оценкой, выполненной на этапе 6, и начертить заново диаграммы А-0 и А0.

Действия:

Перечерчиваем контекстную диаграмму А-0. Обдумываем изложенное в диаграмме и проверяем, сохранился ли в ней смысл. Переписываем, если нужно, цель и точку зрения модели.

Перечерчивая диаграмму А0, обдумываем изложенное в ней. Обращаемся к диаграмме А-0, чтобы удостовериться, что детали диаграммы А0 согласованы с ее контекстом.

Проверяем наличие туннельных стрелок. Проверяем соответствие меток внешних стрелок диаграммы А0 меткам стрелок диаграммы А-0.

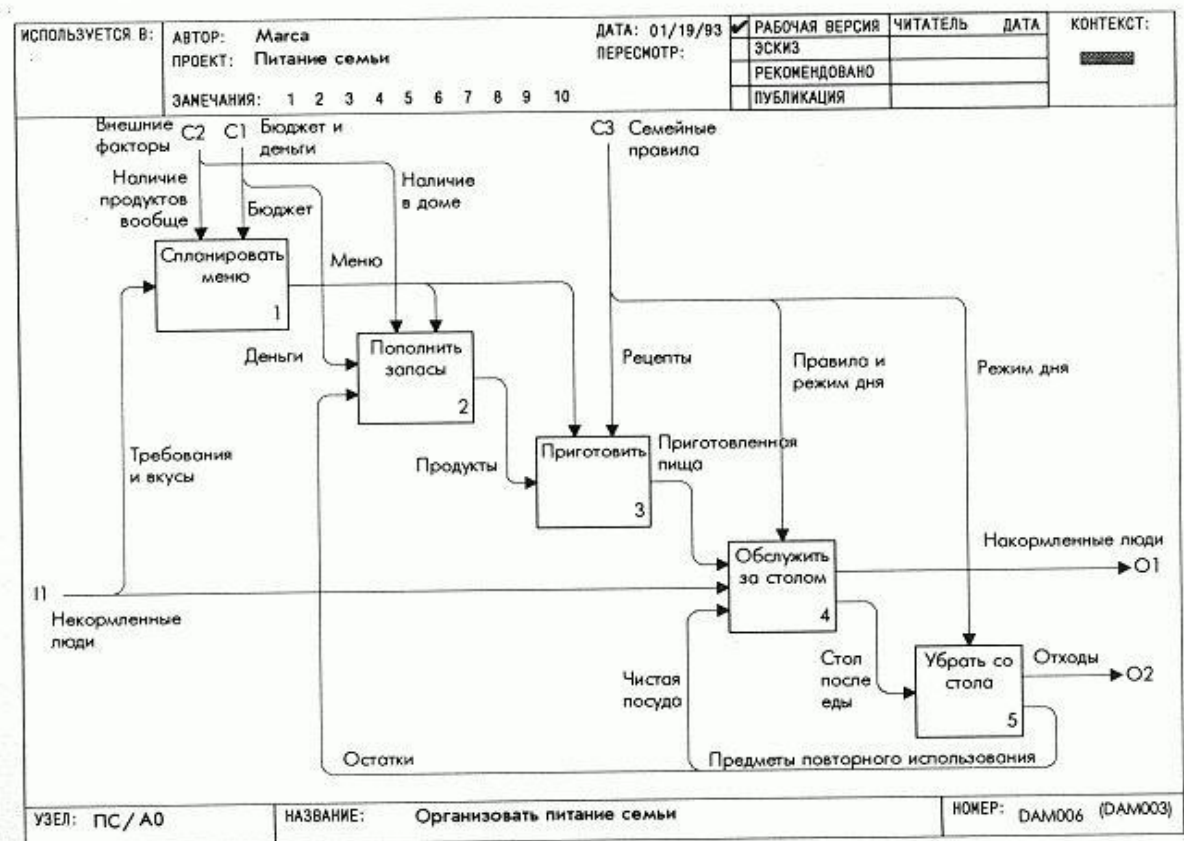
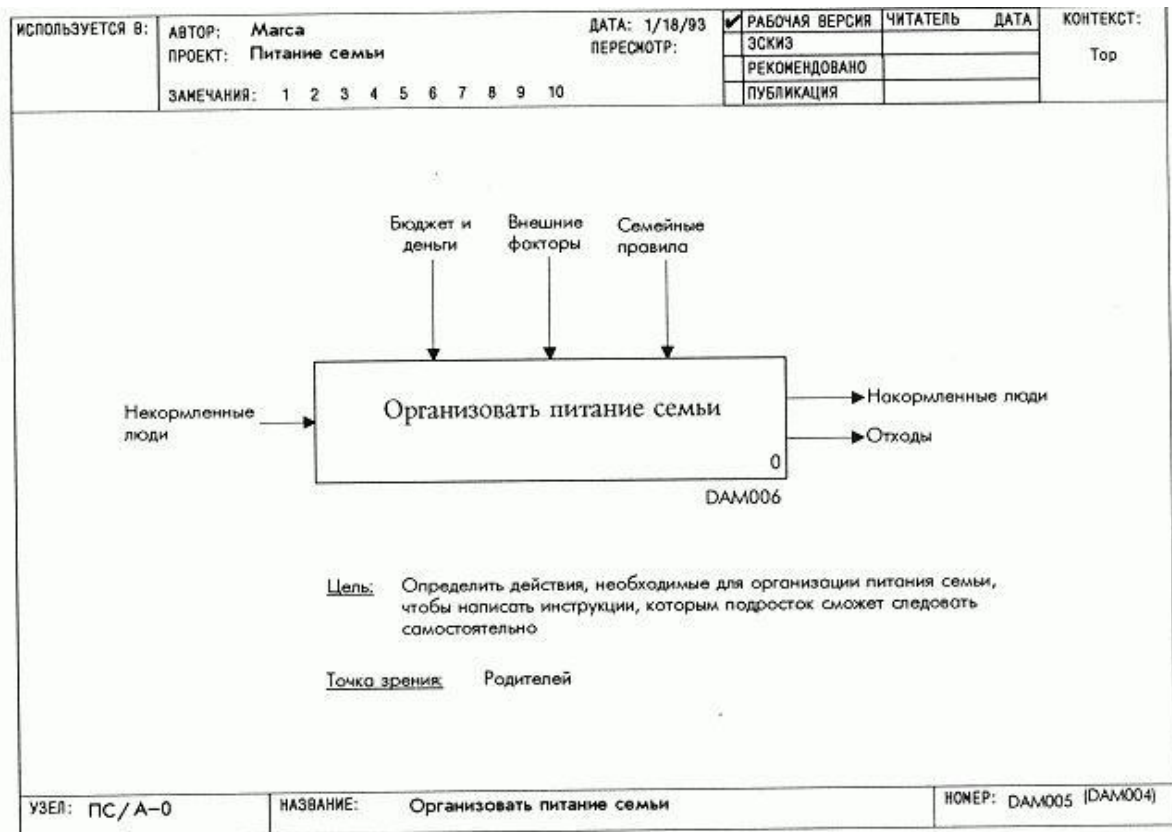


Рис.8. Переделка контекстной диаграммы и диаграммы А0

Контрольные вопросы

При защите лабораторной работы студент должен ориентироваться в проделанной работе, знать:

Цели и задачи создания модели.

Методологию создания модели.

Правила создания модели.

Студент должен уметь изменить модель (добавить новые или удалить элементы) по требованию преподавателя.

Работа с литературой:

Рекомендуемые источники информации (№ источника)			
Основная	Дополнительная	Методическая	Интернет-ресурсы
1-2	1-3	1-3	1-4

Оценочные средства: отчет к лабораторной работе (См.: Фонд оценочных средств)

Лабораторная работа №9

«Стоимостный анализ модели(ABC)»

Форма проведения: лабораторная работа

Цель работы:

Освоение принципов оценки модели. Приобретение практических навыков в оценке модели.

Для оценки модели BPwin предоставляет разработчику такой инструмент как стоимостный анализ, основанный на работах (Activity Based Costing, ABC). ABC является широко распространенной методикой, используемой международными корпорациями и государственными организациями (в т.ч. Департаментом обороны США) для идентификации истинных движителей затрат в организации. Стоимостный анализ представляет собой соглашение об учете, используемое для сбора затрат, связанных с работами, с целью определить общую стоимость процесса. ABC применяется для того, чтобы понять происхождение выходных затрат и облегчить выбор нужной модели работ при реорганизации деятельности предприятия. ABC может проводиться только тогда, когда модель работы последовательная (следует синтаксическим правилам IDEF0), корректная (отражает бизнес), полная (охватывает всю рассматриваемую область) и стабильная (проходит цикл экспертизы без изменений), другими словами, создание модели работы закончено.

При проведении стоимостного анализа сначала задаются единицы измерения времени и денег. Для этого в меню **Model** следует выбрать режим **Model Properties**. В открывшемся диалоговом окне **Model Properties** следует на вкладке **ABC Units** (см. рис. 13) установить денежную единицу (например, рубль) и временную единицу (например, час). Для настройки отображения на экране стоимости работ следует на вкладке **Display** для блока **ABC units** установить переключатель **Cost** (см. рис. 14).

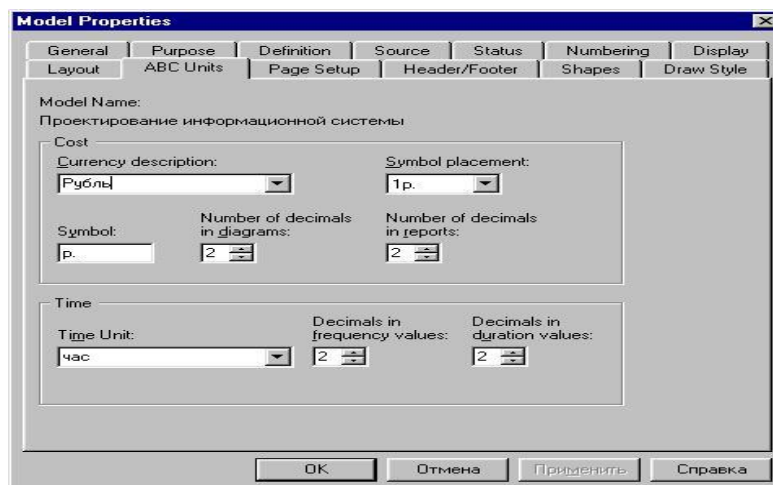


Рис. 13. Настройка единиц измерения валюты и времени.

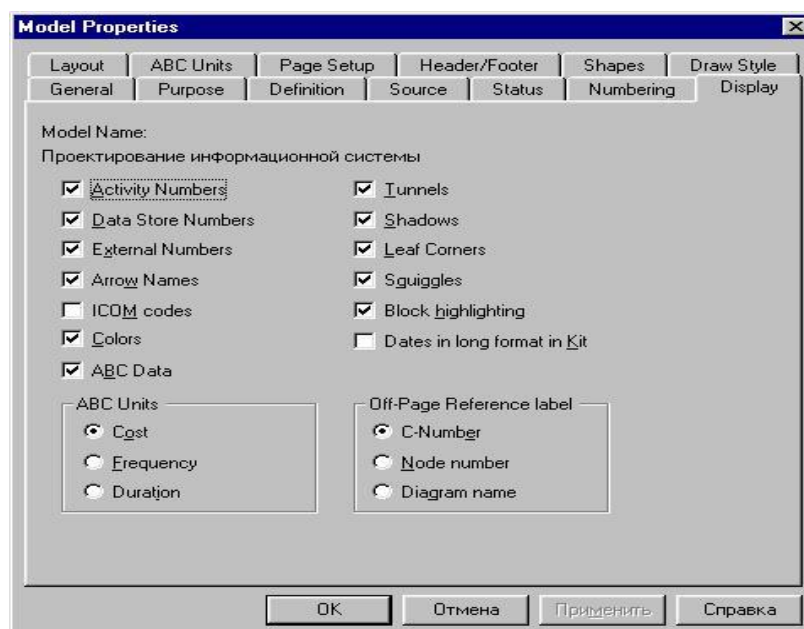


Рис. 14. Настройка отображения стоимости работ.

После этого следует ввести названия центров затрат. Для этого в меню **Dictionary** выбирают режим **Cost Center**. Пример названий центров затрат представлен в таблице 3.

Таблица 3.

Центры затрат (ABC)

Центры затрат	Определение
Заработная плата	Расходы по заработной плате в день
Оборудование	Затраты на амортизацию оборудования в день
Расходные материалы	Стоимость расходных материалов (канцтовары)

Каждой работе в разрезе центров затрат необходимо задать стоимость, для чего следует щелкнуть два раза «мышью» на такой работе. В открывшемся окне **Activity Properties** на вкладке **Costs** устанавливается переключатель Override decompositions (см. рис. 15), устанавливается частота (Frequency) и длительность (Duration) и задается стоимость по центрам затрат, автоматически суммирующимся в стоимость работы. Для редактирования названий центров затрат используется кнопка Cost Center Editor.

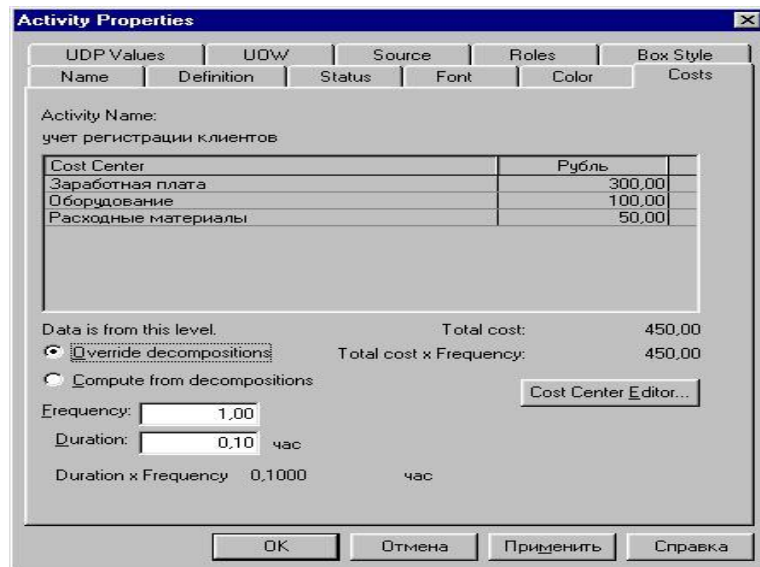


Рис. 15. Диалоговое окно для задания стоимости работ.

1. Выполните команду Model – Model Properties, перейдите на закладку ABC Units и установите единицы измерения денег и времени – белорусские рубли и дни соответственно (рис. 1.16).
2. Перейдите в словарь центров затрат (команда меню Dictionary – - Cost Center) и внесите названия и описания центров затрат согласно таблице 1.5.
3. Для того чтобы стоимость работ отображалась на диаграммах, выполните команду меню Model – Model Properties, перейдите на закладку Display, установите опцию ABC Data и выберите отображение затрат (Cost) (рис. 1.17).
4. Для назначения стоимости работе следует щелкнуть по ней правой кнопкой мыши и выбрать в контекстном меню пункт Cost. В появившемся диалоге необходимо заполнить затраты данной работы по каждому центру затрат, указать ее частоту (Frequency) и длительность (Duration) (рис. 1.18).
5. Заполните затраты по работам на диаграмме Сборка и тестирование компьютеров согласно таблице 1.6.
6. Убедитесь в том, что на диаграмме более высокого уровня была рассчитана стоимость работы Сборка и тестирование компьютеров (рис. 1.19).

Рисунок 1.16 – Настройка единиц измерения ABC

Таблица 1.5 – Центры затрат ABC

Название	Описание
Управление	Затраты на управление, связанные с составлением графика работ, формированием партий компьютеров, контролем над сборкой и тестированием
Рабочая сила	Затраты на оплату рабочих, занятых сборкой и тестированием компьютеров
Компоненты	Затраты на закупку компонентов

7. Сгенерируйте отчет о стоимостных затратах на работу Сборка и тестирование компьютеров. Для этого: ♣ Выполните команду меню Tools – Reports – Activity Cost Report. ♣ Заполните диалог так, как показано на рис. 1.20 и нажмите кнопку Preview.

Рисунок 1.17 – Настройка отображения данных ABC

Рисунок 1.18 – Заполнение затрат по работе

Таблица 1.6 – Стоимость работ на диаграмме A2

Работа	Центр затрат	Стоимость	Длительность	Частота
Отслеживание расписания и управление сборкой и тестированием	Управление	40 000	1,00	1
Сборка настольных компьютеров	Рабочая сила Компоненты	1 200 000 10 000	1,00	12
Сборка ноутбуков	Рабочая сила Компоненты	2 000 000 10 000	1,00	20
Тестирование компьютеров	Рабочая сила	5000	1,00	32



Рисунок 1.19 – Отображение свойств ABC

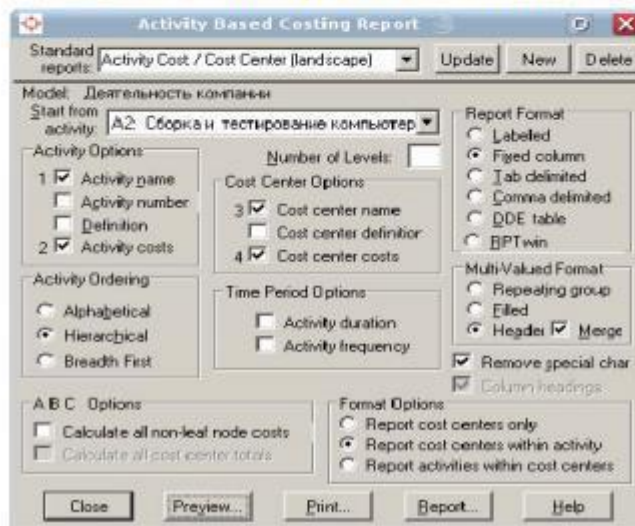


Рисунок 1.20 – Генерация отчета ABC

Контрольные вопросы

При защите лабораторной работы студент должен ориентироваться в проделанной работе, знать:

Цели и задачи стоимостного анализа модели.

Методологию стоимостного анализа модели.

Правила стоимостного анализа модели.

Студент должен уметь изменить модель (добавить новые или удалить элементы) по требованию преподавателя.

Работа с литературой:

Рекомендуемые источники информации (№ источника)			
Основная	Дополнительная	Методическая	Интернет-ресурсы
1-2	1-3	1-3	1-4

Оценочные средства: отчет к лабораторной работе (См.: Фонд оценочных средств)

Лабораторная работа №10

«Разработка вербальной модели предметной области.»

Форма проведения: лабораторная работа

Цель работы:

Освоение принципов вербального описания предметной области. Приобретение практических навыков в создании вербальной модели предметной области.

В иерархии формальных моделей вербальные модели занимают почетное место в основании этой «пирамиды». Такое положение действительно почетно, поскольку вербальные модели — это «альфа» и «омега» многоэтапного процесса моделирования — с этапа синтеза вербальной модели начинается процесс поэтапной формализации и вербальная же (в большинстве случаев) модель формируется на заключительном этапе функционирования модели. Это вызвано вполне понятными причинами — формализм вербальной модели легко воспринимается широким классом потребителей, а синтез вербальной модели (по крайней мере, в первом приближении) может быть осуществлен и специалистом, не обладающим специальными навыками в области построения формальных моделей. Благодаря тому, что языки естественного общения не ограничены рамками узкой предметной области, вербальные модели обладают наивысшей выразительной способностью и часто используются как инструмент интеграции формальных моделей и результатов их применения.

По существу, первичная вербальная модель представляет собой словесный портрет системы и проблемной ситуации, то есть представляет собой документ, аналогичный проекту технического (информационно-поискового и т. п.) задания, разрабатываемого некой организацией-заказчиком. Заметим: процесс синтеза первичной вербальной модели может осуществляться и при участии сторонних (приглашенных) специалистов. К этому шагу приходится прибегать в тех случаях, когда организация не располагает информацией, достаточной для принятия решения или выявления сущности противоречий. Заказчик не всегда в состоянии осознать суть проблем (например, проблем в области управления), с которыми он сталкивается. Находясь внутри системы, заказчик часто пребывает в состоянии информационной изоляции, лишен возможности наблюдать изменения, происшедшие в среде.

Часто на этапе синтеза вербальной модели применяются методы активизации интеллектуальной деятельности специалистов, методы извлечения экспертных знаний, призванные выявить неосознанные алгоритмические схемы функционирования отдельных сотрудников и организации в целом. Здесь могут проводиться в том числе и деловые игры, в ходе которых сторонний специалист пытается выявить алгоритмы функционирования системы, составить схему информационных процессов, информационных контуров управления.

