Федеральное агентство по образованию РФ
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ГОУВПО «АмГУ»)

УТВЕРЖДАЮ
Зав.кафедрой ИиУС
_____А.В.Бушманов
«____»_____2007г.

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС ДИСЦИПЛИНЫ

ПРОЕКТИРОВАНИЕ АСОИУ

для специальности
230102 – Автоматизированные системы обработки информации
и управления

Составитель: А.В.Бушманов

2007

Печатается по решению
редакционно-издательского совета
факультета математики
и информатики
Амурского государственного
университета

Проектирование автоматизированных систем обработки информации и управления для специальности 230102 «Информационные системы и технологии»: учебно-методический комплекс дисциплины. /Бушманов А.В. – Благовещенск. Изд-во Амурского гос. ун-та, 2007г.

ОГЛАВЛЕНИЕ

1. Выписка из государственного образовательного стандарта высшего профессионального образования ……4
2. Рабочая программа ……...5
3. График самостоятельной работы студентов ……16
4. Методические рекомендации по проведению самостоятельной работы студентов ……16
5. Перечень учебников, учебных пособий ……29
6. Краткий конспект лекций ……...31
7. Методические указания по выполнению курсового проекта …….68
8. Методические рекомендации по выполнению лабораторных работ ……74
9. Перечень программных продуктов, используемых в практике выпускников и учебно-методическое пособие …….98
10. Методические указания по применению современных ИТ для преподавания учебной дисциплины ……99
11. Методические указания по организации межсессионного и экзаменационного контроля знаний студентов ……133
12. Комплекты заданий для лабораторных работ и курсовых проектов ……134
13. Фонд тестовых и контрольных заданий для оценки качества заданий по дисциплине ……136
14. Комплект экзаменационных билетов ……141
15. Карта кадровой обеспеченности дисциплины ……..151

1. ВЫПИСКА ИЗ ГОСУДАРСТВЕННОГО ОБРАЗОВАТЕЛЬНОГО СТАНДАРТА ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

Направление подготовки дипломированного специалиста
654600 – Информатика и вычислительная техника

Специальность

220200 (230102) – Автоматизированные системы обработки информации и управления

Квалификация – *инженер*.

СД.10	**Проектирование АСОИУ** Общая характеристика процесса проектирования АСОИУ; структура информационно-логической модели АСОИУ, разработка функциональной модели; исходные данные для проектирования; разработка модели и защита данных; разработка пользовательского интерфейса; разработка проекта распределенной обработки; структура программных модулей; разработка алгоритмов; логический анализ структур АСОИУ; анализ и оценка производительности АСОИУ; управление проектом АСОИУ; проектная документация; инструментальные средства проектирования АСОИУ; типизация проектных решений; графические средства представления проектных решений.	170

РАБОЧАЯ ПРОГРАММА

По дисциплине: Проектирование АСОИУ.

Для специальности: 230102 – Автоматизированные системы обработки информации и управления.

Курс: 5 Семестр: 9

Лекции: 45 (час.) Экзамен 9 семестр.

Практические (семинарские) занятия: нет

Лабораторные занятия: 45 (час.)

Самостоятельная работа: 86 (час.)

Всего: 176 (час.)

Составитель: Бушманов А.В.

Факультет: Математики и информатики.

Кафедра: Информационные и управляющие системы.

1. Цель и задачи дисциплины, ее место в учебном процессе

1.1. Цель преподавания дисциплины

Цель преподавания дисциплины заключается в том, чтобы на основе предшествующих курсов учебного плана дать студентам завершающие знания в области современных научных и практических методов и моделей управления сложными автоматизированными, информационными и организационно-административными системами. Дать знания по проектированию и функционированию систем АСОИУ.

Основные разделы дисциплины определенные в Государственном образовательном стандарте высшего профессионального образования: Общая характеристика процесса проектирования АСОИУ; структура информационно-логической модели АСОИУ, разработка функциональной модели; исходные данные для проектирования; разработка модели и защита данных; разработка пользовательского интерфейса; разработка проекта распределенной обработки; структура программных модулей; разработка алгоритмов; логический анализ структур АСОИУ; анализ и оценка производительности АСОИУ; управление проектом АСОИУ; проектная документация; инструментальные

средства проектирования АСОИУ; типизация проектных решений; графические средства представления проектных решений.

1.2. Задачи изучения дисциплины

Изучая курс «Проектирование АСОИУ», студенты должны получить представление о системах обработки информации и управления, о проектировании систем, получить навыки разработки программ, выяснить какие методики при этом используются.

С целью приобретения навыков в решении основных вопросов проектирования АСОИУ, при их качественной формулировке при изучении дисциплины, необходимы практические шаги в виде курсового проектирования.

По окончании изучения курса студенты должны уметь проектировать и эксплуатировать АСОИУ в самых различных сферах человеческой деятельности, владеть соответствующими навыками.

1.3. Перечень разделов (тем) необходимых дисциплин

1.3.1. Информатика: Количество и качество информации. Единицы измерения информации. Информация и энтропия. Сообщения и сигналы. Кодирование и квантование сигналов. Информационный процесс в автоматизированных системах. Фазы информационного цикла и их модели. Информационный ресурс и его составляющие. Информационные технологии. Технические и программные средства информационных технологий. Основные виды обработки данных.

1.3.2. Электротехника и электроника: Основы электроники и электрические измерения. Элементная база современных электронных устройств. Источники вторичного электропитания. Усилители электрических сигналов. Импульсные и автогенераторные устройства. Основы цифровой электроники. Микропроцессорные средства. Электрические измерения и приборы.

1.3.3. Метрология, стандартизация и сертификация: Основные положения государственной системы стандартизации ГОС. Научная база стандартизации. Определение оптимального уровня унификации и стандартизации. Государственный контроль и надзор за соблюдением требований государственных стандартов. Основные цели и объекты сертификации. Термины и определения в области сертификации.

1.3.4. Безопасность жизнедеятельности: Безопасность функционирования автоматизированных и роботизированных производств. Безопасность в чрезвычайных ситуациях. Управление безопасностью жизнедеятельности. Правовые и нормативно-технические основы управления. Системы контроля требований безопасности и экологичности.

1.3.5. Алгоритмические языки и программирование: Основные этапы решения задач на ЭВМ; критерии качества программы; диалоговые программы; дружественность, жизненный цикл програм-

мы; постановка задачи и спецификация программы; способы записи алгоритма; программа на языке высокого уровня; стандартные типы данных; представление основных структур программирования: итерация, ветвление, повторение; процедуры; типы данных, определяемые пользователем; записи; файлы; динамические структуры данных.

1.3.6. Основы теории управления: общие принципы системной организации; устойчивость, управляемость и наблюдаемость; инвариантность и чувствительность систем управления; математические модели объектов и систем управления; формы представления моделей; методы анализа и синтеза систем управления; цифровые системы управления; использование микропроцессоров и микро-ЭВМ в системах управления; особенности математического описания цифровых систем управления, анализа и синтеза систем управления с ЭВМ в качестве управляющего устройства; программная реализация алгоритмов управления в цифровых системах.

1.3.7. Операционные системы: Назначение и функции операционных систем. Мультипрограммирование. Режим разделения времени. Многопользовательский режим работы. Режим работы и ОС реального времени. Универсальные операционные системы и ОС специального назначения. Классификация операционных систем.

1.3.8. Базы данных: Назначение и основные компоненты системы баз данных; обзор современных систем управления базами данных (СУБД); уровни представления баз данных; понятия схемы и подсхемы; модели данных; иерархическая, сетевая и реляционная модели данных; схема отношения; язык манипулирования данными для реляционной модели; реляционная алгебра и язык SQL; проектирование реляционной базы данных.

1.3.9. Сети ЭВМ и телекоммуникации: Классификация информационно-вычислительных сетей. Способы коммутации. Сети одноранговые и "клиент/сервер". Уровни и протоколы. Эталонная модель взаимосвязи открытых систем. Аналоговые каналы передачи данных. Способы модуляции. Модемы. Цифровые каналы передачи данных. Разделение каналов по времени и частоте. Характеристики проводных линий связи. Спутниковые каналы. Сотовые системы связи. Кодирование информации. Количество информации и энтропия. Самосинхронизирующиеся коды. Способы контроля правильности передачи информации. Алгоритмы сжатия данных. Локальные вычислительные сети. Методы доступа. Множественный доступ с контролем несущей и обнаружением конфликтов. Разновидности сетей Ethernet. Маркерные методы доступа. Сети Token Ring и FDDI. Высокоскоростные локальные сети. Организация корпоративных сетей.

Функции сетевого и транспортного уровней. Алгоритмы маршрутизации. Протоколы TCP/IP. Протоколы управления. Адресация в Internet. Особенности технологий Frame Relay, ATM, SDH.

1.3.10. Методы и средства защиты информации: Политика безопасности. Стандарты безопасности. Криптографические модели. Алгоритмы шифрования. Алгоритмы аутентификации пользователей. Многоуровневая защита корпоративных сетей. Защита информации в сетях. Требования к системам защиты информации.

1.3.11. Организация и планирование производства: Подготовка и организация высокотехнологичного производства; организация вспомогательных цехов и служб предприятия; стратегическое и оперативное планирование производства; методы управления производством и информационное обеспечение; методы разработки и принятия управленческих решений; методы управления персоналом, рациональная организация труда; мотивация, профессиональная адаптация и деловая карьера на предприятии.

1.3.12. Теоретические основы автоматизированного управления: Понятие автоматизированного и автоматического управления; модели и процесс принятия решения; автоматизированные системы управления производством, научным экспериментом, обучением, технологическим процессом; категориальные понятия системного подхода; организационная и функциональная структура систем; методика и примеры формализации систем.

1.3.13. Теория принятия решений: Понятия исследования операций и системного анализа; методологические основы теории принятия решений; задачи выбора решений, отношения, функции выбора, функции полезности, критерии; детерминированные стохастические задачи, задачи в условиях неопределенности; задачи скалярной оптимизации, линейные, нелинейные, дискретные; многокритериальные задачи.

1.3.14. Моделирование систем: Классификация видов моделирования; имитационные модели систем; математические схемы моделирования систем; планирование имитационных экспериментов с моделями систем; формализация и алгоритмизация процессов функционирования систем; концептуальные модели систем; принципы построения моделирующих алгоритмов; статистическое моделирование систем на ЭВМ.

