ФГБОУ ВО БРЯНСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ

ЭКОНОМИЧЕСКИЙ ФАКУЛЬТЕТ

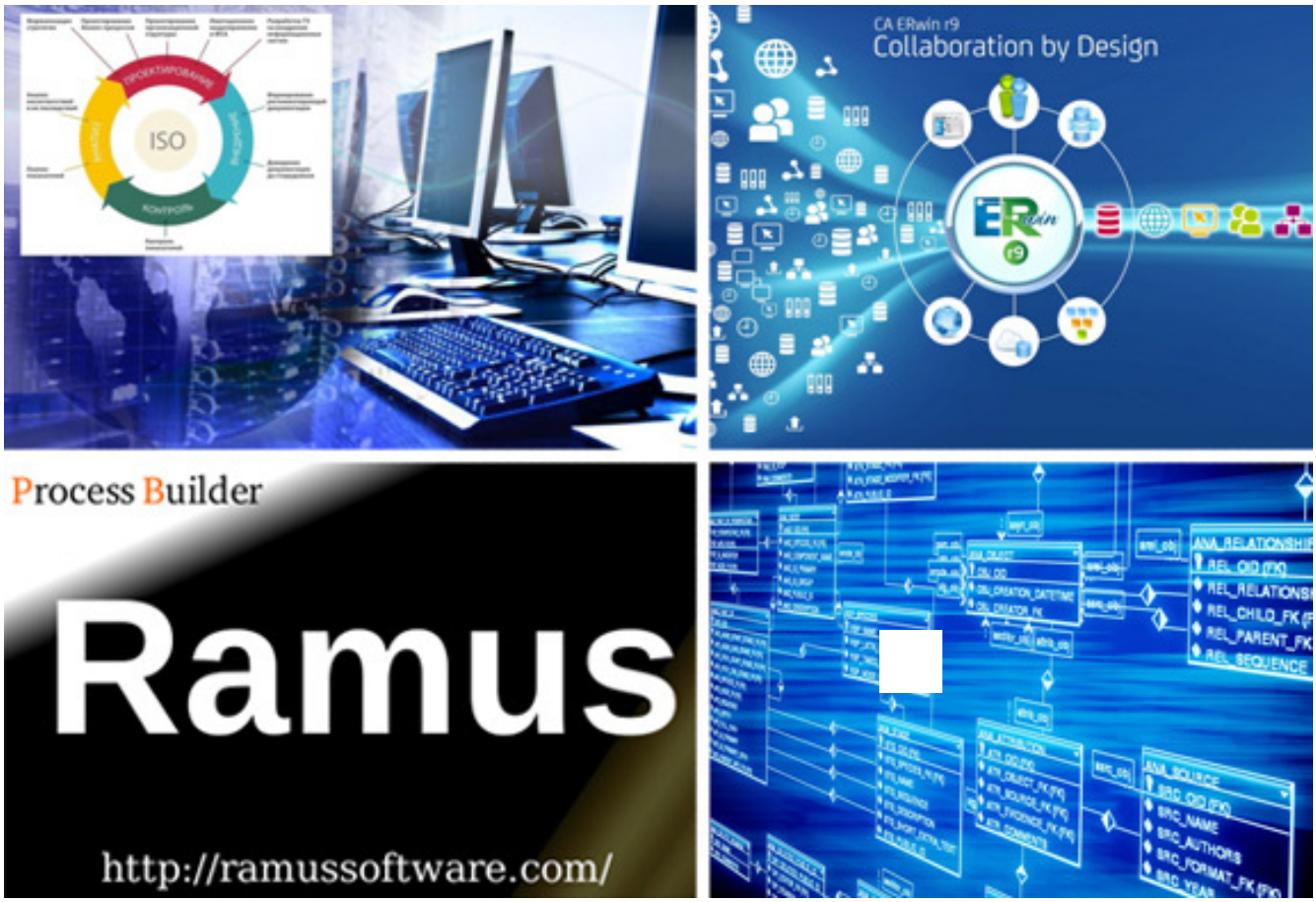
КАФЕДРА ИНФОРМАЦИОННЫХ СИСТЕМ И ТЕХНОЛОГИЙ

ВОЙТОВА Н.А.

Методические указания

к выполнению лабораторных работ

по дисциплине «Проектирование информационных систем»



Брянская область
2015

УДК 004.415.2

Методические указания к выполнению лабораторных работ по дисциплине «Проектирование информационных систем»: методические указания / Сост.: Войтова Н.А. – Брянск: Издательство Брянский ГАУ, 2015. –56 с.

В методических указаниях раскрывается структура и содержание лабораторных работ по дисциплине «Проектирование информационных систем».

Издание окажет помощь студентам профиля Прикладная информатика в экономике при выполнении лабораторных работ по дисциплине «Проектирование информационных систем».

Рекомендовано к изданию Учебно-методическим Советом экономического факультета БГАУ (протокол № 7 от 28.04.2015 г.).

Рецензенты: старший преподаватель кафедры информационных систем и технологий Бишутина Людмила Ивановна

Содержание

ВВЕДЕНИЕ

Пособие направлено на изучение современных методов и средств проектирования информационных систем. Предусматривается изучение CASE-средств, как программного инструмента поддержки проектирования информационных систем (ИС). Курс предусматривает изучение: состава и структуры различных классов экономических ИС как объектов проектирования; современных технологий проектирования ИС и методик обоснования эффективности их применения; содержания стадий и этапов проектирования ИС и их особенностей при использовании различных технологий проектирования; целей и задач проведения предпроектного обследования объектов информатизации; методов моделирования информационных процессов предметной области; классификацию и общие характеристики современных CASE-средств.

Научной основой курса являются методологии системного анализа и моделирования, позволяющие на этапе создания информационной системы решить следующие основные задачи:

- обеспечение требуемой функциональности системы и адаптивности к изменяющимся условиям ее функционирования;
- проектирование реализуемых в системе объектов данных;
- проектирование программ и средств интерфейса (экранных форм, отчетов), которые будут обеспечивать выполнение запросов к данным;
- учет конкретной среды или технологии реализации проекта, а именно: топологии сети, конфигурации аппаратных средств, используемой архитектуры, параллельной обработки, распределенной обработки данных и т.п.

Программой курса предусматривается изучение CASE-инструментов поддержки проектирования информационных систем. Практикум дисциплины включает в себя задания для освоения учащимися инструментальных средств разработки и анализа функциональных и информационных моделей деятельности экономических объектов (предприятий и учреждений), являющихся основой проектирования информационных систем.

Дисциплина «Проектирование информационных систем» имеет целью ознакомить учащихся с информационными технологиями анализа сложных систем и основанными на международных стандартах методами проектирования информационных систем, обучить студентов принципам построения функциональных и информационных моделей систем, проведению анализа полученных результатов, применению инструментальных средств поддержки проектирования экономических информационных систем.

Компетенции, формируемые в результате освоения дисциплины «Проектирование информационных систем»:

ПК-1: способностью проводить обследование организаций, выявлять информационные потребности пользователей, формировать требования к информационной системе.

ПК-4: способностью документировать процессы создания информационных систем на стадиях жизненного цикла.

ПК-9: способностью составлять техническую документацию проектов автоматизации и информатизации прикладных процессов.

ПК-21: способностью проводить оценку экономических затрат и рисков при создании информационных систем.

Ramus - кроссплатформенная система моделирования и анализа бизнес-процессов

Ramus - инструмент для моделирования бизнес-процессов и автоматизированной разработки системы регламентирующей документации, кроссплатформенное программное обеспечение, предназначенное для построения систем управления предприятием.

Ramus полностью поддерживает методологию моделирования бизнес-процессов IDEF0 и DFD, а так же имеет ряд дополнительных возможностей призванных удовлетворить потребности команд разработчиков систем управления предприятиями.

Данный программный продукт создан с целью стать основным инструментом бизнес-аналитиков в проектах по построению или реорганизации систем управления предприятием. К таковым могут относиться: проекты по реинженирингу бизнес-процессов, проекты внедрения процессного управления, проекты построения системы менеджмента качества, проекты построения системы управления знаниями и т.п.

Основными возможностями Ramus являются:

- моделирование процессов (согласно нотаций IDEF0 и DFD);
- разработка систем классификации и кодирования предприятия с внутренними перекрёстными связями, которая также тесно увязывается и с моделями процессов;
- формирование отчётности по моделям и системе классификации, в том числе и отчётности в форме такой регламентирующей документации как должностные инструкции и регламенты процессов;
- генерация сайта, который призван обеспечить доступ к данным моделей процессов, системы классификации и кодирования а также к разнообразнейшей отчётности через веб-интерфейс.

Ramus имеет редактор диаграмм **IDEF0** и **DFD** эргономичность которого находится на уровне не ниже чем у аналогичных продуктов имеющих схожие редакторы. Это проявляется в более лёгкой и быстрой навигации по модели, в более «умном» поведении объектов диаграмм, в поддержке шаблонов диаграмм, в возможности быстрого исправления допущенных ошибок, в том числе и в возможности отмены действий.

Так как, модели процессов реальных предприятий могут содержать многие тысячи разнообразнейших объектов (документы, персонал, функции и т.д.), то в Ramus предусмотрена возможность упорядочено хранить информацию об этих объектах в виде системы классификаторов.

Классификатор в Ramus – это систематизированный перечень наименований объектов с присвоенными кодами.

Классификация объектов значительно упрощает поиск и обработку информации об объектах модели, а так же и об объектах непосредственно на диаграммах процессов не представленных, но, так или иначе, относящихся к процессам предприятия.

Каждый элемент системы классификации, кроме собственно названия, может иметь дополнительные атрибуты, в которых можно упорядочено хранить разнообразнейшую информацию об объекте.

Стоит отметить, что для создания качественной и информативной отчётности по модели, крайне необходимо, чтобы вся информация проекта содержалась упорядочено в виде системы классификации.

Для генерации отчётности в Ramus присутствует редактор отчётности.

Наличествует поддержка шаблонов отчётов в формате XML которые могут быть экспортированы из файла или импортированы в файл.

Совокупность моделей, классификаторов, матричных проекций и отчётов имеющих отношение к одному и тому же предприятию в дальнейшем будем называть Проект.

Просмотр всей информации Проекта может быть осуществлён через веб-браузер. Для этого разработан веб-сервер который выводит информацию Проекта в виде набора HTML страниц, или же, попросту говоря, в виде сайта.

Это существенно упрощает использование и развёртку Ramus, так как избавляет от необходимости установки клиентской версии Ramus на АРМах пользователей, которые имеют доступ только на чтение информации Проекта. Всей или некоторой информации Проекта, что определяется настройками прав доступа.

К любому элементу системы классификации и кодирования можно прикреплять файлы, которые будут доступны для скачивания с сайта Проекта.

Использование технологии Java, при реализации программных модулей, позволяет использовать Ramus под разными видами операционных систем и аппаратных платформ (MS Windows, Linux, Mac OS, и т.д.…).

Ramus может использоваться в **файловом (локальном)** и **сетевом вариантах**.

Сетевая версия Ramus позволяет распределять доступ пользователей к данным.

Сетевая версия Ramus использует стандартизированные протоколы обмена данными, что позволяет интегрировать Ramus с другими системами.

Но и без использования сетевой версии можно разделить работу над Проектом между несколькими разработчиками путём использования функции расщепления Проекта.

В Ramus включена поддержка нескольких языков графического

интерфейса пользователя. Язык интерфейса зависит от региональных настроек операционной системы.

Кроме всего прочего, Ramus поддерживает возможность расширения функциональности с использованием сценариев на языке программирования JavaScript.

Стандарт IDEF0

IDEF0 — Function Modeling — методология функционального моделирования и графическая нотация, предназначенная для формализации и описания бизнес-процессов. Отличительной особенностью IDEF0 является её акцент на соподчинённость объектов. В IDEF0 рассматриваются логические отношения между работами, а не их временная последовательность.

Основные элементы и понятия IDEF0

Графический язык IDEF0 удивительно прост и гармоничен. В основе методологии лежат четыре основных понятия.

Первым из них является понятие **функционального блока** (Activity Box). Функциональный блок графически изображается в виде прямоугольника и олицетворяет собой некоторую конкретную функцию в рамках рассматриваемой системы. По требованиям стандарта название каждого функционального блока должно быть сформулировано в глагольном наклонении (например, «производить услуги», а не «производство услуг») или отглагольным существительным (например, автоматизация учёта услуг)

Типы функциональных блоков в Ramus:

1) комплекс процессов;
2) процесс;
3) под-процесс;
4) операция;
5) действие.

Каждая из четырех сторон функционального блока имеет своё определенное значение (роль), при этом:

1. Верхняя сторона имеет значение «Управление» (Control).
2. Левая сторона имеет значение «Вход» (Input).
3. Правая сторона имеет значение «Выход» (Output).
4. Нижняя сторона имеет значение «Механизм» (Mechanism).

Каждый функциональный блок в рамках единой рассматриваемой системы должен иметь свой уникальный идентификационный номер.

Вторым «китом» методологии IDEF0 является понятие **интерфейсной дуги** (Arrow). Также интерфейсные дуги часто называют потоками или

стрелками. Интерфейсная дуга отображает элемент системы, который обрабатывается функциональным блоком или оказывает иное влияние на функцию, отображенную данным функциональным блоком.

Графическим отображением интерфейсной дуги является однонаправленная стрелка. Каждая интерфейсная дуга должна иметь свое уникальное наименование (Arrow Label). По требованию стандарта, наименование должно быть оборотом существительного.

В IDEF0 различают пять типов стрелок:

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

Вход (Input) — материал или информация, которые используются или преобразуются работой для получения результата (выхода). Допускается, что работа может не иметь ни одной стрелки входа. Каждый тип стрелок подходит к определенной стороне прямоугольника, изображающего работу, или выходит из нее. Стрелка входа рисуется как входящая в левую грань работы. Очень часто сложно определить, являются ли данные входом или управлением. В этом случае подсказкой может служить информация о том, перерабатываются/изменяются ли данные в работе или нет. Если изменяются, то, скорее всего, это вход, если нет — управление.

Выход (Output) — материал или информация, которые производятся работой. Каждая работа должна иметь хотя бы одну стрелку выхода. Работа без результата не имеет смысла и не должна моделироваться. Стрелка выхода

рисуется как исходящая из правой грани работы.

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

Вызов (Call) — специальная стрелка, указывающая на другую модель работы. Стрелка вызова рисуется как исходящая из нижней грани работы. Стрелка вызова используется для указания того, что некоторая работа выполняется за пределами моделируемой системы.

С помощью интерфейсных дуг отображают различные объекты, в той или иной степени определяющие процессы, происходящие в системе. Такими

объектами могут быть элементы реального мира (детали, вагоны, сотрудники и т. д.) или потоки данных и информации (документы, данные, инструкции и т. д.).

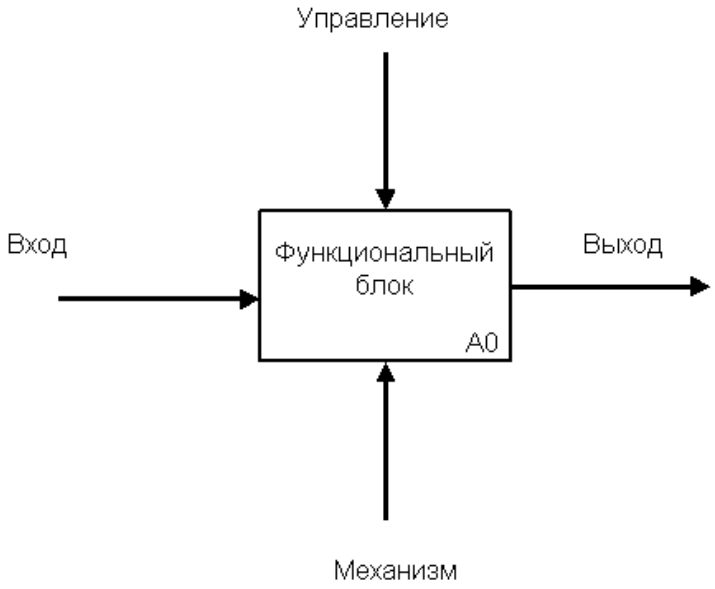


Рис. 1. Функциональный блок

В зависимости от того, к какой из сторон подходит данная интерфейсная дуга, она носит название «входящей», «исходящей» или «управляющей». Кроме того, «источником» (началом) и «приемником» (концом) каждой функциональной дуги могут быть только функциональные блоки. Необходимо отметить, что любой функциональный блок по требованиям стандарта должен иметь по крайней мере одну управляющую интерфейсную дугу и одну исходящую.

Обязательное наличие управляющих интерфейсных дуг является одним из главных отличий стандарта IDEF0 от других методологий классов DFD (Data Flow Diagram) и WFD (Work Flow Diagram).

Третьим основным понятием стандарта IDEF0 является **декомпозиция** (Decomposition). Принцип декомпозиции применяется при разбиении сложного процесса на составляющие его функции. При этом уровень детализации процесса определяется непосредственно разработчиком модели.

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

В процессе декомпозиции, функциональный блок, который в контекстной диаграмме отображает систему как единое целое, подвергается детализации на другой диаграмме. Получившаяся диаграмма второго уровня содержит функциональные блоки, отображающие главные подфункции функционального блока контекстной диаграммы и называется дочерней (Child diagram) по

отношению к нему (каждый из функциональных блоков, принадлежащих дочерней диаграмме соответственно называется дочерним блоком – Child Box). В свою очередь, функциональный блок - предок называется родительским блоком по отношению к дочерней диаграмме (Parent Box), а диаграмма, к которой он принадлежит – родительской диаграммой (Parent Diagram). Каждая из подфункций дочерней диаграммы может быть далее детализирована путем аналогичной декомпозиции соответствующего ей функционального блока.

Важно отметить, что в каждом случае декомпозиции функционального блока все интерфейсные дуги, входящие в данный блок, или исходящие из него фиксируются на дочерней диаграмме. Этим достигается структурная целостность IDEF0 – модели.

Наглядно принцип декомпозиции представлен на рис. 2.