Чтобы понять сущность вербального моделирования, разберемся, для чего создаются вербальные модели. Итак, вербальные модели создаются для:

- 1) получения на материальном носителе вербального описания:
 - структуры системы;
 - отношений между элементами;
 - функций системы и ее компонентов;
 - динамических параметров системы;
 - проблемной ситуации;
 - совокупности целей и задач деятельности;
 - разнообразных ограничений (в том числе — по ресурсам);
 - характеристик среды функционирования и возмущающих воздействий;

2) формирования массива исходных данных, используемых на этапе структурирования и формализации знаний о системе;

3) выявления специфики тезауруса, применяемого в данной предметной области (для внешнего эксперта), и упорядочения системы понятий, подлежащих выражению формальными средствами;

4) выявления неполноты системы знаний и организации процесса их пополнения как за счет внутренних ресурсов системы, так и с привлечением внешних информационных ресурсов;

5) установления характера неопределенностей, с которыми придется столкнуться на этапе синтеза формальной модели;

6) поиска базовых закономерностей и аналогий в смежных отраслях, которые могут быть использованы в дальнейшем.

Таким образом, вербальная модель создается для сокращения неопределенности, компенсации неполноты знаний и формирования гипотезы или набора гипотез. Но первая и главная задача вербального моделирования — **создание вербального описания на материальном носителе.**

Вербальная модель — это не обязательно исключительно текстовый документ — она может содержать в том числе и количественные характеристики, элементы структуризации (например, таблицы, схемы и графики).

В ходе дальнейшей формализации вербальная модель подвергается процедуре структурирования. На этом этапе **устанавливаются группы взаимосвязанных элементов системы** и с необходимой степенью детализации (для решения поставленной задачи) **описываются отношения между ними**, осуществляется **атрибуция элементов системы и данных о них** (устанавливается структура описаний, формулируются требования к точности и т. п.), а также производится группирование данных.

Важным этапом вербального моделирования является этап приведения (стандартизации) терминологии и сокращения избыточности описаний. Результатом выполнения этой процедуры является вербальная модель, построенная в едином стандартизованном тезаурусе, дальнейшее использование которой упрощает решение задач автоматизации процессов анализа и перевода модели на следующий уровень формального представления.

При решении задачи синтеза баз данных и систем информационного обеспечения деловых процессов, данных, полученных на этом этапе, зачастую оказывается достаточно для синтеза макета информационной системы.

По завершении этапа вербального моделирования системы/процесса, при условии, что логическая компонента модели была успешно выделена (не изъята, а именно выделена, маркирована или акцентирована), становится возможен переход на следующий уровень — **уровень логико-лингвистического моделирования.**

Порядок выполнения лабораторной работы.

Дать вербальное описание предметной области по заданию преподавателя с указанием:

- структуры системы;
- отношений между элементами;
- функций системы и ее компонентов;
- динамических параметров системы;
- проблемной ситуации;
- совокупности целей и задач деятельности;
- разнообразных ограничений (в том числе — по ресурсам);
- характеристик среды функционирования и возмущающих воздействий

Контрольные вопросы

1. Что такое вербальная модель предметной области?

2. Для каких целей создается вербальная модель предметной области?

Работа с литературой:

Рекомендуемые источники информации (№ источника)			
Основная	Дополнительная	Методическая	Интернет-ресурсы
1-2	1-3	1-3	1-4

Оценочные средства: отчет к лабораторной работе (См.: Фонд оценочных средств)

Лабораторная работа № 11

«Построение концептуальной базовой модели предметной области.» Форма проведения: лабораторная работа **Цель работы:**

Приобретение практических навыков анализа предметной области, информационных задач и построения концептуальной модели базы данных.

Проектирование базы данных (БД) – одна из наиболее сложных и ответственных задач, связанных с созданием информационной системы (ИС). В результате её решения должны быть определены содержание БД, эффективный для всех её будущих пользователей способ организации данных и инструментальные средства управления данными.

Основная цель процесса проектирования БД состоит в получении такого проекта, который удовлетворяет следующим требованиям:

- корректность схемы БД, т.е. база должна быть гомоморфным образом моделируемой предметной области (ПО), где каждому объекту предметной области соответствуют данные в памяти ЭВМ, а каждому процессу – адекватные процедуры обработки данных;
- обеспечение ограничений ;
- эффективность функционирования ;
- защита данных (от аппаратных и программных сбоев и несанкционированного доступа);
- простота и удобство эксплуатации;
- гибкость, т.е. возможность развития и адаптации к изменениям предметной области и/или требований пользователей.

Техника проектирования баз данных может измениться в целом и в деталях в зависимости от назначения системы. Например, следует различать проектирование систем складирования данных и проектирование так называемых OLTP-систем, ориентируемых на оперативную обработку транзакций. В данном учебном курсе рассматривается проектирование баз данных в основном для OLTP-систем. Именно на таких системах исторически сложилась техника проектирования баз данных.

Процесс проектирования включает в себя следующие этапы:

Концептуальное проектирование – это процедура конструирования информационной модели, не зависящей от каких-либо физических условий реализации.

Логическое проектирование – это процесс конструирования информационной модели на основе существующих моделей данных, не зависимо от используемой СУБД и других условий физической реализации.

Физическое проектирование – это процедура создания описания конкретной реализации БД с описанием структуры хранения данных, методов доступа к данным.

Концептуальное проектирование

Основными задачами концептуального проектирования являются определение предметной области системы и формирование взгляда на ПО с позиций сообщества будущих пользователей БД, т.е. инфологической модели ПО.

Концептуальная модель ПО представляет собой описание структуры и динамики ПО, характера информационных потребностей пользователей в терминах, понятных пользователю и не зависящих от реализации БД. Это описание выражается в терминах не отдельных объектов ПО и связей между ними, а их типов, связанных с ними ограничений целостности и тех процессов, которые приводят к переходу предметной области из одного состояния в другое.

Рассмотрим основные подходы к созданию концептуальной модели предметной области.

Функциональный подход к проектированию БД

Этот метод реализует принцип "от задач" и применяется тогда, когда известны функции некоторой группы лиц и/или комплекса задач, для обслуживания информационных потребностей которых создаётся рассматриваемая БД.

Предметный подход к проектированию БД

Предметный подход к проектированию БД применяется в тех случаях, когда у разработчиков есть чёткое представление о самой ПО и о том, какую именно информацию они хотели бы хранить в БД, а структура запросов не определена или определена не полностью. Тогда основное внимание уделяется исследованию ПО и наиболее адекватному её отображению в БД с учётом самого широкого спектра информационных запросов к ней.

Проектирование с использованием метода "сущность-связь"

Метод "сущность-связь" (entity-relation, ER-method) является комбинацией двух предыдущих и обладает достоинствами обоих. Этап инфологического проектирования начинается с моделирования ПО. Проектировщик разбивает её на ряд локальных областей, каждая из которых (в идеале) включает в себя информацию, достаточную для обеспечения запросов отдельной группы будущих пользователей или решения отдельной задачи (подзадачи). Каждое локальное представление моделируется отдельно, затем они объединяются.

Выбор локального представления зависит от масштабов ПО. Обычно она разбивается на локальные области таким образом, чтобы каждая из них соответствовала отдельному внешнему приложению и содержала 6-7 сущностей.

Сущность – это объект, о котором в системе будет накапливаться информация. Сущности бывают как физически существующие (например, СОТРУДНИК или АВТОМОБИЛЬ), так и абстрактные (например, ЭКЗАМЕН или ДИАГНОЗ).

Для сущностей различают тип сущности и экземпляр. Тип характеризуется именем и списком свойств, а экземпляр – конкретными значениями свойств.

Типы сущностей можно классифицировать как сильные и слабые. Сильные сущности существуют сами по себе, а существование слабых сущностей зависит от существования сильных. Например, читатель библиотеки – сильная сущность, а абонемент этого читателя – слабая, которая зависит от наличия соответствующего читателя. Слабые сущности называют подчинёнными (дочерними), а сильные – базовыми (основными, родительскими).

Для каждой сущности выбираются свойства (атрибуты). Различают:

- Идентифицирующие и описательные атрибуты. Идентифицирующие атрибуты имеют уникальное значение для сущностей данного типа и являются потенциальными ключами. Они позволяют однозначно распознавать экземпляры сущности. Из потенциальных ключей выбирается один первичный ключ (ПК). В качестве ПК обычно выбирается потенциальный ключ, по которому чаще происходит обращение к экземплярам записи. Кроме того, ПК должен включать в свой состав минимально

необходимое для идентификации количество атрибутов. Остальные атрибуты называются описательными и заключают в себе интересующие свойства сущности.

- Составные и простые атрибуты. Простой атрибут состоит из одного компонента, его значение неделимо. Составной атрибут является комбинацией нескольких компонентов, возможно, принадлежащих разным типам данных (например, ФИО или адрес). Решение о том, использовать составной атрибут или разбивать его на компоненты, зависит от характера его обработки и формата пользовательского представления этого атрибута.

- Однозначные и многозначные атрибуты (могут иметь соответственно одно или много значений для каждого экземпляра сущности).

- Основные и производные атрибуты. Значение основного атрибута не зависит от других атрибутов. Значение производного атрибута вычисляется на основе значений других атрибутов (например, возраст студента вычисляется на основе даты его рождения и текущей даты).

Спецификация атрибута состоит из его названия, указания типа данных и описания ограничений целостности – множества значений (или домена), которые может принимать данный атрибут.

Далее осуществляется спецификация связей внутри локального представления. Связи могут иметь различный содержательный смысл (семантику). Различают связи типа "сущность-сущность", "сущность-атрибут" и "атрибут-атрибут" для отношений между атрибутами, которые характеризуют одну и ту же сущность или одну и ту же связь типа "сущность-сущность".

Каждая связь характеризуется именем, обязательностью, типом и степенью. Различают факультативные и обязательные связи. Если вновь порождённый объект одного типа оказывается по необходимости связанным с объектом другого типа, то между этими типами объектов существует обязательная связь (обозначается двойной линией). Иначе связь является факультативной.

По типу различают множественные связи "один к одному" (1:1), "один ко многим" (1:N) и "многие ко многим" (M:N).

Степень связи определяется количеством сущностей, которые охвачены данной связью. Пример бинарной связи – связь между отделом и сотрудниками, которые в нём работают. Примером тернарной связи является связь типа экзамен между сущностями ДИСЦИПЛИНА, СТУДЕНТ, ПРЕПОДАВАТЕЛЬ. Из последнего примера видно, что связь также может иметь атрибуты (в данном случае это Дата проведения и Оценка). Пример ER-диаграммы с указанием сущностей, их атрибутов и связей приведен на рис. 1.



Рисунок 3.1 - Пример ER-диаграммы с однозначными и многозначными атрибутами

В качестве примера возьмем базу данных компании, которая занимается издательской деятельностью.

База данных создаётся для информационного обслуживания редакторов, менеджеров и других сотрудников компании. БД должна содержать данные о сотрудниках компании, книгах, авторах, финансовом состоянии компании и предоставлять возможность получать разнообразные отчёты.

В соответствии с предметной областью система строится с учётом следующих особенностей:

- каждая книга издаётся в рамках контракта;
- книга может быть написана несколькими авторами;
- контракт подписывается одним менеджером и всеми авторами книги;
- каждый автор может написать несколько книг (по разным контрактам);
- порядок, в котором авторы указаны на обложке, влияет на размер гонорара;
- если сотрудник является редактором, то он может работать одновременно над несколькими книгами;
- у каждой книги может быть несколько редакторов, один из них – ответственный редактор;
- каждый заказ оформляется на одного заказчика;
- в заказе на покупку может быть перечислено несколько книг. Выделим базовые сущности этой предметной области:

1. Сотрудники компании. Атрибуты сотрудников – ФИО, табельный номер, пол, дата рождения, паспортные данные, ИНН, должность, оклад, домашний адрес и телефоны. Для редакторов необходимо хранить сведения о редактируемых книгах; для менеджеров – сведения о подписанных контрактах.

1. Авторы. Атрибуты авторов – ФИО, ИНН (индивидуальный номер налогоплательщика), паспортные данные, домашний адрес, телефоны. Для авторов необходимо хранить сведения о написанных книгах.

2. Книги. Атрибуты книги – авторы, название, тираж, дата выхода, цена одного экземпляра, общие затраты на издание, авторский гонорар.

3. Контракты будем рассматривать как связь между авторами, книгами и менеджерами. Атрибуты контракта – номер, дата подписания и участники.

4. Для отражения финансового положения компании в системе нужно учитывать заказы на книги. Для заказа необходимо хранить номер заказа, заказчика, адрес заказчика, дату поступления заказа, дату его выполнения, список заказанных книг с указанием количества экземпляров.

ER–диаграмма издательской компании приведена на рис. 2 (базовые сущности на рисунках выделены полужирным шрифтом).

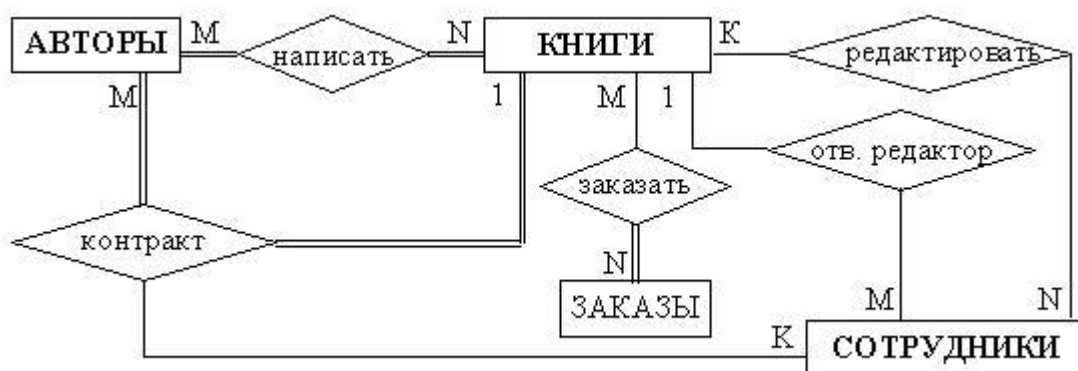


Рисунок 3.2 - ER–диаграмма издательской компании

Анализ информационных задач и круга пользователей системы

Система создаётся для обслуживания следующих групп пользователей:

- администрация (дирекция);
- менеджеры;
- редакторы;
- сотрудники компании, обслуживающие заказы.

Определим границы информационной поддержки пользователей:

1) Функциональные возможности:

- ведение БД (запись, чтение, модификация, удаление в архив);
- обеспечение логической непротиворечивости БД;
- обеспечение защиты данных от несанкционированного или случайного доступа (определение прав доступа);
- реализация наиболее часто встречающихся запросов в готовом виде;
- предоставление возможности сформировать произвольный запрос на языке манипулирования данными.

2) Готовые запросы:

- получение списка всех текущих проектов (книг, находящихся в печати и в продаже);
- получение списка редакторов, работающих над книгами;
- получение полной информации о книге (проекте);
- получение сведений о конкретном авторе (с перечнем всех книг);
- получение информации о продажах (по одному или по всем проектам);
- определение общей прибыли от продаж по текущим проектам;
- определение размера гонорара автора по конкретному проекту

Порядок выполнения лабораторной работы

По заданному преподавателем описанию предметной области построить концептуальную модель базы данных:

- Выделите типы сущностей;
- Выделите типы связей и определите для них показатели кардинальности и степень участия сторон;
- Выделите атрибуты и свяжите их типами сущностей и связей;
- Определите потенциальные и первичные ключи сущностей;
- Нарисуйте ER-диаграмму.

и проанализируйте информационные задачи и группы пользователей.

Контрольные вопросы

1. Какие типы сущностей вы знаете?
2. Какие типы связей вы знаете?
3. Какие виды атрибутов вы знаете?

Работа с литературой:

Рекомендуемые источники информации (№ источника)			
Основная	Дополнительная	Методическая	Интернет-ресурсы
1-2	1-3	1-3	1-4

Оценочные средства: отчет к лабораторной работе (См.: Фонд оценочных средств)

Лабораторная работа №12

«Формирование технического задания на создание конкретной информационной технологии.»

Форма проведения: лабораторная работа

Цель работы:

- Познакомиться с целями создания ТЗ
- Определить состав ТЗ в соответствии со стандартом.
- Научиться формировать ТЗ для конкретных систем и технологий.

На первом этапе формирования информационной системы происходит создание технического задания.

Техническое задание на систему разрабатывается заказчиком при непосредственном участии разработчика.

Техническое задание – документ, утвержденный в соответствующем порядке, определяющий цели, требования и основные исходные данные, необходимые для разработки аналитической информационной системы и содержащий предварительную оценку эффективности системы.

Утвержденное техническое задание является документом, которым разработчики руководствуются на всех этапах создания системы. Изменения, вносимые в техническое задание, должны согласовываться с заказчиком и разработчиком вместе. При разработке технического задания следует:

- установить общую цель создания ЭИС;
- разработать и обосновать требования к информационным подсистемам;
- разработать и обосновать требования к информационной базе, программному обеспечению;
- определить этапы создания системы и сроки их выполнения
- провести предварительный расчет затрат на создание системы и определить уровень экономической эффективности.

При утверждении технического задания разрабатывается график работы и производится полный расчет затрат на разработку системы.

ТЗ содержит следующие разделы, которые могут быть разделены на подразделы:

- общие сведения;
- назначение и цели создания (развития) системы;
- характеристика объектов;
- требования к системе;
- состав и содержание работ по созданию системы;
- порядок контроля и приемки системы;
- требования к составу и содержанию работ по подготовке объекта разработки к вводу системы в действие;
- требования к документированию;
- источники разработки. В ТЗ могут включаться приложения.

В зависимости от вида, назначения, специфических особенностей проекта и условий функционирования системы допускается оформлять разделы ТЗ в виде приложений, вводить дополнительные, исключать или объединять подразделы ТЗ.

В ТЗ на части системы не включают разделы, дублирующие содержание разделов ТЗ в целом.

В разделе “Общие сведения” указывают:

- полное наименование системы и ее условное обозначение;
- шифр темы или шифр (номер) договора;
- наименование компаний разработчика и заказчика (пользователя) системы и их реквизиты;

- перечень документов, на основании которых создается система, кем и когда утверждены эти документы;
- плановые сроки начала и окончания работы по созданию системы;
- сведения об источниках и порядке финансирования работ;
- порядок оформления и предъявления заказчику результатов работ по созданию системы (ее частей), по изготовлению и наладке отдельных средств (технических, программных, информационных) и программно-технических (программно-методических) комплексов системы.

Раздел “Назначение и цели создания (развития) системы” состоит из подразделов:

- назначение системы;
- цели создания системы.

В подразделе “Назначение системы” указывают вид деятельности системы (управление, проектирование и т. п.) и перечень объектов информатизации (объектов), на которых предполагается ее использовать.

В подразделе “Цели создания системы” приводят наименования и требуемые значения технических, технологических, производственно-экономических или других показателей объекта информатизации, которые должны быть достигнуты в результате создания ИС, и указывают критерии оценки достижения целей создания системы.

В разделе “Характеристики объекта информатизации” приводят:

- краткие сведения об объекте информатизации или ссылки на документы, содержащие такую информацию;
- сведения об условиях эксплуатации объекта автоматизации.

Раздел “Требования к системе” состоит из следующих подразделов:

- требования к системе в целом;
- требования к функциям (задачам), выполняемым системой;
- требования к видам обеспечения.

Состав требований к системе, включаемых в данный раздел ТЗ на ИС, устанавливают в зависимости от вида, назначения, специфических особенностей и условий функционирования конкретной системы.

В подразделе “Требования к системе в целом” указывают:

- требования к структуре и функционированию системы;
- требования к численности и квалификации персонала системы и режиму его работы;
- показатели назначения;
- требования к надежности;
- требования безопасности;
- требования к эргономике и технической эстетике;
- требования к эксплуатации, техническому обслуживанию, ремонту и хранению компонентов системы;
- требования к защите информации от несанкционированного доступа;
- требования по сохранности информации при авариях;
- требования к защите от влияния внешних воздействий;
- требования к патентной чистоте;
- требования по стандартизации и унификации;
- дополнительные требования.

В требованиях к структуре и функционированию системы приводят:

- перечень подсистем, их назначение и основные характеристики, требования к числу уровней иерархии и степени централизации системы;
- требования к способам и средствам связи для информационного обмена между компонентами системы;
- требования к характеристикам взаимосвязей создаваемой системы со смежными системами, требования к ее совместимости, в том числе указания о

способах обмена информацией (автоматически, пересылкой документов, по телефону и т. п.);

- требования к режимам функционирования системы;
- требования по диагностированию системы;
- перспективы развития, модернизации системы.