1.3.15. Информационные технологии: Общая классификация видов информационных технологий и их реализация в промышленности, административном управлении, обучении; модели информационных процессов передачи, обработки, накопления данных; системный подход к решению функциональных задач и к организации информационных процессов; глобальная, базовая и кон-

кретные информационные технологии; особенности новых информационных технологий; модели, методы и средства их реализации, объектно-ориентированные среды, функциональное и логическое программирование, информационные технологии в распределенных системах, технологии разработки программного обеспечения.

1.3.16. Сетевые технологии: Классификация сетей; интеграция информационного сервиса пользователей; концепция архитектуры открытых систем как основа построения цифровых сетей интегрального обслуживания (ISDN); основные этапы построения сетей; иерархия моделей процессов в сетях; АТМ-технология; анализ и синтез топологической структуры магистральной и локальной сети; административное и оперативное управление сетью; управление режимами коммутации.

1.3.17. Технологии программирования: Основные этапы решения задач на ЭВМ; критерии качества программы; диалоговые программы; дружественность, жизненный цикл программы; постановка задачи и спецификация программы; способы записи алгоритма; стандартные типы данных; представление основных структур программирования; типы данных, определяемые пользователем; записи; файлы; динамические структуры данных; списки; программирование рекурсивных алгоритмов; способы конструирования программ; модульные программы.

1.3.18. Системы искусственного интеллекта: Искусственный интеллект как научное направление, представление знаний, рассуждений и задач; эпистомологическая полнота представления знаний и эвристически эффективные стратегии поиска решения задач; модели представления знаний: алгоритмические, логические, сетевые и продукционные модели; сценарии; экспертные системы: классификация и структура; инструментальные средства проектирования, разработки и отладки; этапы разработки; примеры реализации.

1.3.19. Системы реального времени: Аппаратурная среда, устройство связи с объектом; методы и средства обработки асинхронных событий; концепция процесса; ядро реального времени; механизмы синхронизации и взаимодействия процессов; языки программирования реального времени; программирование синхронной и асинхронной обработки данных.

1.3.20. Надежность, эргономика и качество АСОИУ: Основные понятия теории надежности; элементы, модели, функции, системы; основные расчетные модели для оценки показателей надежности аппаратуры, организация и проведение испытаний на надежность, модели надежности программного обеспечения, методы обеспечения надежности; виды избыточности, характеристика человека как звена АСОИУ.

2. Содержание дисциплины

2.1. Федеральный компонент
СД Цикл специальных дисциплин

2.2. Лекционные занятия

2.2.1. Введение. Основные направления проектирования Автоматизированных систем обработки информации и управления. Проблемы проектирования АСОИУ. Мировые концепции развития АСОИУ.(3 часа)

2.2.2. Тема 1. Понятие системы и ее характеристики. Способы классификации систем. (2 часа)

2.2.3. Тема 2. Системный и функциональный подход к проектированию АСОиУ. (2 часа)

2.2.4. Тема 3. Представление информации. Фазы управления. (2 часа)

2.2.5. Тема 4. Требования к разрабатываемым системам. (2 часа)

2.2.6. Тема 5. Требования, характеризующие различные количественные показатели информации. (2 часа)

2.2.7. Тема 6. Требования к временным характеристикам АСОИУ. (2 часа)

2.2.8. Тема 7. Требования к информации. (2 часа)

2.2.9. Тема 8. Технико-экономические требования к системам. (2 часа)

2.2.10. Тема 9. Состав и структура АСОИУ. Функциональные подсистемы. Обеспечивающие и управляющие подсистемы.(2 часа)

2.2.11. Тема 10. Разработка концептуальной модели будущей системы. (2 часа)

2.2.12. Тема 11. Технико-экономического обоснование и разработка технического задания на проект системы. (2 часа)

2.2.13. Тема 12. Предпроектное обследование. (2 часа)

2.2.14. Тема 13. Этап разработки проекта. (2 часа)

2.2.15. Тема 14. Этап ввода в эксплуатацию системы. Документация проекта. (4 часа)

2.2.16. Тема 15. Диаграммы потоков данных как средство анализа и синтеза алгоритма работы систем автоматизации. Принципы построения диаграмм потоков данных. Использование диаграмм в процессе разработки и реализации проекта АСОиУ. (4 часа)

2.2.17. Тема 16. Методология функционального проектирования. (2 часа)

2.2.18. Тема 17. Технология «клиент-сервер». (2 часа)

2.2.19. Тема 18. Жизненный цикл программного обеспечения ИС. Модели жизненного цикла программного обеспечения. Методология RAD. (2 часа)

2.2.20. Тема 19. Объектно-ориентированное программирование в процессах разработки систем АСОИУ. (2 часа)

2.3. Лабораторные занятия

2.3.1. Теоретическое введение в предметную область проектирования систем. (2 часа)

2.3.2. Методология IDEF0. (2 часа)

2.3.3. Дополнение моделей процессов диаграммами DFD и WorkFlow (IDEF3). (2 часа)

2.3.4. Отчеты в BPWin. (3 часа)

2.3.5. Методология IDEF1X. (3 часа)

2.3.6. Создание логической модели. (3часа)

2.3.7. Отчеты в ERWin. (3 часа)

2.3.8. Генерация кода в ERWin. (3 часа)

2.3.9. Введение в Case – пакет Rational Rose. (3 часа)

2.3.10. Диаграммы вариантов использования. (3 часа)

2.3.11. Диаграммы классов. (3 часа)

2.3.12. Диаграммы взаимодействия. (3 часа)

2.3.13. Диаграммы состояний. (3 часа)

2.3.14. Диаграммы пакетов, компонентов и размещения. (3 часа)

2.3.15. Генерация исходных текстов программ. (3 часа)

2.3.16. Обратное проектирование. (3 часа)

2.4. Курсовое проектирование

2.4.1. Курсовой проект включает в себя анализ, проектирование и реализацию систем автоматизации. Тематика проекта, как правило, связана с темой дипломной работы или научно-исследовательской работой студента.

2.4.2. Примерные темы курсовых проектов:

2.4.2.1. Проектирование автоматизированной системы контроля доступа в помещение Амурского комплексного научно-исследовательского института с применением пластиковых карточек.

2.4.2.2. Проектирование подсистемы складского учета.

2.4.2.3. Прямое адаптивное управление объектами с запаздыванием нейтрального типа с использованием нестационарных линейных компенсаторов.

2.4.2.4. Разработка программно-информационного и аппаратно-технического обеспечения автоматизированной системы связи между Управлением Федерального казначейства и его подразделениями по Амурской области.

2.4.2.5. Проектирование корпоративной сети малой ГИС общего назначения с использованием технологий Intranet/Internet.

2.4.2.6. Автоматизированная система учета оборота сырья и расчета себестоимости изготовленной продукции цеха мороженного ОАО «Хладокомбинат».

2.4.2.7. Проектирование информационной системы «Учет и анализ дорожно-транспортных происшествий» отдела административной практики при ГИБДД Амурской области.

2.4.2.8. Автоматизация документооборота Администрации города Благовещенска.

2.4.2.9. Проектирование информационной системы авиационного агентства «Амуркруизавиа».

2.4.2.10. Проектирование автоматизированной подсистемы управления отделом маркетинга ОАО «Благовещенский молочный комбинат».

2.4.2.11. Проектирование информационной системы отдела отчетности и статистики при ФАО ЦЭС.

2.4.2.12. Разработка системы электронного документооборота контрольного отдела Администрации Амурской области.

2.5. Самостоятельная работа студентов

Самостоятельная работа студента, это работа над рекомендованной литературой, контроль осуществляется оценкой курсового проекта.

Самостоятельная работа включает в себя изучение следующих разделов:

2.5.1. Общая характеристика процесса проектирования АСОИУ.

2.5.2. Разработка функциональной модели АСОИУ.

2.5.3. Исходные данные для проектирования, стандарты.

2.5.4. Проектирование пользовательского интерфейса.

2.5.5. Задачи информационного обеспечения, структура информационно-логической модели АСОИУ.

2.5.6. Распределенная обработка данных.

2.5.7. Структура программных модулей, разработка алгоритмов, логический анализ структур АСОИУ.

2.5.8. Разработка моделей защиты данных.

2.5.9. Автоматизация проектирования CASE – средствами.

2.5.9.1. Возможности и практика работы с пакетом BPWin, изучение структуры пакета и возможности использования при проектировании.

2.5.9.2. Возможности и практика работы с пакетом ERWin, изучение структуры пакета и возможности использования при проектировании.

2.5.9.3. Возможности и практика работы с пакетом Rational Rose 2003, изучение структуры пакета и возможности использования при проектировании.

2.6. Вопросы к экзамену

Понятие системы и ее характеристики.

Способы классификации систем.

Системный и функциональный подход к проектированию АСОИУ.

Основные характеристики управления предприятием.

Требования к разрабатываемым системам.

Требования к информации.

Основные методы позволяющие избежать информационных ошибок.
Безопасность в АСОИУ.
Технико-экономические требования АСОИУ.
Разработка концептуальной модели будущей системы.
Разработка ТЗ при создании АСОИУ.
ТЭО на разработку АСОИУ.
Понятие о технологической операции проектирования.
Методы проектирования САУ.
Выбор подходящих типовых проектных решений.
Обобщенная технологическая схема жизненного цикла ИС.
Функциональные подсистемы ИС.
Обеспечивающие подсистемы ИС.
Состав стадий и этапов канонического проектирования ИС.
Состав и содержание работ на предпроектной стадии создания ИС.
Технологическая сеть выполнения процесса работ на этапе «Анализ материалов обследования.
Состав и содержание работ на стадии техно-рабочего проектирования. Технологическая сеть.
Состав и содержание работ на стадиях внедрения, эксплуатации и сопровождения проекта.
Классификация экономической информации.

Экзамен предусматривает два теоретических вопроса.
Экзаменуемый студент должен подтвердить знание фундаментальных основ:

- ПАСОИУ (Проектирование Автоматизированных Систем Обработки Информации и Управления);
- знание и использование методологии проектирования АСОИУ;
- умение строить модели различных этапов жизненного цикла программного продукта.