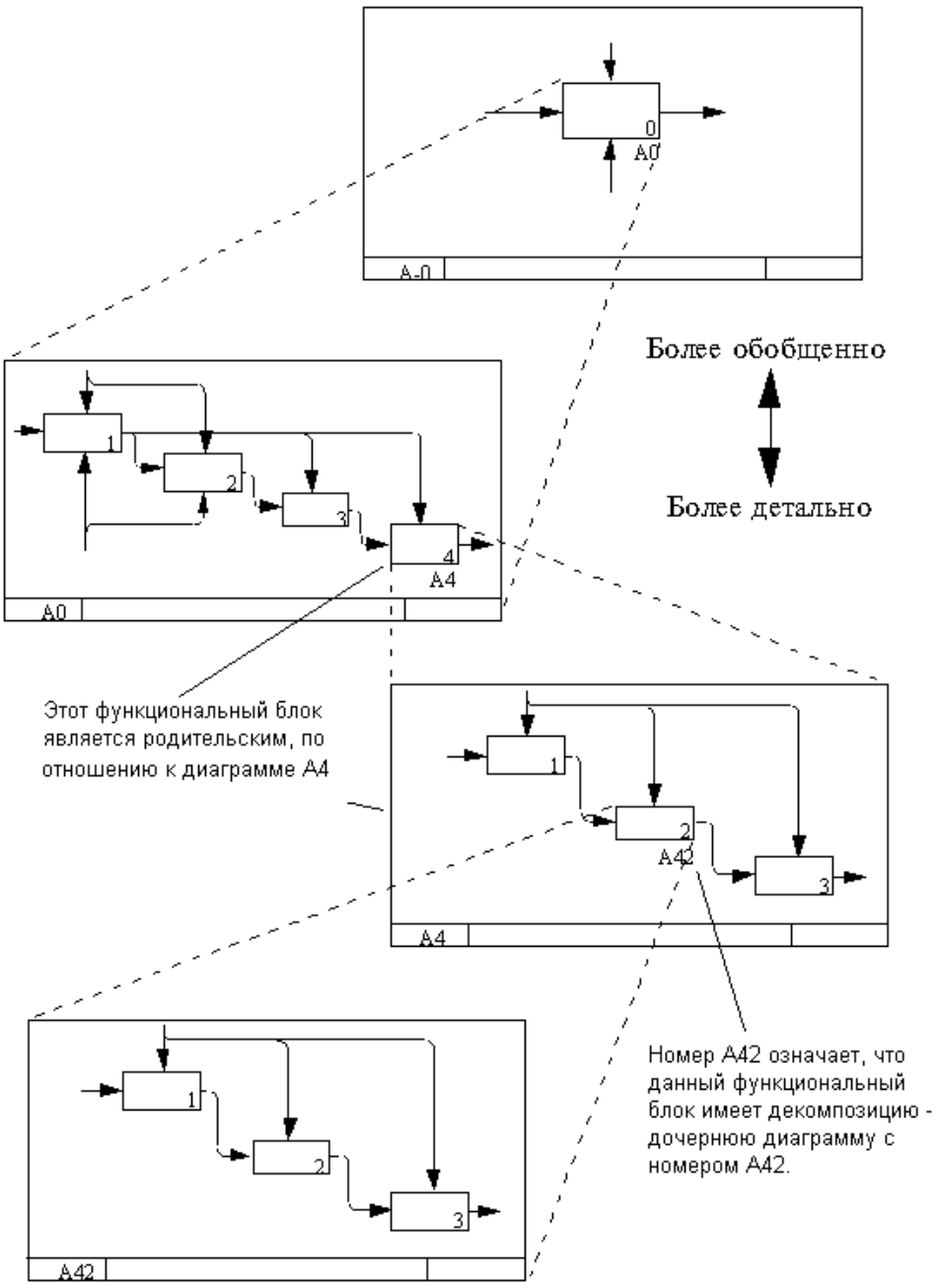


Рис.2. Декомпозиция функционального блока

Принципы ограничения сложности IDEF0-диаграмм. Обычно IDEF0-модели несут в себе сложную и концентрированную информацию, и для того, чтобы ограничить их перегруженность и сделать удобочитаемыми, в соответствующем стандарте приняты соответствующие ограничения сложности:

- Ограничение количества функциональных блоков на диаграмме тремя-шестью. Верхний предел (шесть) заставляет разработчика использовать иерархии при описании сложных предметов, а нижний предел (три) гарантирует, что на соответствующей диаграмме достаточно деталей, чтобы оправдать ее создание;
- Ограничение количества подходящих к одному функциональному блоку (выходящих из одного функционального блока) интерфейсных дуг четырьмя.

Разумеется, строго следовать этим ограничениям вовсе необязательно, однако, как показывает опыт, они являются весьма практичными в реальной работе.

Разновидности IDEF .

1. IDEF0 - методология функционального моделирования. С помощью наглядного графического языка IDEF0, изучаемая система предстает перед разработчиками и аналитиками в виде набора взаимосвязанных функций (функциональных блоков - в терминах IDEF0). Как правило, моделирование средствами IDEF0 является первым этапом изучения любой системы.

2. IDEF1 – методология моделирования информационных потоков внутри системы, позволяющая отображать и анализировать их структуру и взаимосвязи.

3. IDEF1X (IDEF1 Extended) – методология построения реляционных структур. IDEF1X относится к типу методологий “Сущность-взаимосвязь” (ER – Entity-Relationship) и, как правило, используется для моделирования реляционных баз данных, имеющих отношение к рассматриваемой системе.

4. IDEF2 – методология динамического моделирования развития систем. В связи с весьма серьезными сложностями анализа динамических систем от этого стандарта практически отказались, и его развитие приостановилось на самом начальном этапе. Однако в настоящее время присутствуют алгоритмы и их компьютерные реализации, позволяющие превращать набор статических диаграмм IDEF0 в динамические модели, построенные на базе “раскрашенных сетей Петри” (CPN – Color Petri Nets).

5. IDEF3 – методология документирования процессов, происходящих в системе, которая используется, например, при исследовании технологических процессов на предприятиях. С помощью IDEF3 описываются сценарий и

последовательность операций для каждого процесса. IDEF3 имеет прямую взаимосвязь с методологией IDEF0 – каждая функция (функциональный блок) может быть представлена в виде отдельного процесса средствами IDEF3.

6. IDEF4 – методология построения объектно-ориентированных систем. Средства IDEF4 позволяют наглядно отображать структуру объектов и заложенные принципы их взаимодействия, тем самым позволяя анализировать и оптимизировать сложные объектно-ориентированные системы.

7. IDEF5 – методология онтологического исследования сложных систем. С помощью методологии IDEF5 онтология системы может быть описана при помощи определенного словаря терминов и правил, на основании которых могут быть сформированы достоверные утверждения о состоянии рассматриваемой системы в некоторый момент времени. На основе этих утверждений формируются выводы о дальнейшем развитии системы и производится её оптимизация. В рамках этой статьи мы рассмотрим наиболее часто используемую методологию функционального моделирования IDEF0.

Диаграмма потоков данных

DFD — общепринятое сокращение от англ. Data Flow Diagrams — диаграммы потоков данных. Так называется методология графического структурного анализа, описывающая внешние по отношению к системе источники и адресаты данных, логические функции, потоки данных и хранилища данных, к которым осуществляется доступ.

Диаграмма потоков данных (data flow diagram, DFD) — один из основных инструментов структурного анализа и проектирования информационных систем, существовавших до широкого распространения UML. Несмотря на имеющее место в современных условиях смещение акцентов от структурного к объектно-ориентированному подходу к анализу и проектированию систем, «старинные» структурные нотации по-прежнему широко и эффективно используются как в бизнес-анализе, так и в анализе информационных систем.

Исторически сложилось так, что для описания диаграмм DFD используются две нотации — Йордана (Yourdon) и Гейна-Сарсона (Gane-Sarson), отличающиеся синтаксисом.

Информационная система принимает извне потоки данных. Для обозначения элементов среды функционирования системы используется понятие внешней сущности. Внутри системы существуют процессы преобразования информации, порождающие новые потоки данных. Потоки данных могут поступать на вход к другим процессам, помещаться (и извлекаться) в накопители данных, передаваться к внешним сущностям.

Модель DFD, как и большинство других структурных моделей — иерархическая модель. Каждый процесс может быть подвергнут декомпозиции, то есть разбиению на структурные составляющие, отношения между которыми в той же нотации могут быть показаны на отдельной диаграмме. Когда достигнута требуемая глубина декомпозиции — процесс нижнего уровня сопровождается мини-спецификацией (текстовым описанием).

Кроме того, нотация DFD поддерживает понятие подсистемы — структурного компонента разрабатываемой системы.

Нотация DFD — удобное средство для формирования контекстной диаграммы, то есть диаграммы, показывающей разрабатываемую АИС в коммуникации с внешней средой. Это — диаграмма верхнего уровня в иерархии диаграмм DFD. Ее назначение — ограничить рамки системы, определить, где заканчивается разрабатываемая система и начинается среда. Другие нотации, часто используемые при формировании контекстной диаграммы — диаграмма SADT, Диаграмма вариантов использования.

External Entity (Внешняя сущность) представляет собой материальный предмет или физическое лицо, представляющее собой источник или приемник информации, например, заказчики, персонал, поставщики, клиенты, склад. Определение некоторого объекта или системы в качестве внешней сущности указывает на то, что она находится за пределами границ анализируемой ИС. В процессе анализа некоторые внешние сущности могут быть перенесены внутрь диаграммы анализируемой ИС, если это необходимо, или, наоборот, часть процессов ИС может быть вынесена за пределы диаграммы и представлена как внешняя сущность. Внешняя сущность обозначается квадратом, расположенным как бы "над" диаграммой и бросающим на нее тень, для того, чтобы можно было выделить этот символ среди других обозначений.

Process (Системы и подсистемы) - при построении модели сложной ИС она может быть представлена в самом общем виде на так называемой контекстной диаграмме в виде одной системы как единого целого, либо может быть декомпозирована на ряд подсистем. Процесс представляет собой преобразование входных потоков данных в выходные в соответствии с определенным алгоритмом. Физически процесс может быть реализован различными способами: это может быть подразделение организации (отдел), выполняющее обработку входных документов и выпуск отчетов, программа, аппаратно реализованное логическое устройство и тд.

Моделирование данных

CA ERwin® Data Modeler Community Edition — это бесплатное средство моделирования, которое является базовой версией флагманского продукта CA

ERwin Data Modeler Standard Edition. Это решение предоставляет множество базовых функций моделирования данных с ограничением до 25 объектов моделей, в том числе для проектирования и создания баз данных, сравнения моделей, определения стандартов и др.

Решение CA ERwin Data Modeler Community Edition помогает организациям управлять сложной инфраструктурой данных с помощью следующих основных возможностей:

- Визуализация сложных структур данных. Модели данных создаются автоматически, что дает простое графическое представление для визуализации сложных структур баз данных.
- Создание проектов баз данных. Создавайте проекты баз данных непосредственно на основе визуальных моделей, что позволяет повысить эффективность и уменьшить число ошибок.
- Определение стандартов. Повторно используемые стандарты, такие как шаблоны моделей, домены, стандарты именования, улучшают качество и эффективность.
- Сравнение моделей и баз данных. Функция Complete Compare осуществляет сравнение моделей, скриптов и баз данных, выводя все различия на экран (в версии Community Edition данные доступны только для чтения).
- Отчетность и публикация. Простой, интуитивно понятный интерфейс Report Designer позволяет создавать отчеты в виде текстовых и HTML-файлов, которые могут содержать диаграммы и метаданные.

CA ERwin Data Modeler Community Edition — это простое в использовании средство для управления небольшими базами данных и получения навыков моделирования баз данных. Моделирование данных повышает производительность за счет простой в использовании графической среды, которая упрощает проектирование и обслуживание баз данных, автоматизирует множество трудоемких задач и улучшает коммуникацию в организации, помогая повысить эффективность и качество данных, сокращая при этом расходы.

Совместная работа над моделями данных позволяет реализовать эффективную коммуникацию между лицами, отвечающими за бизнес и технические вопросы, помогая обеспечивать согласование бизнес-требований с технической реализацией бизнес-данных. Визуальные инструменты проектирования позволяют разработчикам баз данных решать задачи проектирования и устранять проблемы до каких-либо существенных вложений ресурсов, что помогает организации быстрее реагировать на растущие потребности бизнеса, выявляя влияние изменений на информационные активы

и обеспечивая быструю реакцию на непрерывные изменения и быстрый рост среды данных.

Поддерживаемые среды

CA ERwin Data Modeler Community Edition работает в следующих средах:

- Windows XP
- Windows 2003 и 2008 Server SP2
- Windows Vista
- Windows 7
- Windows 8

и поддерживает следующие среды баз данных:

- Oracle
- SQL Server
- DB2 UDB
- MySQL
- ODBC
- Sybase

ERwin имеет два уровня представления модели - **логический** и **физический**.

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

Лабораторная работа №1
Создание контекстной диаграммы

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

Деятельность компании состоит из следующих элементов:

- менеджер по работе с клиентами принимают заказы клиентов;
- операторы группируют заказы по типам клиентов;
- операторы собирают (согласно заказов) и тестируют компьютеры;
- операторы упаковывают компьютеры согласно заказам;
- кладовщик отгружает клиентам заказ.

Компания использует приобретенную бухгалтерскую ИС, которая позволяет оформить заказ, счет и отследить платежи по счетам.

1. Запустите программу Ramus. После запуска программы на экране появится окно начала работ Выберите опцию "**Создать**" и нажмите "**ОК**".
2. Внесите имя автора, название проекта, название модели и выберите опцию "IDEF0" (рис. 3).
3. На следующем шаге укажите, что модель используется "отделом стратегического планирования и развития" (рис. 4).
4. В описании проекта укажите "Это учебная модель, описывающая деятельность компании" (рис. 5), перейдите к следующему шагу.

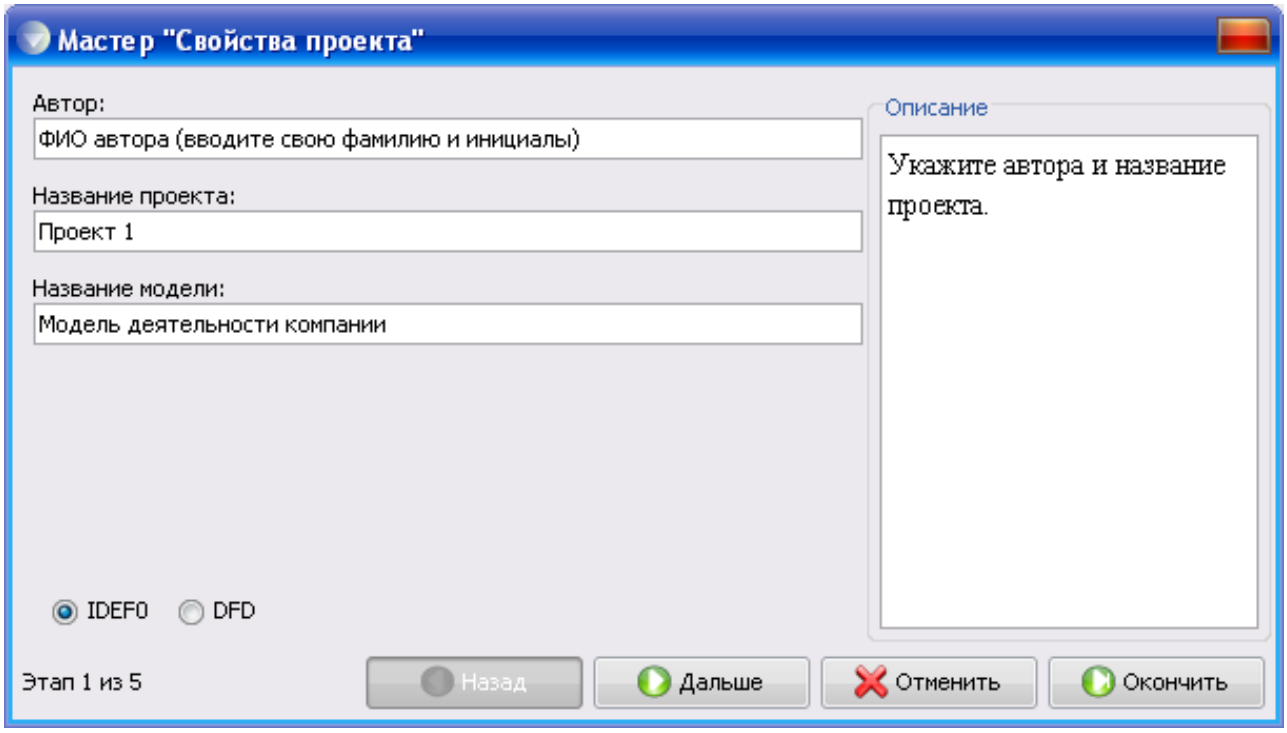


Рис. 3. Внесение первоначальных сведений
о проекте (этап 1)

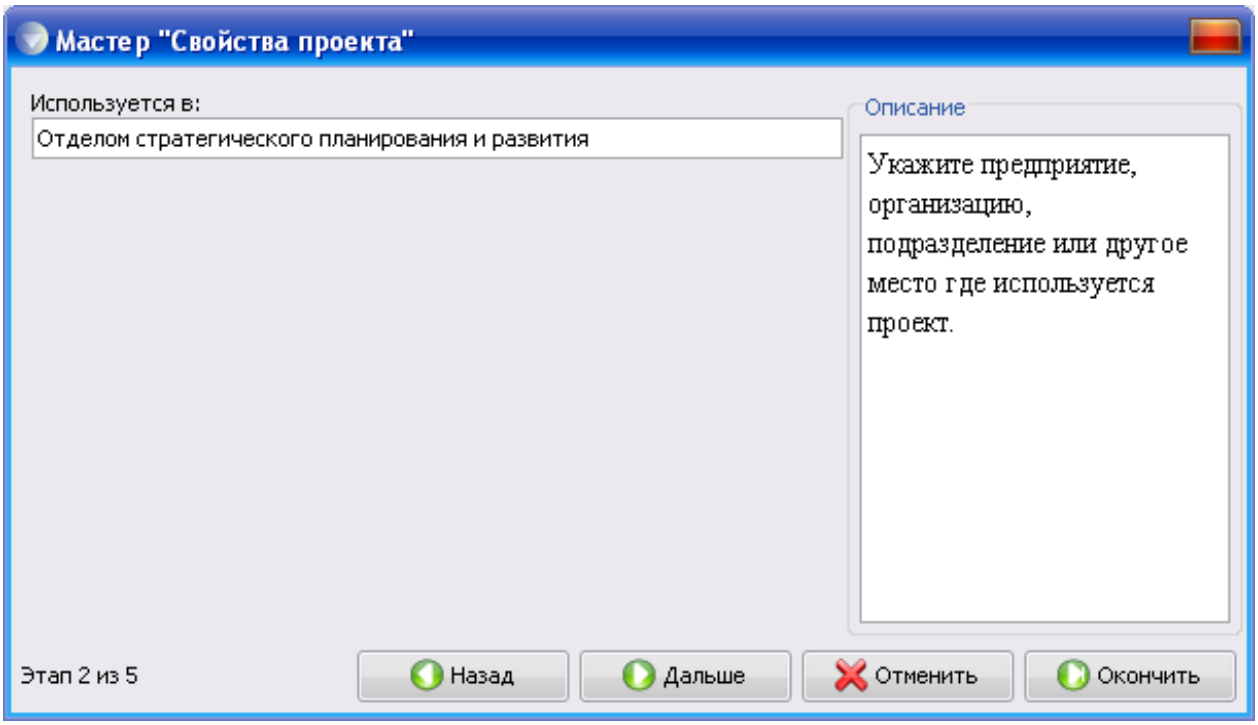


Рис. 4. Внесение первоначальных сведений о проекте (этап 2)

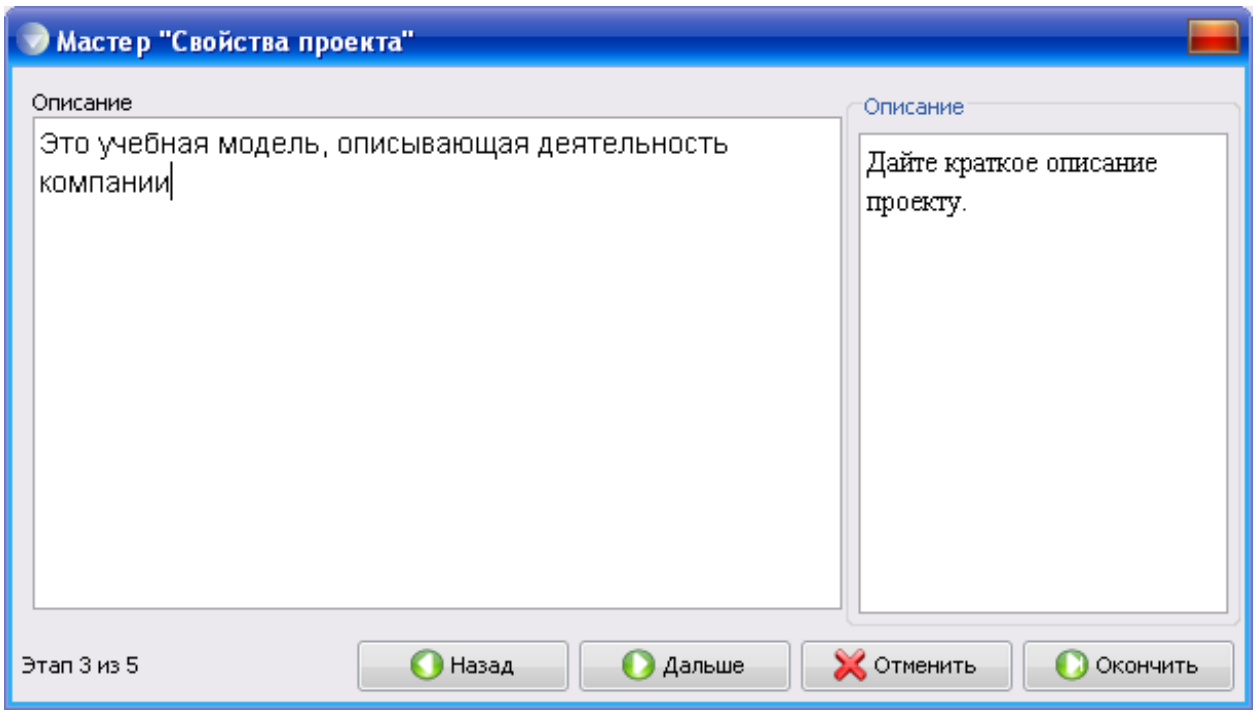


Рис. 5. Внесение первоначальных сведений о проекте (этап 3)

5. Раздел "классификаторы" оставьте незаполненным и нажмите "Дальше".