В требованиях к численности и квалификации персонала на ИС приводят:

- требования к численности персонала (пользователей) ИС;
- требования к квалификации персонала, порядку его подготовки и контроля знаний и навыков;
- требуемый режим работы персонала ИС.

В требованиях к показателям назначения ИС приводят значения параметров, характеризующие степень соответствия системы ее назначению.

В требования к надежности включают:

- состав и количественные значения показателей надежности для системы в целом или ее подсистем;
- перечень аварийных ситуаций, по которым должны быть регламентированы требования к надежности, и значения соответствующих показателей;
- требования к надежности технических средств и программного обеспечения;
- требования к методам оценки и контроля показателей надежности на разных стадиях создания системы в соответствии с действующими нормативно-техническими документами.

В требования по безопасности включают требования по обеспечению безопасности при поставке, наладке, эксплуатации и обслуживании системы.

В требования по эргономике и технической эстетике включают показатели ИС, задающие необходимое качество взаимодействия человека с машиной и комфортность условий работы персонала.

В требования к защите информации от несанкционированного доступа включают требования, установленные действующей в отрасли и информационной среде заказчика.

В требованиях по сохранности информации приводят перечень событий: аварий, отказов технических средств (в том числе - потеря питания) и т. п., при которых должна быть обеспечена сохранность информации в системе.

В требованиях по патентной чистоте указывают перечень стран, в отношении которых должна быть обеспечена патентная чистота системы и ее частей.

В дополнительные требования включают специальные требования по усмотрению разработчика или заказчика системы.

В подразделе “Требование к функциям (задачам)”, выполняемым системой, приводят:

- по каждой подсистеме перечень функций, задач или их комплексов (в том числе обеспечивающих взаимодействие частей системы), подлежащих автоматизации;
- при создании системы в две или более очереди - перечень функциональных подсистем, отдельных функций или задач, вводимых в действие в 1-й и последующих очередях;
- временной регламент реализации каждой функции, задачи (или комплекса задач);
- требования к качеству реализации каждой функции (задачи или комплекса задач), к форме представления выходной информации, характеристики необходимой точности и времени выполнения, требования одновременности выполнения группы функций, достоверности выдачи результатов;

- перечень и критерии отказов для каждой функции, по которой задаются требования по надежности.

В подразделе “Требования к видам обеспечения” в зависимости от вида системы приводят требования к математическому, информационному, лингвистическому, программному, техническому, метрологическому, организационному, методическому и другим видам обеспечения системы.

Для информационного обеспечения системы приводят требования:

- к составу, структуре и способам организации данных в системе;
- к информационному обмену между компонентами системы;
- к информационной совместимости со смежными системами;
- по применению систем управления базами данных;
- к структуре процесса сбора, обработки, передачи данных в системе и представлению данных;
- к защите данных;
- к контролю, хранению, обновлению и восстановлению данных;

Для лингвистического обеспечения системы приводят требования к применению в системе языков программирования высокого уровня, языков взаимодействия пользователей и технических средств системы, а также требования к кодированию и декодированию данных, к языкам ввода-вывода данных, языкам манипулирования данными, средствам описания предметной области, к способам организации диалога.

Для программного обеспечения системы приводят перечень покупных программных средств, а также требования:

- к зависимости программных средств от операционной среды;
- к качеству программных средств, а также к способам его обеспечения и контролю;

Для технического обеспечения системы приводят требования:

- к видам технических средств, в том числе к видам комплексов технических средств, программно-технических комплексов и других комплектующих изделий, допустимых к использованию в системе;
- к функциональным, конструктивным и эксплуатационным характеристикам средств технического обеспечения системы.

В требованиях к метрологическому обеспечению приводят:

- предварительный перечень измерительных каналов;
- требования к точности измерений параметров и (или) к метрологическим характеристикам измерительных каналов;
- требования к метрологической совместимости технических средств системы;
- перечень управляющих и вычислительных каналов системы, для которых необходимо оценивать точностные характеристики;
- требования к метрологическому обеспечению технических и программных средств, входящих в состав измерительных каналов системы, средств, встроенного контроля, метрологической пригодности измерительных каналов и средств измерений, используемых при наладке и испытаниях системы;
- вид метрологической аттестации (государственная или ведомственная) с указанием порядка ее выполнения и организаций, проводящих аттестацию.

Для организационного обеспечения приводят требования:

- к структуре и функциям подразделений, участвующих в функционировании системы или обеспечивающих эксплуатацию;
- к организации функционирования системы и порядку взаимодействия персонала ИС и персонала объекта информатизации;
- к защите от ошибочных действий персонала системы.

Раздел “Состав и содержание работ по созданию (развитию) системы” должен содержать перечень стадий и этапов работ по созданию системы, сроки их выполнения, перечень организаций - исполнителей работ, ссылки на документы, подтверждающие согласие этих организаций на участие в создании системы, или запись, определяющую ответственного (заказчик или разработчик) за проведение этих работ.

В данном разделе также приводят:

- перечень документов предъявляемых по окончании соответствующих стадий и этапов работ;
- вид и порядок проведения экспертизы технической документации (стадия, этап, объем проверяемой документации, организация-эксперт);
- программу работ, направленных на обеспечение требуемого уровня надежности разрабатываемой системы (при необходимости);
- перечень работ по метрологическому обеспечению на всех стадиях создания системы с указанием их сроков выполнения и организаций-исполнителей (при необходимости).

В разделе “Порядок контроля и приемки системы” указывают:

- виды, состав, объем и методы испытаний системы и ее составных частей;
- общие требования к приемке работ по стадиям, порядок согласования и утверждения приемочной документации;

В разделе “Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие” необходимо привести перечень основных мероприятий и их исполнителей, которые следует выполнить при подготовке проекта к вводу ИС в действие.

В перечень основных мероприятий включают:

- приведение поступающей в систему информации (в соответствии с требованиями к информационному и лингвистическому обеспечению);
- создание условий функционирования проекта, при которых гарантируется соответствие создаваемой системы требованиям, содержащимся в ТЗ;
- создание необходимых для функционирования системы подразделений и служб;
- сроки и порядок комплектования штатов и обучения персонала.

В разделе “Требования к документированию” приводят:

- согласованный разработчиком и Заказчиком системы перечень

подлежащих разработке комплектов и видов документов; перечень документов, выпускаемых на машинных носителях;

- при отсутствии государственных стандартов, определяющих требования к документированию элементов системы, дополнительно включают требования к составу и содержанию таких документов.

В разделе “Источники разработки” должны быть перечислены документы и информационные материалы, на основании которых разрабатывалось ТЗ и которые должны быть использованы при создании системы.

Порядок выполнения лабораторной работы

По заданному преподавателем описанию предметной области оформить техническое задание обратив особое внимание на подробное описание Требований к системе в целом.

Контрольные вопросы

1. Каково назначение ТЗ
2. На каких этапах разработки используется ТЗ?

3. Кто участвует в формировании ТЗ?

Работа с литературой:

Рекомендуемые источники информации (№ источника)			
Основная	Дополнительная	Методическая	Интернет-ресурсы
1-2	1-3	1-3	1-4

Оценочные средства: отчет к лабораторной работе (См.: Фонд оценочных средств)

Лабораторная работа № 13**«Разработка схемы автоматизированного процесса обработки информации прикладной информационной технологии.»**

Форма проведения: **лабораторная работа**

Цель работы:

Приобрести практические навыки в разработке схемы автоматизированного процесса обработки информации прикладной информационной технологии.

Реализация информационных технологий основана на рационально организованном технологическом процессе обработки экономической информации.

Технологический процесс автоматизированной обработки экономической информации представляет собой совокупность операций, выполняемых в строго регламентированной последовательности, начиная от сбора (регистрации) исходной информации до передачи результатной информации заинтересованным пользователям для выполнения функций управления.

Под **операцией** в данном случае понимается комплекс действий, выполняемых над информацией на одном рабочем месте.

Разработка технологического процесса должна обеспечить максимальную автоматизацию процессов обработки информации при использовании различных технологических средств и высокую достоверность получения результатной информации при минимальных трудовых и стоимостных затратах.

Состав операций и последовательность их выполнения зависят от характера решаемых экономических задач и имеющегося комплекса технических средств.

Характер задач в основном определяется объемами обрабатываемой информации, периодичностью решения, а также сложностью алгоритмов ее преобразования. Решение задачи на каждом уровне обработки может выполняться разными способами с применением различных вычислительных средств и методов контроля.

При определении состава операций технологического процесса создается возможность выбора наиболее рационального способа обработки информации.

В технологическом процессе автоматизированной обработки экономической информации при решении различных задач прослеживаются **три основных этапа**: первичный, основной и заключительный.

Операции **первичного этапа** связаны со сбором и подготовкой первичных документов к машинной обработке. От качества выполнения этих операций во многом зависит эффективность и достоверность непосредственной автоматизированной обработки.

Операции **основного этапа** обеспечивают ввод и обработку первичных документов, а также вывод результатных документов в соответствии с заданным алгоритмом решения экономических задач.

На **заключительном этапе** выполняются операции контроля и выпуска результатных документов с целью проверки качества их обработки, а также рассылки различным потребителям.

Так как при выполнении отдельных операций технологического процесса обработки информации могут быть использованы различные средства вычислительной техники и в различных сочетаниях, на практике существует множество вариантов технологических процессов. Эффективность проектирования автоматизированной обработки экономической информации требует их стандартизации.

Под **стандартизацией технологии автоматизированной обработки информации** понимается разработка комплекса детализированных и максимально унифицированных схем технологических процессов, в которых строго установлены состав и последовательность выполнения операций.

Ориентация на стандартные технологические процессы приводит к улучшению качества проектирования, облегчению и ускорению внедрения проектов автоматизированной обработки данных. Поэтому подробнее рассмотрим типовую схему технологии обработки информации при решении экономической задачи, представленную на рис. 5.1.

Первичный этап начинается со сбора первичных документов от различных источников информации (экономических служб предприятий или фирм), кодирования необходимых реквизитов, а также контроля полноты и качества их заполнения экономистами (опер. 1). Если обнаруживаются ошибки, то экономисты делают запросы к источникам информации (опер. 3).

В противном случае по ним производится подсчет и регистрация контрольных сумм (опер. 4). Контрольные суммы необходимы в дальнейшем для проверки правильности ввода данных в ЭВМ и в зависимости от структуры первичного документа они могут подсчитываться по каждой строке, графе или по отчету в целом.

Первичные данные и промежуточные результаты по задачам могут быть представлены в электронном виде, а при вводе машиночитаемых документов используются сканирующие устройства.

Основной этап начинается со ввода первичных данных с клавиатуры, их контроля (арифметического, логического и синтаксического) и, независимо от результатов контроля, записи информации в файл или информационную базу (опер. 5).

Арифметический контроль заключается в проверке баланса данных в строках и графах первичного документа с применением заранее подсчитанных контрольных сумм. При логическом контроле проверяются тип данных, допустимые отклонения значений от наиболее вероятных, предельные значения данных, а также различные смысловые проверки с использованием специальных алгоритмов. С помощью синтаксического контроля проверяется структура заполнения документа различными данными.

При этом осуществляется проверка полноты ввода и своевременная корректировка первичных документов на магнитных носителях, если ошибка связана с набором информации.

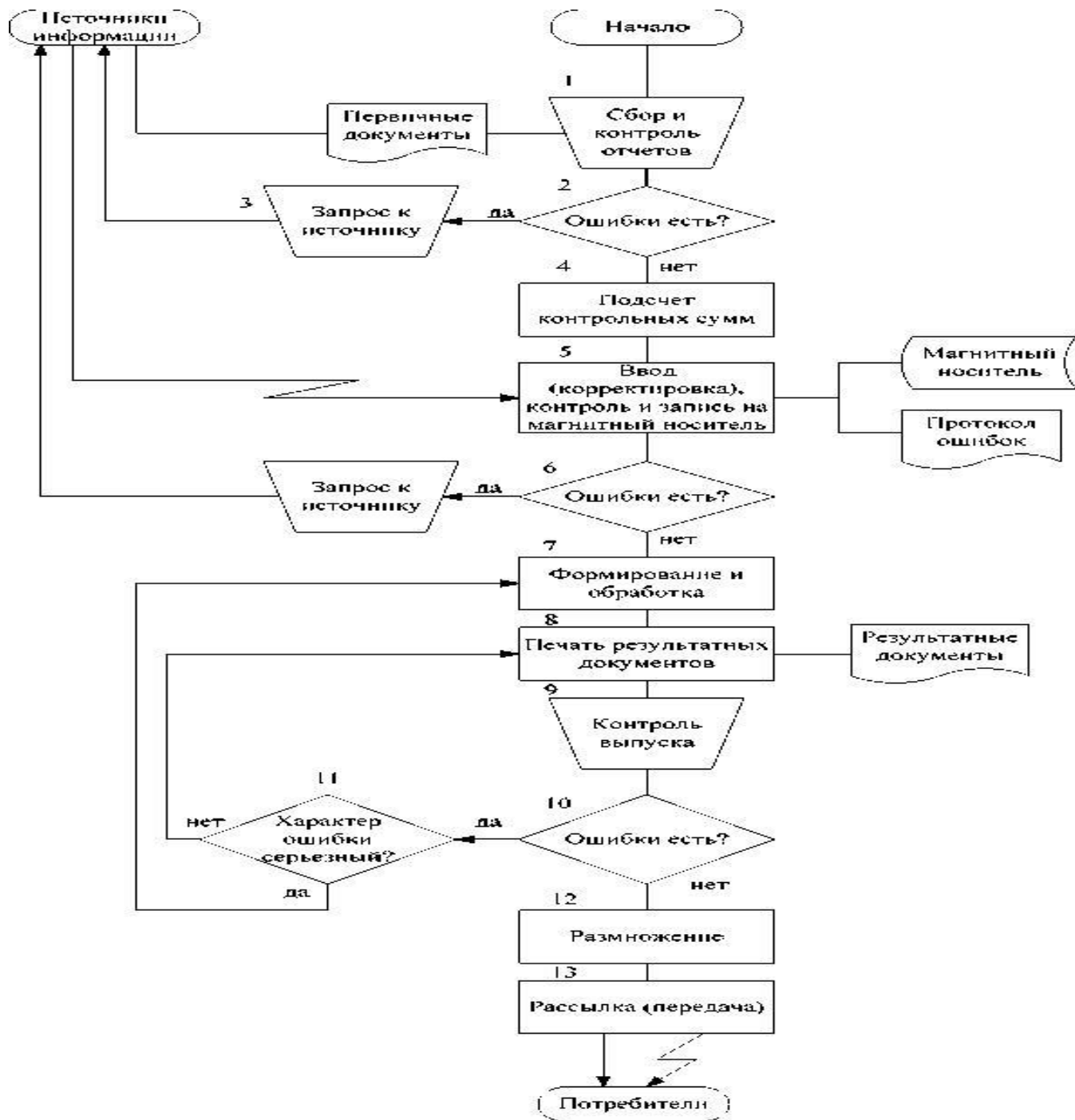


Рисунок 5.1. Типовая схема технологического процесса обработки экономической информации

Если при контроле обнаружилось нарушение, связанное с содержательной стороной представленного первичного документа, то производится печать протокола ошибок, в котором указывается характер нарушений, конкретная ошибка и ее местонахождение в файле или информационной базе.

На основании этого протокола делается запрос к источникам информации (опер. 3), где выясняются ошибки и исправленные данные поступают по каналам связи (факс или электронная почта) для последующей корректировки информационной базы (опер. 5). Только после внесения всех изменений и проверки контроля полноты записи первичных документов в информационной базе приступают к формированию и обработке результатных данных (опер. 7) согласно заданному алгоритму решения задачи, а затем осуществляется печать результатных документов (опер. 8).

На заключительном этапе технологического процесса обработки информации экономистами осуществляется контроль выпуска результатных документов (опер. 9), состоящий в проверке на полноту вхождения исходных данных в заданный разрез обработки и качества их печати. Учитывая характер ошибки (опер. 11), формирование и обработка повторяются или результатные данные печатаются заново.

Затем осуществляется размножение результатных данных с помощью различных копировально-множительных средств (опер. 12) и рассылка их в виде документов и магнитных носителей или передача их по каналам связи различным заинтересованным потребителям (опер. 13).

Рассмотренная типовая схема предусматривает, что первичная информация от источника может сразу поступать на ввод и контроль (опер. 5) в зависимости от того, каким образом она будет передаваться (по факсу или по электронной почте).

Основными документами, регламентирующими технологический процесс обработки экономической информации, являются технологические и инструкционные карты.

Технологическая карта представляет собой описание последовательно выполняемых операций технологического процесса по каждой экономической задаче.

На каждую операцию технологического процесса разрабатывается **инструкционная карта**, в которой указываются сведения об исходной информации, о конечных результатах и о порядке выполнения конкретной работы.

Порядок выполнения лабораторной работы

По заданному преподавателем описанию предметной области разработать схему автоматизированного процесса обработки информации прикладной информационной технологии.

Контрольные вопросы

1. Что такое технологический процесс автоматизированной обработки экономической информации?
2. Три основных этапа автоматизированной обработки экономической информации.
3. Что такое технологическая карта?

Работа с литературой:

Рекомендуемые источники информации (№ источника)			
Основная	Дополнительная	Методическая	Интернет-ресурсы
1-2	1-3	1-3	1-4

Оценочные средства: отчет к лабораторной работе (См.: Фонд оценочных средств)

Лабораторная работа № 14

«Разработка информационно-логической модели и алгоритмов обработки данных»

Форма проведения: **лабораторная работа**

Цель работы:

Овладеть навыками работы в Egwin, построения логической модели заданной предметной области. Построить логическую модель данных выбранной предметной области в нотации IDEF1X.

Описание процесса создания модели данных с использованием ERWIN

Физическая и логическая модель данных

ERwin имеет два уровня представления модели - логический и физический.

Логический уровень - это абстрактный взгляд на данные, на нем данные представляются так, как выглядят в реальном мире, и могут называться так, как они называются в реальном мире, например "Автор", "Издательство" или "Авторский гонорар". Объекты модели, представляемые на логическом уровне, называются сущностями и атрибутами (подробнее о сущностях и атрибутах будет рассказано ниже). Логическая модель данных может быть построена на основе другой логической модели, например на основе модели процессов. Логическая модель данных является универсальной и никак не связана с конкретной реализацией СУБД.

Физическая модель данных, напротив, зависит от конкретной СУБД, фактически являясь отображением системного каталога. В физической модели содержится информация о всех объектах БД. Поскольку стандартов на объекты БД не существует (например, нет стандарта на типы данных), физическая модель зависит от конкретной реализации СУБД. Следовательно, одной и той же логической модели могут соответствовать несколько разных физических моделей. Если в логической модели не имеет значения, какой конкретно тип данных имеет атрибут, то в физической модели важно описать всю информацию о конкретных физических объектах - таблицах, колонках, индексах, процедурах и т. д. Разделение модели данных на логические и физические позволяет решить несколько важных задач.

Масштабирование. Создание модели данных, как правило, начинается с создания логической модели. После описания логической модели, проектировщик может выбрать необходимую СУБД и ERwin автоматически создаст соответствующую физическую модель. На основе физической модели ERwin может сгенерировать системный каталог СУБД или соответствующий SQL-скрипт. Этот процесс называется прямым проектированием (Forward Engineering). Тем самым достигается масштабируемость - создав одну логическую модель данных, можно сгенерировать физические модели под любую поддерживаемую ERwin СУБД. С другой стороны, ERwin способен по содержимому системного каталога или SQL-скрипту воссоздать о физическую и логическую модель данных (Reverse Engineering). На основе полученной логической модели данных можно сгенерировать физическую модель для другой СУБД и затем сгенерировать ее системный каталог. Следовательно, ERwin позволяет решить задачу по переносу структуры данных с одного сервера на другой. Например, можно перенести структуру данных с Oracle на Informix (или наоборот) или перенести структуру dbf-файлов в реляционную СУБД, тем самым облегчив решение по переходу от файл-серверной к клиент-серверной ИС. Заметим, однако, что формальный перенос структуры "плоских" таблиц на реляционную СУБД обычно неэффективен. Для того чтобы извлечь выгоды от перехода на клиент-серверную технологию, структуру данных следует модифицировать. Процессы прямого и обратного проектирования будут рассмотрены ниже.

Для переключения между логической и физической моделью данных служит список выбора в левой части панели инструментов Erwin (рис. 1).



рис. 1. Переключение между логической и физической моделью

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

Создание логической модели базы данных

Уровни логической модели

Различают три уровня логической модели, отличающихся по глубине представления информации о данных:

- * диаграмма сущность-связь (Entity Relationship Diagram, ERD);
- * модель данных, основанная на ключах (Key Based model, KB);
- * полная атрибутивная модель (Fully Attributed model, FA).