2.7. Оценочные критерии

При оценке знаний на экзамене учитывается:

- правильность и осознанность изложения содержания ответа на вопросы, полнота раскрытия понятий и закономерностей, точность употребления и трактовки общенаучных и специальных терминов;
- степень сформированности интеллектуальных и научных способностей экзаменуемого;
- самостоятельность ответа;
- речевая грамотность и логическая последовательность ответа.

Оценка "отлично":

- полно раскрыто содержание вопросов в объеме программы и рекомендованной литературы;

- четко и правильно даны определения и раскрыто содержание концептуальных понятий, закономерностей, корректно использованы научные термины;
- для доказательства использованы различные теоретические знания, выводы из наблюдений и опытов;
- ответ самостоятельный, исчерпывающий, без наводящих дополнительных вопросов, с опорой на знания, приобретенные в процессе специализации по выбранному направлению информатики.

Оценка "хорошо":

- раскрыто основное содержание вопросов;
- в основном правильно даны определения понятий и использованы научные термины;
- ответ самостоятельный;
- определения понятий неполные, допущены нарушения последовательности изложения, небольшие неточности при использовании научных терминов или в выводах и обобщениях, исправляемые по дополнительным вопросам экзаменаторов.

Оценка "удовлетворительно":

- усвоено основное содержание учебного материала, но изложено фрагментарно, не всегда последовательно;
- определение понятий недостаточно четкое;
- не использованы в качестве доказательства выводы из наблюдений и опытов или допущены ошибки при их изложении;
- допущены ошибки и неточности в использовании научной терминологии, определении понятий.

Оценка "неудовлетворительно":

- ответ неправильный, не раскрыто основное содержание программного материала;
- не даны ответы на вспомогательные вопросы экзаменаторов;
- допущены грубые ошибки в определении понятий, при использовании терминологии.

3. Литература

3.1. Основная литература:

3.1.1. Управляющие вычислительные комплексы: Учеб. пособие/Под ред. Н.Л. Прохорова. -М.:Финансы и статистика, 2003. -352 С.

3.1.2. Администрирование локальных сетей Windows NT/2000/.NET: Учеб. пособие/Под ред. Г.А.Егорова -М.:Финансы и статистика, 2003. -480 С.

3.1.3. Архитектура компьютерных систем и сетей: Учеб. пособие/Под ред. В.И. Лойко. -М.:Финансы и статистика, 2003. -256 С.

3.1.4. Системный анализ в управлении: Учеб. пособие/Под ред. А.А. Емельянова. -М.:Финансы и статистика, 2002. -368 С.
3.1.5. А.А. Сабитов. Базы данных. Методическое пособие. – Новосибирск, 2002.

3.2. Дополнительная литература:

3.2.1 А.М. Вендеров CASE – технологии. Современные методы и средства проектирования информационных систем. – М.: Финансы и статистика, 1998.
3.2.2. Г.Н. Калянов. CASE. Структурный системный анализ (автоматизация и применение). М., "Лори", 1996.
3.2.3. А. Вишневский. Сетевые технологии Windows 2000 для профессионалов –СПб: Питер, 2000. -592 С.
3.2.4. Т. Кватрани. Визуальное моделирование с помощью Rational Rose 2002 и UML: Пер. с англ. – М.: Издательский дом «Вильямс», 2003. -192 С.
3.2.5. С.А.Трофимов. CASE – технологии: практическая работа в Rational Rose. Изд. 2-е. – М.: Бином – Пресс, 2002. -288 С.
3.2.6. О. Ю. Сабинин, В.В. Зверев. "Символьное имитационное моделирование технических систем". Приборы и системы управления, №3, 1997.
3.2.7. Т.Конноли, К.Бегг, А.Страчан. Базы данных: проектирование, реализация, сопровождение. Пер. с англ. М.: издательский дом «Вильямс», 2000, 1120 с.
3.2.8. К.Дж.Дейт. Введение в системы баз данных. 6-е издание. К., М.; СПб.: Издательский дом «Вильямс», 2000. 848 с.
3.2.9. С. Д. Кузнецов. Проектирование и разработка корпоративных информационных систем. Центр Информационных Технологий, 1998.
3.2.10. С. В. Маклаков “ERwin и BPwin. CASE-средства разработки информационных систем.” – М.: ДИАЛОГ-МИФИ, 2000 – 256 с.

3.3. Литература для самостоятельного изучения.

3.3.1. Создание информационной системы предприятия. "Computer Direct", 1996, N2.
3.3.2. Международные стандарты, поддерживающие жизненный цикл программных средств. М., МП "Экономика", 1996.
3.3.3. Ю.В. Новоженов. Объектно-ориентированные технологии разработки сложных программных систем. М., 1996.

4. Учебно – методическая (технологическая) карта дисциплины

Номер недели	Номер темы	изучаемые на лекцииВопросы,	Занятия		и методические пособияИспользуемые наглядные	Самостоятельная работа студентов		Форма контроля
			Практические	Лабораторные		Содержание	Часы	
1	2	3	4	5	6	7	8	9
1	Вв[1]	2.2.1	-	2.3.1	3.1.1	2.5.1	4	злр [2]
2	1	2.2.2	-	2.3.2	3.2.10	2.5.9.1	4	
3	2	2.2.3	-	2.3.3	3.2.10	2.5.9.1	4	злр
4	3	2.2.4	-	2.3.4	3.2.10	2.5.9.2	4	
5	4	2.2.5	-	2.3.4	3.2.10	2.5.9.2	4	злр
6	5	2.2.6	-	2.3.5	3.2.10	2.5.9.3	4	
7	6	2.2.7	-	2.3.5	3.2.10	2.5.9.3	4	злр
8	7	2.2.8	-	2.3.6	3.2.10	2.5.9.3	4	
9	8	2.2.9	-	2.3.7	3.2.10	2.5.9.3	4	злр
10	9	2.2.10	-	2.3.7	3.2.10	2.5.2	4	
11	10	2.2.11	-	2.3.8	3.2.10	2.5.2	4	злр,
12	11	2.2.12	-	2.3.9	3.2.4	2.5.3	4	сб.[3]
13	12	2.2.13	-	2.3.10	3.2.4	2.5.3	5	злр
14	13	2.2.14	-	2.3.11	3.2.4	2.5.4	5	
15	14	2.2.15	-	2.3.12	3.2.4	2.5.4	4	
16	15	2.2.16	-	2.3.13	3.2.5	2.5.5	4	
17	16	2.2.17	-	2.3.14	3.2.5	2.5.5	5	сб.
18	17	2.2.18	-	2.3.14	3.2.5	2.5.6	5	
19	18	2.2.19	-	2.3.15	3.2.5	2.5.7	5	
20	19	2.2.20	-	2.3.16	3.2.5	2.5.8	5	зач.[4]

1. Введение,
2. Защита отчета о выполнении лабораторной работы,
3. Зачет по лабораторным работам,

4. Собеседование по результатам.

3. ГРАФИК САМОСТОЯТЕЛЬНОЙ РАБОТЫ СТУДЕНТОВ

Содержание	Объем в часах	Сроки и форма контроля
Изучение основных этапов проведения проектирования в BPWin 44.1 и ERWin Data Modeler 4.1. 1. Инструментальная среда BPWin 4.1: Общее описание интерфейса BPWin 4.1; Создание новой модели; Установка цвета и шрифта объектов; Model Explorer – навигатор модели процессов. 2. Создание модели данных с помощью ERWin Data Modeler: Интерфейс ERWin. Уровни отображения модели; Подмножества модели и сохраняемые отображения; Modeler Explorer – навигатор модели данных.	15 час. 15 час.	Собеседование (8 неделя) Собеседование (8 неделя)

4. МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ПРОВЕДЕНИЮ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СТУДЕНТОВ

Выполнение самостоятельной работы по курсу «Проектирование АСОИУ», имеет целью выработку у студентов навыков, в том числе и в следующих направлениях:

1. Применение соответствующих методологий для разработки АСОИУ и программного обеспечения;
2. Применение соответствующего программного инструментария.

В настоящее время, в условиях бурно изменяющихся экономических факторов перед управляющими предприятиями все чаще встает задача реорганизации деятельности. Целью такой реорганизации может быть реорганизация штатной структуры, внедрение новой технологии, автоматизация (внедрение новой информационной системы) и т.д. – все то, что может резко по-

высить эффективность работы и поможет выжить в условиях нестабильного внешнего окружения. Бессмысленно планировать изменения и улучшения, если нет понимания существующего положения дел. Поэтому, первым шагом при проведении реорганизации является описание того, как в настоящий момент работает предприятие. Это не такая простая задача, как может показаться на первый взгляд. Никто из сотрудников, включая руководителя, не может полностью и достаточно подробно описать бизнес-процессы организации. Рядовые сотрудники, возможно, хорошо представляют, что происходит на их конкретном рабочем месте, но плохо знают, как работают их коллеги и, тем более, не представляют, как работает организация в целом. Руководитель хорошо знает, как работает предприятие в общем, но не в состоянии вникнуть в особенности деятельности на каждом рабочем месте. Следовательно, для того, чтобы получить адекватное описание функциональности организации нужно аккумулировать знание многих людей в единой модели. Такая модель может помочь найти слабые места в организации бизнеса и явиться основой для оценки стоимости производства продукции или обслуживания клиентов и, затем, может служить основой для построения идеальной модели – такого конечного состояния бизнес-процессов, к которому следует перейти, чтобы добиться необходимого результата.

Целостную и достаточно подробную модель можно получить, пользуясь специальными методологиями структурного анализа, такими как IDEF. IDEF0 была впервые предложена в конце шестидесятых Дугласом Россом (тогда она называлась SADT – Structured Analysis and Design Technique). Первоначально методология SADT предназначалась для моделирования технологических процессов, но вот уже более 20 лет она успешно применяется во всем мире сотнями компаний в самых разных областях деятельности. Согласно синтаксису IDEF0 модель представляет собой совокупность иерархически выстроенных диаграмм, каждая из которых является описанием какого-либо процесса (activity). Построение модели начинается с описания функциональности моделируемой системы в целом (контекстная диаграмма). Взаимодействие с окружающим миром описывается в терминах входа (данные или объекты, потребляемые или изменяемые процессом), выхода (основной результат деятельности процесса, конечный продукт), управления (стратегии и процедуры, которыми руководствуется процесс) и механизмов (ресурсы, необходимые для процесса).