6. В следующем диалоговом окне нажмите "Окончить" и перейдите к рабочему интерфейсу программы.

Через меню Диаграмма/Свойства модели можно отредактировать метаданные модели, а именно: название модели, описание, место ее использования.

Активируйте окно модели, кликнув на область моделирования. Создайте контекстную диаграмму, нажав на соответствующую кнопку выбора элемента и после этого щёлкнув по пустой области построения диаграммы.

Перейдите в режим редактирования контекстной диаграммы, нажав правой кнопкой мыши на объекте и выбрав опцию "Редактировать активный элемент". В закладке "Название" введите "Деятельность компании".

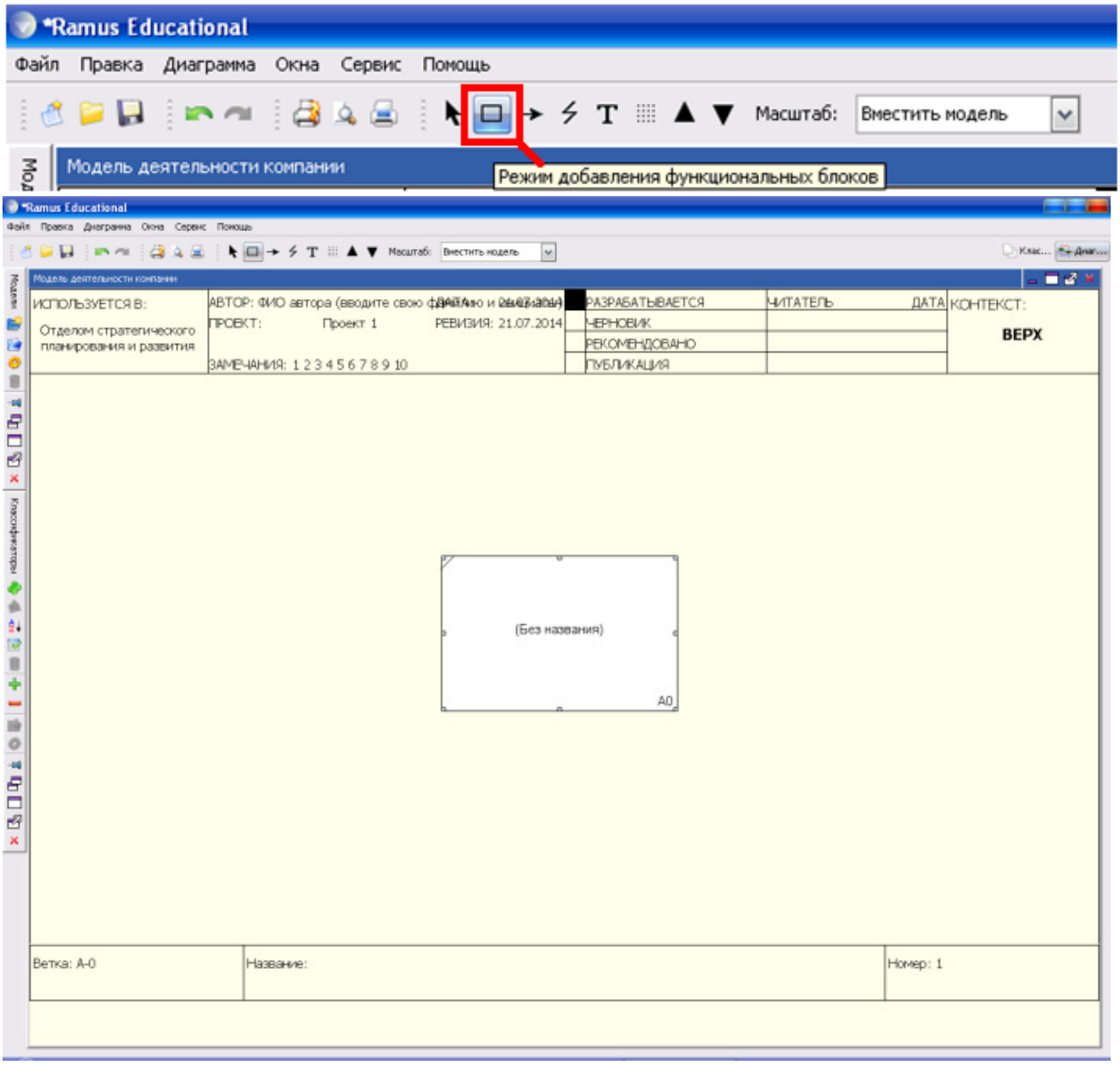


Рис. 6. Результат выполнения указанных действий

Создайте стрелки на контекстной диаграмме в соответствие с информацией, приведенной в таблице 1.

Для создания стрелок необходимо перейти в режим построения стрелок с помощью кнопки , навести курсор на исходную точку стрелки (левая, верхняя и нижняя граница области построения модели или правая граница контекстной диаграммы), после того, как область будет подсвечена черным цветом, кликнуть один раз и аналогичным образом обозначить конец стрелки (правая, верхняя и нижняя граница контекстной диаграммы или правая граница области построения модели). Перемещать стрелки и их названия можно по принципам стандартного механизма drag&drop.

Таблица 1

Описание стрелок контекстной диаграммы

НАЗВАНИЕ	"СМЫСЛОВАЯ НАГРУЗКА"	ТИП
Персонал	Обслуживание системы (реализация управления деятельстью компании на всех этапах выполнения работ)	Механизм
Звонки клиентов	Запросы информации, заказы, необходимость технической поддержки и т.д.	Вход
Правила и процедуры	Правила продаж, инструкции по сборке, процедуры тестирования, критерии производительности, различные законодательльные акты, ГОСТы, стандарты и т. д.	Управляющее воздействие
Документы на получение проданных товаров со склада	Договор на заключение сделки купли-продажи, счёт на оплату, накладная (по которой производится выдача товаров со склада), гарантийный талон.	Выход

После того как все стрелки созданы необходимо выровнять их относительно блока диаграммы - для этого вызвать контекстное меню (щелкнув по блоку работ) и выбрать Центровать присоединённые стрелки.

На рис. 7 представлен результат построения контекстной диаграммы.

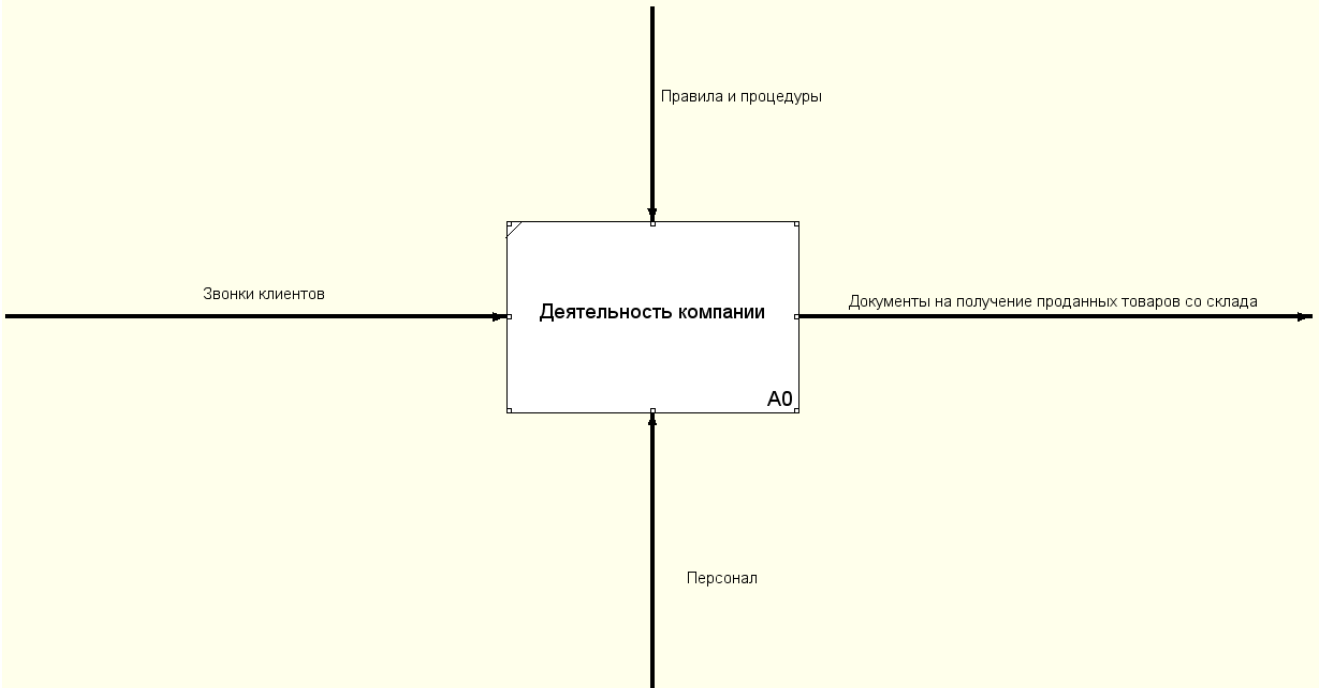


Рис. 7. Контекстная диаграмма

Лабораторная работа №2
Создание классификатора и диаграммы декомпозиции

Создание классификатора

Для того что бы создать классификатор необходимо перейти в соответствующий режим, для чего нужно щелкнуть ЛКМ по кнопке Классификаторы в правом верхнем углу (на панели инструментов).

Создайте элемент классификатора «Роли», в который внесите следующие элементы:

- менеджер по работе с клиентами;
- персонал производственного отдела;
- кладовщик.

Результаты выполнения указанных действий представлены на рис. 8.

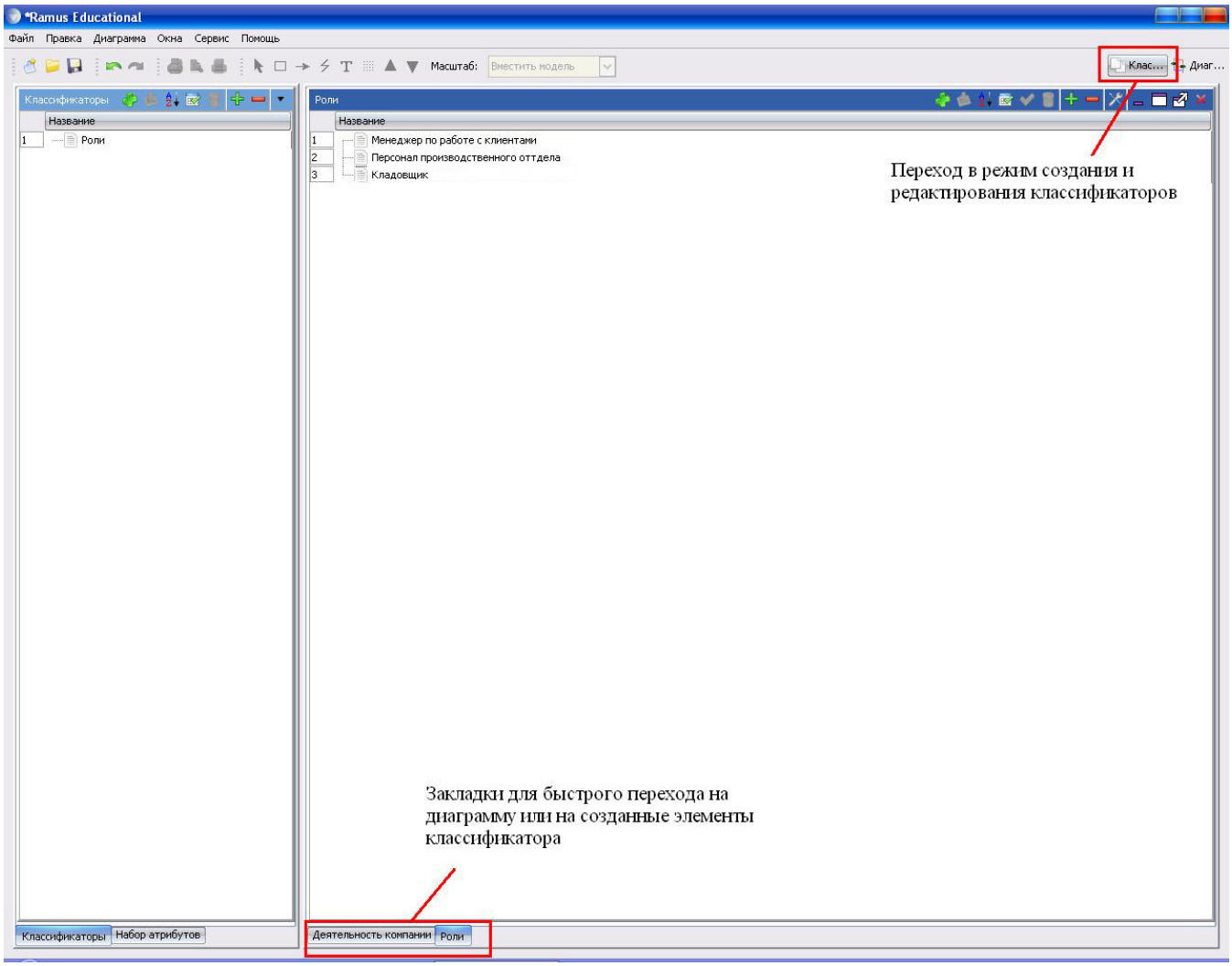


Рис. 8. Создание классификатора

Воспользовавшись закладкой в нижней части окна программы перейдите к контекстной диаграмме.

Создание диаграммы декомпозиций

1. Выберите кнопку перехода на уровень ниже ▾ на панели инструментов.

2. В диалоговом окне укажите число работ на диаграмме нижнего уровня - "3", а нотацию декомпозиции - *IDEF* (рис. 9), затем нажмите ОК. Автоматически будет создана диаграмма декомпозиции.

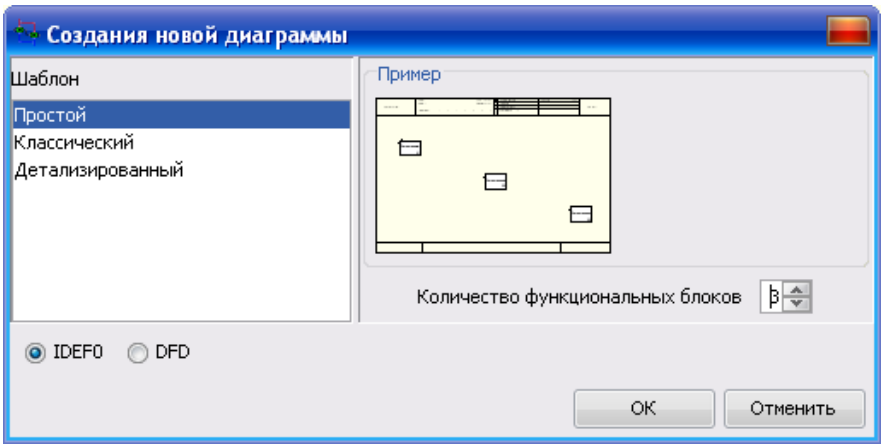


Рис. 9. Диалоговое окно декомпозиции работ

3. Правой кнопкой мыши щелкните по 1-ой работе, выберите "Редактировать активный элемент" и на вкладке "Название" укажите имя работы. Повторите операцию для всех трех работ.

4. Перейдите в режим рисования стрелок. Произведите связывание граничных стрелок с функциональными объектами, как показано на рис. 10. Для связывания граничных стрелок наводите курсор на сами стрелки, а не на границы области построения моделей.

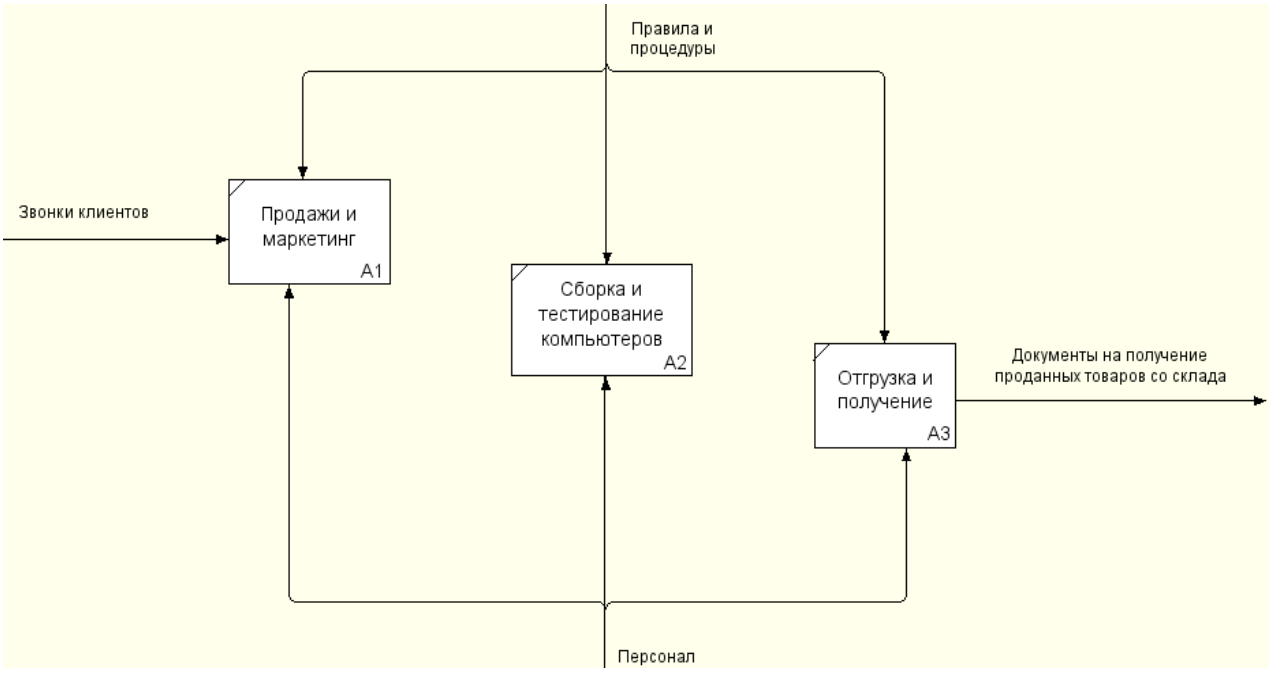


Рис. 10. Связывание граничных стрелок
на диаграмме декомпозиции

5. Правой кнопкой мыши щёлкните по ветви стрелки "Сборка и тестирование компьютеров", переименуйте ее в "Правила сборки и тестирования" (рис. 12).