Диаграмма сущность-связь представляет собой модель данных верхнего уровня. Она включает сущности и взаимосвязи, отражающие основные бизнес-правила предметной области. Такая диаграмма не слишком детализирована, в нее включаются основные сущности и связи между ними, которые удовлетворяют основным требованиям, предъявляемым к ИС. Диаграмма сущность-связь может включать связи многие-ко-многим и не включать описание ключей. Как правило, ERD используется для презентаций и обсуждения структуры данных с экспертами предметной области.

Модель данных, основанная на ключах, - более подробное представление данных. Она включает описание всех сущностей и первичных ключей и предназначена для представления структуры данных и ключей, которые соответствуют предметной области. **Полная атрибутивная модель** - наиболее детальное представление структуры данных: представляет данные в третьей нормальной форме и включает все сущности, атрибуты и связи.

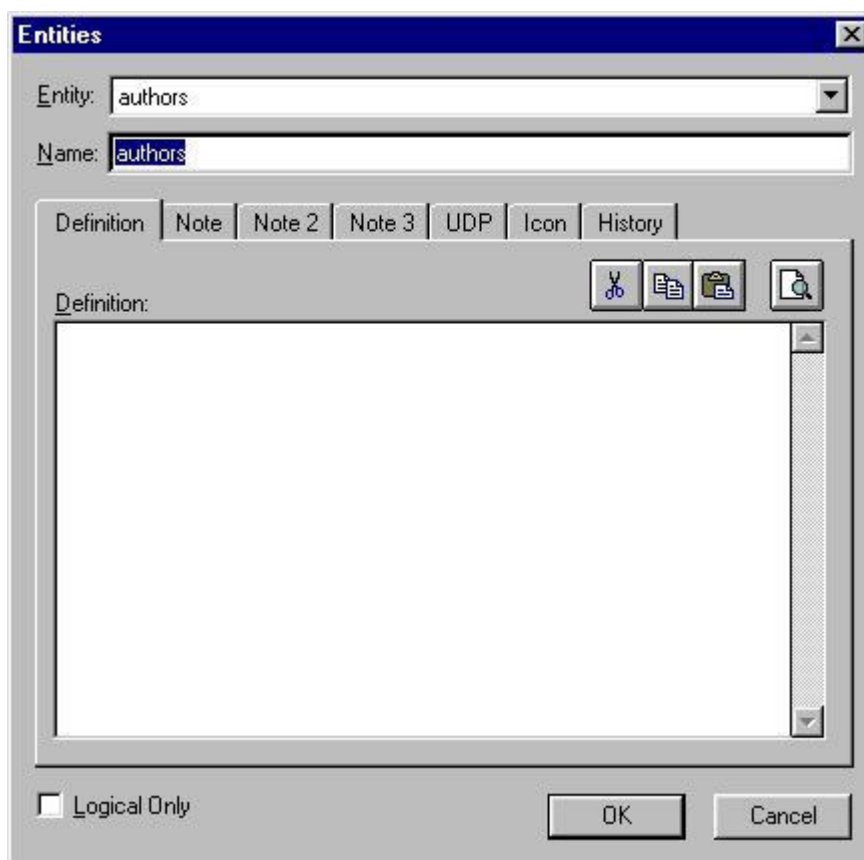
Сущности и атрибуты

Основные компоненты диаграммы Erwin - это сущности, атрибуты и связи. Каждая сущность является множеством подобных индивидуальных объектов, называемых экземплярами. Каждый экземпляр индивидуален и должен отличаться от всех остальных экземпляров. Атрибут выражает определенное свойство объекта. С точки зрения БД (физическая модель) сущности соответствует таблица, экземпляру сущности - строка в таблице, а атрибуту - колонка таблицы.

Построение модели данных предполагает определение сущностей и атрибутов, т. е. необходимо определить, какая информация будет храниться в конкретной сущности или атрибуте. Сущность можно определить как объект, событие или концепцию, информация о которых должна сохраняться. Сущности должны иметь наименование с четким смысловым значением, именоваться существительным в единственном числе, не носить "технических" наименований и быть достаточно важными для того, чтобы их моделировать. Именование сущности в единственном числе облегчает в дальнейшем чтение модели. Фактически имя сущности дается по имени ее экземпляра. Примером может быть сущность **Автор** (но не **Авторы!**) с атрибутами **Номер автора**, **Фамилия автора** и **Адрес автора**. На уровне физической модели ей может соответствовать таблица **Authors** с колонками **au_id**, **au_lname** и **address**.

Для внесения сущности в модель необходимо (убедившись предварительно, что вы находитесь на уровне логической модели - переключателем между логической и физической моделью служит раскрывающийся список в правой части панели инструментов) "кликнуть" по кнопке сущности на панели инструментов (ERwin Toolbox), затем "кликнуть" по тому месту на диаграмме, где необходимо расположить новую

сущность. Щелкнув правой кнопкой мыши по сущности и выбрав из всплывающего меню пункт Entity Properties, можно вызвать диалог Entities, в котором определяются имя, описание и комментарии сущности.



Каждая сущность должна быть полностью определена с помощью текстового описания в укладке Definition. Закладки Note, Note 2, Note 3, UDP (User Defined Properties - Свойства, определенные пользователем) служат для внесения дополнительных комментариев и определений к сущности. В прежних версиях ERwin закладкам Note2 и Note3 соответствовали окна Query и Sample.

Закладка Definition используется для ввода определения сущности. 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 (CREATE COMMENT on entity_name).

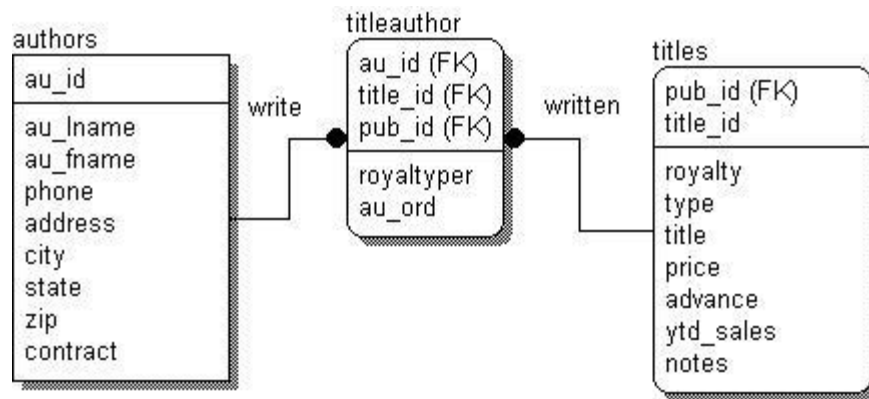
Закладка Note позволяет добавлять дополнительные замечания о сущности, которые не были отражены в определении, введенном в закладке Definition. Здесь можно ввести полезное замечание, описывающее какое-либо бизнес-правило или соглашение по организации диаграммы.

В закладке Note 2 можно документировать некоторые возможные запросы, которые, как ожидается, будут использоваться по отношению к сущности в БД. При переходе к физическому проектированию, записанные запросы помогут принимать такие решения в отношении проектирования, которые сделают БД более эффективной.

Связи

Связь является логическим соотношением между сущностями. Каждая связь должна именоваться глаголом или глагольной фразой (Relationship Verb Phrases). Имя связи выражает некоторое ограничение или бизнес-правило и облегчает чтение диаграммы, например:

- * Каждый АВТОР <пишет> КНИГИ;
- * Каждый КНИГА <написана> АВТОРОМ.



Связь показывает, какой именно автор написал какие книги и какие именно книги были написаны каким автором. По умолчанию имя связи на диаграмме не показывается. Для отображения имени следует в контекстном меню, которое появляется, если щелкнуть левой кнопкой мыши по любому месту диаграммы, не занятому объектами модели, выбрать пункт Relationship Display и затем включить опцию Verb Phrase.

На логическом уровне можно установить идентифицирующую связь один-ко-многим, связь многие-ко-многим и неидентифицирующую связь один-ко-многим (соответственно это кнопки слева направо в палитре инструментов).

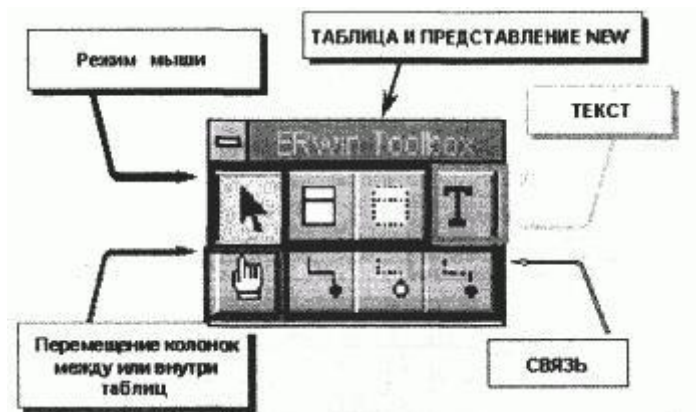


рис.1

На логическом уровне можно установить идентифицирующую связь один-ко-многим, связь многие-ко-многим и неидентифицирующую связь один-ко-многим (соответственно это кнопки слева направо в палитре инструментов).

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 сущность в зависимую. Зависимая сущность изображается прямоугольником со скругленными углами (сущность **titleauthor** на рис. 2). Экземпляр зависимой сущности определяется только через отношение к родительской сущности, т. е. в структуре на рис. 2 информация о гонораре и о порядковом номере автора не может быть внесена и не имеет смысла без информации об авторе, которому этот гонорар предназначен. При установлении идентифицирующей связи атрибуты первичного ключа родительской 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 внешний ключ - (FK).

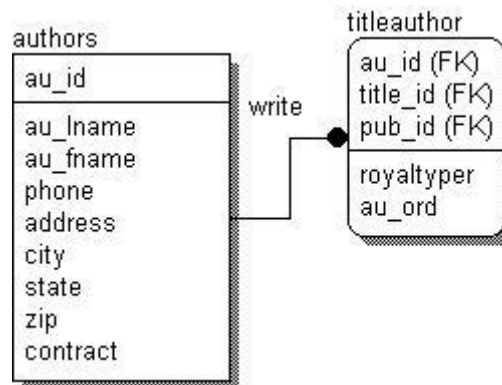


рис.2

В дальнейшем, при генерации схемы БД, атрибуты первичного ключа получают признак NOT NULL, что означает невозможность внесения записи в таблицу заказов без информации о номере клиента.

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

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

Для создания новой связи следует:

- * установить курсор на нужной кнопке в палитре инструментов (идентифицирующая или неидентифицирующая связь) и нажать левую кнопку мыши (рис. 1);
- * щелкнуть сначала по родительской, а затем по дочерней сущности.

Форму линии связи можно изменить. Для этого нужно захватывать мышью нужную линию связи и переносить ее с места на место, пока линия не начнет выглядеть лучше. В палитре инструментов кнопка соответствует идентифицирующей связи, кнопка связи многие-ко-многим и кнопка соответствуют неидентифицирующей связи.

Для редактирования свойств связи следует "кликнуть" правой кнопкой мыши по связи и выбрать на контекстном меню пункт Relationship Editor.

Ключи

Каждый экземпляр сущности должен быть уникален и отличаться от других атрибутов.

Первичный ключ (primary key) - это атрибут или группа атрибутов, однозначно идентифицирующая 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 диалог Attribute Editor для того, чтобы сделать его атрибутом первичного ключа, нужно включить флажок Primary Key в нижней части закладки General. На диаграмме неключевой атрибут можно внести в состав первичного ключа, воспользовавшись режимом переноса атрибутов (кнопка в палитре инструментов).

Выбор первичного ключа может оказаться непростой задачей, решение которой может

повлиять на эффективность будущей ИС. В одной сущности могут оказаться несколько атрибутов или наборов атрибутов, претендующих на роль первичного ключа. Такие претенденты называются **потенциальными ключами (candidate key)**. Ключи могут быть **сложными**, т. е. содержащими несколько атрибутов. Сложные первичные ключи не требуют специального обозначения - это список атрибутов выше горизонтальной линии.

Рассмотрим сущность **authors**. Атрибуты, претендующие на роль первичных ключей:

- au_id - идентификатор автора
- au_lname + au_fname + zip - фамилия, имя автора и энергичность
- address - адрес проживания автора.



Для того чтобы стать первичным, потенциальный ключ должен удовлетворять ряду требований:

Уникальность. Два экземпляра не должны иметь одинаковых значений возможного ключа. Этому требованию удовлетворяет **идентификатор автора**.

Компактность. Сложный возможный ключ не должен содержать ни одного атрибута, удаление которого не приводило бы к утрате уникальности. Тогда второй ключ, претендующий на роль первичного не является компактным, так как удаление энергичности не приведет к утрате уникальности.

Таким образом, после прохождения всех вышеприведенных этапов была получена следующая модель данных (рис.3):

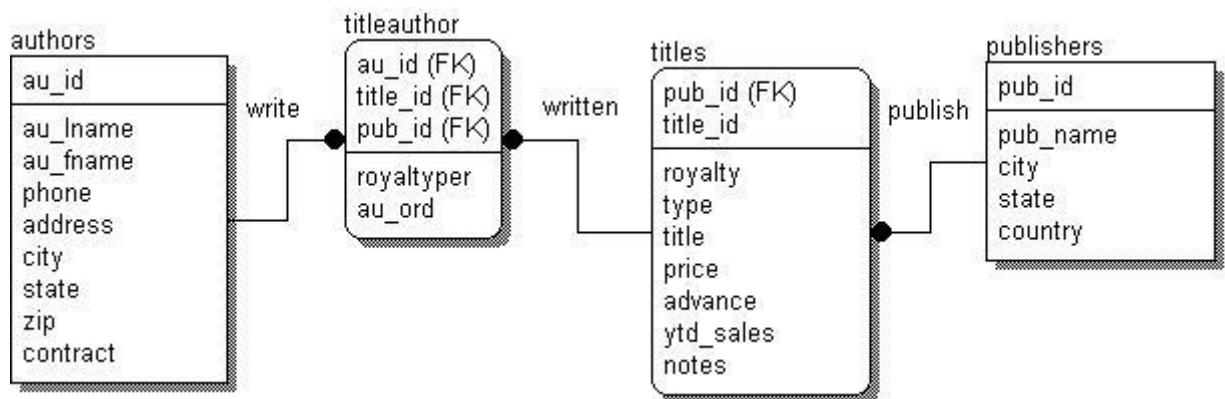


рис.3

Контрольные вопросы

1. Что такое бизнес-процесс?
2. Каковы основные компоненты функциональной модели?
3. Что представляют собой методологии функционального моделирования?
4. Что такое сценарии?

Работа с литературой:

Рекомендуемые источники информации (№ источника)			
Основная	Дополнительная	Методическая	Интернет-ресурсы
1-2	1-3	1-3	1-4

Оценочные средства: отчет к лабораторной работе (См.: Фонд оценочных средств)

Лабораторная работа № 15**«Создание физической модели базы**

данных.» Форма проведения: лабораторная работа **Цель работы:**

Научиться разрабатывать физическую модель базы данных.

Уровни физической модели**Различают два уровня физической модели:**

- * трансформационная модель (Transformation Model);
- * модель СУБД (DBMS Model).

Физическая модель содержит всю информацию, необходимую для реализации конкретной БД. Трансформационная модель содержит информацию для реализации отдельного проекта, который может быть частью общей ИС и описывать подмножество предметной области. ERwin поддерживает ведение отдельных проектов, позволяя проектировщику выделять подмножество модели в виде предметных областей (Subject Area). Трансформационная модель позволяет проектировщикам и администраторам БД лучше представлять, какие объекты БД хранятся в словаре данных, и проверить, насколько физическая модель данных удовлетворяет требованиям к ИС. Модель СУБД автоматически генерируется из трансформационной модели и является точным отображением системного каталога СУБД. ERwin непосредственно поддерживает эту модель путем генерации системного каталога.

Выбор сервера

Физический уровень представления модели зависит от выбранного сервера. Для выбора СУБД служит редактор Target Server (меню Server/Target Server... доступно только на физическом уровне) (рис. 3).

ERwin поддерживает практически все распространенные СУБД, всего более 20 реляционных и нереляционных БД. Для выбора СУБД нужно щелкнуть по соответствующей кнопке рядом с именем СУБД.

Диалог Target Server позволяет задать тип данных и опцию NULL для новых колонок, а также правила ссылочной целостности, принимаемые по умолчанию. Тип данных можно выбрать в раскрывающемся списке Default Datatype, который автоматически заполняется типами данных, поддерживаемых выбранным сервером. Установка правил ссылочной целостности по умолчанию была рассмотрена в пункте "Связи".

Группа кнопок Default Non-Key Null Option позволяет разрешить или запретить значения NULL для неключевых колонок.

Окно выбора Allow special chars in names позволяет разрешить или запретить использование специальных символов и пробелов в именах таблиц. Эта опция действует только для тех СУБД, которые поддерживают использование специальных символов.

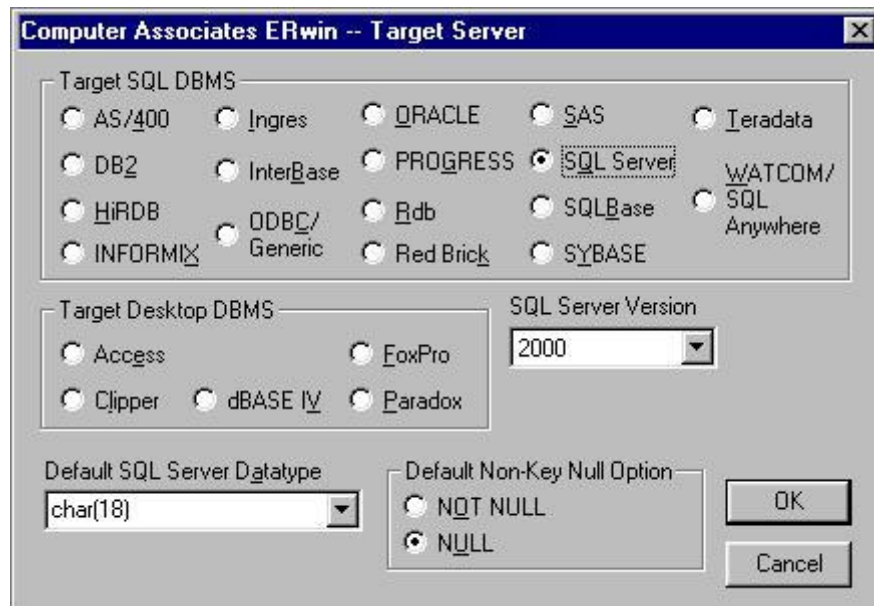


рис.4

По умолчанию ERwin генерирует имена таблиц и индексов по шаблону на основе имен соответствующих сущностей и ключей логической модели. Окна Table Name Macro и Index Name Macro позволяют изменить шаблон генерации имен, заданный по умолчанию. В дальнейшем имена таблиц и индексов можно изменить вручную.

Кнопка Reset Names вызывает диалог Globally Reset DBMS Property (рис. 5), 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. Для автоматического преобразования следует в ответ на запрос нажать Yes.

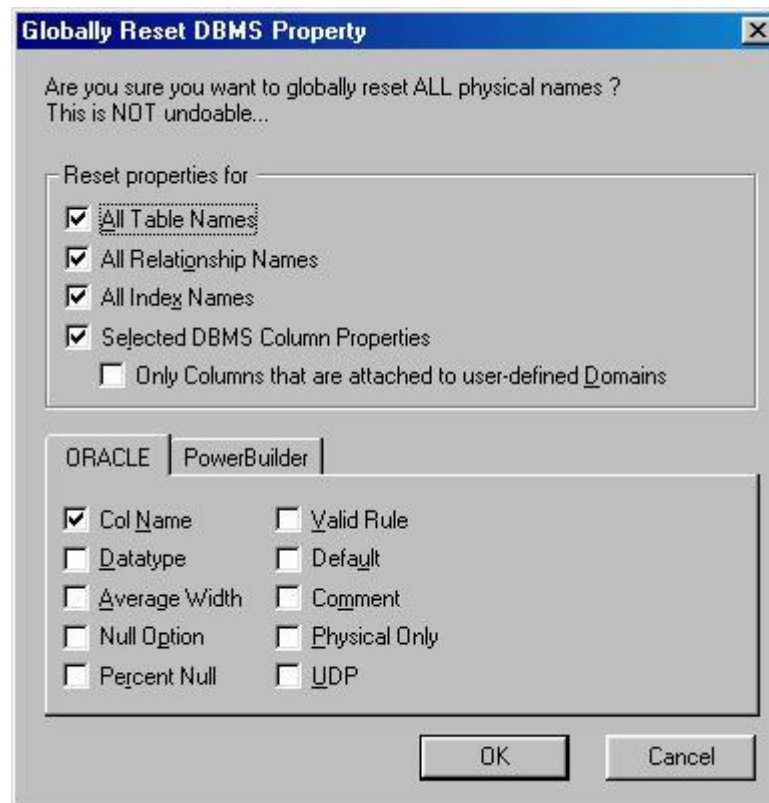


рис.5. Диалог Globally Reset DBMS Property

Прямое и обратное проектирование

Процесс генерации физической схемы БД из логической модели данных называется прямым проектированием (Forward Engineering). 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 (Reverse Engineering). 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/Forward Engineer/Schema Generation или нажать кнопку на панели инструментов. Появляется диалог Schema Generation (рис. 6).