Кроме этого, в контекст входит описание цели моделирования, области (описания того, что будет рассматриваться как компонент системы, а что как внешнее воздействие) и точки зрения (позиции, с которой будет строиться модель). Обычно в качестве точки зрения выбирается точка зрения лица или объекта, ответственного за работу моделируемой системы в целом. Работы на диаграммах изображаются в виде прямоугольников (функциональные блоки). Каждая работа изображает какую-либо функцию или задачу и именуется глаголом или глагольной фразой, обозначающей действие, например «Изготовление изделия», «Обслуживание клиента» и т.д. Стрелки помечаются суще-

ствительным и обозначают объекты или информацию, связывающую работы между собой и с внешним миром.

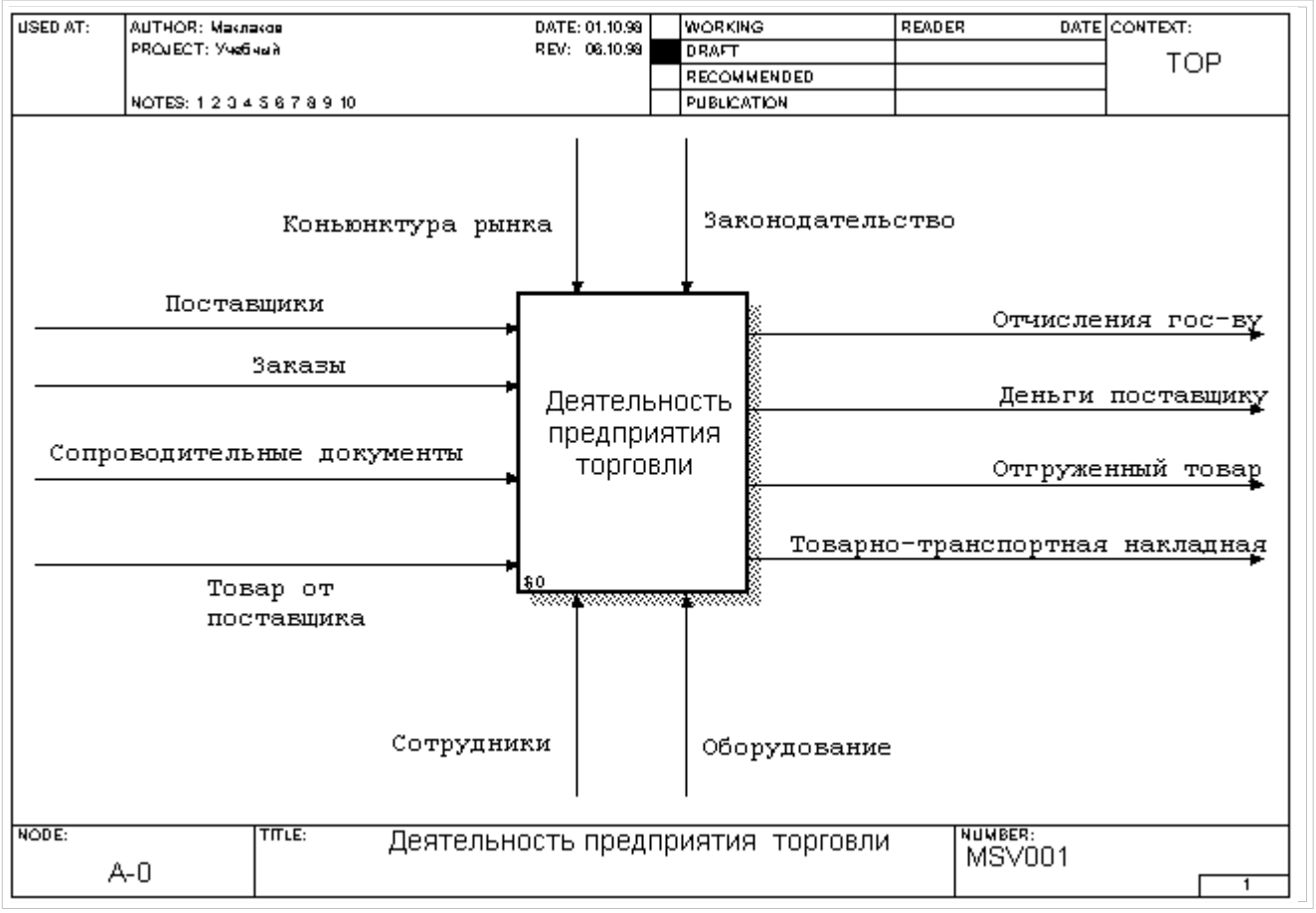


Рис.1 . Пример контекстной диаграммы.

После описания контекста проводится функциональная декомпозиция-система разбивается на подсистемы и каждая подсистема описывается в том же синтаксисе, что и система в целом. Затем каждая подсистема разбивается на более мелкие и так до достижения нужного уровня подробности. В результате такого разбиения, каждый фрагмент системы изображается на отдельной диаграмме декомпозиции (рис.2). Диаграмма декомпозиции предназначена для детализации работы. В отличие от моделей, отображающих структуру организации, работа на диаграмме верхнего уровня в IDEF0 – это не элемент управления нижестоящими работами. Работы нижнего уровня – это то же самое, что работа верхнего уровня, но в более детальном изложении. После каждого сеанса декомпозиции автором диаграммы формируется папка – набор документов, в который входит сама диаграмма, дополнительные отчеты и т.д. Папка направляется эксперту предметной области (т.е. человеку, хорошо разбирающемуся в моделируемом фрагменте деятельности предприятия) для проведения экспертизы. На уровне контекстной диаграммы это может быть управляющий предприятия, на уровне первой декомпозиции – начальник отдела и так далее вплоть до рядового исполнителя. Прежде чем декомпозировать далее, на текущем уровне необходимо внести в диаграмму все замечания экспертов. Таким образом, каждый из экспертов дополняет мо-

дель в той ее части, в которой он наиболее компетентен. В результате получается полностью адекватная системе модель, которая позволяет наглядно представить существующие недостатки, перенаправить и усовершенствовать бизнес – процессы, провести анализ стоимости производства, а также послужить основой для создания информационной системы.

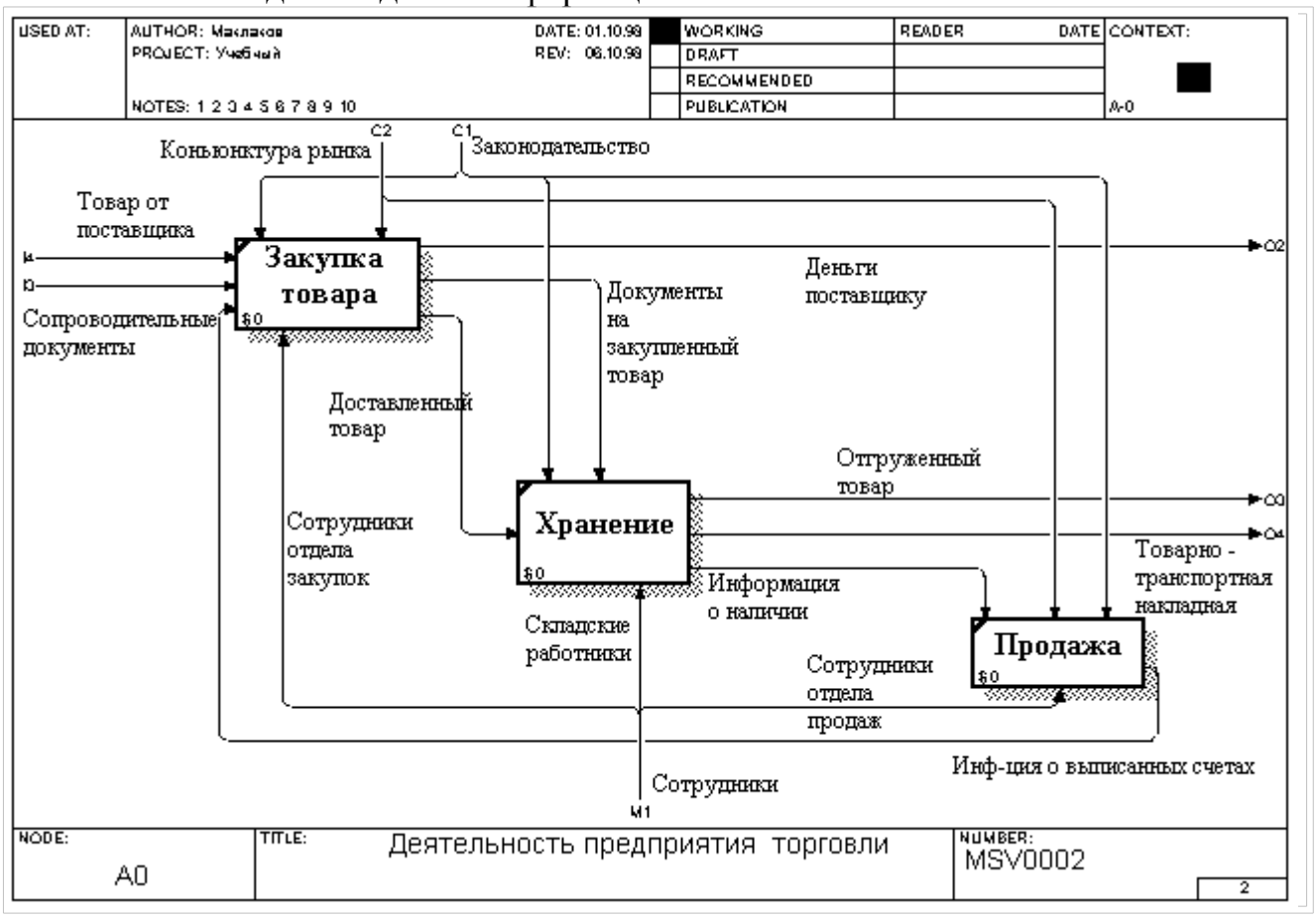


Рис. 2. Пример диаграммы декомпозиции.