Далее необходимо конкретизировать роли персонала, отвечающего за выполнение конкретного блока работ. Правой кнопкой мыши щелкните по ветви стрелки механизма работы "Продажи и маркетинг" и в открывшемся диалоговом окне на закладке «Поток» сначала нажмите Очистить, а потом Добавить. Выберите «Роли» и поставьте галочку возле «Менеджер по работе с клиентами» (рис. 11) и нажмите Ок.

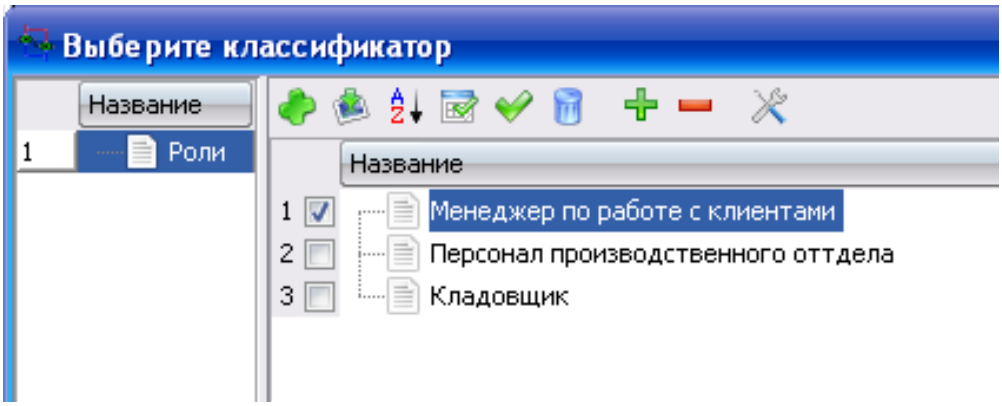


Рис. 11. Назначение названия стрелки из классификатора

Аналогично обозначить все остальные стрелки механизма (рис 12).

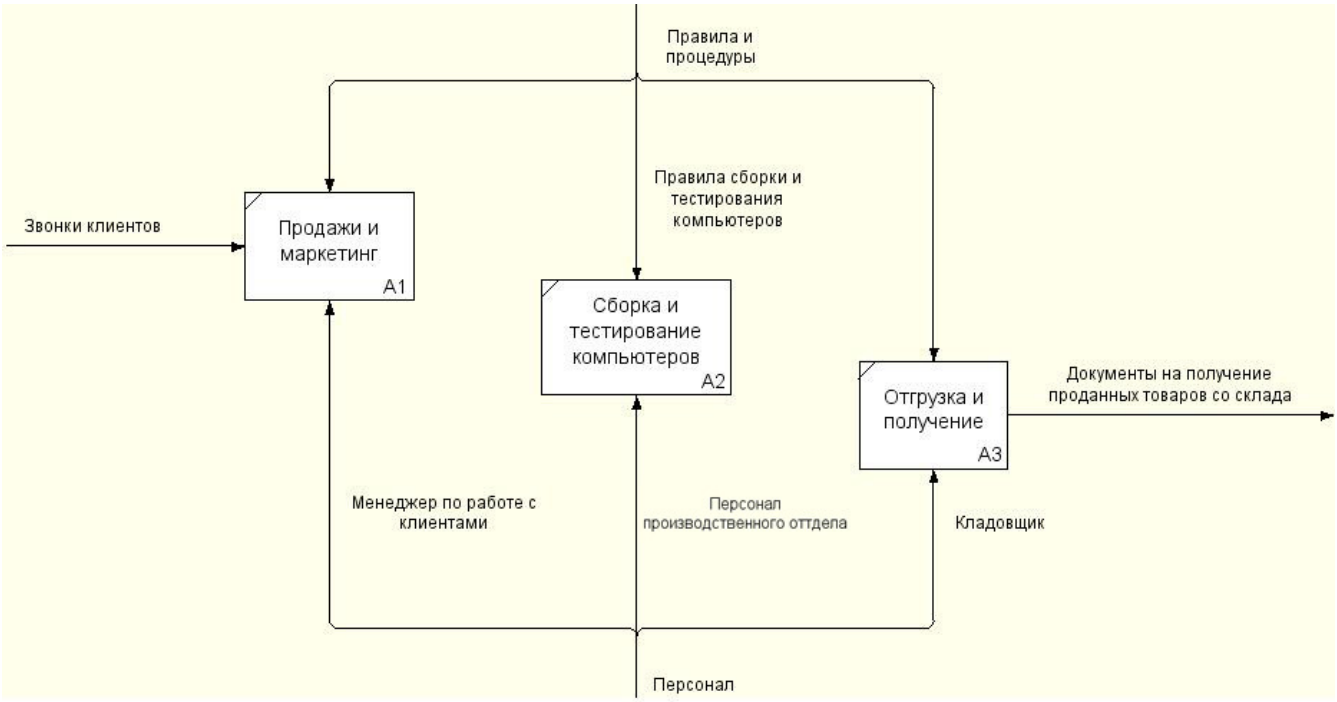


Рис. 12. Присвоение названий ветвям стрелок диаграммы декомпозиции

6. Создайте новые внутренние стрелки, как показано на рисунке (рис. 13).

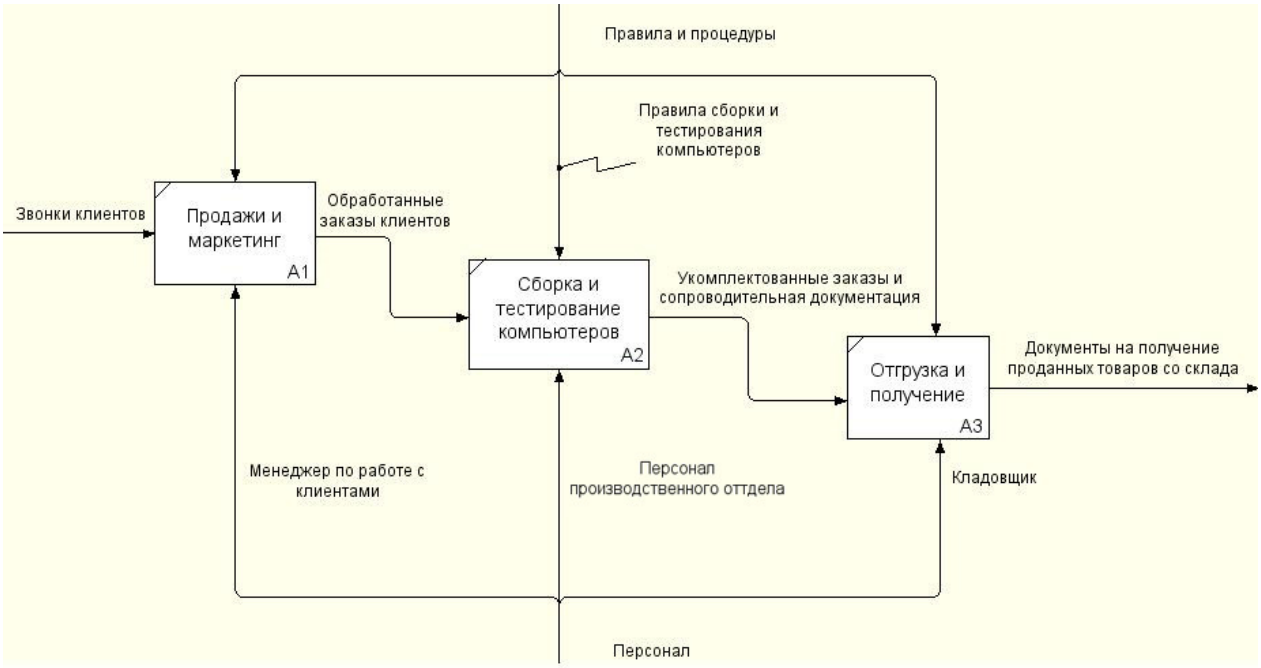


Рис. 13. Внутренние стрелки диаграммы декомпозиции

7. Создайте новую граничную стрелку "Маркетинговые материалы", выходящую из работы "Продажи и маркетинг" Эта стрелка автоматически не попадает на диаграмму верхнего уровне и имеет квадратные скобки у окончания ——▣. Щелкните правой кнопки мыши по квадратным скобкам и выберите в контекстном меню "Туннель" одну их двух опций: Создать стрелку и Обозначить туннель круглыми скобками, в нашем случае - первый вариант (рис. 14).

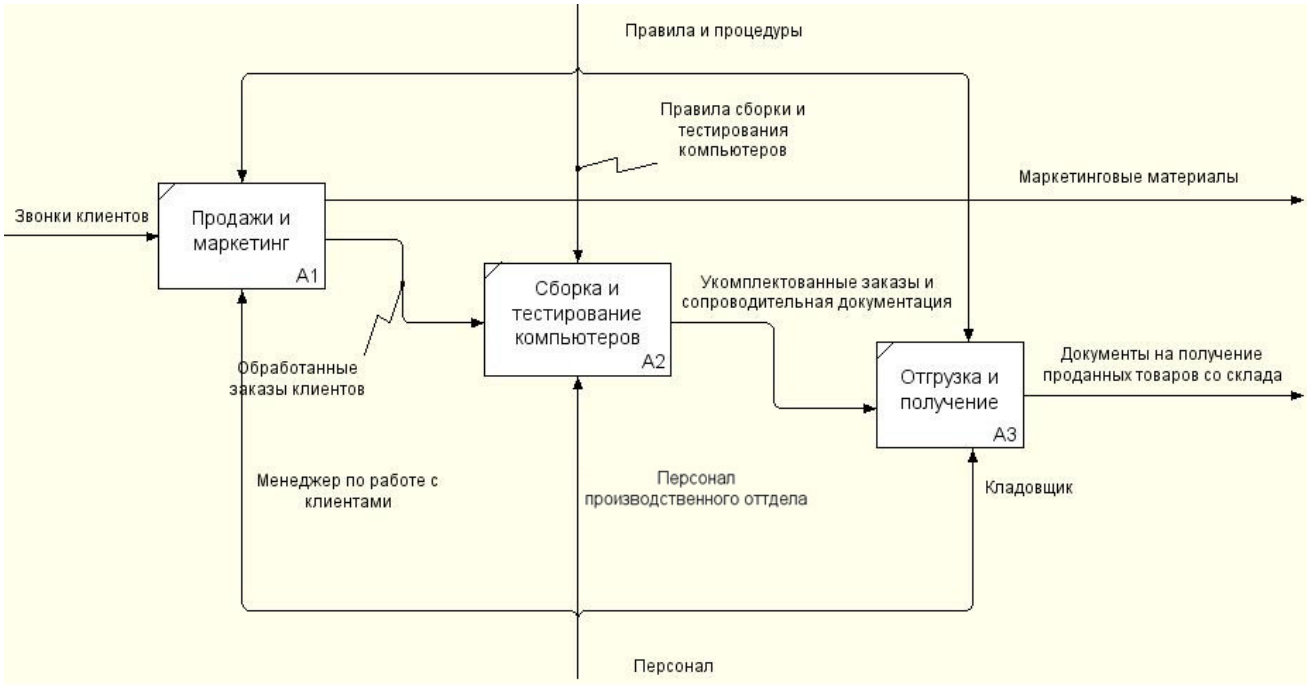


Рис. 14. Результат туннелирования стрелок

После этого необходимо доработать классификатор, добавив в Персонал производственного отдела две роли: Диспетчер и Тестировщики (рис. 15).

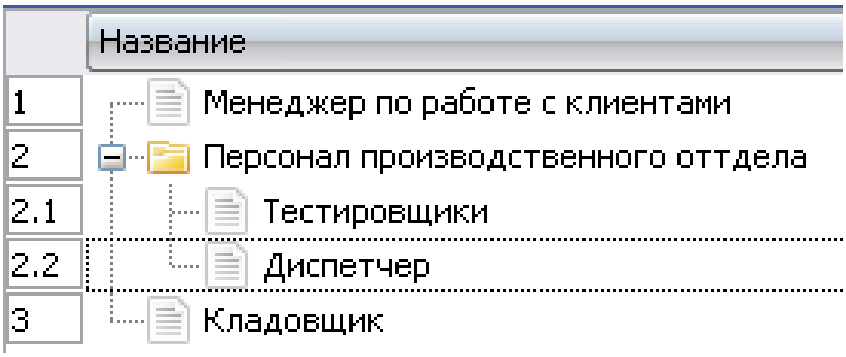


Рис. 15. Редактирование классификатора

Лабораторная работа №3
Создание диаграммы декомпозиций второго уровня

Декомпозируем работу "Сборка и тестирование компьютеров". В результате проведенного анализа получена следующая информация о процессе:

Производственный отдел получает заказы от отдела клиентов по мере их поступления.

- диспетчер координирует работу сборщиков, сортирует заказы, группирует и дает указания на отгрузку компьютеров, когда они готовы;
- каждые 2 часа диспетчер группирует заказы - отдельно для настольных компьютеров и ноутбуков - и направляет их 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. Тестировщик тестируют каждый компьютер и, в случае необходимости, заменяет неисправные компоненты.
- Тестировщики направляют результаты тестирования диспетчеру, который на основании этой информации принимает решение о передаче компьютеров, соответствующих группе заказов, на отгрузку.

1. На основе информации из таблиц 1 и 2 внесите новые работы и стрелки на диаграмму декомпозиции А2.
2. Далее следует разнести стрелки, перенесённые с диаграммы верхнего уровня (как показано на рис. 16).

Таблица 1

Описание функциональных блоков диаграммы декомпозиции A2

НАЗВАНИЕ ФУНКЦИОНАЛЬНОГО БЛОКА	ОПИСАНИЕ
Отслеживание расписания и управление сборкой и тестирование	Просмотр заказов, установка расписания выполнения заказов, просмотр результатов тестирования, формирования 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.

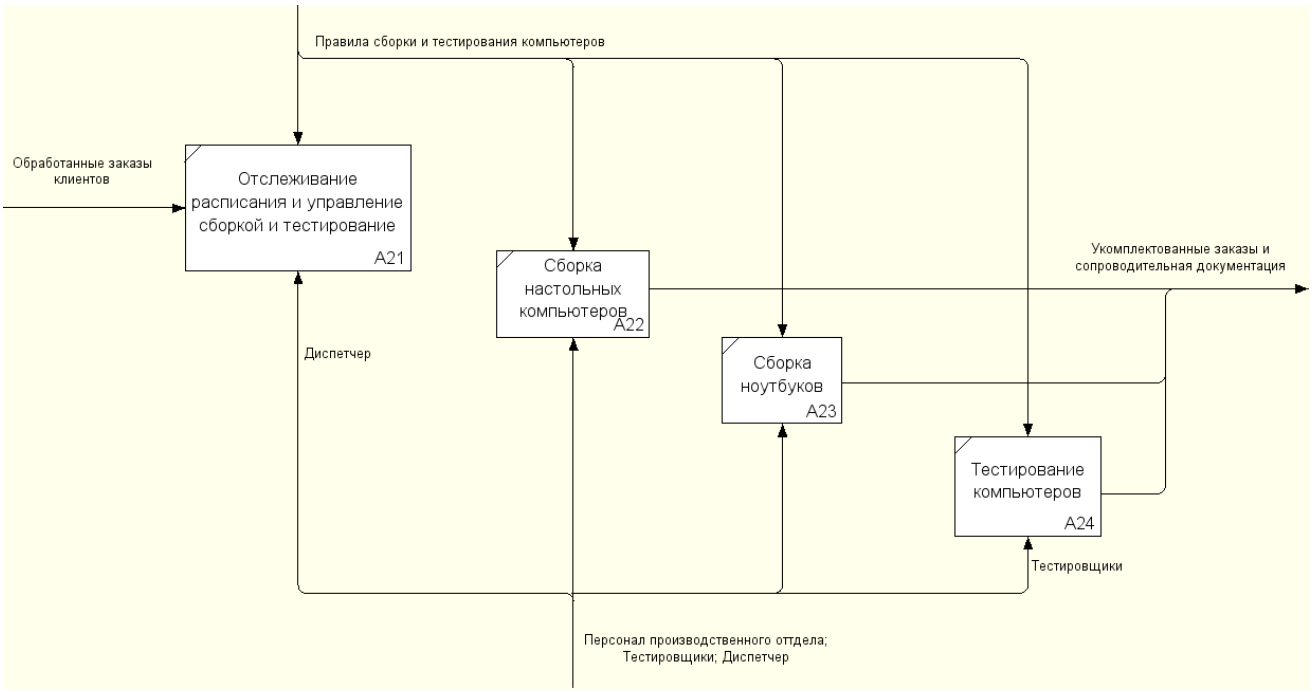


Рис. 16. Связывание граничных стрелок на диаграмме декомпозиции A2

Таблица 2

Описание стрелок диаграммы декомпозиции A2

НАЗВАНИЕ СТРЕЛКИ	НАЧАЛО СТРЕЛКИ	ТИП НАЧАЛА СТРЕЛКИ	ОКОНЧАНИЕ СТРЕЛКИ	ТИП ОКОНЧАНИЯ СТРЕЛКИ
Диспетчер (назначить из классификатора)	Персонал производственного отдела	Механизм (ветка стрелки)	Отслеживание расписания и управление сборкой и тестированием	Механизм
Заказы на настольные компьютеры	Отслеживание расписания и управление сборкой и тестированием	Выход	Сборка настольных компьютеров	Вход
Заказы на ноутбуки	Отслеживание расписания и управление сборкой и	Выход	Сборка компьютеров	Вход

	тестированием			
Компоненты	*Туннелированная* стрелка	Вход	Сборка настольных компьютеров	Вход
			Сборка ноутбуков	Вход
			Тестирование компьютеров	Вход
Настольные компьютеры	Сборка настольных компьютеров	Выход	Тестирование компьютеров	Вход
Ноутбуки	Сборка ноутбуков	Выход	Тестирование компьютеров	Вход
Результаты тестирование	Тестирование компьютеров	Выход	Отслеживание расписания и управление сборкой и тестированием	Вход
Тестировщики (назначить из классификатора)	Персонал производственного отдела	Механизм (ветка стрелки)	Тестирование компьютеров	Механизм
Указание передать компьютеры на отгрузку	Отслеживание расписания и управление сборкой и тестированием	Выход	Тестирование компьютеров	Управляющее воздействие

3. Произведите туннелирование и связку граничных стрелок, если это необходимо. Результат представлен на рис. 17.