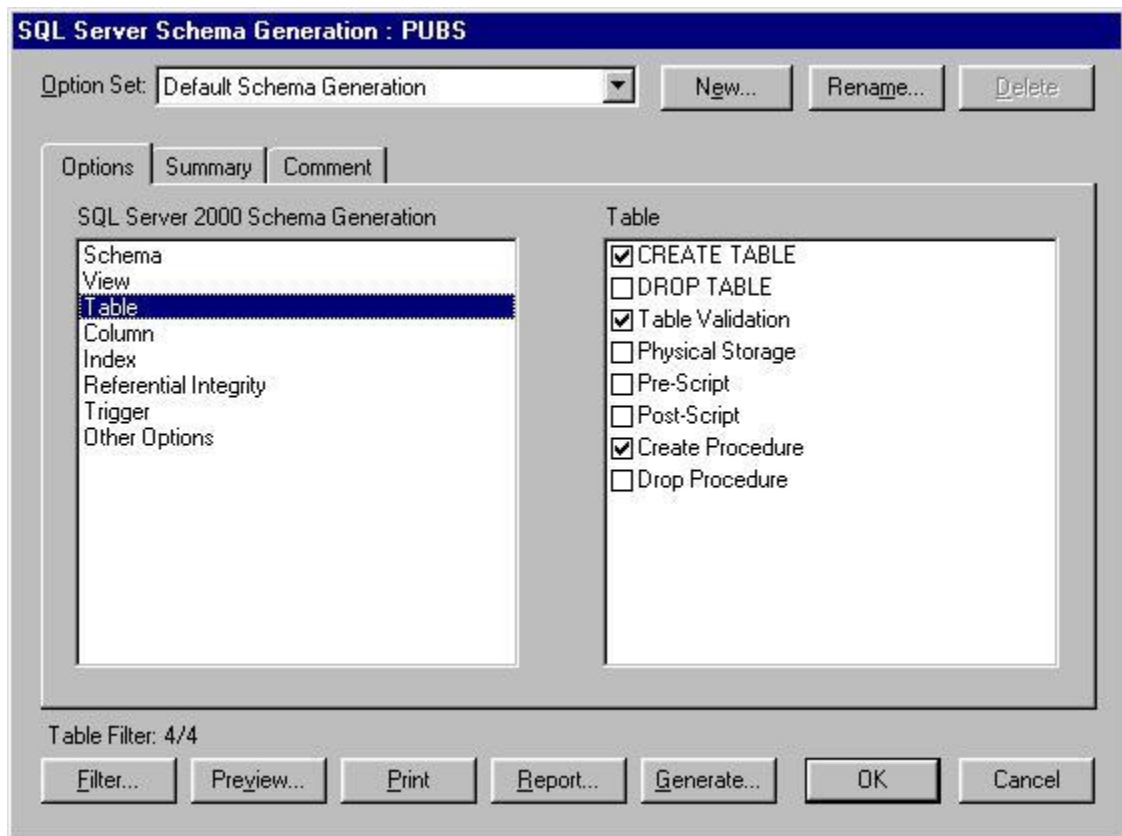


рис.6

Диалог Schema Generation 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 (см. рис. 6).

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 Option Set, кнопки New, Rename и Delete) и использован многократно.

Кнопка **Generate** запускает процесс генерации схемы. Возникает диалог связи с БД (рис. 7), устанавливается сеанс связи с сервером и начинается выполнение SQL-скрипта. При этом возникает диалог Generate Database Schema (рис. 8).

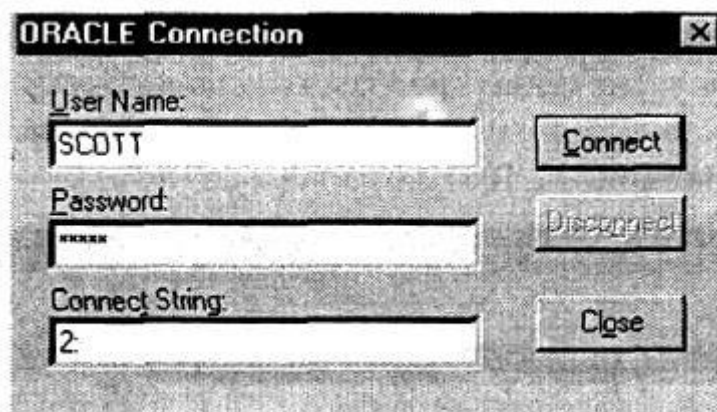


Рис.7. Диалог связи с БД

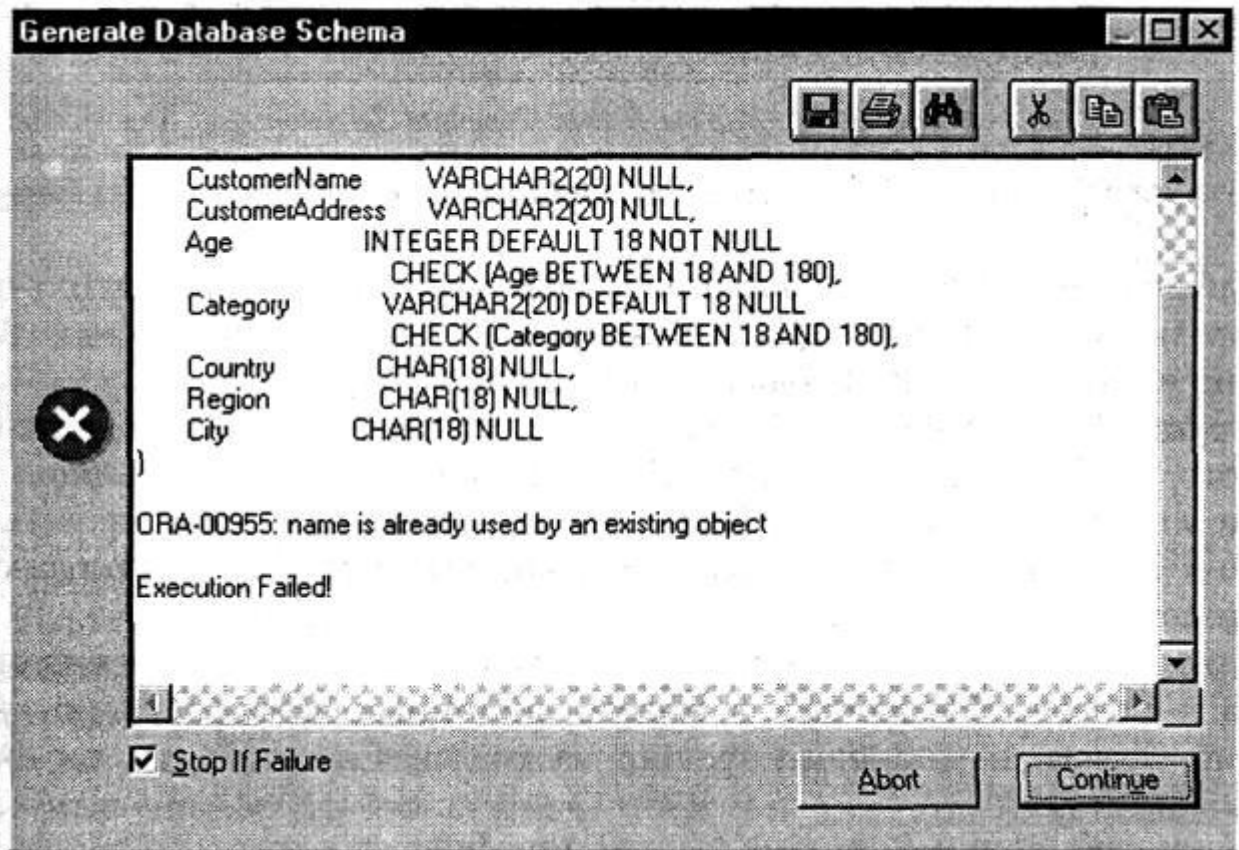


Рис.8. Диалог Generate Database Schema

По умолчанию в диалоге Generate Database Schema включена опция Stop If Failure. Это означает, что при первой же ошибке выполнение скрипта прекращается. Щелкнув по кнопке Continue, можно продолжить выполнение. Кнопка Abort прерывает выполнение. При выключенной опции Stop If Failure скрипт будет выполняться, несмотря на встречающиеся ошибки.

Для выполнения обратного проектирования следует выбрать пункт меню Tasks/Reverse Engineer

После выполнения всех вышеописанных действий получим физическую модель данных, представленную на рис.9 и [SQL - скрипт](#)

:

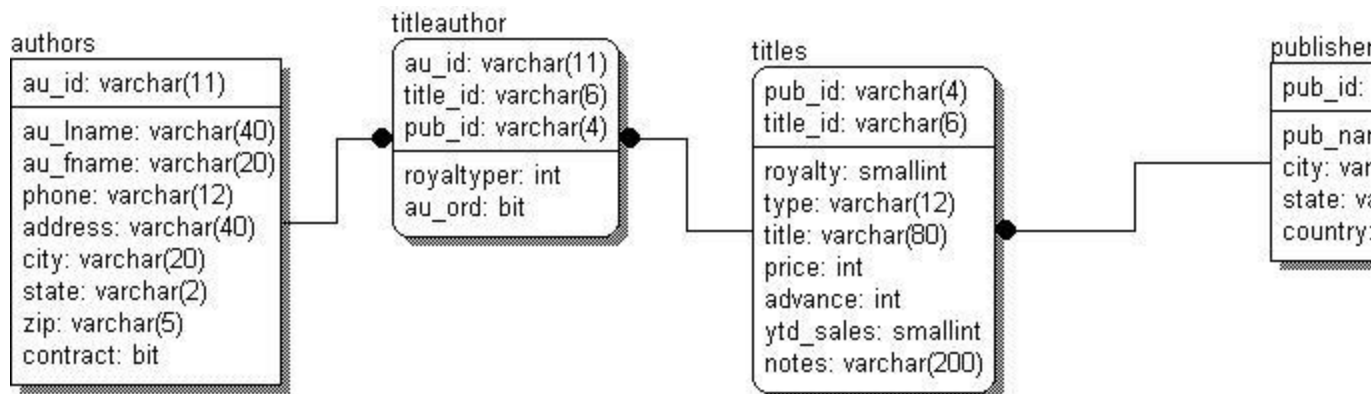


рис.9 Физическая модель данных

Контрольные вопросы

1. Что такое бизнес-процесс?
2. Каковы основные компоненты функциональной модели?
3. Что представляют собой методологии функционального моделирования?
4. Что такое сценарии?

Работа с литературой:

Рекомендуемые источники информации (№ источника)			
Основная	Дополнительная	Методическая	Интернет-ресурсы
1-2	1-3	1-3	1-4

Оценочные средства: отчет к лабораторной работе (См.: Фонд оценочных средств)

Лабораторная работа № 16

«Разработка диалогового интерфейса пользователя предметной области.» Форма проведения: лабораторная работа **Цель работы:**

Научиться разрабатывать диалоговый интерфейс пользователя предметной области.

Пользовательский интерфейс – совокупность программных и аппаратных средств, обеспечивающих взаимодействие пользователя с компьютером. **Основа взаимодействия** – диалоги.

Диалог – регламентированный обмен информацией между человеком и компьютером, осуществляемый в реальном масштабе времени и направленный на совместное решение конкретной задачи: обмен информацией и координация действий. Каждый диалог состоит из отдельных процессов ввода-вывода, которые физически обеспечивают связь пользователя и компьютера.

Обмен информацией осуществляется передачей сообщений и управляющих сигналов.

Сообщение – порция информации, участвующая в диалоговом обмене.

Виды сообщений:

- входные сообщения, которые генерируются человеком с помощью средств ввода: клавиатуры, манипуляторов (мышь, и. т.п.);
- выходные сообщения, которые генерируются компьютером в виде текстов, звуковых сигналов и/или изображений и выводятся пользователю на экран монитора или другие устройства вывода информации.

Пользователь генерирует сообщения типа:

- запрос информации,
- запрос помощи,
- запрос операции или функции,
- ввод или изменение информации,
- выбор поля кадра.

Получает в ответ:

- подсказки или справки,
- **информационные сообщения**, не требующие ответа,
- приказы, требующие действий,

- сообщения об ошибках, нуждающиеся в ответных действиях,
- изменение формата кадра.

Основные устройства, обеспечивающие выполнение операций ввода-вывода.

Для вывода сообщений:

- монохромные и цветные мониторы – вывод оперативной текстовой и графической информации;
- принтеры – получение «твердой копии» текстовой и графической информации;
- графопостроители – получение твердой копии графической информации;
- синтезаторы речи – речевой вывод;
- звукогенераторы – вывод музыки.

Для ввода сообщений:

- клавиатура – текстовый ввод;
- планшеты – графический ввод;
- сканеры – графический ввод;
- манипуляторы, световое перо, сенсорный экран – позиционирование и выбор информации на экране.

Типы интерфейсов

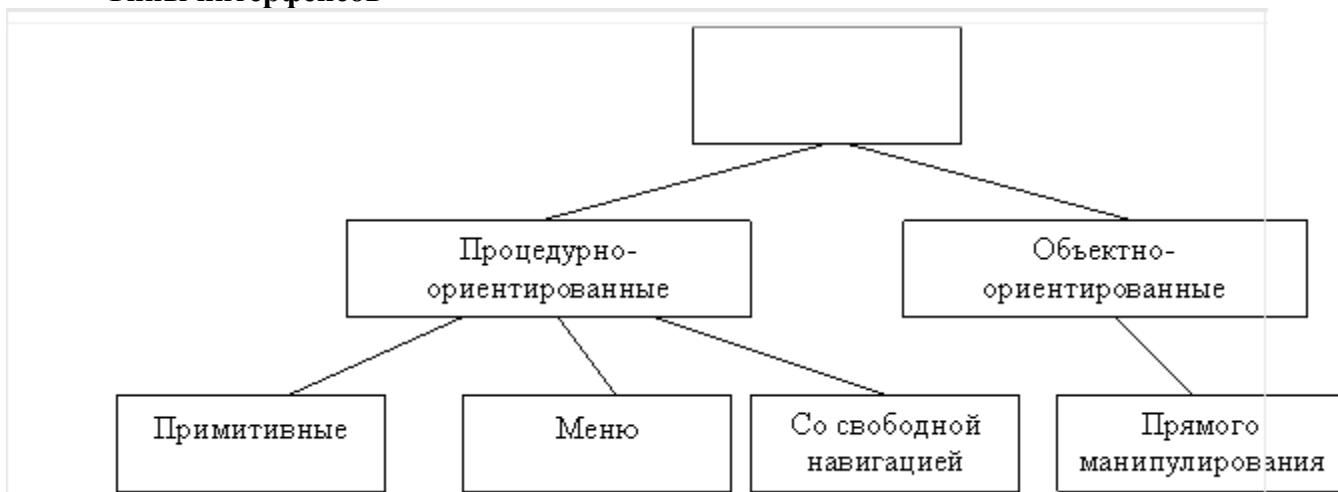


Рисунок 7.1 – Типы интерфейсов

Процедурно-ориентированные интерфейсы используют модель взаимодействия с пользователем, основанную на понятиях «процедура» и «операция». Программное обеспечение предоставляет пользователю возможность выполнения некоторых *действий*, для которых пользователь определяет соответствующие данные и следствием выполнения которых является получение желаемых результатов.

Объектно-ориентированные интерфейсы используют модель взаимодействия с пользователем, ориентированную на манипулирование *объектами* предметной области. Пользователю предоставляется возможность напрямую взаимодействовать с каждым объектом и инициировать выполнение операций, в процессе которых взаимодействуют несколько объектов. Задача пользователя формулируется как целенаправленное изменение некоторого объекта, имеющего внутреннюю структуру, определенное содержание и внешнее символическое или графическое представление. Например, модель реальной системы или процесса, база данных, текст и т. д. Элементы интерфейсов данного типа включены в пользовательский интерфейс Windows. Например, пользователь может «взять» файл и «переместить» его в другую папку. Таким образом, он инициирует выполнение операции перемещения файла.

Примитивным называют интерфейс, который организует взаимодействие с пользователем в консольном режиме. Такой интерфейс реализует конкретный сценарий работы программного обеспечения, например: ввод данных – решение задачи – вывод

результата. Обычно используются при обучении программированию или когда программа реализует одну функцию.

Интерфейс-меню позволяет выбирать необходимые операции из специального списка, выводимого ему программой. Эти интерфейсы предполагают реализацию множества сценариев работы, последовательность действий в которых определяется пользователем. Различают одноуровневые и иерархические меню.

Интерфейсы со свободной навигацией называют *графическими пользовательскими интерфейсами*. Интерфейсы этого типа ориентированы на использование экрана в графическом режиме с высокой разрешающей способностью.

Графические интерфейсы поддерживают концепцию интерактивного взаимодействия с программным обеспечением, осуществляя визуальную обратную связь с пользователем и возможность прямого манипулирования объектами и информацией на экране.

В отличие от интерфейса-меню интерфейс со свободной навигацией обеспечивает возможность осуществления любых допустимых в конкретном состоянии операций, доступ к которым возможен через различные интерфейсные компоненты. Например, окна программ, реализующих интерфейс Windows, обычно содержат:

- меню различных типов: ниспадающее, кнопочное, контекстное;
- разного рода компоненты ввода данных.

Причем выбор следующей операции в меню осуществляется как мышью, так и с помощью клавиатуры.

Особенность интерфейса со свободной навигацией - способность изменяться в процессе взаимодействия с пользователем, предлагая выбор только тех операций, которые имеют смысл в конкретной ситуации. Реализуют интерфейсы со свободной навигацией, используя событийное программирование и объектно-ориентированные библиотеки, что предполагает применение визуальных сред разработки программного обеспечения.

Интерфейс прямого манипулирования предполагает, что взаимодействие пользователя с программным обеспечением осуществляется посредством выбора и перемещения *пиктограмм*, соответствующих объектам предметной области. Для реализации таких интерфейсов также используют событийное программирование и объектно-ориентированные библиотеки.

Этапы разработки пользовательского интерфейса. Разработка пользовательского интерфейса включает те же основные этапы, что и разработка программного обеспечения:

- постановка задачи – определение типа интерфейса и общих требований к нему;
- анализ требований и определение спецификаций – определение сценариев использования и пользовательской модели интерфейса;
- проектирование – проектирование диалогов и их реализация в виде процессов ввода-вывода;
- реализация – программирование и тестирование интерфейсных процессов.

При проектировании пользовательских интерфейсов необходимо учитывать психофизические особенности человека, связанные с восприятием, запоминанием и обработкой информации.

Исследованием принципов работы мозга человека занимается *когнитивная психология*.

Информация о внешнем мире поступает в наш мозг в огромных количествах. Часть мозга, которую условно можно назвать «процессором восприятия», постоянно без участия сознания перерабатывает ее, сравнивая с прошлым опытом, и помещает в хранилище уже в виде зрительных, звуковых и прочих образов. Любые внезапные или просто значимые для нас изменения в окружении привлекают наше внимание, и тогда интересующая нас информация поступает в кратковременную память. Если же наше

внимание не было привлечено, то информация в хранилище пропадает, замещаясь следующими порциями.

В каждый момент времени фокус внимания может фиксироваться в одной точке. Поэтому, если возникает необходимость «одновременно» отслеживать несколько ситуаций, то обычно фокус перемещается с одного отслеживаемого элемента на другой. При этом внимание «рассредоточивается», и какие-то детали могут быть упущены.

Обработка процессором восприятия требует некоторого времени и, если сигнал выдается в течение времени, меньшем времени обработки, то наш мозг его не воспринимает.

Восприятие во многом основано на мотивации. Например, если человек голоден, то он в первую очередь будет замечать все съедобное, а если устал – то, войдя в комнату, он в первую очередь увидит диван или кровать.

В процессе переработки информации мозг сравнивает поступающие данные с предыдущими.