С появлением персональных компьютеров с графическими интерфейсами, на рынке стали предлагаться многочисленные средства, автоматизирующие построение структурных моделей (CASE- средства) . Одним из таких CASE- средств является Platinum BPwin 2.5 (До слияния Logic Works с Platinum Technology inc. выпускался под именем Logic Works BPwin). От инструментов подобного типа BPwin выгодно отличается тем, что поддерживает помимо IDEF0 методологии DFD и IDEF3 и тесно интегрируется с другими программными продуктами:

- с широко известным инструментом моделирования данных ERwin (Platinum Technology inc.)
- ситемой управления и хранения проектов ModelMart (Platinum Technology inc.)
- специализированным генератором отчетов по модели RPTwin (Platinum Technology inc.)
- системой имитационного моделирования BPSimulator (System Modeling Corporation);
- инструментом стоимостного анализа EasyABC (ABC Technologies).

Каждая из трех методологий, поддерживаемых в BPwin, позволяет рассмотреть различные стороны деятельности предприятия. IDEF0 представляет собой функциональную модель, которая предназначена для описания бизнес-процессов на предприятии, позволяет понять какие объекты или информация служат сырьем для процессов, какие результаты производят работы, что является управляющими факторами и какие ресурсы для этого необходимы.

Диаграммы потоков данных (Data flow diagramming, DFD) используются для описания документооборота и обработки информации. Подобно IDEF0, DFD представляет модельную систему как сеть связанных меж собой работ. Их можно использовать как дополнение к модели IDEF0 для более наглядного отображения текущих операций документооборота в корпоративных системах обработки информации. DFD описывают функции обработки информации, документы, объекты, а также сотрудников или отделы, которые участвуют в обработке информации. Синтаксис DFD включает помимо работ и стрелок дополнительные элементы: внешнюю сущность, которая служит для изображения внешних по отношению к проектируемой системе объектов, (например, клиент, отдел кадров, справочники) и хранилище данных - "склад" информационных объектов. Хранилищем данных может быть база данных, файл или архив бумажных документов. Хранилище данных – это как бы «замороженные» данные, позволяющие отобразить отсрочку в передаче объектов и информации от одной работы к другой.

Наличие в диаграммах DFD элементов для описания источников, приемников и хранилищ данных позволяет более эффективно и наглядно описать процесс документооборота. Однако, для описания логики взаимодействия информационных потоков более подходит IDEF3, называемая также workflow diagramming, - методология моделирования, использующая графическое описание информационных потоков, взаимоотношений между процессами обработки информации и объектов, являющихся частью этих процессов. Перекресток – специфический для IDEF3 элемент – позволяет описать последовательность выполнения работ, очередность их запуска и завершения. С помощью диаграмм Workflow можно описывать сценарии действий сотрудников организации, например, последовательность обработки заказа или события, которые необходимо обработать за конечное время. Каждый сценарий сопровождается описанием процесса и может быть использован для документирования каждой функции, моделируемой в методологии IDEF0. Если в одной модели необходимо осветить специфические стороны бизнес-процессов предприятия, BPwin позволяет переключиться на любой ветви модели с IDEF0 на нотацию IDEF3 или DFD и создать смешанную модель. BPwin имеет мощный инструмент навигации - Model Explorer, который позволяет представить смешанную модель в виде дерева диаграмм и существенно облегчает навигацию по модели. В версии BPwin 2.5 с помощью Model Explorer можно методом Drag&Drop переносить и копировать работы вместе со всеми соответствующими стрелками как внутри модели так и между моде-

лями. Работы IDEF0 показываются в Model Explorer зеленым цветом, DFD – желтым и IDEF3 – синим.

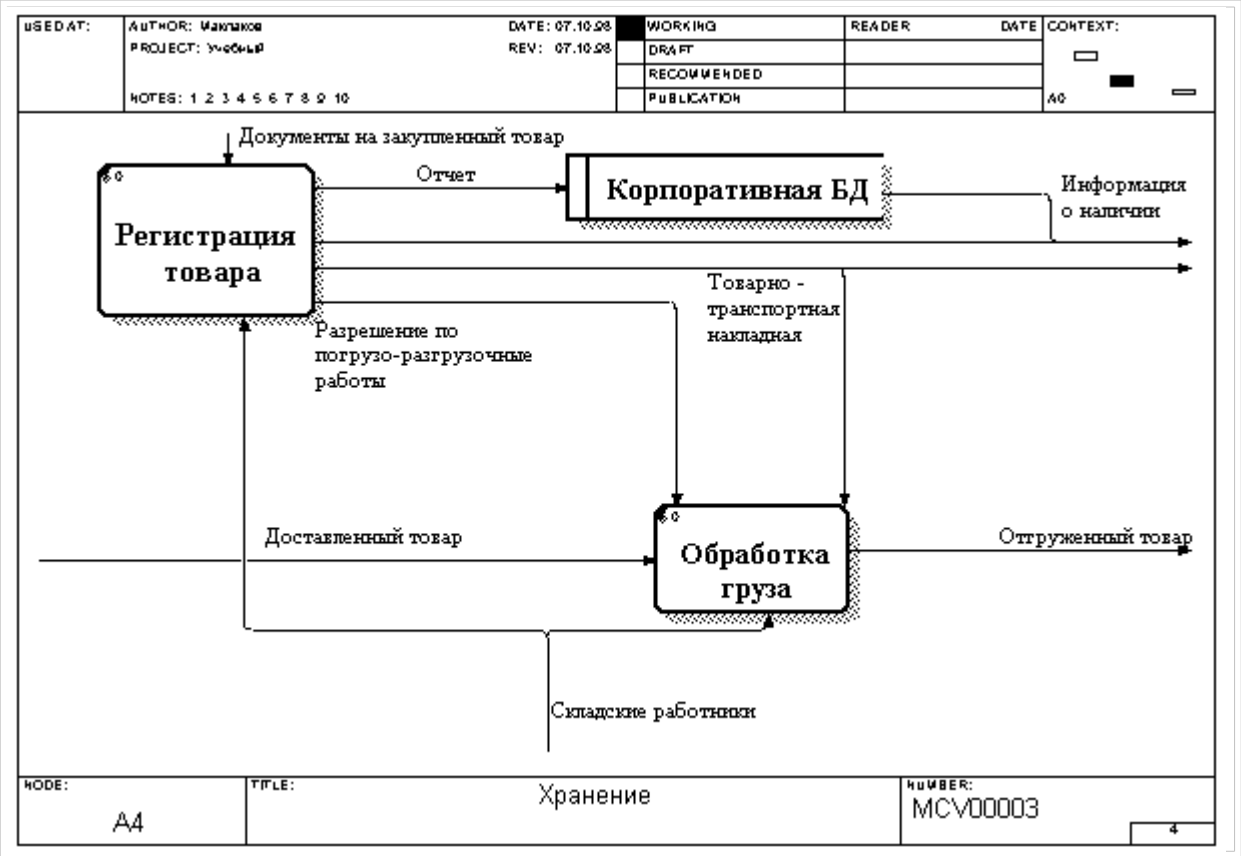


Рис. 3. Пример диаграммы DFD.

BPwin позволяет эффективно манипулировать моделями- сливать и расщеплять, документировать посредством генерации отчетов (всего 7 типов отчетов, поддерживаются запоминаемые определяемые пользователем стандартные отчеты) и т.д.

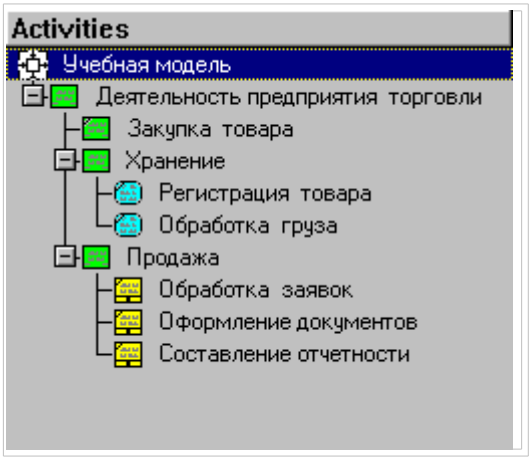


Рис. 4. Представление смешанной модели в Model Explorer.

BPwin в состоянии поддерживать синтаксис IDEF0, IDEF3и DFD. Частично ошибки анализируются на этапе внесения новых объектов (некоторые ошибочные элементы просто невозможно внести в диаграмму), частично – детектируются в специальном отчете - Model Consistency Report.

Обычно в целях реорганизации предприятия сначала строится функциональная модель существующей организации работы -«AS-IS» (как есть). На основе модели «AS-IS» достигается консенсус между различными единицами бизнеса по тому, «кто что сделал», и что каждая единица бизнеса добавляет в процесс. Модель «AS-IS» позволяет выяснить «что мы делаем сегодня» перед тем, как перепрыгнуть на то, «что мы будем делать завтра». Анализ функциональной модели позволяет понять, где находятся наиболее слабые места, в чем будут состоять преимущества новых бизнес-процессов и насколько глубоким изменениям подвергнется существующая структура организации бизнеса. Детализация бизнес-процессов позволяет выявить недостатки организации даже там, где функциональность на первый взгляд кажется очевидной. Признаком неэффективной деятельности могут быть бесполезные, неуправляемые и дублирующиеся работы, неэффективной документооборот (нужный документ не оказывается в нужном месте в нужное время), отсутствие обратных связей по управлению (на проведение работы не оказывает влияние ее результат) и входу (объекты или информация используются нерационально) и т.д.

Найденные в модели «AS-IS» недостатки можно исправить при создании модели «TO-BE» (как будет) - модели новой организации бизнес-процессов. Модель «TO-BE» нужна для анализа альтернативных/лучших путей выполнения работы и документирования того, как компания будет делать бизнес в будущем. Как правило, строятся несколько моделей «TO-BE», из которых по какому-либо критерию выбирается наилучшая. Проблема состоит в том, что таких критериев много и непросто определить важнейший. Для того, чтобы определить качество созданной модели с точки зрения эффективности бизнес- процессов, необходима система метрики, то есть качество следует оценивать количественно.

BPwin предоставляет аналитику два инструмента для оценки модели – стоимостной анализ, основанный на работах (Activity Based Costing, ABC) и свойства, определяемые пользователем (User Defined Properties, UDP). ABC является широко распространенной методикой, используемой международными корпорациями и государственными организациями (в том числе Департаментом обороны США) для идентификации истинных движителей затрат в организации.