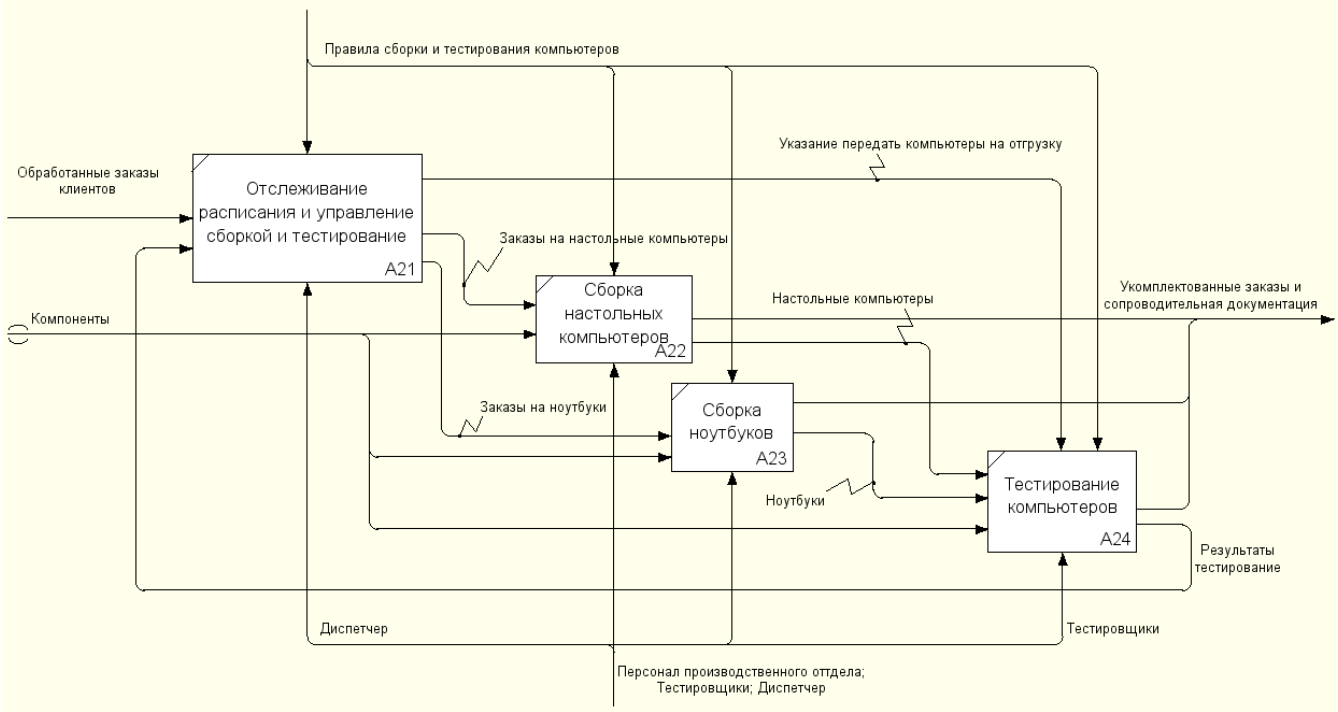


Рис. 17. Результат декомпозиции процесса «Сборка и тестирование»

Лабораторная работа №4
Создание диаграммы DFD

1. Создайте контекстную диаграмму процесса "Оформление заказов" (Файл / Новый проект).

2. Декомпозируйте созданную контекстную диаграмму "Оформление заказов", для чего в диалоговом окне выберите количество элементов декомпозиции - 2, тип диаграммы - **DFD**. Нажмите "ОК" и внесите в диаграмму DFD имена работ:

- Проверка и внесение клиента
- Внесение заказа

3. Создайте классификаторы:

- Список клиентов
- Список продуктов
- Список заказов
- Заявки на заказ

4. Внесите в модель соответствующие хранилища данных при помощи кнопки (рис. 18), а также внешнюю ссылку "**Заявки на заказ**", используя кнопку .

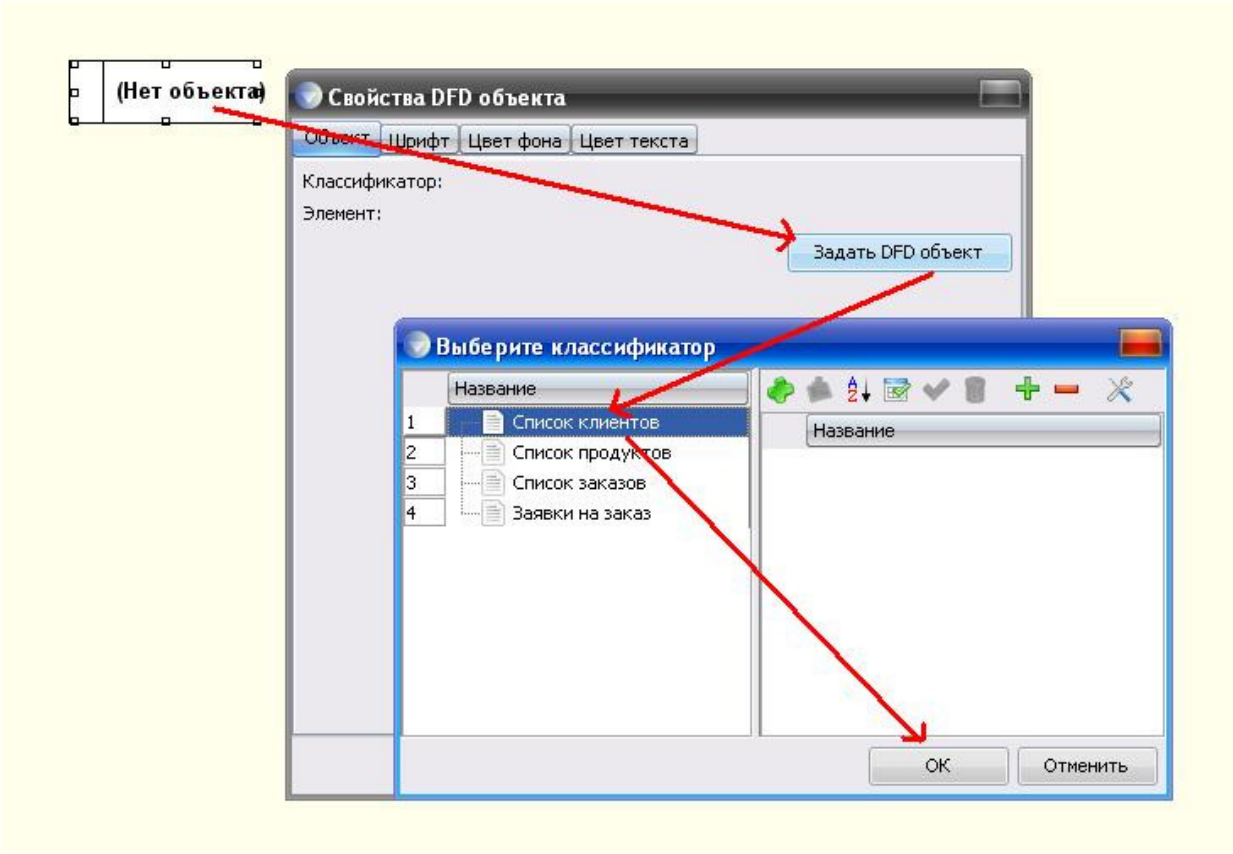


Рис. 18. Внесение хранилища данных

5. На основе следующей информации постройте DFD-модель процесса "Оформление заказов":

- Процесс "Оформление заказов" состоит из двух подпроцессов: проверка и внесение клиентов, и внесение заказов. Для выполнения этих процессов необходим список клиентов, список продуктов и для регистрации результатов выполнения процессов реестр списка заказов. Проверка и внесение клиентов в базу данных клиентов осуществляется на основе информации из заявок на заказ, а также после анализа информации в списке клиентов.
- Внесение заказов производится только при наличии информации о соответствующем клиенте в списке клиентов и только на те товары, которые занесены в список продуктов компании. Существуют возможность использовать ранее созданные заказы, сохраненные в списке заказов.
- Имейте в виду, что связь между некоторыми функциональными объектами и хранилищами данных может быть двунаправленной (исходящая и входящая стрелки).

6. Сверьте построенную Вами модель с моделью на рис. 19.

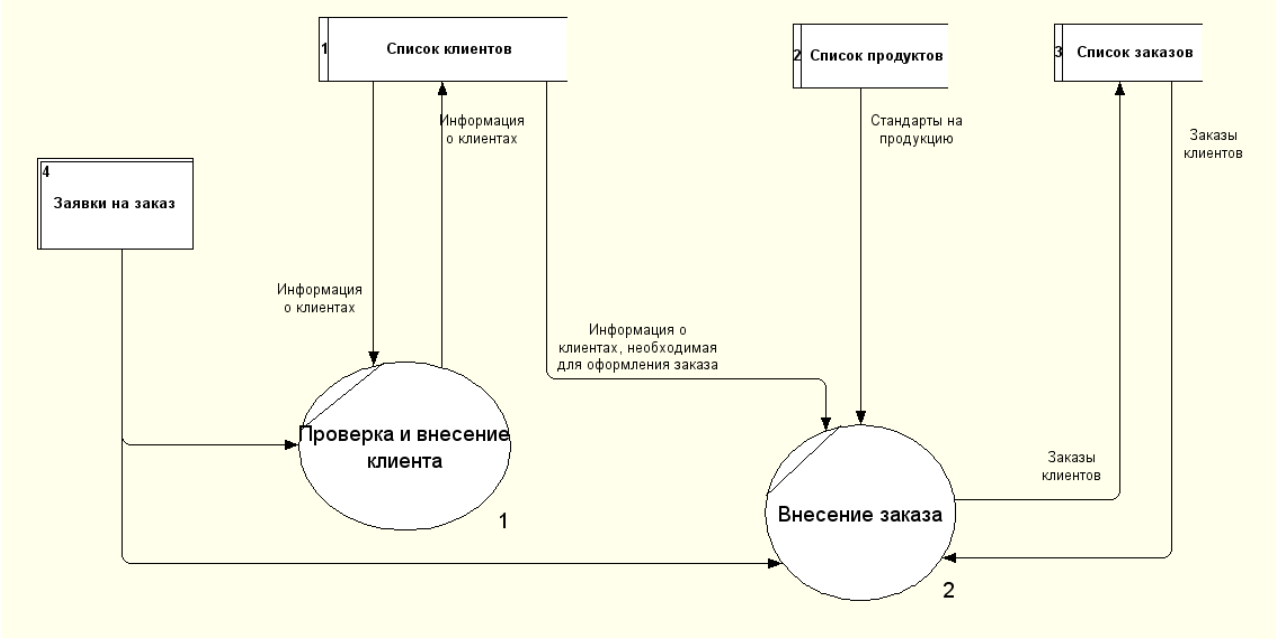


Рис. 19. DFD-диаграмма декомпозиции процесса оформления заказа

Лабораторная работа №5
Построение ERD-диаграмм

Задача: Построить ERD-диаграммы (логическую и физическую модели) компании по сборке компьютеров и ноутбуков.

Построение модели данных начинается с выделения сущностей данной предметной области. В нашем случае были выделены следующие сущности:

- клиент - организация, которая покупает компьютеры;
- заказ - список компьютеров, которые покупает клиент;
- компьютер;
- комплектующие - то, из чего собирают компьютеры;
- сотрудник - сотрудник предприятия, собирающий конкретный компьютер.

Далее необходимо определить связи между сущностями:

- **Клиент - Заказ.** Один клиент может делать несколько заказов. При этом если данные о клиенте имеются в базе данных, то он сделал минимум один заказ. Поэтому мощность связи - P. Связь идентифицирующая, т.к. заказ без клиента существовать не может;
- **Заказ - Компьютер.** В рамках одного заказа клиент может заказать несколько компьютеров, но как минимум заказ должен состоять из одного компьютера. Поэтому мощность связи - P. Связь идентифицирующая, т.к. компьютер без заказа существовать не может;
- **Компьютер - Комплектующие.** В состав одного компьютера входит много различных комплектующих; один и тот же тип комплектующего может входить в состав разных компьютеров. Мощность связи - много ко многим. В IDEF1X такой тип связи отсутствует, поэтому вводим промежуточную (ассоциативную) сущность - **Конфигурация**. Связь в обоих случаях идентифицирующая, т.к. конфигурация компьютера не может существовать без привязки к самому компьютеру и к комплектующим;
- **Комплектующие - Тип комплектующих.** Поскольку перечень типов комплектующих, которые могут быть установлены в компьютер, ограничен, но используется очень часто, то мы приняли решение создать еще одну сущность - Тип комплектующих. Мощность связи - P. Связь идентифицирующая;
- **Компьютер - Сотрудник.** Каждый компьютер собирается каким-то одним сотрудником. Какие-то сотрудники могут собирать множество компьютеров. Мощность связи - N. Тип связи - неидентифицирующая, поскольку экземпляр сущности Компьютер уже может существовать, но за

ним еще может быть не закреплен ни один сотрудник. Именно из этих же соображений в свойствах этой связи мы выбрали переключатель "Nulls Allowed" (на диаграмме это отображается в виде незакрашенного ромбика со стороны сущности-родителя.

Атрибуты сущностей представлены в таблице:

Таблица 3

Атрибуты сущностей

АТРИБУТ	ОПИСАНИЕ
Клиент	
Код клиента	Уникальный номер для идентификации связи
Наименование клиента	Наименование организации, которая делает заказ на поставку компьютеров
Телефон	Номер телефона заказчика
Реквизиты	Реквизиты организации-заказчика: адреса, номера счетов т.п.
Заказ	
Код заказа	Уникальный номер для идентификации связи
Дата оформления	Дата оформления заказа
Дата выполнения	Дата выполнения заказа
Сумма заказа	Общая стоимость заказанных компьютеров
Компьютер	
Код компьютера	Уникальный номер для идентификации связи
Серийный номер	Серийный номер компьютера
Тип компьютера	Тип компьютера (стационарный или ноутбук)
Конфигурация	
Код строки	Уникальный номер для идентификации связи
Количество	Количество комплектующих определённого вида
Комплектующие	
Код комплектующего	Уникальный номер для идентификации связи
Модель	Модель комплектующего
Производитель	Фирма-производитель комплектующего
Цена	Стоимость комплектующего
Описание	Характеристики комплектующего
Тип комплектующих	
Код типа	Уникальный номер для идентификации связи
Название типа	Название типа комплектующего
Сотрудник	
Код сотрудника	Уникальный номер для идентификации связи
Фамилия ИО	Фамилия и инициалы сотрудника
Должность	Должность сотрудника

Построение ERD-диаграмм

1. Запустите программу CA ERwin Data Modeler. CA ERwin Data Modeler (далее ERwin) - CASE-средство для проектирования и документирования баз данных, которое позволяет создавать, документировать и сопровождать базы данных, хранилища и витрины данных.

2. Меню File/New… Откроется диалоговое окно New Model. Во фрейме Type установите опцию Logical/Physical, а во фрейме Target Server (Целевой сервер базы данных) установите в раскрывающемся окне списка систему управления базой данных SQL Server, и ее версию 2012 (рис. 20). Нажмите Ок:

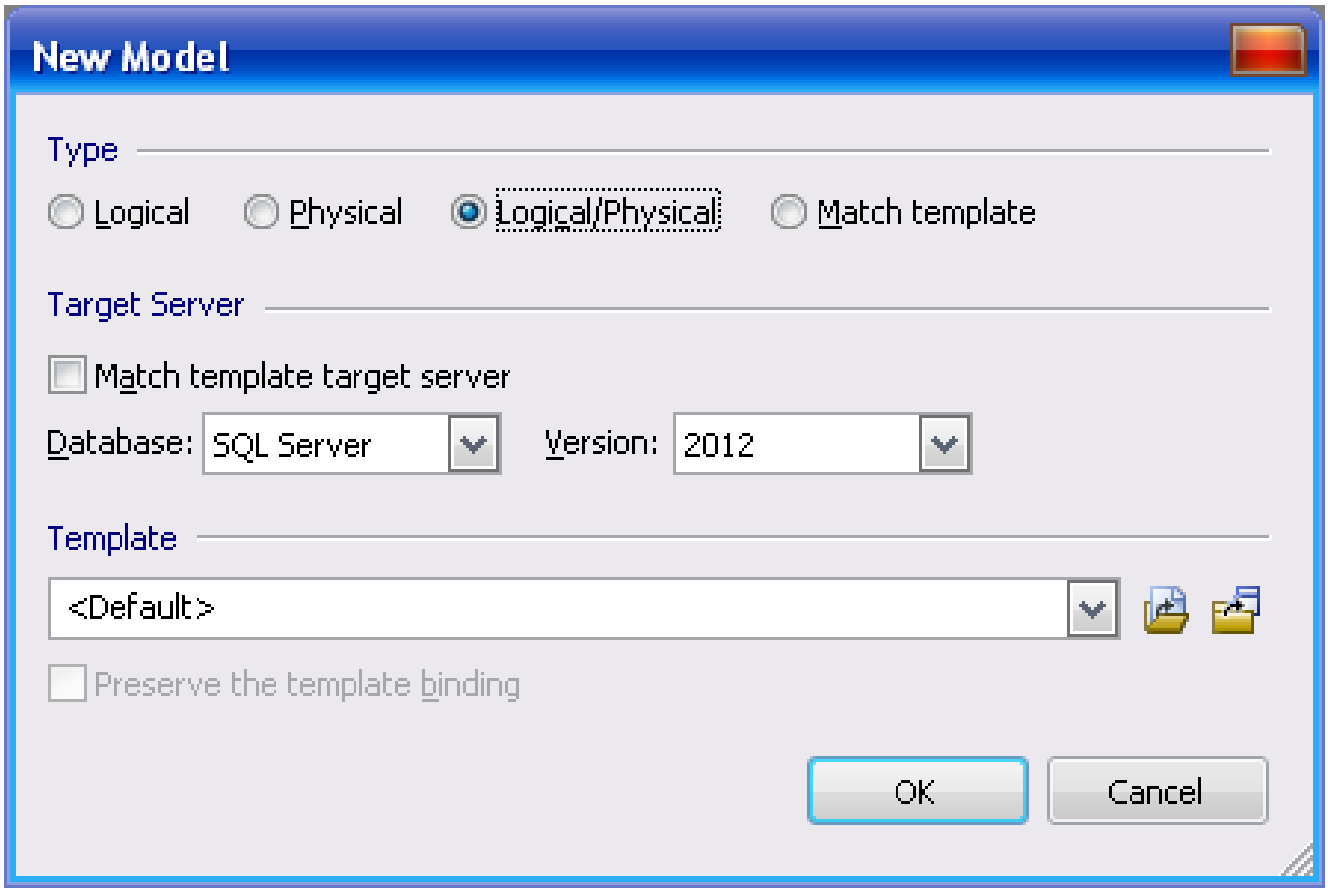


Рис. 20. Первоначальные настройки создаваемой модели

Автоматически создается незаполненная модель данных и вы получили доступ к интерфейсу среды ERwin.

3. В меню Diagram/Diagrams… Откроется диалоговое окно для настройки создаваемой модели. Первоначально в структуре созданной модели имеется только одна диаграмма, названная по умолчанию ER_Diagram_163 - переименовать её в «Логическая модель» и добавить ещё одну диаграмму, назвав её «Физическая модель» (рис. 21).