При смене кадра мозг на некоторое время блокируется: он «осваивает» новую картинку, выделяя наиболее существенные детали. А значит, если необходима быстрая реакция пользователя, то резко менять картину не стоит.

Краткосрочная память – самое «узкое» место «системы обработки информации» человека. Ее емкость приблизительно равна 7 ± 2 несвязанных объектов. Не востребованная информация хранится в краткосрочной памяти не более 30 с.

При проектировании интерфейсов следует иметь в виду, что подавляющему большинству людей сложно, например, запомнить и ввести на другом экране число, содержащее более 5 цифр (7-2), или некоторое сочетание букв.

Долговременная память человека – хранилище информации с неограниченной емкостью и временем хранения. Однако доступ к этой информации непрост: по всей вероятности, механизмы извлечения информации из памяти имеют ассоциативный характер. Специальная методика запоминания информации (*мнемоника*) использует именно это свойство памяти: для запоминания информации ее «привязывают» к тем данным, которые память уже хранит и позволяет легко получить.

Поскольку доступ к долговременной памяти затруднен, целесообразно рассчитывать не на то, что пользователь вспомнит нужную информацию, а на то, что он ее узнает. Поэтому интерфейс типа меню так широко используется.

Особенности восприятия цвета. Цвет в сознании человека ассоциируется с эмоциональным фоном. Теплые цвета: красный, оранжевый, желтый человека возбуждают, а холодные: синий, фиолетовый, серый – успокаивают. Причем цвет является очень сильным раздражителем, поэтому применять цвета в интерфейсе необходимо крайне осторожно.

Обилие оттенков привлекает внимание, но быстро утомляет. Поэтому не стоит ярко раскрашивать окна, с которыми пользователь будет долго работать. Необходимо учитывать и индивидуальные особенности восприятия цветов человеком, например, каждый десятый человек плохо различает какие-то цвета, поэтому в ответственных случаях необходимо предоставить пользователю возможность настройки цветов.

Особенности восприятия звука. В интерфейсах звук обычно используют с разными целями: для привлечения внимания, как фон, обеспечивающий некоторое состояние пользователя, как источник дополнительной информации. Следует учитывать, что большинство людей очень чувствительны к звуковым сигналам, особенно, если последние указывают на наличие ошибки. Поэтому при создании звукового сопровождения целесообразно предусматривать возможность его отключения.

Субъективное восприятие времени. Человеку свойственно субъективное восприятие времени. Занятый человек обычно времени не замечает. Зато в состоянии ожидания время тянется бесконечно, что связано с тем, что в это время мозг оказывается в

состоянии информационного вакуума. К аналогичному состоянию приводит усталость: информация поступает, но больше обрабатывается, а потому и ход времени замедляется.

При ожидании более 1-2 с пользователь может отвлечься, «потерять мысль», что неблагоприятно сказывается на результатах работы и увеличивает усталость, так как каждый раз после ожидания много сил тратится на включение в работу.

Сократить время ожидания можно, заняв пользователя, но не отвлекая его от работы. Например, можно предоставить ему какую-либо информацию для обдумывания. По возможности целесообразно выводить пользователю промежуточные результаты: во-первых, он будет занят их обдумыванием, во-вторых, по ним он сможет оценить будущие результаты, и отменит операцию, если они его не удовлетворяют.

Для «развлечения» пользователя используется анимация. *Например, в Windows при копировании файлов демонстрируется «ролик» с летающими листочками. Следует иметь в виду, что, когда какую-либо анимацию смотришь первый раз, то это интересно, а когда в течение получаса наблюдаешь, как «летают» листочки при получении информации из Интернета,, то это начинает раздражать.*

Чтобы уменьшить раздражение, возникающее при ожидании, необходимо соблюдать основное правило: информировать пользователя, что заказанные им операции потребуют некоторого времени выполнения. Для этого используют индикаторы оставшегося времени, анимированные объекты, как в Интернете, и изменение формы курсора мыши на песочные часы. Очень важно точно обозначить момент, когда система готова продолжать работу. Обычно для этого используют значительные изменения внешнего вида экрана.

Взаимодействие пользователя с интерфейсом будет определяться не только *физическими возможностями и особенностями* человека, но и пользовательской моделью интерфейса.

Модели пользовательского интерфейса:

- модель программиста;
- модель пользователя;
- программная модель.

Программист, разрабатывая пользовательский интерфейс, исходит из того, управление какими операциями ему необходимо реализовать, как это осуществить, не затрачивая ни существенных ресурсов компьютера, ни своих сил и времени. Его интересуют функциональность, эффективность, технологичность, внутренняя стройность и другие, не связанные с удобством пользователя характеристики программного обеспечения. Поэтому большинство интерфейсов существующих программ вызывают серьезные нарекания пользователей.

Пользовательская модель интерфейса – это совокупность обобщенных представлений конкретного пользователя или некоторой группы пользователей о процессах, происходящих во время работы программы или программной системы. Эта модель базируется на *особенностях* опыта конкретных пользователей, который характеризуется:

- уровнем подготовки в предметной области разрабатываемого программного обеспечения;
- интуитивными моделями выполнения операций в этой предметной области;
- уровнем подготовки в области владения компьютером;
- устоявшимися стереотипами работы с компьютером.

Для построения пользовательской модели необходимо изучить перечисленные выше особенности опыта предполагаемых пользователей программного обеспечения. С этой целью используют опросы, тесты и даже фиксируют последовательность действий, осуществляемых в процессе выполнения некоторых операций, на пленку.

Приведение в соответствие моделей пользователя и программиста, а также построение на их базе программной модели интерфейса задача непростая.

Интуитивные модели выполнения операций в предметной области должны стать основой для разработки интерфейса, а потому в большинстве случаев их необходимо не менять, а уточнять и совершенствовать.

Критерии оценки интерфейса пользователем. Основные критерии интерфейсов пользователя:

- простота освоения и запоминания операций системы – конкретно оценивают время освоения и продолжительность сохранения информации в памяти;
- скорость достижения результатов при использовании системы – определяется количеством вводимых или выбираемых мышью команд и настроек;
- субъективная удовлетворенность при эксплуатации системы (удобство работы, утомляемость и т. д.).

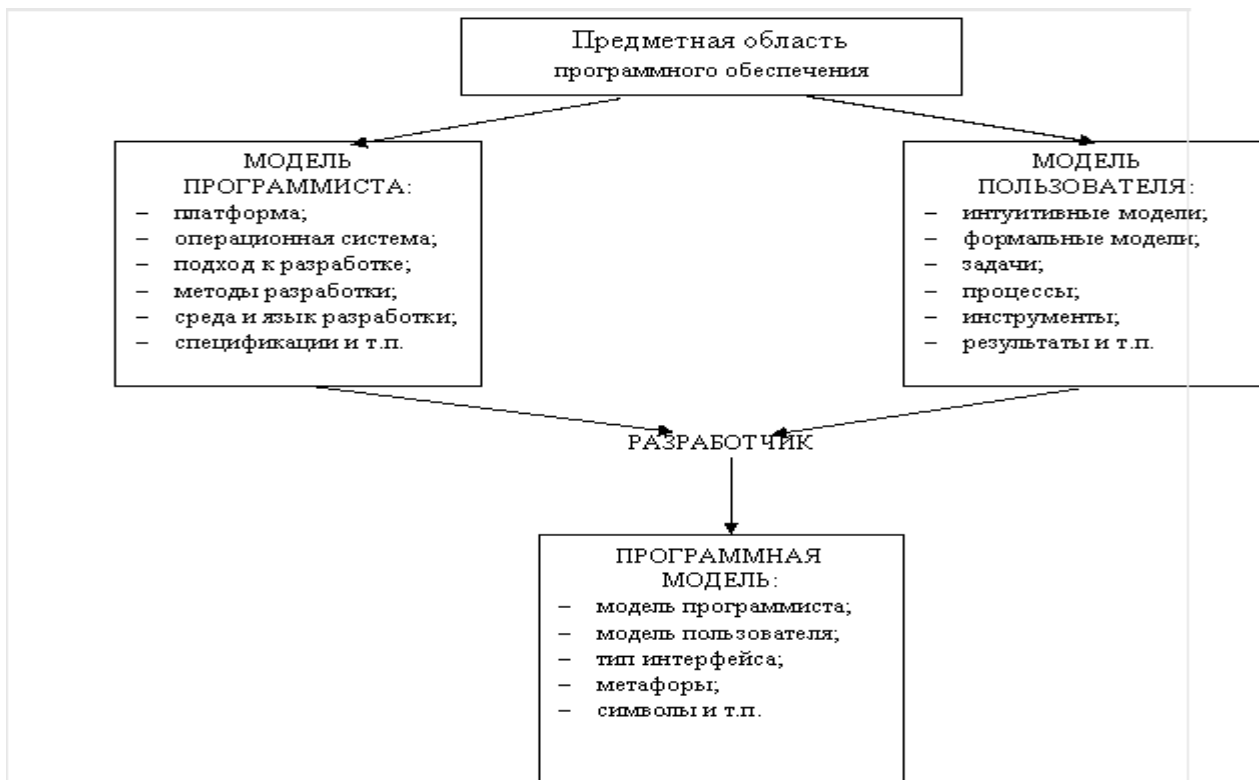


Рисунок 7.2 – Процесс проектирования пользовательского интерфейса

Наилучшими характеристиками для пользователей-профессионалов обладают интерфейсы со свободной навигацией, а для пользователей-непрофессионалов – интерфейсы прямого манипулирования. Замечено, что при выполнении операции копирования файлов большинство профессионалов используют оболочки типа Far, а непрофессионалы – «перетаскивание объектов» Windows.

Типы диалога. Тип диалога определяет, кто из «собеседников» управляет процессом обмена информацией.

Различают два типа диалога:

- управляемые программой;
- управляемые пользователем.

Диалог, *управляемый программой*, предусматривает наличие жесткого, линейного или древовидного, т. е. включающего возможные альтернативные варианты, сценария диалога, заложенного в программное обеспечение. Такой диалог обычно сопровождают большим количеством подсказок, которые уточняют, какую информацию необходимо вводить на каждом шаге.

Диалог, *управляемый пользователем*, подразумевает, что сценарий диалога зависит от пользователя, который применяет систему для выполнения необходимых ему операций. При этом система обеспечивает возможность реализации различных пользовательских сценариев.

Форма диалога. Никакой диалог невозможен, если не существует языка, понятного «собеседникам». Описание языка, на котором ведется диалог, включает определение его *синтаксиса* – правил, определяющих допустимые конструкции (слова, предложения) языка или его форму, и *семантики* – правил, определяющих смысл синтаксически корректных конструкций языка или его содержание.

Различают три формы диалога:

- фразовую,
- директивную,
- табличную.

Фразовая форма предполагает «общение» с пользователем на естественном языке или его подмножестве. Содержание диалога составляют повелительные, повествовательные и вопросительные предложения и ответы на вопросы.

Чаще всего используют диалоги, предполагающие односложные ответы.

Например:

Программа: Введите свой возраст (полных лет):

Пользователь:

При обработке фраз оперируют понятием словоформа.

Словоформа – отрезок текста между двумя соседними пробелами или знаками препинания.

Морфологический анализ - обработка словоформ вне связи с контекстом.

Два метода морфологического анализа:

- **декларативный** – предполагает, что в словаре находятся все возможные словоформы каждого слова, тогда анализ сводится к поиску словоформы в словаре. Данный метод обеспечивает возможность обработки сообщений, состоящих из строчных и прописных букв в произвольной комбинации, при чем как латинского, так и русского или других алфавитов;

- **процедурный** – предполагает выделение в текущей словоформе основу, которую затем идентифицируют.

После распознавания словоформ осуществляют синтаксический анализ сообщения, по результатам которого определяют его синтаксическую структуру, т. е. выполняют разбор предложения.

Далее выполняют *семантический анализ*, т. е. определяют смысловые отношения между словоформами. При этом выделяют главные предикаты, определяющие смысл предложения.

Интерфейс, реализующий фразовую форму диалога, должен: преобразовывать сообщения из естественно-языковой формы в форму внутреннего представления и обратно, выполнять анализ и синтез сообщений пользователя и системы, отслеживать и запоминать пройденную часть диалога.

Недостатки фразовой формы:

- большие затраты ресурсов;
- отсутствие гарантии однозначной интерпретации формулировок;
- необходимость ввода длинных грамматически правильных фраз.

Достоинство фразовой формы – свободное общение с системой.

Директивная форма - использование команд (директив) *специально разработанного формального языка*.

Команда – предложение этого языка, описывающее комбинированные данные, которые включают идентификатор иницируемого процесса и, при необходимости, данные для него.

Команду можно вводить:

- в виде строки текста, специально разработанного формата (команды MS DOS в командной строке);
- нажатием некоторой комбинации клавиш (комбинации «быстрого доступа» Windows-приложений);
- посредством манипулирования мышью («перетаскиванием» пиктограмм);
- комбинацией второго и третьего способов.

Достоинства директивной формы:

- небольшой объем вводимой информации;
- гибкость – возможности выбора операции, ограничивается набором допустимых команд;
- ориентация на диалог, управляемый пользователем;
- использование минимальной области экрана или не использование ее вообще;
- возможность совмещения с другими формами.

Недостатки директивной формы:

- практическое отсутствие подсказок на экране, что требует запоминания вводимых команд и их синтаксиса;
- почти полное отсутствие обратной связи о состоянии инициированных процессов;
- необходимость навыков ввода текстовой информации или манипуляций мышью;
- отсутствие возможности настройки пользователем.

Директивная форма удобна для пользователя-профессионала, который обычно быстро запоминает синтаксис часто используемых команд или комбинации клавиш. Достоинства формы (гибкость и хорошие временные характеристики) проявляются в этом случае особенно ярко.

Табличная форма – пользователь выбирает ответ из предложенных программой. Язык диалога имеет простейший синтаксис и однозначную семантику, что достаточно легко реализовать. Форма удобна для пользователя, так как выбрать всегда проще, что существенно для пользователя-непрофессионала. Эту форму можно использовать, если множество возможных ответов на конкретный вопрос конечно. Если количество возможных ответов велико (более 20), то применение табличной формы может оказаться нецелесообразным.

Достоинства табличной формы:

- наличие подсказки;
- сокращение количества ошибок ввода: пользователь не вводит информацию, а указывает на нее;
- сокращение времени обучения пользователя;
- возможность совмещения с другими формами;
- в некоторых случаях возможность настройки пользователем.

Недостатки табличной формы:

- необходимость наличия навыков навигации по экрану;
- использование сравнительно большой площади экрана для изображения визуальных компонентов;
- интенсивное использование ресурсов компьютера, связанное с необходимостью постоянного обновления информации на экране.

Типы и формы диалога выбирают независимо друг от друга: любая форма применима для обоих типов диалогов.

Синхронные - диалоги, происходящие в процессе нормальной работы программного обеспечения.

Асинхронные – диалоги, возникающие по инициативе системы или пользователя при нарушении сценария нормального процесса. Их используют для выдачи экстренных сообщений от системы или пользователя.

Разработка диалогов. Стадии проектирования и реализации диалогов:

- определение множества необходимых диалогов, их основных сообщений и возможных сценариев – проектирование *абстрактных диалогов*;
- определение типа и формы каждого диалога, а также синтаксиса и семантики используемых языков – проектирование *конкретных диалогов*;
- выбор основных и дополнительных устройств и проектирование процессов ввода-вывода для каждого диалога, а также уточнение передаваемых сообщений – проектирование *технических диалогов*.

Основа абстрактных диалогов – идеология технологического процесса, для автоматизации которого предназначается программный продукт.

Кроме сценариев используют *диаграммы состояний интерфейса* или *графы диалога*.

Граф диалога – ориентированный взвешенный граф, каждой вершине которого сопоставлена конкретная картина на экране (*кадр*) или определенное состояние диалога, характеризующееся набором доступных пользователю действий. Дуги, исходящие из вершин, показывают возможные изменения состояний при выполнении пользователем указанных действий. Дуги, исходящие из вершин, показывают возможные изменения состояний при выполнении пользователем указанных действий. В качестве весов дуг указывают условия переходов из состояния в состояние и операции, выполняемые во время перехода.

Каждый маршрут на графе соответствует возможному варианту диалога.

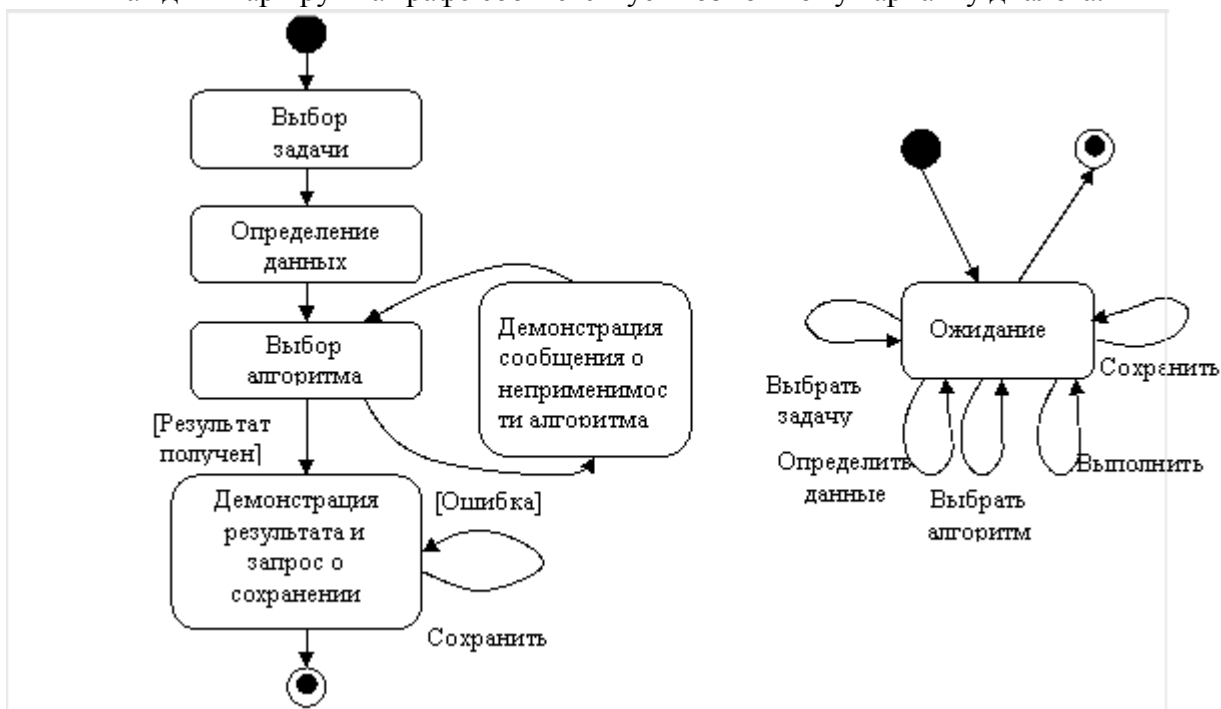


Рисунок 7.3 – Графы абстрактного диалога:

а – диалог, управляемый системой; б – диалог, управляемый пользователем

Графические пользовательские интерфейсы поддерживаются операционными системами Windows, Apple Macintosh, OS/2 и т. д. Для таких интерфейсов разработаны наборы стандартных компонентов взаимодействия с пользователем для каждой операционной системы.

Интерфейсы строятся по технологии WIMP: W – Windows (окна), I – Icons (пиктограммы), M – Mouse (мышь), P – Pop-up (всплывающие или выпадающие меню). Основные элементы графических интерфейсов: окна, пиктограммы, компоненты ввода-

вывода и мышью, которую используют в качестве указывающего устройства и устройства прямого манипулирования объектами на экране.

Окна. *Окно* – прямоугольная, ограниченная рамкой область физического экрана. Окно может менять размеры и местоположение в пределах экрана.

5 категорий окон:

- основные окна (окна приложений);
- дочерние или подчиненные окна;
- окна диалога;
- информационные окна;
- окна меню.

Окно приложения Windows содержит: рамку, ограничивающую рабочую область окна, строку заголовка с кнопкой системного меню и кнопками выбора представления окна и выхода, строку меню, пиктографическое меню (панель инструментов), горизонтальные и вертикальные полосы прокрутки и строку состояния.

Дочернее окно Windows используют в многодокументных программных интерфейсах (MDI). Это окно не содержит меню. В строке заголовка – специальное имя, идентифицирующее связанный с ним документ или файл. Пиктограммы всех дочерних окон одинаковы.

Диалоговое окно Windows используют для просмотра и задания различных режимов работы, необходимых параметров или другой информации.