Стоимостной анализ представляет собой соглашение об учете, используемое для сбора затрат, связанных с работами, с целью определить общую стоимость процесса. Стоимостной анализ основан на модели работ, поскольку количественная оценка невозможна без детального понимания в функциональности предприятия. Обычно ABC применяется для того, чтобы понять происхождение выходных затрат и облегчить выбор нужной модели работ при реорганизации деятельности предприятия (Business Process Re-engineering, BPR). С помощью стоимостного анализа можно решить такие задачи как определение действительной стоимости производства продукта, определение действительной стоимости поддержки клиента, идентификация работ, которые стоят больше всего (те, которые должны быть улучшены в

первую очередь), обеспечение менеджеров финансовой мерой предлагаемых изменений т.д.

ABC может проводиться только когда модель работы последовательная (следует синтаксическим правилам IDEF0), корректная (отражает бизнес), полная (охватывает всю рассматриваемую область) и стабильная (проходит цикл экспертизы без изменений). Эта методика включает основные понятия, такие как объект затрат (причина, по которой работа выполняется, обычно, основной выход работы, стоимость работ есть стоимость объектов затрат), движитель затрат (характеристики входов и управлений работы, которые влияют на то, как выполняется и как долго длится работа) и центры затрат (центры затрат можно трактовать как статьи расхода). При проведении стоимостного анализа в BPwin сначала задаются единицы измерения времени и денег, затем описываются центры затрат (cost centers) и, наконец для каждой работы на диаграмме декомпозиции назначаются продолжительность (duration), частота проведения данной работы в рамках общего процесса (frequency) и суммы по каждому центру затрат, то есть задается стоимость каждой работы по каждой статье расхода (рис. 5).

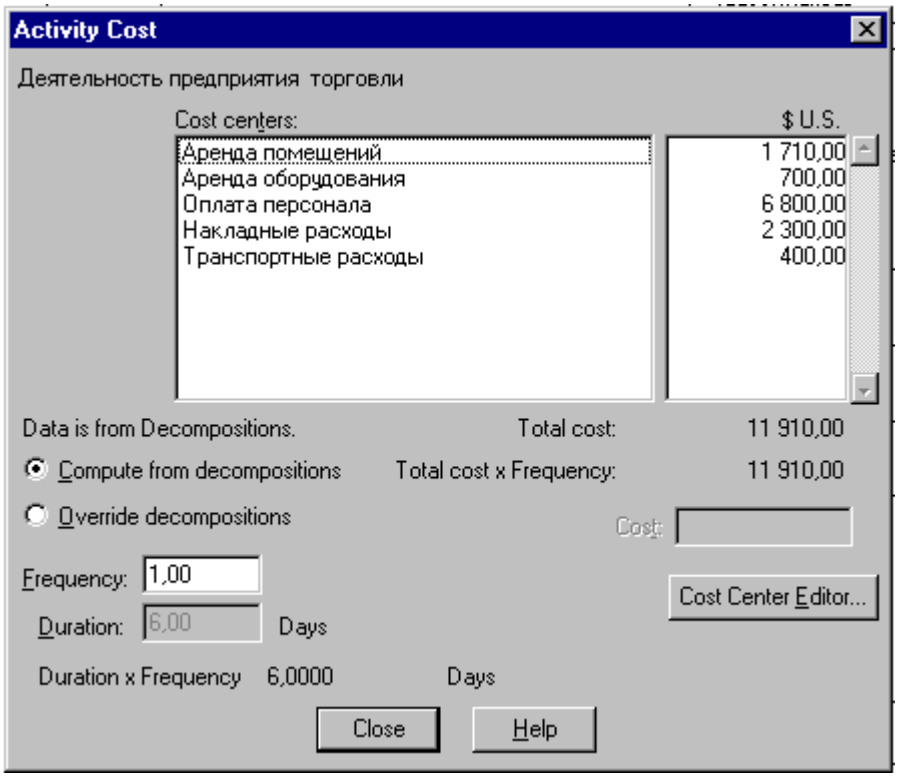


Рис. 5. Задание стоимости работ в диалоге Activity Cost.

Затраты вышестоящей работы определяется как сумма затрат дочерних работ по каждому центру затрат (режим Compute from Decompositions). Это достаточно упрощенный принцип подсчета справедлив, если работы выполняются последовательно. Если схема выполнения более сложная, можно отказаться от подсчета и задать итоговые суммы вручную (Override Decompositions) или воспользоваться специализированным средством стои-

мостного анализа EasyABC. Результаты стоимостного анализа наглядно представляются на специальном отчете BPwin - Activity Cost Report.

ABC позволяет оценить стоимостные и временные характеристики системы. Если стоимостных показателей недостаточно, имеется возможность внесения собственных метрик - свойств, определенных пользователем (User Defined Properties, UDP). Имеется возможность задания 18 различных типов UDP, в том числе управляющих команд и массивов, объединенных по категориям. Например, категория «Загрязнение окружающей среды», может объединять свойство «загрязнение воды» типа Real Number и свойство «загрязнение воздуха» типа Integer List с предварительно определенной областью значений (1, 2, 3, 4, 5). Каждой работе можно поставить в соответствие набор UDP и проанализировать результат в специальном отчете Diagram Object Report.

Все вышеперечисленные возможности BPwin»а делают его лидером среди недорогих CASE-инструментов, предназначенных для анализа деятельности предприятия и проектирования информационных систем.

Структура данных и будущее приложение

Ни одну область деятельности человека, поддерживаемую информационными технологиями, невозможно представить себе без использования баз данных, помогающих получить быстрый доступ к информации, увеличивая тем самым продуктивность работы. Клиент-серверные приложения, получившие в последнее время широкое распространение, построены на основе баз данных; приложения Internet и intranet могут получать доступ к базам данных, открывая широкие возможности для публикации информации, необходимой широкому кругу пользователей. Большинство клиент-серверных систем в данный момент представляют собой приложения по оперативной обработке транзакций (On-Line Transaction Processing, OLTP), которые служат для быстрой обработки и сохранения данных. Примерами таких приложений могут служить системы выписки счетов, регистрации и учета продукции и т.п. В то же время в последние годы значительное внимание уделяется построению хранилищ данных (data warehousing) - это базы данных специального назначения, складирующие всю информацию предприятия. Хранилища данных лежат в основе так называемых систем оперативного анализа данных (On-Line Analisys Processing, OLAP), которые позволяют принимать решения и помогают планировать стратегию развития предприятия. Успех любого приложения зависит от того, насколько хорошо смоделирована и разработана база данных приложения, поэтому разработке базы данных необходимо уделить много внимания.

База данных создается в несколько этапов, на каждом из которых необходимо согласовывать структуру данных с заказчиком и, что самое важное, подвергать созданную структуру данных экспертизе внутри команды, которая создает систему. Поэтому представление данных должно быть простым и понятным всем заинтересованным лицам. Именно по этой причине, наибольшее распространение получило представление базы данных под названием "сущность-отношение" (entity-relationship), которое также известно как

ER-диаграмма. Модели, представленные в виде ER-диаграмм, крайне просты и удобны для понимания. Фрагмент такой модели изображен на рис. 6.

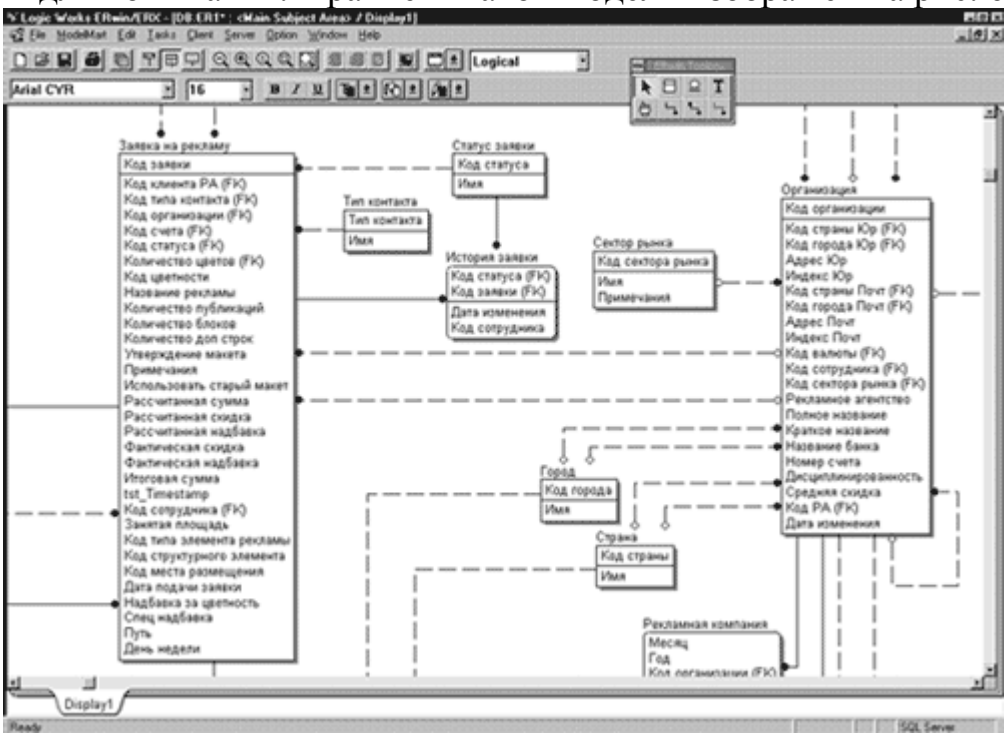


Рис. 6. Модель базы данных разрабатываемого приложения, представленная в виде ER-диаграммы, построенной в среде ERwin.

ER-диаграммы были приняты в качестве основы для создания стандарта IDEF1X. Предварительный вариант этого стандарта был разработан в военно-воздушных силах США и предназначался для увеличения производительности при разработке компьютерных систем. В 1981 году этот стандарт был формализован и опубликован организацией ICAM (Integrated Computed Aided Manufacturing), и с тех пор является наиболее распространенным стандартом для создания моделей баз данных по всему миру.