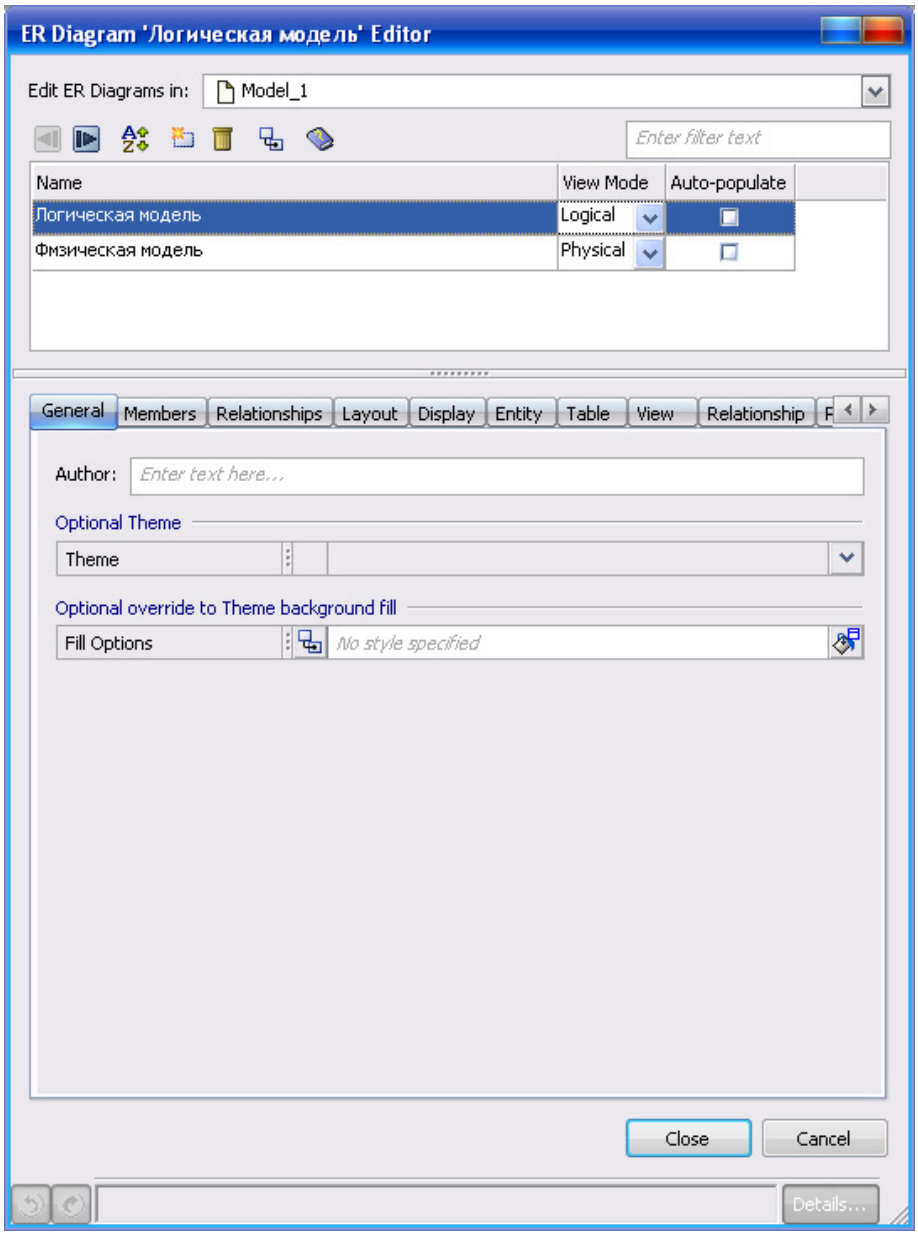


Рис. 21. Структура модели

Построение логической модели данных

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. Объекты модели, представляемые на логическом уровне, называются сущностями и атрибутами. Логическая модель данных является универсальной и никак не связана с конкретной реализацией СУБД.

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 желательно 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. Для внесения сущности в модель необходимо "кликнуть" по кнопке сущности на панели инструментов (Entity) , затем "кликнуть" по тому месту на диаграмме, где необходимо расположить новую сущность. Внесите имя сущности «Клиент».

После этого необходимо вызвать диалог Attribute Properties (из контекстного меню сущности Заказчик) – для определения атрибутов сущности. Добавление нового атрибута осуществляется щелчком по кнопке . Для каждого атрибута следует указать Имя, Домен (логический, дата/время, числовой или строковый), тип данных (определяется исходя из выбранного домена) и определить является ли атрибут первичным или внешним ключом. Результат внесения атрибутов для сущности «Клиент» представлен на рис. 22.

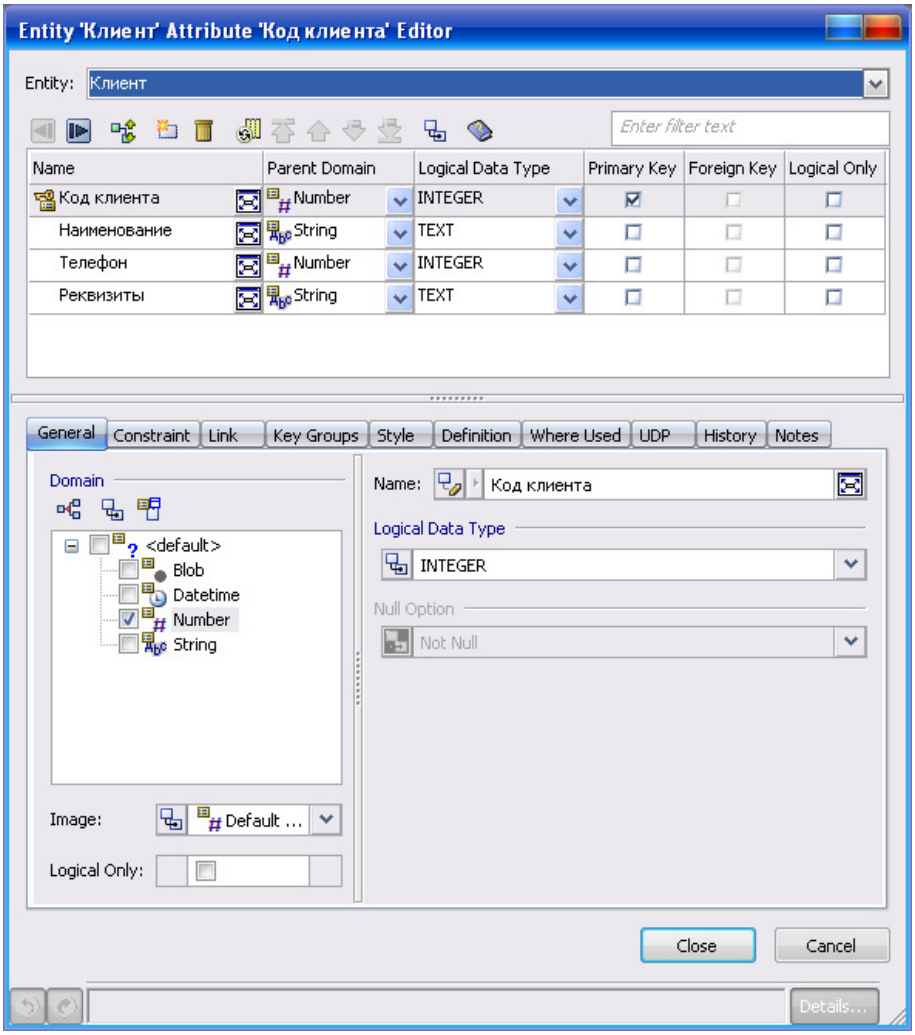


Рис. 22. Внесение атрибутов для сущности «Клиент»

После того как будут созданы все сущности, выделенные в предметной области необходимо обозначить связи между ними при помощи соответствующих кнопок панели инструментов . После того как будут созданы связи – в структуре сущностей появятся внешние ключи.

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 Properties, перейти на закладку Relationship и затем включить опции Displey Logical Relationship Name и Displey Logical Physical Name (рис. 23).

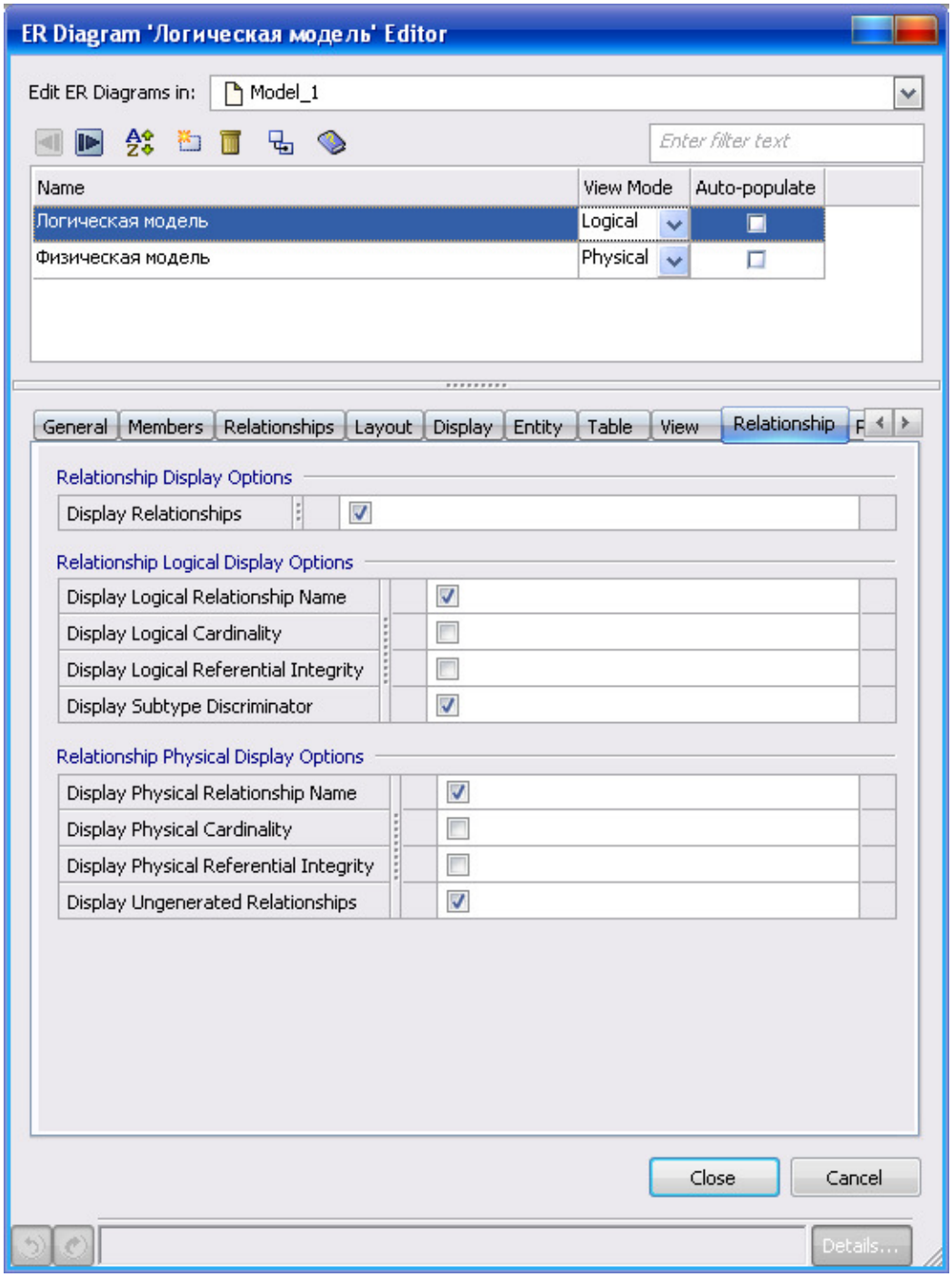


Рис. 23. Включение видимости названий связей

Итоговый вид логической модели представлен на рис. 24.

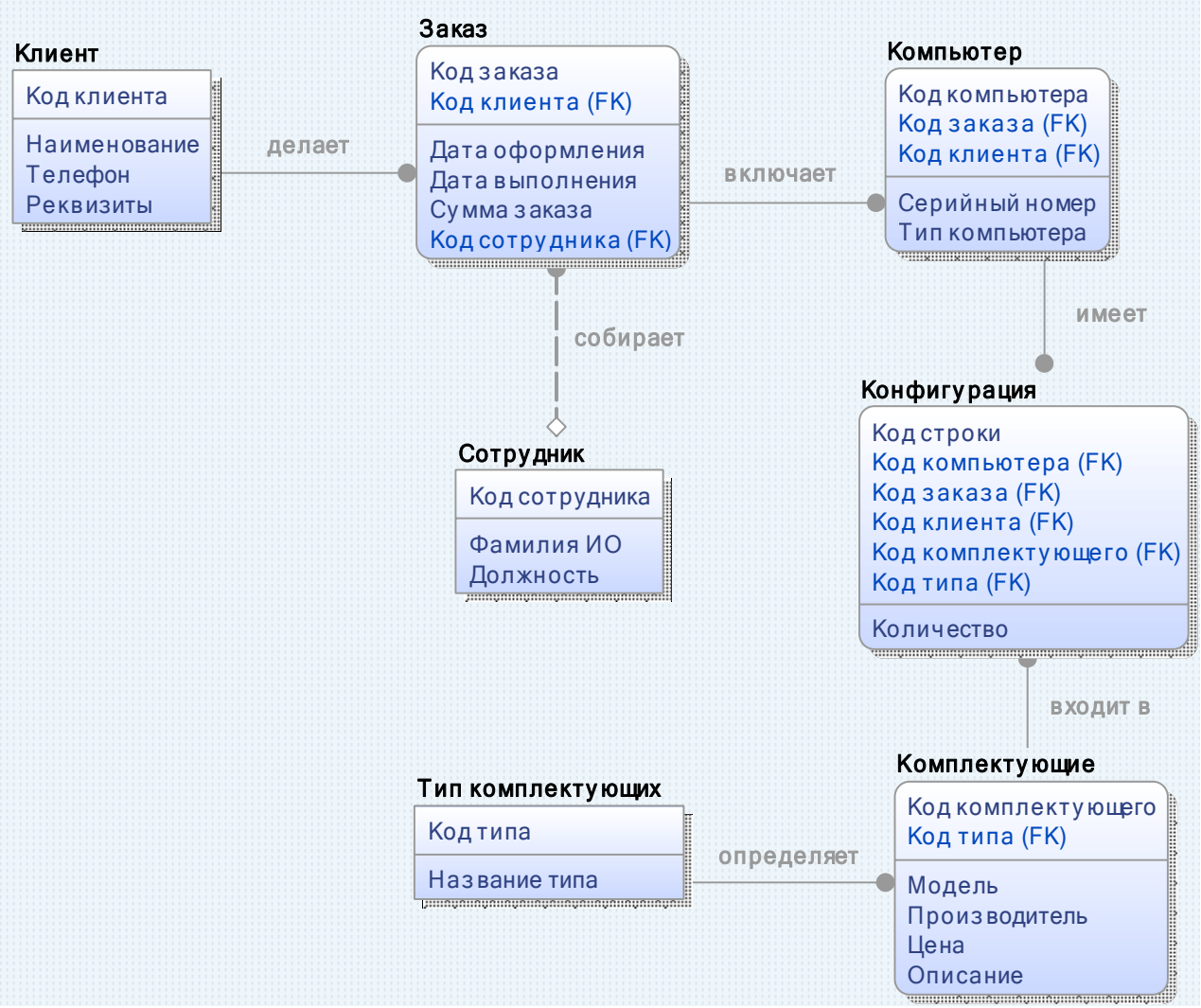


Рис. 24. Логическая модель

Построение физической модели данных

Физическая модель данных 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.

Для автоматического трансформирования физической модели из логической необходимо в контекстном меню, (вызванном щелчком левой

кнопкой мыши по любому месту диаграммы, не занятому объектами модели) выбрать пункт Properties и поставить галочки в колонке Auto-populate (рис. 25).

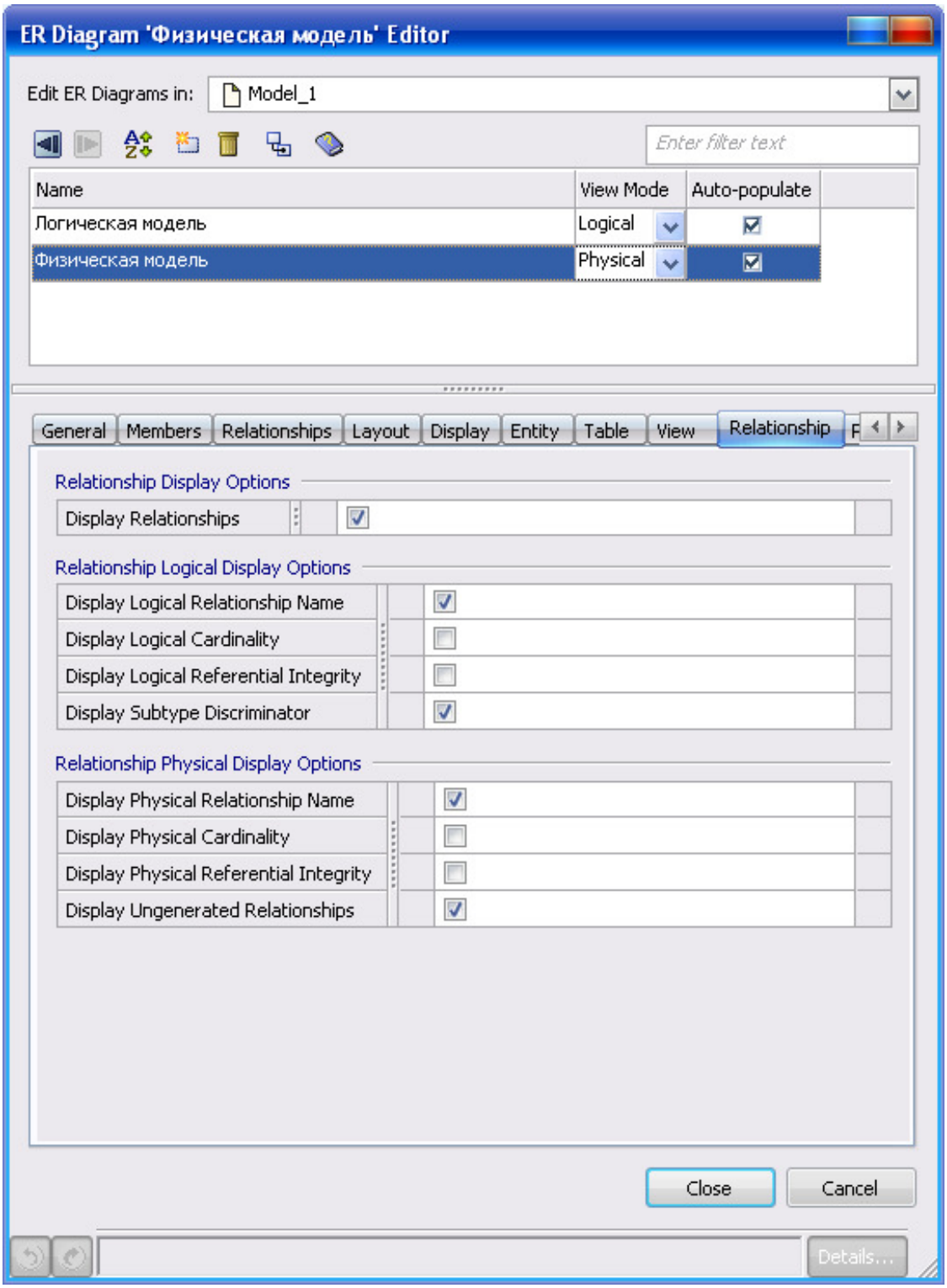


Рис. 25. Диалог «Свойства физической модели»

Лабораторная работа №6

Анализ и описание информационных потоков предметной области «Приёмная комиссия ВУЗа»

Задание: требуется представить процесс приёма, регистрации и дальнейшей обработки документов абитуриентов в виде диаграмм методологии IDEF0.