Оно может содержать:

- строку заголовка с кнопкой системного меню;
- компоненты, обеспечивающие пользователю возможность ввода или выбора ответа;
- вспомогательные компоненты, обеспечивающие подсказку (поле просмотра или кнопка справки).

Размер окна не изменяем, но по экрану его можно перемещать.

Информационные окна двух типов:

- окна сообщений;
- окна помощи.

Окна сообщений содержат: заголовок с кнопкой системного меню, текст сообщения, одна или несколько кнопок реакции пользователя (Yes, No, Cancel).

Окно помощи содержит: меню, полосы прокрутки, информационная область, аналогично окну приложения, но имеет узкоспециальное назначение.

Окна меню Windows используют как открывающиеся панели иерархического меню или как контекстные меню.

Каждой строке окна меню может соответствовать:

- команда;
- меню следующего уровня, что обеспечивается стрелкой;
- окно диалога, что обозначается тремя точками.

Добавляется указание клавиш быстрого вызова.

Пиктограммы. Пиктограмма – небольшое окно с графическим изображением, отражающим содержимое буфера, с которым она связана.

Виды пиктограмм:

- программные, связанные с соответствующей программой;
- пиктограммы дочерних окон, обеспечивающие доступ к различным документам;
- пиктограммы панели инструментов, дублируют доступ к соответствующим функциям через меню, обеспечивая их быстрый доступ;
- пиктограммы объектов, для прямого манипулирования объектами.

Прямое манипулирование изображением. *Прямое манипулирование изображением* – это возможность замены команды воздействия на некоторый объект физическим действием в интерфейсе, осуществляемым с помощью мыши. При этом

любая область экрана рассматривается как адресат, который может быть активизирован при подведении курсора и нажатии клавиши мыши.

По реакции на воздействие различают типы адресатов:

- указание и выбор (развертывание пиктограмм, определение активного окна);
- буксировка и «резиновая нить» (перенос объекта или его границ);
- экранные кнопки и «скользящие» барьеры (выполнение или циклически повторяемых действий (выполнение некоторых операций или рисование, подразумеваемых при активизации определенной области экрана - кнопки)).

Динамический визуальный сигнал - изменение изображения на экране (курсор мыши при выполнении конкретных операций, изменения изображения кнопки).

Компоненты ввода-вывода. Интерфейсы включают несколько меню: основное или «ниспадающее» иерархическое меню, пиктографические меню (панели инструментов) и контекстные меню для разных ситуаций. Любое из указанных меню представляет собой компонент ввода-вывода, реализующий диалог с пользователем, используя табличную форму.

Иерархическое меню используют, чтобы организовать выполняемые программным обеспечением операции, если их число превышает в соответствии с рекомендациями фирмы IBM), и обеспечить пользователю их обзор. Панели инструментов и контекстное меню применяют для обеспечения быстрого доступа к часто используемым командам, обеспечивая пользователю возможность относительно свободной навигации.

Другие формы ввода-вывода:

- фразовая,
- табличная,
- смешанная.

Диалоги обоих типов:

- управляемые пользователем,
- управляемые системой.

Реализация диалогов, управляемых пользователем. Для реализации применяют меню различных видов:

- основное,
- панели инструментов,
- контекстные и кнопочные.

Как альтернативу меню целесообразно использовать директивную форму диалога, поставив в соответствие основным командам определенные комбинации клавиш. Целесообразно предусмотреть возможность управления меню клавиатурой, если большую часть времени работы с системой пользователь вводит текст или данные, т. е. взаимодействует с клавиатурой.

Меню. Меню проектируют на основе графов диалогов разрабатываемого программного обеспечения. Если число операций не превышает 5, то обычно используют кнопки. Если число операций не более 9-10, то – одноуровневое меню. Если число операций более 10, то используют «ниспадающее» двухуровневое иерархическое меню.

Ниспадающее меню. Первый уровень иерархического меню должен содержать имена основных групп операций.

Традиционно (обычно в текстовых и графических редакторах):

1. пункт Файл,
 2. пункт Правка,
 3. пункт Вид,
- последний пункт Справка.

Количество уровней иерархического меню не должно превышать 2-3 (сложно искать). Число операций в окне не должно превышать 7-8 операций.

Если число операций превышает 70-80. Разработчики Microsoft Word предложили *адаптивное* иерархическое меню, где содержимое окна меню второго уровня постоянно

меняется, отображая только те операции, которые использует пользователь. Если пользователь не находит нужной операции, то через несколько секунд или при нажатии специальной кнопки Word демонстрирует окно меню полностью.

Возможность прямого манипулирования, предусмотренная в WIMP интерфейсах, позволяет разрабатывать для приложений объектно-ориентированные интерфейсы прямого манипулирования.

Интерфейсы используют директивную форму диалога: ввод команды осуществляется при выполнении определенных действий с пиктограммой объекта мышью. Основными элементами этих интерфейсов являются: метафоры, объекты, представления объектов и технологии Drag and Drop («перетаскил и бросил»).

Метафоры. *Метафоры* – мысленный перенос свойств или признаков одного объекта на другой, чем-то аналогичный первому. Использование метафор в интерфейсах предполагает активизацию имеющегося у пользователя опыта.

Интерфейс прямого манипулирования должен обеспечить пользователю среду, содержащую знакомые элементы, с которыми пользователь не раз встречался в профессиональной деятельности или в быту, и предоставлять ему возможность манипулирования отдельными объектами. (Метафора “Выбрасывание мусора” – для удаления файлов).

Похожие элементы должны вести себя похожим образом, элементы, выделенные одним цветом, должны находиться в определенной связи друг с другом.

Целесообразно не делать изображения слишком реалистичными, чтобы не обмануть ожидания пользователя.

Метафоры и анимация. При реализации метафор все большая роль уделяется средствам мультимедиа, в основном анимации. Используя мультипликацию, можно не только развлекать пользователя, но и “готовить” его к смене кадров, сокращая время, необходимое для адаптации к изменившейся ситуации.

Программа, реализующая анимационные интерфейсы, никогда не простаивает, так как во время ожидания ввода команды пользователя она продолжает отображать соответствующие кадры. В основе таких программ лежит *временное программирование*. В отличие от событийного программирования, которое позволяет связывать изображение на экране с внешними и внутренними событиями в системе, временное программирование обеспечивает изменение проецируемой *последовательности кадров* в зависимости от состояния [моделируемых процессов](#) и действий пользователя.

Объекты интерфейса прямого манипулирования и их представления.

Три основные типа объектов интерфейсов прямого манипулирования:

- объекты-данные,
- объекты контейнеры,
- объекты устройства.

Объекты-данные снабжают пользователя информацией (тексты, изображения, электронные таблицы, музыка, видео). В рамках операционной системы таким объектам соответствуют приложения, которые запускаются при раскрытии объекта.

Объекты-контейнеры могут манипулировать своими внутренними объектами, в том числе и другими контейнерами (копировать их или сортировать в любом порядке). К типичным контейнерам относятся папки, корзины. При раскрытии контейнера демонстрируются сохраняемые им компоненты, и появляется возможность ими манипулировать. Компоненты могут обозначаться пиктограммами или представляться в виде таблицы.

Объекты-устройства представляют устройства, существующие в реальном мире: телефоны, факсы, принтеры и т. д. Их используют для обозначения этих устройств в абстрактном мире интерфейса. При раскрытии такого объекта можно увидеть его настройки.

Каждому объекту соответствует одно окно. В исходном состоянии это окно представлено пиктограммой, но при необходимости его можно раскрыть и выполнить требуемые операции, например настройки объекта. Окно объекта в раскрытом состоянии может содержать меню и панели инструментов. Пиктограмме же должно соответствовать контекстное меню, содержащее перечень операций над объектом.

Имя пиктограммы формируют по своему для каждого типа объектов. Пиктограммам объектов-данных присваивают имена, соответствующие именам хранимых данных, а тип данных кодируется самой пиктограммой. Имя пиктограммы-контейнера или пиктограммы устройства обозначает сам объект, а потому не зависит от содержимого.

Различие между типами объектов является условным, так как один и тот же объект в разных ситуациях может вести себя то, как объект-данные, то, как объект-устройство, то, как объект-контейнер (принтер – объект-устройство, может обладать свойствами объекта-контейнера, может содержать объекты-данные в очереди на печать; представление в виде пиктограммы, окна очереди на печать, окна настроек; имя представления целесообразно указывать в заголовке окна объекта).

Технология Drag and Drop. Основные принципы прямого манипулирования, описанные в руководстве по разработке пользовательских интерфейсов фирмы IBM:

- результат перемещения объекта должен соответствовать ожиданиям пользователя;
- пользователи не должны неожиданно терять информацию;
- пользователь должен иметь возможность отменить неправильное действие.

Основные принципы [визуализации](#) операции прямого манипулирования:

- исходное выделение – используется в качестве обратной связи пользователю, чтобы сообщить ему, что объект захвачен, в Windows с этой целью используется выделение цветом;
- визуализация перемещения – используется для идентификации выполняемого действия;
- целевое выделение – используется для идентификации пункта назначения, показывая, таким образом, куда «упадет» объект, если его отпустить в текущий момент времени;
- визуализация действия – используется для обозначения времени ожидания завершения операции, обычно с этой целью применяют анимацию или изменение формы курсора на «песочные часы».

Существует два вида пунктов назначения: один принимает объект, а другой его копию (Пользователь «бросает» документ в «корзину» – уничтожается сам документ, а если на принтер, то передается копия документа).

Проектирование интерфейсов прямого манипулирования. Проектирование выполняется на основе графов диалога, разработанных для конкретного программного обеспечения, и включает следующие процедуры:

- формирование *множества объектов предметной области*, которое должно быть представлено на экране, причем в качестве основы в этом случае используют не варианты использования, а концептуальную модель предметной области;
- анализ *объектов*, определение их *типов и представлений*, а также перечня операций с этими объектами;
- уточнение *взаимодействия объектов* и построение матрицы *прямого манипулирования*;
- определение *визуальных представлений* объектов;
- разработка *меню окон объектов и контекстных меню*;
- создание *прототипа* интерфейса;
- тестирование на *удобство использования*.

Интеллектуальные элементы пользовательских интерфейсов

Элементы пользовательских интерфейсов: Мастер, Советчик, Агент. Сделано множество попыток создания социализированного пользовательского интерфейса. В основе такого интерфейс лежит идея создания персонифицированного, т. е. «имеющего личность», интерфейса. Развлекающие программы, такие как Cats(кошки) и Dogs(собаки), реализующие сложное поведение домашних животных в разных ситуациях, показывают, что технически это вполне решаемая задача.

Советчики. Представляют собой форму подсказки. Их можно вызвать с помощью меню справки, командной строки окна или из всплывающего меню. Советчики помогают пользователям в выполнении конкретных задач.

Мастера. Программу-мастер используют для выполнения общераспространенных, но редко выполняемых отдельным пользователем задач (установка программ или оборудования). Выполнение подобных действий требует от пользователя принятия сложных взаимосвязанных решений, последовательность которых диктует программа-мастер. Интеллектуальные Мастера способны на каждом шаге демонстрировать в окне просмотра результаты ответов пользователя на предыдущие вопросы, помогая последнему сориентироваться в ситуации.

Мастер реализует последовательный или древовидный сценарий диалога. Его целесообразно использовать для решения хорошо структурированных, последовательных задач.

При этом необходимо:

- предоставить пользователю возможность возврата на предыдущий шаг;
- предусмотреть возможность отмены работы Мастера;
- нумеровать шаги и сообщать пользователю количество шагов Мастера, особенно, если таких шагов больше трех;
- пояснить пользователю каждый шаг;
- по возможности демонстрировать результат уже выполненных операций на каждом шаге.

Программные агенты. Используются для выполнения рутинной работы. Основными функциями Агентов-Помощников являются: наблюдение, поиски управление. Различают:

программы-агенты, настраиваемые на выполнение указанных задач; программы-агенты, способные обучаться (фиксируя действия пользователя (по типу магнитофона)).

Порядок выполнения лабораторной работы

По заданному преподавателем описанию предметной области:

1. Изучить основные аспекты предметной области.
2. Описать контекст использования системы.
3. Описать бизнес-роли пользователей.
4. Составить сценарии работы пользователей в виде схемы алгоритма, описывающие весь процесс использования системы для выполнения конкретной задачи.
5. Разработать полную схему экранов системы.
6. Спроектировать абстрактные диалоги системы в виде графа диалога.
7. Определить типы и формы каждого диалога.

Контрольные вопросы

- 1 Стадии проектирования и реализации диалогов.
2. Типы диалога.
3. Технология Drag and Drop.
4. Форма диалога.

Работа с литературой:

Рекомендуемые источники информации (№ источника)			
Основная	Дополнительная	Методическая	Интернет-ресурсы
1-2	1-3	1-3	1-4

Оценочные средства: отчет к лабораторной работе (См.: Фонд оценочных средств)

Лабораторная работа № 17

«Разработка мониторинговых форм и документов отображения аналитической и отчетной информации. Разработка регламента формирования БД и обработки информации.»

Форма проведения: **лабораторная работа**

Цель работы:

Целью данной лабораторной работы является разработка мониторинговых форм и документов отображения аналитической и отчетной информации. Знакомство с разработкой регламента формирования БД и обработки информации.

Access предоставляет возможность вводить данные как непосредственно в таблицу, так и с помощью форм. Форма в БД - это структурированное окно, которое можно представить так, чтобы оно повторяло форму бланка. Формы создаются из набора отдельных элементов управления.

Внешний вид формы выбирается в зависимости от того, с какой целью она создается. Формы Access позволяют выполнять задания, которые нельзя выполнить в режиме таблицы. Формы позволяют вычислять значения и выводить на экран результат. Источником данных для формы являются записи таблицы или запроса.

Форма предоставляет возможности для:

- ввода и просмотра информации базы данных;
- изменения данных;
- печати;
- создания сообщений.

Способы создания форм:

1. Конструктор форм (предназначен для создания формы любой сложности).
2. Мастер форм (позволяет создавать формы различные как по стилю, так и по содержанию).
3. Автоформа: в столбец (многостраничная – поля для записи выводятся в один столбец, в форме одновременно отображаются данные для одной записи).
4. Автоформа: ленточная (все поля записи выводятся в одну строку, в форме отображаются все записи).
5. Автоформа: табличная (отображение записей осуществляется в режиме таблицы).
6. Автоформа: сводная таблица.
7. Автоформа: сводная диаграмма.
8. Диаграмма (создается форма с диаграммой, построенной Microsoft Graph).
9. Сводная таблица (создается форма Access, отображаемая в режиме сводной таблицы Excel).

Алгоритм создания форм следующий:

1. Открыть окно БД.
2. В окне БД выбрать вкладку Формы.
3. Щелкнуть на пиктограмме Создать, расположенной на панели инструментов окна БД.

4. В появившемся диалоговом окне «Новая форма» Выбрать способ создания формы и источник данных.

5. Щелкнуть на кнопке ОК.

Создание формы с помощью Мастера

Вызвать Мастер форм можно несколькими способами. Один из них – выбрать Мастер форм в окне диалога Новая форма и щелкнуть на кнопке ОК. Откроется окно диалога Создание форм, в котором необходимо отвечать на вопросы каждого текущего экрана Мастера и щелкать на кнопке Далее.

В первом окне необходимо выбрать поля из источника данных (таблиц или запросов). Для этого надо открыть список Таблицы и запросы, щелкнув на кнопку, справа. Затем доступные поля требуется перевести в Выбранные поля, выделив их и щелкнув на кнопку >>.

Например, выберем источник – таблицу Студенты и все ее поля, а затем необходимо щелкнуть на кнопке Далее.

В этом окне надо выбрать внешний вид формы, например в один столбец и щелкнуть Далее.

После выбора стиля формы (например, официальный), требуется перейти в последнее окно, щелкнув на кнопке Далее. В последнем окне Мастера требуется ввести имя формы и указать дальнейшие действия: Открыть форму для просмотра и ввода данных; Изменить макет формы.

После ввода имени формы (например, Студенты), выбора режима: «Открыть форму для просмотра и ввода данных» и щелчка на кнопке Готово, получим следующую форму для ввода и просмотра записей в таблицу Студенты.

Создание формы с помощью Конструктора

Для создания формы Студенты необходимо выполнить следующие действия:

1. Запустить программу Microsoft Access и открыть БД.

2. В окне БД выбрать вкладку Формы. Выполнить щелчок по кнопке Создать. Появится диалоговое окно Новая форма. В этом окне необходимо выбрать из списка пункт Конструктор. Затем в списке "Выберите в качестве источника данных таблицу или запрос" выбрать имя таблицы (например, Студент). Выполнить щелчок по кнопке ОК. На экране появится окно Форма 1.

3. Если на экране отсутствует список полей выбранной для построения формы таблицы, выбрать пункт меню Вид / Список полей.

4. Поля из списка переместить на форму (по одному или предварительно выделив с использованием клавиши Shift и мыши, для выделения всех полей выполнить двойной щелчок мышью на заголовке окна Список полей).

5. Разместить поля на форме в нужных местах по разработанному образцу.

6. Перемещение полей и их имен по форме производится следующим образом:

- выделить объект (поле с именем) щелчком мыши. Вокруг него появятся маркеры перемещения и изменения размеров. Перемещать поле можно вместе с привязанным к нему именем или отдельно от него;

- для перемещения поместить указатель мыши на квадратик, находящийся в левом верхнем углу элемента. Указатель мыши в виде ладони позволяет перемещать объект вместе с привязанным к нему именем, в виде ладони с вытянутым указательным пальцем - перемещает один объект;

- нажать кнопку мыши и, удерживая ее, буксировать поле или его имя в нужное место в форме. Затем отпустить кнопку мыши;

- для изменения надписи, связанной с полем необходимо выполнить на ней двойной щелчок мышью. В открывшемся диалоговом окне Надпись выбрать вкладку Макет и выполнить необходимые изменения. Затем закрыть окно;

- для изменения размеров поместить курсор на размерные маркеры, при этом курсор примет вид двунаправленной стрелки. Нажать кнопку мыши, буксировать в нужном направлении, затем отпустить кнопку мыши;

- для удаления поля выделить его, нажать клавишу Delete или выбрать команду Правка / Удалить.

7. Сохранить форму, выбрав из меню Файл команду Сохранить как, и в открывшемся окне выбрать режим сохранения «в текущей базе данных», затем щелчок по кнопке ОК.

8. Просмотреть форму в режиме Конструктора, выполнив щелчок по кнопке Открыть.

9. Если вид формы не удовлетворяет, открыть форму в режиме Конструктор и внести необходимые изменения, затем сохранить форму Файл — Сохранить или выполнить щелчок по пиктограмме Сохранить.

Порядок выполнения лабораторной работы

1. Создать сложные формы для ввода, редактирования и отображения данных БД “Продажи”. Каждая форма используется для отображения данных из двух связанных таблиц. Для ввода и отображения данных из таблиц, находящихся на стороне отношения ∞ (многие), использовать подчиненные ленточные формы. Для ввода и отображения данных из таблиц, находящихся на стороне отношения 1 (один) использовать формы в один столбец.

2. Создать макросы для открытия форм, созданных в п.1 данной работы, и выполнения запросов, созданных в п. 7.1 и 7.2 лабораторной работы 6.

3. Создать форму управления, содержащую кнопки выполнения запросов и открытия форм. Установить необходимые свойства формы. Выводить на экран эту форму при открытии базы данных “Продажи”.

Создание формы

Для создания формы нужно выполнить следующие действия:

1. перейти на вкладку **Формы**;
2. щелкнуть по кнопке **Создать**;
3. в диалоговом окне «Новая форма» выбрать режим создания формы и источник данных (таблицу или запрос).

Для помещения в форму конкретного элемента управления нужно:

1. нажать соответствующую кнопку на панели элементов;
2. поместить указатель в то место формы, где будет находиться левый верхний угол ЭУ.
3. нарисовать мышью на форме прямоугольную рамку для размещения ЭУ.

Создание сложной формы

1. создать с помощью *Мастера форм* форму (главную) в один столбец для главной таблицы (на стороне отношения 1);
2. создать с помощью *Мастера форм* ленточную форму (подчиненную) для подчиненной таблицы (на стороне отношения ∞);
3. открыть главную форму в режиме Конструктора;
4. включить в текущую форму подчиненную форму.

Создание кнопочной формы управления

1. создать макросы для открытия *Форм* и *Запросов*;
2. перейти на вкладку *Формы*;
3. выбрать режим *Создание формы в режиме конструктора*;
4. разместить на форме необходимое количество ЭУ *Кнопка* для открытия созданных форм и запросов;
5. изменить текст на кнопке в соответствии ее назначением;

6. для каждой кнопки в окне *Свойств* на вкладке *События* выбрать событие *Нажатие кнопки* и выбрать для реакции на него нужный макрос.
7. Установить свойства формы управления на вкладке *Макет*:
 - *Область выделения* – нет;
 - *Кнопки перехода* – нет;
 - *Разделительные линии* – нет.