С развитием компьютерных технологий и появлением CASE-моделирования (Computer Aided Software Engineering) возникла потребность в инструментах, которые бы поддерживали стандарты моделирования. Современный инструмент моделирования баз данных должен удовлетворять ряду требований. Позволять разработчику сконцентрироваться на самом моделировании, а не на проблемах с графическим отображением диаграммы. Инструмент должен автоматически размещать сущности на диаграмме, иметь развитые и простые в управлении средства визуализации и создания представлений модели.

- Инструмент должен проверять диаграмму на согласованность, автоматически определяя и разрешая несоответствия. Однако инструмент должен быть настраиваемым и при желании предоставлять разработчику некоторую свободу в действиях и право самому разрешать несоответствия или отступления от методологии.

- Инструмент моделирования должен поддерживать как логическое, так и физическое моделирование.
- Современный инструмент должен автоматически генерировать базу данных на СУБД назначения.

Все современные инструменты моделирования в той или иной степени удовлетворяют перечисленным выше общим требованиям, однако в этой статье речь пойдет об инструменте моделирования баз данных ERwin версии 3.5, продукте компании CA/Logic Works. Выбор инструмента не случаен, т.к. на нынешний момент ERwin является наиболее мощным средством для разработки структуры данных как на логическом, так и на физическом уровне. Следует отметить, что существует несколько модификаций ERwin, каждая из которых, помимо моделирования, предназначена для выполнения специфических целей. Здесь мы рассмотрим ERwin3.5/ERX, который предназначен для работы именно с системами управления базами данных. Остальные члены семейства ERwin предназначены для использования с инструментами разработки клиентской части приложения, такими, как Power Builder, Visual Basic и прочими. Продукт CA/Logic Works ERwin 3.5 был выпущен в феврале этого года и сразу же получил признание широкого круга пользователей за многие усовершенствования по сравнению с предыдущими версиями, которые в него были внесены. Этот инструмент моделирования полностью поддерживает стандарт IDEF1X и является лидером на рынке инструментов разработки баз данных.

Разработка в среде ERwin

Обычно разработка модели базы данных состоит из двух этапов: составление логической модели и создание на ее основе физической модели. ERwin полностью поддерживает такой процесс, он имеет два представления модели: логическое (logical) и физическое (physical). Таким образом, разработчик может строить логическую модель базы данных, не задумываясь над деталями физической реализации, т.е. уделяя основное внимание требованиям к информации и бизнес-процессам, которые будет поддерживать будущая база данных. ERwin имеет очень удобный пользовательский интерфейс, позволяющий представить базу данных в самых различных аспектах. Например, ERwin имеет такие средства визуализации как "хранимое представление" (stored display) и "предметная область" (subject area). Хранимые представления позволяют иметь несколько вариантов представления модели, в каждом из которых могут быть подчеркнуты определенные детали, которые вызвали бы перенасыщение модели, если бы они были помещены на одном представлении. Предметные области помогают вычленить из сложной и трудной для восприятия модели отдельные фрагменты, которые относятся лишь к определенной области, из числа тех, что охватывает информационная модель. Интерфейс среды разработки ERwin представлен на рисунке.

Возможности редактирования и визуализации в среде ERwin весьма широки, так, например, создание отношений возможно при помощи перетаскивания атрибута из одной сущности в другую. Такое редактирование моде-

ли позволяет вносить изменения и проводить нормализацию быстрее и эффективнее, чем с использованием других инструментов. Для того, чтобы добавить новый элемент на диаграмму, его просто нужно выбрать на панели инструментов (Toolbox) и перенести в нужное место диаграммы. Добавив новую сущность на диаграмму, в нее можно добавить атрибуты, не открывая никаких редакторов, а просто ввести их названия прямо на диаграмме. Таким образом, ERwin позволяет значительно снизить время на создание самой диаграммы и сконцентрироваться на самих задачах, стоящих перед разработчиком.

ERwin имеет мощные средства визуализации модели, такие, как использование различных шрифтов, цветов и отображение модели на различных уровнях, например, на уровне описания сущности, на уровне первичных ключей сущности и т.д. Эти средства ERwin значительно помогают при презентации модели в кругу разработчиков системы или сторонним лицам.

Возможность использования модели ERwin одновременно для логического и физического представления данных позволяет по окончании работы получить полностью документированную модель.ERwin, как и инструмент моделирования бизнес-процессов BPwin, интегрирован с генератором отчетов фирмы CA/Logic Works - RPTwin. Это средство позволяет получать подробные отчеты по модели, освещая самые различные ракурсы и аспекты. Инструмент RPTwin поставляется вместе с ERwin и имеет богатый набор встроенных отчетов, позволяющих получать многогранную информацию по модели. Документирование структуры данных является очень важной частью моделирования, т.к. это позволяет другим разработчикам или лицам, которые будут сопровождать систему, быстрее начать ориентироваться во внутренней структуре и понимать назначение компонентов.

Как уже говорилось, ERwin является не только инструментом для дизайна баз данных, он также поддерживает автоматическую генерацию спроектированной и определенной на физическом уровне структуры данных. ERwin 3.5 поддерживает широчайший спектр серверных и настольных СУБД. В этот список входят такие продукты, как Microsoft SQL Server, Oracle, Sybase, DB2, INFORMIX, Red Brick, Teradata, PROGRESS, Microsoft Access, FoxPro, Clipper и многие другие. Для каждой из перечисленных СУБД в ERwin предусмотрено присоединение по "родному" для этой СУБД протоколу и поддержка всех средств управления данными, присущих этой СУБД. Инструмент имеет богатый и гибкий макроязык, позволяющий создавать сценарии (pre- и postscripts), которые будут выполняться до и после генерации определенного объекта на СУБД назначения. С помощью этого макроязыка можно также сгенерировать на СУБД назначения тысячи строк шаблонов, хранимых процедур и триггеров. ERwin не поддерживает моделирования механизмов защиты базы данных, однако при помощи макроязыка можно автоматически выдать права на объект, пользуясь языком определения прав, который используется в конкретной СУБД.

ERwin имеет средство, выполняющее задачу, обратную генерации, что называется "обратная разработка" (reverse engineering). Т.е. ERwin может

присоединиться к СУБД, получить всю информацию о структуре базы данных и отобразить ее в графическом интерфейсе, сохранив все сущности, связи, атрибуты и прочие свойства. Таким образом, можно переносить существующую структуру данных с одной платформы на другую, а также исследовать структуру существующих баз данных.

Но и это еще не все. ERwin имеет средство Complete-Compare, которое является единственным на данный момент средством интерактивной разработки. ERwin демонстрирует разногласия между моделью и базой данных, эти несоответствия можно переносить или оставлять без изменений. При помощи этого средства можно все изменения модели вносить в базу данных автоматически без необходимости контроля за соответствием модели и базы данных "вручную", при этом существующие данные не будут затронуты.

Начиная с версии 3.5 ERwin, поддерживает многомерное моделирование, которое используется при построении хранилищ данных. Производительность OLAP-приложений определяется, в основном, качеством дизайна хранилища данных, поэтому критически важно при разработке хранилища иметь инструмент, который бы поддерживал распространенные технологии. ERwin поддерживает две технологии моделирования хранилищ данных: звезда (star) и снежинка (snowflake).

ERwin тесно интегрирован с другими продуктами CA/Logic Works. Словарь данных, созданный при анализе бизнес-процессов при помощи инструмента BPwin, может быть использован как основа для построения модели базы данных. Однако взаимосвязь между этими двумя инструментами двусторонняя, модели BPwin и ERwin можно постоянно поддерживать в согласованном состоянии. Интеграция этих двух продуктов очень важна с точки зрения их совместного использования при разработке программного обеспечения, т.к. отпадает необходимость в повторном выполнении действий и процесс создания словаря данных становится практически автоматическим.

Мы рассмотрели лишь незначительную часть средств, которые облегчают и помогают разработку в среде ERwin; остальные средства выходят за рамки данной статьи и являются предметом более специального и детального обсуждения.

Коллективная разработка

При коллективной разработке базы данных возникает проблема синхронизации моделей, созданных различными разработчиками. Такая проблема особенно актуальна, когда над одной моделью работает большое количество разработчиков. Для эффективного решения этой проблемы CA/Logic Works предлагает продукт ModelMart, который представляет собой репозитарий для хранения моделей. Этот репозитарий имеет средства защиты модели, позволяет гибко управлять различными моделями или версиями модели, предоставляет единый активный словарь данных, которым можно централизованно управлять и использовать его в нескольких моделях сразу.

ModelMart не зависит от используемой платформы и прост в установке и администрировании. Использование такого коллективного репозитария позволяет масштабировать разработку моделей с помощью ERwin до масштабов предприятия. ModelMart также может предоставлять репозитарий для хранения и управления моделями BPwin, начиная с версии 2.02. Этот факт также говорит в пользу разработки информационных систем с использованием BPwin и ERwin.

5. ПЕРЕЧЕНЬ УЧЕБНИКОВ И УЧЕБНЫХ ПОСОБИЙ

1. Клейменов М.С. Системный подход к проектированию сложных систем //Журнал д-ра Добба. 1999. – №1. – С.9-14.
2. Норенков И.П. Основы автоматизированного проектирования, М. Изд. МГТУ им. Баумана, 2000г.
3. Серия ГОСТ 34.ХХХ "Информационная технология. Комплекс стандартов на автоматизированные системы".
4. ГОСТ Р ИСО/МЭК 12207-99 "Информационная технология. Процессы жизненного цикла программных средств".
5. РД 50–680–88 Методические указания. Автоматизированные системы. Основные положения.
6. Э.Е. Кудряшова. САПР ТП как система интегрированной среды. Интегриро-ванные автоматизированные системы CAD/CAM/CAE: Учеб. пособие / Волгоград: ВолгГТУ, 1998. - 148с.
7. Э.Е. Кудряшова. Новые информационные технологии в автоматизированных системах обработки ннформации и управления: Учеб. пособие/ Волгоград: ВолгГТУ, 2000г. -88с.
8. Маклаков С.В. BPWin, ERWin. CASE – средства разработки информационных систем. –М.: ДИАЛОГ-МИФИ, 1999.
9. Грекул В.И., Денищенко Г.Н., Коровкина Н.Л. Проектирование информационных систем. /Интернет-университет информационных технологий - ИНТУИТ.ру, 2005.
10. Амриш К.И., Ахмед Х.З. Разработка корпоративных Java-приложений с использованием J2EE и UML. /Пер. с англ. -М.: "Вильямс", 2002. - 272 с.
11. Боггс У., Боггс М. UML и Rational Rose./ - М.: "ЛОРИ", 2000. - 582 с.
12. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений на С++. - М.: "Бином", СПб: "Невский диалект", 1999. - 560 с.
13. Буч Г., Рамбо Дж., Джекобсон А. Язык UML. Руководство пользователя. /- М.: ДМК, 2000. - 432 с.
14. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж. Приемы объектно-ориентированного проектирования. /Паттерны проектирования. - СПб: "Питер", 2001.- 368 с.
9. Гома Х. UML. Проектирование систем реального времени, параллельных и распределенных приложений. - М.: "ДМК Пресс", 2002. - 704 с.