Схема информационных потоков, анализируемого участка учёта представлена на рис. 26.

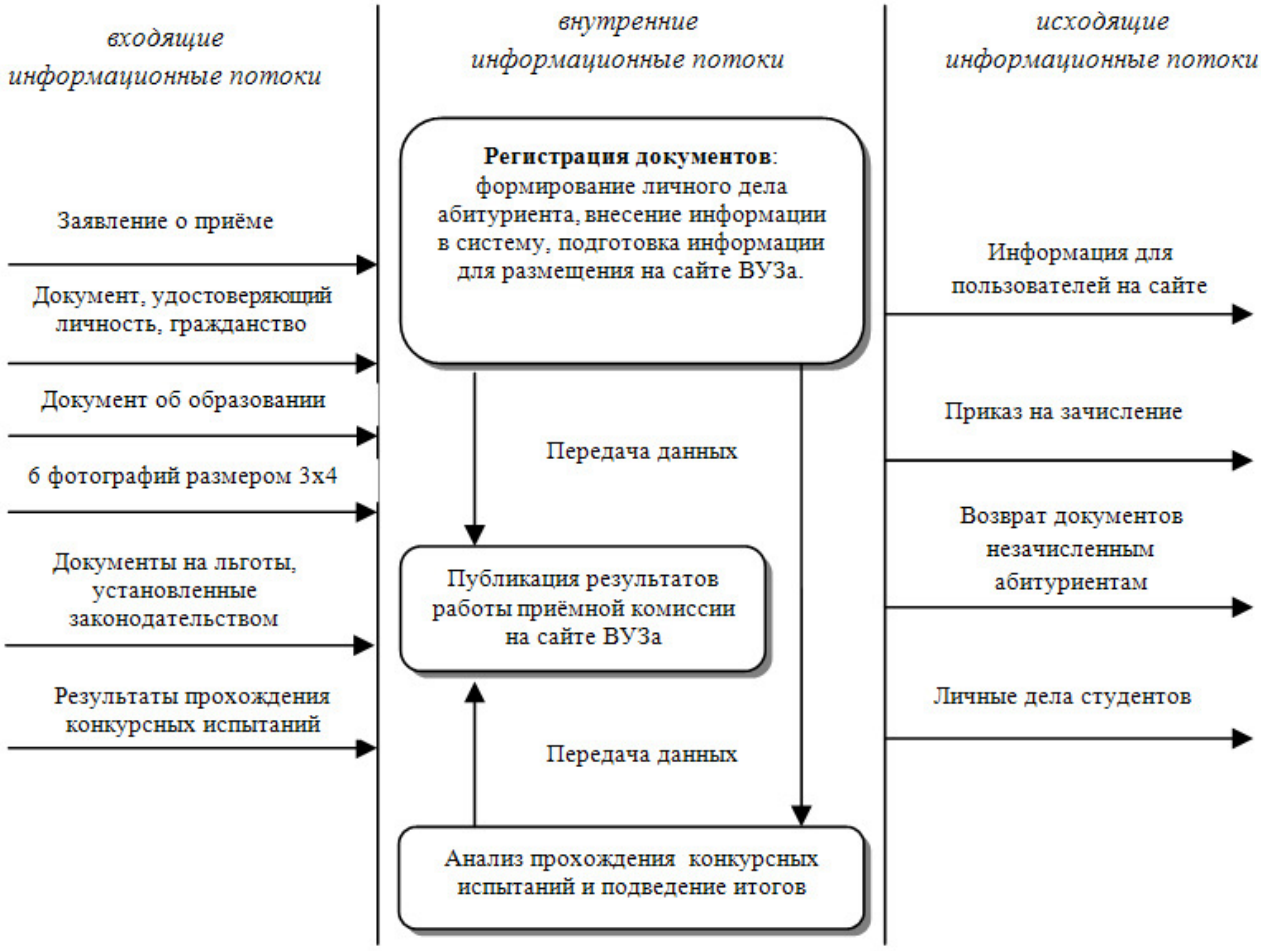


Рис. 26. Схема информационных потоков

Лабораторная работа №7

Пример решения задачи: проектирование ИС «Телефонный справочник»

Описание предметной области

Вашей задачей является создание телефонного справочника организации. Организация имеет различные подразделения. Каждое из них может иметь собственные подотделы. Один сотрудник может иметь несколько телефонных номеров и, наоборот, один телефон могут иметь несколько сотрудников. Необходимо создать справочник для поиска по подразделениям, сотрудникам и телефонам.

Классы объектов

- Сотрудники (Фамилия, Имя, Отчество, Подразделение).
- Подразделения (Наименование подразделения).
- Телефоны (Номер телефона, Тип телефона, Сотрудник).

Развитие постановки задачи

Нужно учесть, что один сотрудник может работать в разных подразделениях. Например, сотрудники в подразделении «ответственные за пожарную безопасность» работают и в других подразделениях (по основному месту работы).

Построение UML-диаграмм

Прежде всего, необходимо определить действующие лица, которые будут взаимодействовать с системой: управлять ею и использовать в конечном итоге. В данном случае – необходимо определить лицо, которое будет работать с телефонным справочником. Назовём его – Пользователь.

После этого необходимо определить прецеденты – то есть «блоки работ» внутри проектируемой системы, применяемы для обработки и генерации информационных потоков. В данном случае прецеденты будут определяться обозначенными в задании классами объектов:

Таким образом диаграмма прецедентов (или вариантов использования) будет иметь следующий вид (рис. 27):

Для того, чтобы лучше понять как реализуются вышеобозначенные варианты использования необходимо рассмотреть диаграммы последовательности (рис. 28, 29, 30) каждого из этих вариантов, где более подробно описаны потоки событий.

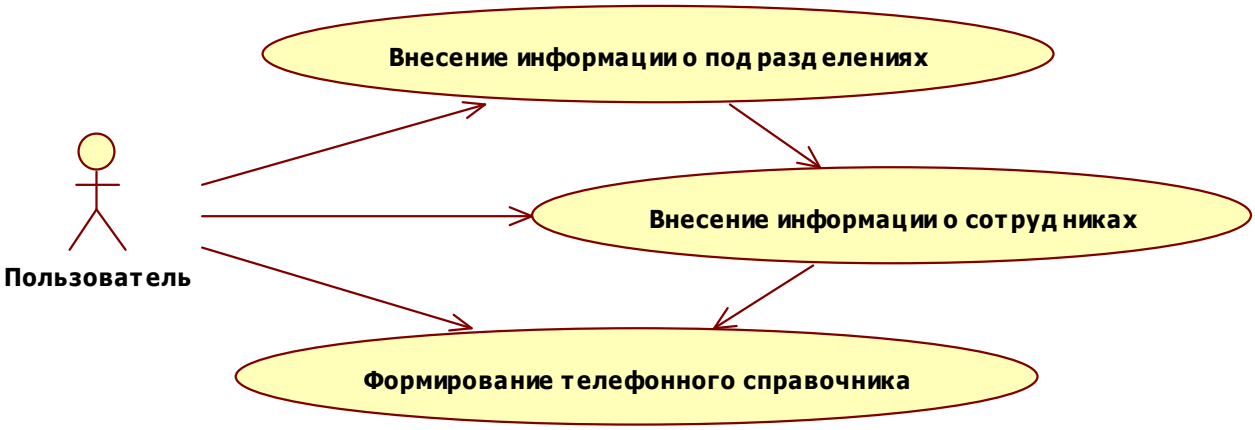


Рис. 27. Диаграмма вариантов использования

Диаграмма последовательности «Внесение информации о подразделениях» (рис. 28) рассматривает комплекс действий, которые необходимо провести для того, чтобы ввести в систему данные о новом подразделение или отредактировать уже имеющиеся в системе данные. Данные вводятся через экранную форму, при этом пользователь выбирает одно из действий над системой: ввод нового объекта или редактирование уже имеющегося:

Рис. 28. Диаграмма последовательности «Внесение информации о подразделениях»

Диаграмма последовательности «Внесение информации о сотрудниках» (рис. 29) аналогично предыдущей диаграмме, описывает действия ввода нового объекта и редактирования уже имеющихся, только в данном случае в качестве объекта выступает – сотрудник, сведения о подразделении к которому принадлежит сотрудник выбирается из БД.

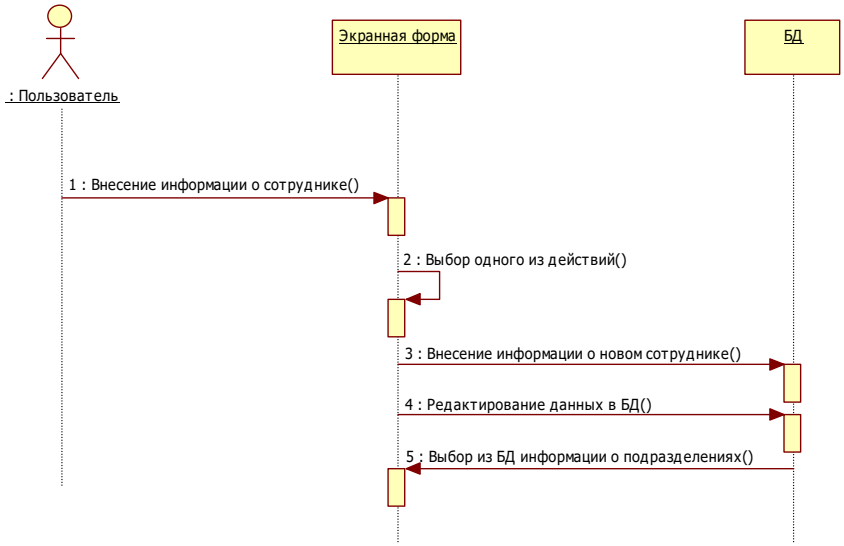


Рис. 29. Диаграмма последовательности «Внесение информации о сотрудниках»

Диаграмма последовательности «Внесение информации о телефонах» (рис. 30) так же описывает ввод новых данных и редактирование старых – в данном случае данных о номерах телефонов сотрудников:

Рис. 30. Диаграмма последовательности «Формирование телефонного справочника»

Построение диаграмм методологии IDEF0

Эти же события необходимо рассмотреть с точки зрения моделирования бизнес-процессов, соответственно прибегнув к помощи CASE-средств.

Таким образом, необходимо построить функциональную модель (методология IDEF0) предназначенную для описания существующих процессов в проектируемой информационной системе «Телефонный справочник» (рис. 31). В рамках методологии IDEF0 (Integration Definition for Functional Modeling) процесс представляется виде набора элементов - работ, которые взаимодействуют между собой, а так же показываются информационные, трудовые и производственные ресурсы, потребляемые каждой работой.

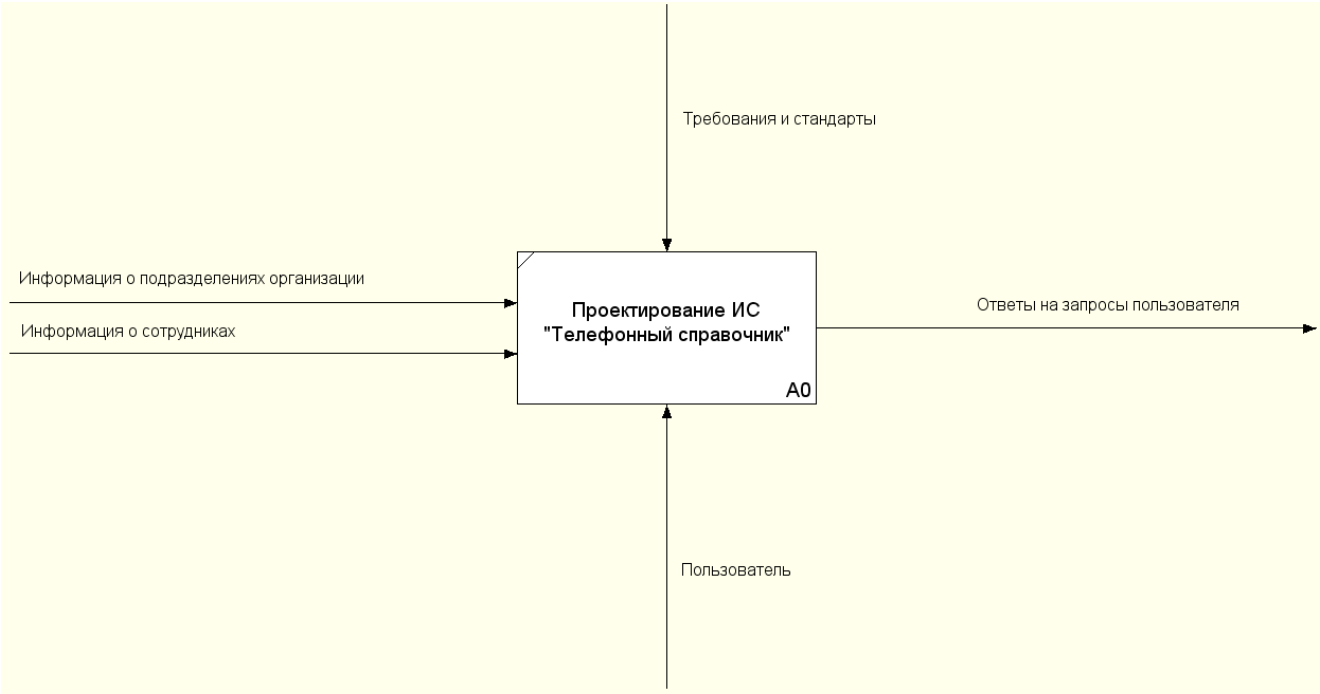


Рис. 31. Контекстная диаграмма «Проектирование ИС «Телефонный справочник»

Рассматривая данную диаграмму видно, что система представляет собой электронный телефонный справочник. Но по данной схеме нельзя сказать, как движется информация внутри проектируемой ИС, поэтому необходимо её декомпозировать на три работы, каждая из которых - отдельный участок учёта и обработки информации (рис. 32):

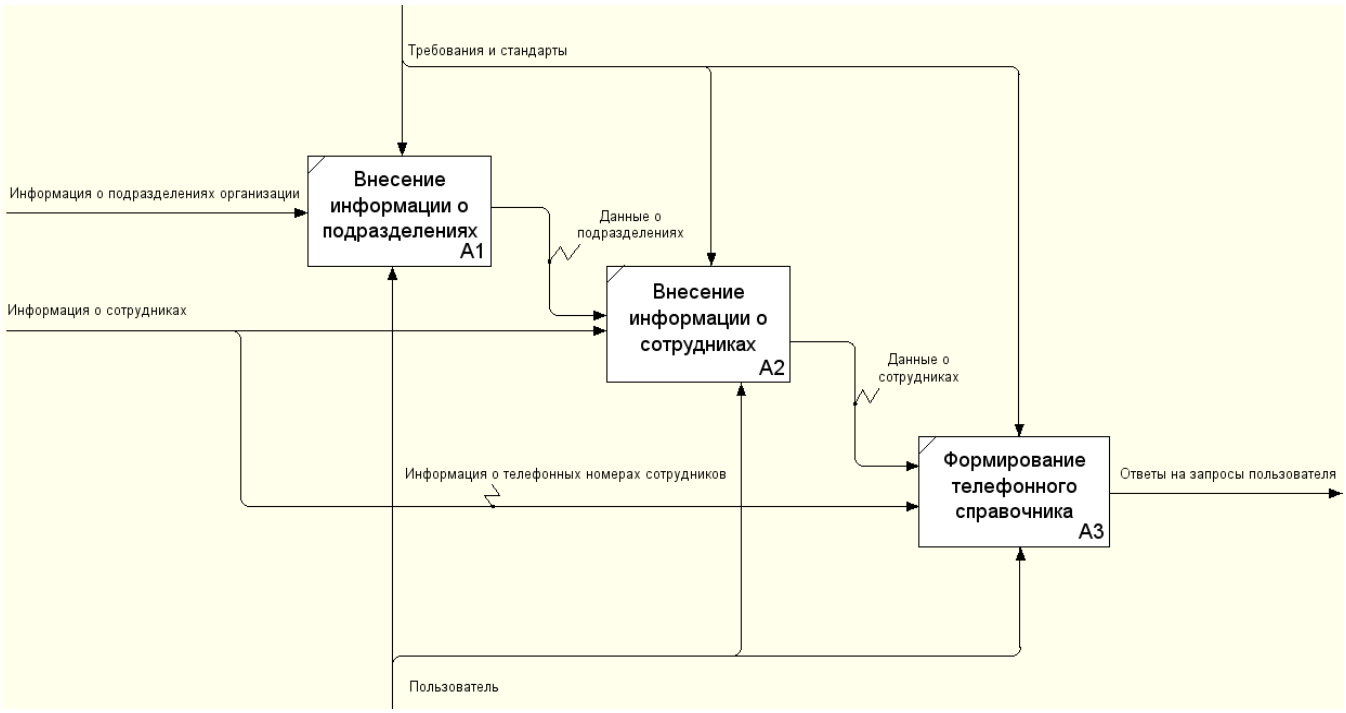


Рис. 32. Диаграмма декомпозиции «Проектирование ИС «Телефонный справочник»

Построение ERD-диаграмм

Далее следует перейти к проектированию логической модели. На логическом уровне данные представляются так, как они выглядят в реальном мире. Модель логического уровня является универсальной и не связана с конкретной базой данных. Первым шагом при создании логической модели БД является построение ERD-диаграммы.

ERD-диаграмма состоит из трёх частей:

1) сущности;
2) атрибуты;
3) взаимосвязи.

Сущностями являются существительные, атрибутами - модификаторы или прилагательные, взаимосвязями - глаголы. ERD-диаграмма позволяет рассмотреть систему целиком и выяснить требования, необходимые для её разработки, касающиеся хранения информации.

Рассмотрим процесс построения логической модели проектирования ИС «Телефонный справочник». Первым этапом является определение сущностей и атрибутов. В базе данных проектируемой ИС будут храниться сведения о подразделениях организации, сотрудниках и телефонных номерах - следовательно сущностями будут: «Подразделение», «Сотрудник» и «Телефон» (табл. 4, 5,6):

Таблица 4

Атрибуты сущности «Подразделение»

АТРИБУТ	ОПИСАНИЕ
Код подразделения	Уникальный номер для идентификации связи
Наименование подразделения	Наименование подразделения организации

Таблица 5

Атрибуты сущности «Сотрудник»

АТРИБУТ	ОПИСАНИЕ
Код сотрудника	Уникальный номер для идентификации связи
Фамилия	Фамилия сотрудника
Имя	Имя сотрудника
Отчество	Отчество сотрудника

Таблица 6

Атрибуты сущности «Телефон»

АТРИБУТ	ОПИСАНИЕ
Номер телефона	Уникальный номер для идентификации связи, номер телефона сотрудника
Тип телефона	Тип телефона (например, рабочий, личный, домашний и т.п.)
Код сотрудника	Внешний ключ

Далее необходимо определить типы связей между сущностями:

- «Подразделения» - «Сотрудники» - связь типа "многие ко многим", т. е. в одном подразделении может работать несколько сотрудников и один сотрудник может совмещать работу в нескольких подразделениях. Согласно требованиям нормализации данных такой тип связи недопустим, поэтому вводим промежуточную (ассоциативную) сущность – «Вид занятости» (атрибуты которой представлены в табл. 7).
- «Сотрудники» - «Телефоны» - связь типа "один ко многим", т. е. одному сотруднику может принадлежать несколько телефонных номеров, но каждый телефонный номер принадлежит только одному сотруднику.