Создание простого макроса

1. в окне базы данных выбрать вкладку *Макросы* и нажать кнопку *Создать*;
2. в окне макроса выбрать из раскрывающегося списка столбца «Макрокоманда» необходимую макрокоманду (например, *Открыть форму* или *Открыть Запрос*);
3. задать аргументы этой макрокоманды (*Имя формы* или *Имя запроса*)
4. сохранить макрос под соответствующим именем.

Связывание макросов с формами

1. открыть форму в режиме конструктора;
2. выделить необходимый элемент (например, кнопку), с реакцией на событие которого (например, нажатие кнопки), будет связано выполнение макроса;
3. в окне свойств этого элемента управления выбрать вкладку *События*;
4. в раскрывающемся списке для строки, соответствующей выбранному событию (например, нажатию кнопки), выбрать имя нужного макроса;
5. сохранить изменения в форме.

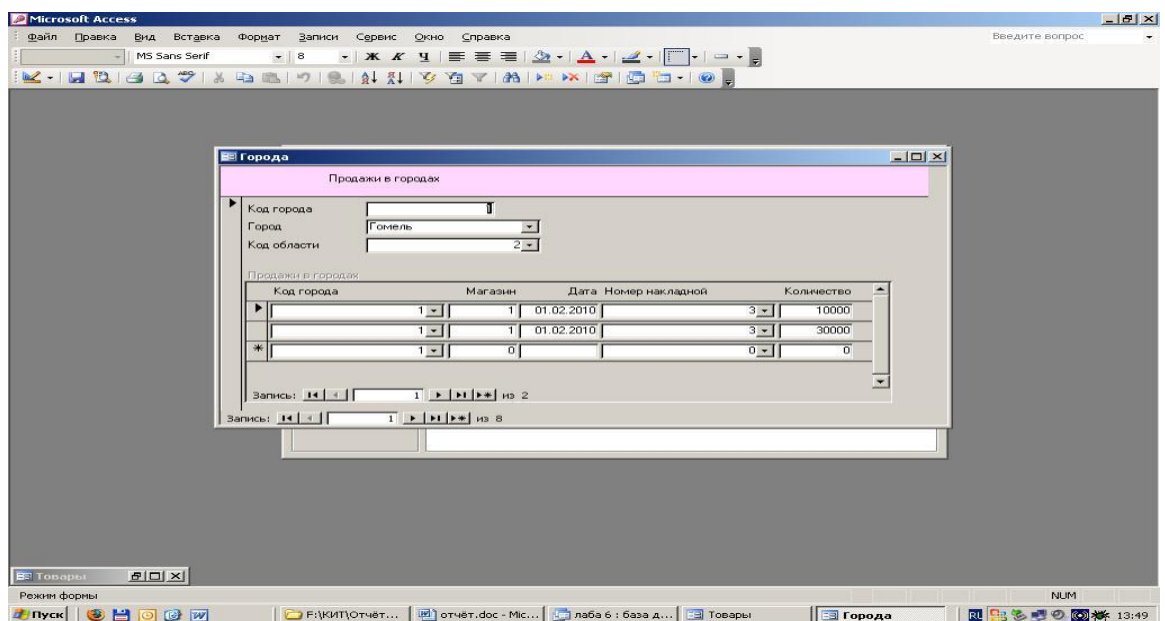


Рисунок 8.1

Накладные

Номер накладной:
 Код товара:
 Цена:

Код города	Магазин	Дата	Номер накладной	Количество
2	2	02.02.2010	1	2000
5	2	04.02.2009	1	55000
0	0		1	0

Записи: 1 из 2
 Запись: 1 из 5

Избранное

Рисунок 8.2

Области

Код области:
 Область:

Код города	Город
4	Брест
5	Кобрин
(Счетчик)	

Запись: 1 из 2
 Запись: 1 из 4

Рисунок 8.3

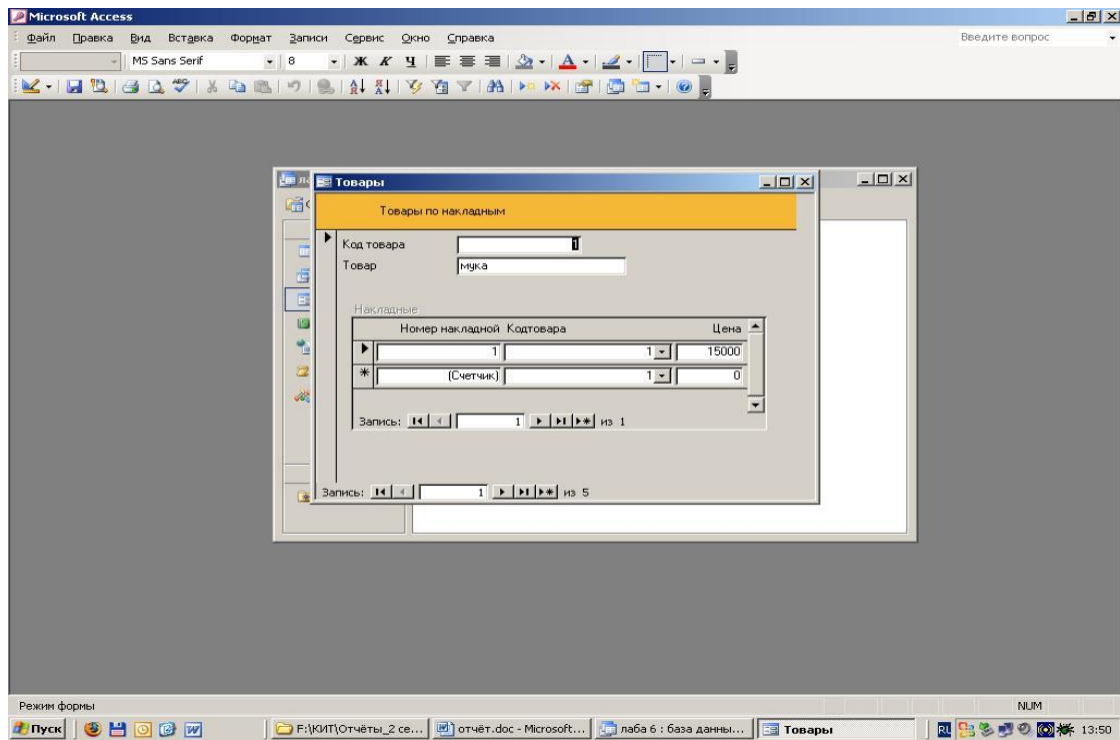


Рисунок 8.4

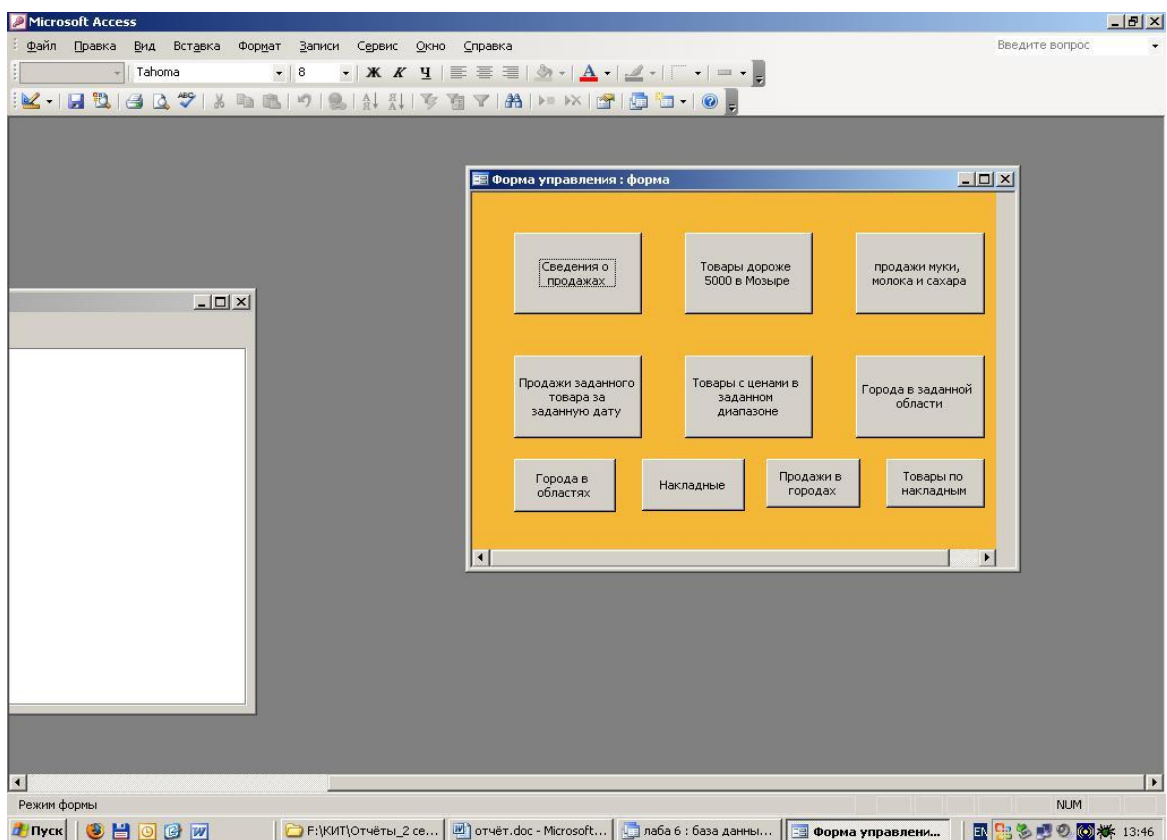


Рисунок 8.5

Регламент формирования БД и обработки информации может быть сформирован с помощью ТПР (типовые проектные решения). ТПР в области ЭИС представляет комплект технической документации, содержащей проектное решение по части объекта проектирования и предназначенные для многократного использования в процессе

разработки, внедрения и функционирования ЭИС с целью уменьшения трудоёмкости и разработки, затрат на создание ЭИС.

ТПР разрабатывают при наличии однородных объектов управления, для которых создание ТПР является экономически целесообразным.

ТПР делится на следующие виды:

- простые
- комбинированные

Простые ТПР охватывают вид обеспечения, комбинированный два и более .

Объектами проектирования для простых ТПР являются:

- ТПР по информационному обеспечению (БД ,классификаторы технико-экономической и нормативно-справочной информации)
- ТПР по программному обеспечению(программы общего и специального назначения)
- ТПР по организационному обеспечению(инструкции определяющих функции управления)
- ТПР по постановке задачи

Общие требования предъявляемые к ТПР:

1. ТПР должен обеспечивать возможность их использования при создании ЭИС ,при минимальном участии разработчиков ТПР

2.Сроки применения ТПР должны быть увязаны со сроками определяемые обновлением используемых в ЭИС технических и программных средств и нормативно технических документов

3. ТПР используемая в конкретном ЭИС должна удовлетворять требованиям стандартов, распространяемых на ЭИС и её части

ТПР должен удовлетворять одному или нескольким из следующих свойств:

6. Обладать способностью удовлетворять все возможные потребности в рамках своей функциональной ориентации

7. Допускать адаптацию к конкретным условиям применение путём изменения параметров

8. Допускать возможность выбора нужной комбинации ТПР в любом конкретном применении

9. Обладать возможностью адаптации к различным техническим средствам

ТПР может подразделяться на три класса:

1. Класс задачи- охватывает комплекс алгоритмов и программ реализующих определенные функции управления. ТПР этого класса разделяется на подклассы ,которые в свою очередь состоят из комплексов задач(по модульному принципу).При этом предусматривается разработка модулей алгоритмов и модулей программ.

ТПР по задачам имеет следующие особенности:

- а) Они построены по модульному принципу
- б) Они обладают настраиваемостью программ, т.е. возможностью выбора определённых алгоритмов из числа заложенных в ТПР решений
- в) ТПР по задачам снабжена развитой системой диагностики широкого класса ошибок возможных при генерировании и эксплуатации рабочих программ

2.Класс техника- определяет состав ,порядок размещения и использования технических средств и содержит два вида решений:

- Методические рекомендации по составу, порядку, размещению технических средств

- типовые схемы применения периферийных технических средств

3.Класс персонал- предназначен для регламентации действий персонала в нормативных условиях функционирования ЭИС .В их задачи входит обеспечение персонала должностными инструкциями с указанием их прав, обязанности и ответственности.

ОСОБЕННОСТИ на этапах (разработка концепции ЭИС, тех задание, тех проект, раб док-ия) производится оценка функциональной полноты используемых ТПР. Возможно 3 варианта: 1 применение ТПР без доработки 2 модификация ТПР за счет включения дополнит модулей 3 разработка оригинальных программных модулей. Анализ функц полноты: выбор ТПР “задача” в последовательности: на основании обследования предприятия сравнивается имеющаяся ТПР с перечнем задач, анализ входных, выходных форм.

Порядок выполнения лабораторной работы

По заданному преподавателем описанию предметной области разработать регламент формирования БД и обработки информации (использовать результаты предыдущих работ).

Контрольные вопросы

1. Для чего используется Конструктор форм.
2. Для чего используется Мастер форм.
3. Какие возможности предоставляет Форма.
4. Особенности ТПР по задачам.
5. Общие требования предъявляемые к ТПР.
6. Классы ТПР.

Работа с литературой:

Рекомендуемые источники информации (№ источника)			
Основная	Дополнительная	Методическая	Интернет-ресурсы
1-2	1-3	1-3	1-4

Оценочные средства: отчет к лабораторной работе (См.: Фонд оценочных средств)

Лабораторная работа № 18

«Разработка организационно-технической и эксплуатационной документации АИТ предметной области.»

Форма проведения: **лабораторная работа**

Цель работы:

Целью данной лабораторной работы является разработка организационно-технической и эксплуатационной документации АИТ предметной области.

Общие требования к документированию

Документы должны быть представлены на бумажном виде (оригинал) и на магнитном носителе (копия). Исходные тексты программ - только на магнитном носителе (оригинал). Возможно предоставление комплекта документации и текстов программ на компакт-дисках.

Все документы должны быть оформлены на русском языке. Состав документов на общее программное обеспечение, поставляемое в составе АИС, должен соответствовать комплекту поставки компании - изготовителя.

Перечень подлежащих разработке документов

В ходе создания Подсистемы должен быть подготовлен и передан Заказчику комплект документации в составе:

- проектная документация и материалы техно-рабочего проекта на разработку Подсистемы;

- конструкторская, программная и эксплуатационная документация на Подсистему;
- сопроводительная документация на поставляемые программно-аппаратные средства в комплектности поставки заводом-изготовителем;
- предложения по организации системно-технической поддержки функционирования Подсистемы.

Состав и содержание комплекта документации на Подсистему может быть уточнен на стадии проектирования.

Подготовленные документы должны удовлетворять требованиям государственных стандартов и рекомендаций по оформлению, содержанию, форматированию, использованию терминов, определений и надписей, обозначений программ и программных документов.

Порядок выполнения лабораторной работы

По заданному преподавателем описанию предметной области разработать организационно-техническую и эксплуатационную документацию АИТ предметной области (использовать результаты предыдущих работ).

Контрольные вопросы

1. Каковы общие требования к документированию.
2. Каков перечень подлежащих разработке документов.

Работа с литературой:

Рекомендуемые источники информации (№ источника)			
Основная	Дополнительная	Методическая	Интернет-ресурсы
1-2	1-3	1-3	1-4

Оценочные средства: отчет к лабораторной работе (См.: Фонд оценочных средств)

6. КРИТЕРИИ ОЦЕНИВАНИЯ КОМПЕТЕНЦИЙ

Оценка «отлично» выставляется студенту, если он продемонстрировал глубокие, исчерпывающие знания и творческие способности в понимании, изложении и использовании учебно-программного материала; логически последовательные, содержательные, полные, правильные и конкретные ответы на все поставленные вопросы и дополнительные вопросы преподавателя; свободное владение основной и дополнительной литературой, рекомендованной учебной программой.

Оценка «хорошо» выставляется студенту, если он продемонстрировал твердые и достаточно полные знания всего программного материала, правильное понимание сущности и взаимосвязи рассматриваемых процессов и явлений; последовательные, правильные, конкретные ответы на поставленные вопросы при свободном устранении замечаний по отдельным вопросам; достаточное владение литературой, рекомендованной учебной программой.

Оценка «удовлетворительно» выставляется студенту, если он продемонстрировал твердые знания и понимание основного программного материала; правильные, без грубых ошибок ответы на поставленные вопросы при устранении неточностей и несущественных ошибок в освещении отдельных положений при наводящих вопросах преподавателя; недостаточное владение литературой, рекомендованной учебной программой.

Оценка «неудовлетворительно» выставляется студенту, если он продемонстрировал неправильные ответы на основные вопросы, допущены грубые

ошибки в ответах, непонимание сущности излагаемых вопросов; неуверенные и неточные ответы на дополнительные вопросы.

7. МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ, ОПРЕДЕЛЯЮЩИЕ ПРОЦЕДУРЫ ОЦЕНИВАНИЯ ЗНАНИЙ, УМЕНИЙ, НАВЫКОВ И (ИЛИ) ОПЫТА ДЕЯТЕЛЬНОСТИ, ХАРАКТЕРИЗУЮЩИХ

7.1 ЭТАПЫ ФОРМИРОВАНИЯ КОМПЕТЕНЦИЙ

Текущая аттестация студентов проводится преподавателями, ведущими лабораторные занятия по дисциплине, в следующей форме: отчет письменный по заданию преподавателя.

Допуск к лабораторным работам происходит при наличии у студентов печатного варианта отчета. Защита отчета проходит в форме доклада студента по выполненной работе и ответов на вопросы преподавателя.

Отчет включает в себя следующие разделы: титульный лист с названием работы; цель работы; краткие теоретические сведения; описание результатов лабораторной работы (скриншоты); вывод из работы, включающий в себя описание проделанной работы.

Оценку «отлично» студент получает, если оформление отчета соответствует установленным требованиям, правильно отвечает на предложенные преподавателем контрольные вопросы, правильно отвечает на дополнительные вопросы по теме лабораторной работы.

Оценку «хорошо» студент получает, если оформление отчета соответствует установленным требованиям, правильно отвечает на предложенные преподавателем контрольные вопросы.

Оценку «удовлетворительно» студент получает без беседы с преподавателем, если оформление отчета соответствует установленным требованиям.

Отчет может быть отправлен на доработку в следующих случаях:

- полностью не соответствует установленным требованиям;
- не раскрыта суть работы.

8. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

8.1. Рекомендуемая литература

8.1.1. Основная литература

1. Основы информационных технологий [Электронный ресурс]/ С.В. Назаров [и др.]. Электрон. текстовые данные - М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. - 530 с. - Режим доступа: <http://www.iprbookshop.ru/52159>.
2. Богданова С.В. Информационные технологии [Электронный ресурс]: учебное пособие для студентов высших учебных заведений/ Богданова С.В., Ермакова А.Н. - Электрон. текстовые данные. - Ставрополь: Ставропольский государственный аграрный университет, Сервисшкола, 2014. - 211 с. - Режим доступа: <http://www.iprbookshop.ru/48251>

8.1.2. Дополнительная литература:

1. Веретехина С.В. Информационные технологии. Пакеты программного обеспечения общего блока «ИТ-инструментарий» [Электронный ресурс]: учебное пособие/ Веретехина С.В., Веретехин В.В.— Электрон. текстовые данные.— М.: Русайнс,

2015.— 44 с.— Режим доступа: <http://www.iprbookshop.ru/48895>.

2. Избачков Ю.С., Петров В.Н., Васильев А.А., Телина И. С. Информационные системы: Учебник для ВУЗов. 3-е изд. – СПб.: Питер, 2011

112

3. Вдовенко Л.А. Информационная система предприятия; Учебное пособие. М.: Вузовский учебник; ИНФРА-М. 2011. – 237 с.

8.1.3. Методическая литература:

1. Методические указания по выполнению лабораторных работ по дисциплине «Информационные технологии».
2. Методические рекомендации для студентов по организации самостоятельной работы по дисциплине «Информационные технологии».
3. Методические указания к контрольной работе по дисциплине «Информационные технологии».

8.1.4. Интернет-ресурсы:

1. Университетская библиотека online. <http://www.biblioclub.ru>.
2. ЭБС «IPRbooks». <http://www.iprbookshop.ru>.
3. Электронная библиотека СКФУ.. <http://catalog.ncstu.ru>.
4. Государственная публичная научно- техническая библиотека России. (ГПНТБ России). www.gpntb.ru.