Таблица 7

Атрибуты сущности «Вид занятости»

АТРИБУТ	ОПИСАНИЕ
Номер строки	Уникальный номер для идентификации связи
Код сотрудника	Внешний ключ
Код подразделения	Внешний ключ
Вид занятости	Вид занятости сотрудника (основное место работы или совместительство)

Таким образом, логическая модель будет иметь следующий вид (рис. 33):

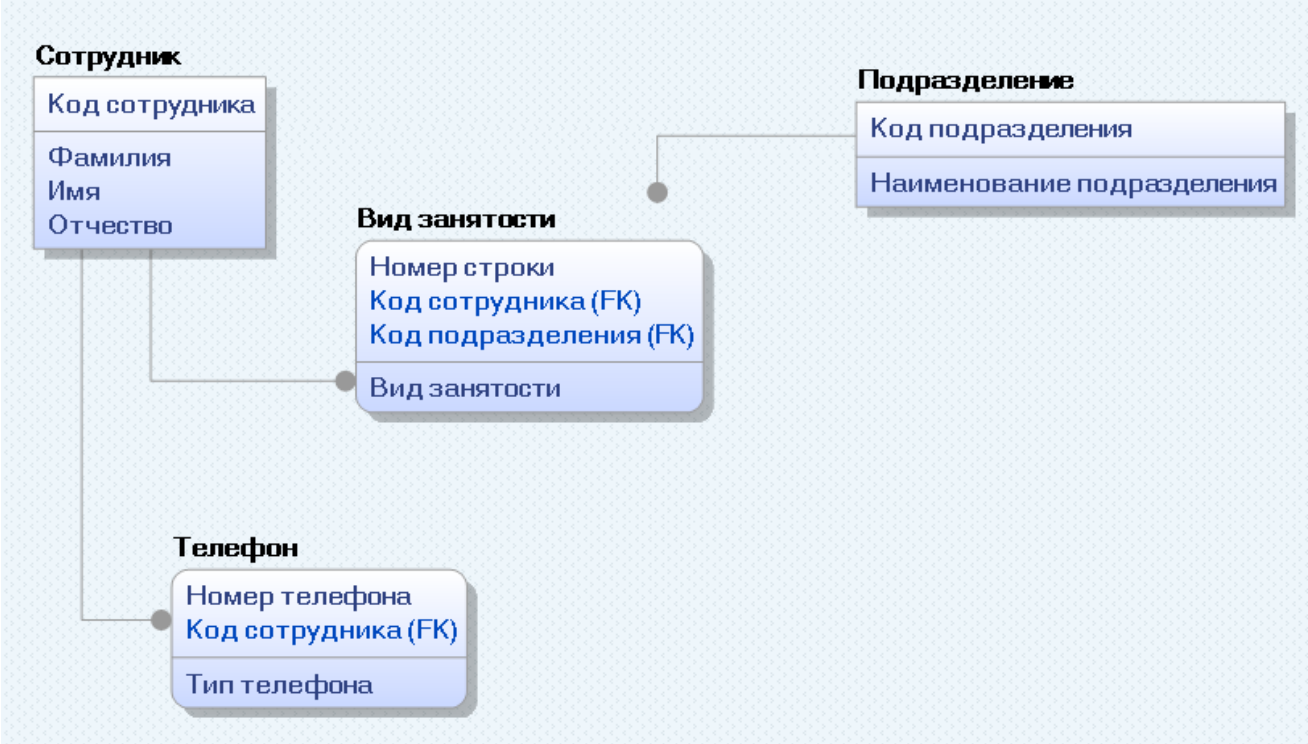


Рис. 33. Логическая модель данных

Задание: на основании спроектированной логической модели данных создайте БД «Телефонный справочник» в MS Access. Реализуйте в ней соответствующие запросы на поиск и выборку данных:

- поиск номера телефона по коду сотрудника;
- выборка данных о телефонных номерах сотрудников определённого подразделения.

Лабораторная работа №8

Расчёт трудоёмкости разработки

Задание: *определить трудоёмкость разработки спроектированной в ходе выполнения лабораторной работы №7 ИС «Телефонный справочник».*

Расчёт трудоёмкости разработки осуществляется на основе вариантов использования и содержит в себе следующие блоки работ (рис. 34):

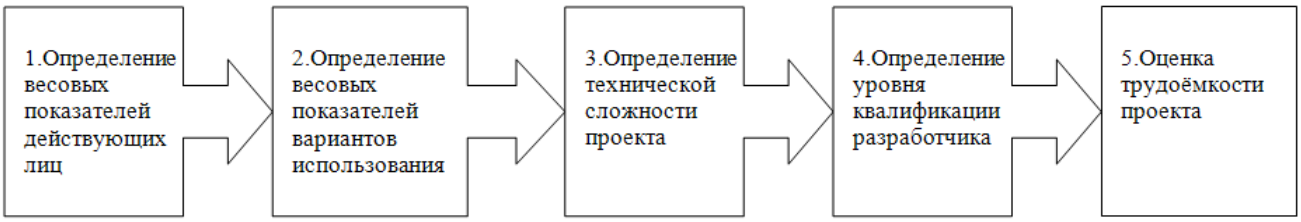


Рис. 34. Схема определения трудоёмкости проекта

Первый блок предполагает определение весовых показателей действующих лиц. Все действующие лица системы делятся на 3 типа: простые, средние, сложные.

Простое действующее лицо представляет собой внешнюю систему с четко определенным программным интерфейсом.

Среднее – это личность, пользующаяся текстовым интерфейсом.

Сложное – это личность, пользующаяся графическим пользовательским интерфейсом. Типы действующих лиц и их весовые коэффициенты представлены в табл. 8:

Таблица 8

Весовые коэффициенты действующих лиц

Тип действующего лица	Весовой коэффициент
Простое	1
Среднее	2
Сложное	3

В данном случае (рис. 27) в системе имеется одно действующее лицо – Пользователь. Предполагается, что работать с системой пользователь будет посредствам программной оболочки через графический интерфейс, значит тип данного действующего лица – сложное и соответственно весовой коэффициент 3.

«Общий весовой показатель» будет равен: **A**=1*3=3

Второй блок - «Определение весовых показателей вариантов использования». Все варианты использования делятся на три типа: простые, средние и сложные в зависимости от количества транзакций в потоках событий (основных и альтернативных). В данном случае под транзакцией понимается

атомарная последовательность действий, которая выполняется полностью или отменяется. Описание типов вариантов использования и их весовые показатели представлены в табл. 9:

Таблица 9

Весовые коэффициенты вариантов использования

Тип варианта использования	Описание	Весовой коэффициент
Простой	до 3 транзакций	5
Средний	от 4 до 7 транзакций	10
Сложный	более 7 транзакций	15

Все варианты использования, представленные в системе по типу определены как «средние», в соответствии с этим им были присвоены следующие весовые коэффициенты (табл. 10):

Таблица10

Сложность вариантов использования

Вариант использования	Тип варианта использования	Весовой коэффициент
Внесение информации о подразделениях	Средний	10
Внесение информации о сотрудниках	Средний	10
Формирование телефонного справочника	средний	10

Общее количество вариантов использования каждого типа умножается на соответствующий весовой коэффициент, затем вычисляется общий весовой показатель. Таким образом, общий весовой показатель равен: **ИС**=3*10=30.

В результате показатель **ИИСР**=А+ИС=3+30=33.

Далее необходимо определить техническую сложность, то есть перейти к **третьему блоку** работ. Техническая сложность проекта (TCF) вычисляется с учетом показателей технической сложности (табл. 11):

Таблица 11

Характеристика проекта

Показатель	Вес	Значение	Значение с учётом веса
T1	2	4	8
T2	1	3	3
T3	1	5	5

T4	1	1	1
T5	1	0	0
T6	0,5	5	2,5
T7	0,5	5	2,5
T8	2	0	0
T9	1	4	4
T10	1	5	5
T11	1	3	3
T12	1	5	5
T13	1	1	1
СУММА	-		40

Значение TCF вычисляется по формуле:

TCF = 0,6 + (0,01 * (∑Ti * Beci))

и в данном случае будет равна:

TCF=0.6+(0.01*(∑Ti*Beci))=0.6+(0.01*40)=1,0

Четвёртый блок предполагает определение уровня квалификации разработчика. Оценка уровня квалификации проводится по восьми различным показателям (табл. 12):

Таблица 12

Определение уровня квалификации разработчика

Показатель	Название показателя	Вес	Значение	Значение с учётом веса
F1	Знакомство с технологиями	1,5	3	4,5
F2	Опыт разработки приложений	0,5	3	1,5
F3	Опыт использования объектно-ориентированного подхода	1	3	3
F4	Наличие ведущего аналитика	0,5	4	2
F5	Мотивация	1	5	5
F6	Стабильность требований	2	4	8
F7	Частичная занятость	-1	0	0
F8	Сложность языка проектирования	-1	3	-3
СУММА	-			21

Уровень квалификации разработчиков (EF) вычисляется по формуле:

EF = 1,4 + (-0,03 * (∑Fi * Beci)).

В данном случае:

EF=1.4+(-0.03(∑Fi*Beci))=1.4+(-0.03*21)=1.4-0.63=0.77

В результате получаем окончательное значение ИСР:

ИСР=ИИСР*TCF*EF=33*1,0*0,77=25,41

Таким образом, оценив все выше перечисленные показатели, можно перейти к «Оценке трудоёмкости проекта»: В качестве начального значения предлагается использовать 20 Чел.часов на 1 ИСР. Таким образом общее время разработки составляет 25,41*20=508,2 Чел.часов, что составляет 12,7 недель при 40 часовой рабочей неделе (приблизительно 3 месяца).

Лабораторная работа №9

Определение затрат на разработку

Рассчитаем стоимостные показатели, влияющие на разработку информационной системы.

Таблица 13

Расчет электроэнергии для восьмичасового рабочего дня

Наименование	кол-во	кВт/час	кВт в сутки (примерно)	кВт в месяц
Компьютер	1	0,25	2	44
Освещение	2	0,36	5,76	126,72
ИТОГО:				168,72

Стоимость 1 кВт/ч для предприятия - 4,65 рублей. Трудоемкость 3 месяца. Затраты на электроэнергию равны:

За весь период разработки проекта, затраты на электроэнергию составят:

$$З_{эл} = 168,72 * 4,65 * 3 = 2353 \text{ руб.}$$

Так же следует рассчитать заработную плату (табл.14). Проект выполнялся усилиями одного разработчика под контролем руководителя. При этом будем считать, что заработная плата разработчика будет рассчитываться исходя из оклада и времени, затраченного на разработку, а заработная плата руководителя будет обозначена единой суммой за весь период руководства проектом разработки ИС.

Таблица 14

Расчет заработной платы

Должность	Оклад, руб.	Количество выработанного времени	Сумма, руб.	Налоги с ФОТ (30,28%)
Руководитель проекта	-	3 мес.	3000	908
Разработчик ИС «Телефонный справочник»	4000	3 мес.	12000	3634
ИТОГО:			15000	4542
Всего расходов:			19542	

Расчет суммы амортизационных отчислений. Годовая сумма амортизационных отчислений рассчитывается по формуле:

$$A = \frac{Ф * N_A}{100\,\%}$$

где Ф – первоначальная стоимость основных фондов по видам, руб.;

N_A – норма амортизации по видам основных фондов, в %.

Годовую сумму амортизационных отчислений отразим в таблице10.

Таблица 15

Расчет годовой суммы амортизационных отчислений

Элементы основных фондов	Кол-во	Стоимость, руб.	Сумма руб.	Норма амортизации, %	Амортизационные отчисления, руб.
Компьютер	1	30000	30000	20	6000
Помещение	5 $м^2$	2000	10000	3	300
ИТОГО:					6300

Исходя из того, что трудоемкость разработки составляет 3 месяца, рассчитаем амортизацию оборудования за этот период по формуле:

$$A_{факт} = \frac{A_{год} * T_{факт}}{12},$$

где $T_{факт}$ – трудоемкость, мес.

$$A_{факт} = \frac{6300 * 3}{12} = 1575 \text{ руб.}$$

Таблица 16

Затраты на расходные материалы и инструменты

Наименование	Расход, шт.	Стоимость, руб.	Сумма, руб.
Интернет соединение	1	350	350
Диски CD-R	1	20	60
ИТОГО			410

Так же при расчете затрат необходимо учесть накладные расходы. Они составляют 10% от общих затрат на разработку электронного учебно-методического пособия.

Накладные расходы - дополнительные к основным затратам расходы, необходимые для обеспечения процесса разработки, связанные с управлением, обслуживанием, содержанием и эксплуатацией оборудования плюс ненормированные расходы.

Проведём расчет затрат на разработку электронного варианта учебно-методического пособия (табл. 17).

Таблица 17

Затраты на разработку

№ п/п	Статьи расходов	Затраты (руб.)
1	Фонд оплаты труда + налог с ФОТ	19542
2	Амортизационные отчисления	1575
3	Затраты на электроэнергию	2353
4	Расходные материалы и инструменты	410
5	Накладные расходы	2388
ИТОГО		26262

Таким образом, суммарные затраты на создание системы составляют 26262 руб. Структура затрат представлена на рис. 34

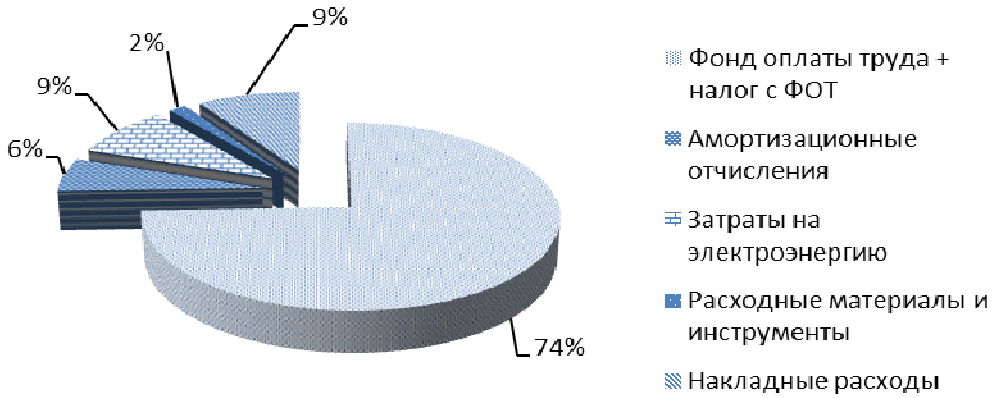


Рис. 34. Структура затрат на разработку

Исходя из полученных данных можно сделать вывод, что большая часть затрат приходится на оплату труда и налоги с ФОТ – 19542 руб., что составляет 74 % всех затрат; амортизационные отчисления составили 1575 руб. , 6% общих затрат; затраты на электроэнергию - 2353 руб., 9% всех затрат; накладные расходы - 2388 руб.- 9%. Меньшая доля затрат приходится на Расходные материалы и инструменты- 410 руб., что составило 2 % всех затрат.

Литература

1. Барановская, Т.П. Информационные системы и технологии в экономике/ Т.П. Барановская, В.И. Лойко, М.И. Семенов, А.И. Трубилин. – М.: Финансы и статистика, 2009.
2. Вендров, А.М. Проектирование программного обеспечения экономических информационных систем: учебник / А.М. Вендров. – М.: Финансы и статистика, 2011.
3. Войтова, Н.А. Электронный вариант учебно-методического пособия по дисциплине «Проектирование информационных систем» [Электронный ресурс]: учебно-методическое пособие / Н.А. Войтова. – Электрон. дан. – Брянск: БГАУ, 2015. – Режим доступа: http://moodle.bgsha.com/ – Загл. с экрана.
4. Войтова, Н.А. Методические указания к выполнению лабораторных работ по дисциплине «Проектирование информационных систем» / Н.А. Войтова. – Брянск: БГАУ, 2015.
5. Гайдамакин, Н.А. Автоматизированные информационные системы, базы и банки данных / Н.А. Гайдамакин. – М.: Гелиос АРВ, 2008.
6. Грекул, В. И. Проектирование информационных систем / В. И. Грекул. – М.: Интернет-Университет Информационных Технологий, 2008
7. Дубейковский, В.И. Практика функционального моделирования с AIIFusion Process Modeler / В.И. Дубейковский. – М.: ДИАЛОГ МИФИ, 2011.
8. Ильина, О.П. Информационные технологии бухгалтерского учета / О.П. Ильина. – Питер, 2002.
9. Калянов, Г.Н. CASE структурный системный анализ (автоматизация и применение) / Г.Н. Калянов. – М.: ЛОРИ, 1996. 242с.
10. Казин, Ф.А. Проектный менеджмент в вузе. Учебные кейсы / под ред. Ф.А. Казина, Н.Р. Тойвонена [Электронный ресурс] – СПб.: НИУ ИТМО, 2012. – 182 с. – Режим доступа: http://window.edu.ru/resource/221/7822 СПб.: НИУ ИТМО, 2012
11. Карминский, А.М. Информатизация бизнеса / А.М. Карминский, С.А. арминский, П.В. Нестеров, Б.В. Черников. – М., “Финансы и статистика”, 2010.
12. Карпова, Т. Базы данных / Т. Карпова. – Питер, 2012.
13. Кетков, Ю.Л. Практика программирования: Visual Basic, C++Builder, Delphi / Ю.Л. Кетков, А.Ю. Кетков. – СПб.: БХВ Петербург, 2012.
14. Лещев, Д.В. Создание интерактивного WEB-сайта: учебный курс / Д.В. Лещев. – СПб.:Питер,2010.

15. Маклаков, С.В. Создание информационных систем с AIIFusion Modeling Suite / С.В. Маклаков. – М.: ДИАЛОГМИФИ, 2010.
16. Никифоров С.В. Введение в сетевые технологии / С.В. Никифоров. – М.: Финансы и статистика, 2011.
17. Омельченко, Л.Н. Visual FoxPro 8 / Л.Н. Омельченко. – СПб.: БХВ Петербург,2010.
18. Орлов, С.А. Технологии разработки программного обеспечения: Учебник для вузов / С.А. Орлов. – СПб.: Питер, 2010.
19. Петров, В.Н. Информационные системы / В.Н. Петров // учебник для вузов. – СПб.:Питер, 2012.
20. Тельнов, Ю.Ф. Проектирование экономических информационных систем: Учебник/; Под ред. Ю.Ф. Тельнова. – М.: Финансы и статистика, 2011.
21. Федоров, А. Базы данных / А. Федоров, Н. Елманова. – М. Компьютер пресс, 2011.
22. Чекалов, А.П. Базы данных: от проектирования до разработки приложений / А.П. Чекалов. –СПб.: БХВ-Петербург, 2010.
23. Черемных, С.В. Структурный анализ систем: IDEF-технологии / С.В. Черемных, И.О. Семенов, В.С. Ручкин. – М.: Финансы и статистика, 2005.
24. Ярочкин, В.И. Информационная безопасность / В.И. Ярочкин. – М.: Летописец, 2010.

Учебное издание

Войтова Надежда Александровна

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к выполнению лабораторных работ

по дисциплине «Проектирование информационных систем»

для студентов направления подготовки 09.03.03 Прикладная информатика, очной и заочной формы обучения

Компьютерный набор произвела Войтова Н.А.

Редактор Павлютина И.П.

Подписано к печати Формат 60х84. 1/16. Бумага печатная
П.л.3,5. Тираж 50 экз. Изд.№ 4102

Издательство Брянский ГАУ
243365, Брянская обл., Выгоничский р-н, п. Кокино, БГАУ