10.Грехем И. Объектно-ориентированные методы. Принципы и практика.- М.: "Вильямс", 2004. - 880 с.

11.Коналлен Дж. Разработка Web-приложений с использованием UML. /Пер. с англ. - М.: "Вильямс", 2001. - 288 с.

12.Кьоу Дж., Джеанини М. Объектно-ориентированное программирование. Учебный курс. - СПб: "Питер", 2005.- 238 с.

13.Ларман К. Применение UML и шаблонов проектирования - М.: "Вильямс", 2001. - 496 с.

14.Ларман К. Применение UML и шаблонов проектирования. 2-е издание. - М.: "Вильямс", 2002. - 624 с.

15.Леоненков А.В. Самоучитель UML. - СПб.: "БХВ - Петербург", 2001. - 304 с.

16.Леоненков А.В.Самоучитель UML. 2-е издание. - СПб.: "БХВ-Петербург", 2004. - 432 с.

17.Леффингуэлл Д., Уидриг Д. Принципы работы с требованиями к программному обеспечению. Унифицированный подход. - М.: "Вильямс", 2002. - 448 с.

18.Нейбург Э.Дж., Максимчук Р.А. Проектирование баз данных с помощью UML. - М.: "Вильямс", 2002. - 288 с.

19.Рамбо Дж., Якобсон А., Буч Г. UML: специальный справочник - СПб: "Питер", 2001. - 656 с.

20.Розенберг Д., Скотт К. Применение объектного моделирования с использованием UML и анализ прецедентов. - М.: "ДМК Пресс", 2002. - 160 с.

21.Санблэд С., Санблэд С. Разработка масштабируемых приложений для Microsoft Windows. Мастер-класс. - М.: ИТД "Русская редакция", 2002. - 416 с.

22.Фаулер М., Скотт К. UML. Основы. - СПб: "Символ-Плюс", 2002. - 192 с.

23.Шаллоуей А., Тротт Дж.Р. Шаблоны проектирования. Новый подход к объектно-ориентированному анализу и проектированию. - М.: "Вильямс", 2002. - 288 с.

24.Шмуллер Д. Освой самостоятельно UML за 24 часа. - М.: "Вильямс", 2002. - 352 с.

25.Якобсон А., Буч Г., Рамбо Дж. Унифицированный процесс разработки программного обеспечения. - СПб: "Питер", 2002. - 496 с.

26.Grand M. Patterns in Java. A Catalog of Reusable Design Patterns Illustrated with UML. - Wiley & Sons, 1998. Vol. 1, - 468 p. Vol. 2, - 356 p.

27.Starr L. Executable UML. How to build class models. - Prentice Hall PTR, 2002. - 418 p.

28.Wampler B.E. The Essence Object-Oriented Programming with Java and UML. - Addison-Wesley, 2002, - 290 p.

- 6. КРАТКИЙ КОНСПЕКТ ЛЕКЦИЙ

Лекция 1. Общая характеристика процесса проектирования АСОИУ.

Современные информационные технологии предоставляют широкий набор способов реализации АСОИУ, выбор которых осуществляется на основе требований со стороны предполагаемых пользователей, которые, как правило, изменяются в процессе разработки. Для теории принятия решений процесс проектирования системы – это процесс принятия проектно-конструкторских решений, направленных на получение версии системы, удовлетворяющей требования заказчика.

Под *проектом* будем понимать проектно-конструкторскую и технологическую документацию, в которой представлено описание проектных решений по созданию и эксплуатации системы в конкретной программно-технической среде.

Под *проектированием системы* понимается процесс преобразования входной информации об объекте проектирования, о методах проектирования и об опыте проектирования объектов аналогичного назначения в соответствии с ГОСТом в проект АСОИУ. С этой точки зрения проектирование АСОИУ сводится к последовательной формализации проектных решений на различных стадиях жизненного цикла системы: предпроектного анализа требований, технического и рабочего проектирования, внедрения и эксплуатации АСОИУ.

Осуществление проектирования системы предполагает использование проектировщиками определенной технологии проектирования, соответствующей масштабу и особенностям разрабатываемого проекта.

Технология проектирования системы – это совокупность методологии и средств проектирования системы, а также методов и средств организации проектирования (управление процессом создания и модернизации проекта системы).

В основе технологии проектирования лежит технологический процесс, который определяет действия, их последовательность, состав исполнителей, средства и ресурсы, требуемые для выполнения этих действий. *Технологический процесс* проектирования системы в целом делится на совокупность последовательно-параллельных, связанных и соподчиненных цепочек действий, каждое из которых может иметь свой предмет.

Проектирование системы – трудоемкий, длительный и динамический процесс. Технологии проектирования, применяемые в настоящее время, предполагают поэтапную разработку системы. Этапы по общности могут разделятся в стадии. Совокупность стадий и этапов, которые проходит система в своем развитии о момента принятия решения о создании системы до момента прекращения функционирования системы, называется *жизненным циклом системы.*

Суть содержания жизненного цикла разработки системы в различных подходах одинакова и сводится к выполнению следующих стадий:

Планирование и анализ требований (предпроектная стадия) – системный анализ. Исследование и анализ существующей системы, определение требова-

ний к создаваемой системе, оформление технико-экономического обоснования и технического задания на разработку системы.

Проектирование (техническое проектирование, логическое проектирование). Разработка в соответствии со сформулированными требованиями состава автоматизируемых функций и состава обеспечивающих подсистем, оформление технического проекта системы.

Реализация проекта (рабочее проектирование, физическое проектирование, программирование). Разработка и настройка программ, наполнение баз данных, создание рабочих инструкций для персонала, оформление рабочего проекта.

Внедрение (тестирование, опытная эксплуатация). Комплексная отладка подсистем, обучение персонала, поэтапное внедрение системы в эксплуатацию, оформление акта о приемо-сдаточных испытаниях системы.

Эксплуатация системы (сопровождение, модернизация). Сбор рекламаций и статистики о функционировании системы, исправление ошибок и недоработок, оформление требований к модернизации системы и ее выполнение.

Лекция 2. Разработка функциональной модели АСОИУ.

Предпроектная стадия, включающая проведение необходимых исследований, работ по формированию структуры АСОИУ и получении в конечном счете технического задания на проектирование может проходить по двум принципиально различным вариантам.

Первый – классический фундаментальный подход. Связан с проведением исследований по вскрытию законов, на основе которых функционирует исследуемая система обработки информации и управления. Далее, вследствие выявленных фундаментальных законов, строится формальная модель, связывающая входы, выходы системы, влияние внешних воздействий на нее. Таким образом получаем формально-математическое представление системы, которое и может в дальнейшем служить основой для функционального, инфологического и техно-рабочего проектирования АСОИУ.

Второй подход, достаточно часто применяемый при построении АСОИУ, включает в себя проведение информационного обследования объекта (предметной области), выявление основных информационных потоков, построения, как правило, имитационной модели функционирования объекта и далее выход также на инфологическое и техно-рабочее проектирование.

Первый подход в большей степени гарантирует получение высококачественной разработки, но требует больших интеллектуальных и финансовых затрат. Второй подход обеспечивает, на первый взгляд, меньшие затраты, но вероятней всего за счет неудачных и малоэффективных решений, общие затраты при реализации работ по второму варианту могут быть и значительно больше, чем в первом случае.

Однако, выбор схемы решения всегда остается за разработчиком.

Далее приводятся некоторые моменты, связанные с использованием структурного анализа.

Структурным анализом SADT (Structured Analysis and Design Technique), принято называть метод исследования системы с помощью ее графического модельного представления, которое начинается с общего обзора и последующей детализации, в иерархическую структуру со все большим числом уровней. Для таких методов характерно: разбиение на уровни абстракции с ограничением числа элементов на каждом из уровней (обычно от 3 до 9); ограниченный контекст, включающий лишь существенные на каждом уровне детали; дуальность данных и операций над ними; использование строгих формальных правил записи; последовательное приближение к конечному результату.

Анализ является первым этапом создания АСОИУ, на котором требования заказчика уточняются, формализуются и документируются. Фактически на этом этапе дается ответ на вопрос: «Что должна делать будущая система?». Именно здесь лежит ключ к успеху всего проекта.

Структурный анализ начинается с исследования того, как организована система управления предприятием, с обследования функциональной и информационной структуры системы управления. По результатам обследования аналитик на первой стадии анализа строит обобщенную логическую модель исходной предметной области, отображающую ее функциональную структуру, особенности основной деятельности и информационное пространство, в котором эта деятельность осуществляется. Используя специальную терминологию, можно сказать, что аналитик строит модель «как есть».

Вторая стадия работы, к которой привлекаются заинтересованные представители заказчика, а при необходимости и независимые эксперты, состоит в анализе модели «как есть», выявлении ее недостатков и узких мест, определение путей совершенствования системы управления на основе выделенных критериев качества.

Третья стадия анализа, содержащая элементы проектирования, – создание усовершенствованной обобщенной логической модели, отображающей реорганизованную предметную область или ее часть, которая подлежит автоматизации. Эту модель можно назвать моделью «как надо», т.е. здесь происходит формализация системы.

На ряду со структурным подходом существует и более мощный подход называемый объектно-ориентированным ООП. Эта методология создана для проектирования больших и сложных систем и имеет ряд преимуществ перед структурным подходом.

ООП базируется на создании объектной модели, которая описывает структуру объектов, составляющих систему, их атрибуты, операции, взаимосвязи с другими объектами. Цель разработки объектной модели – описать объекты, составляющие в совокупности проектируемую систему, а также выявить и указать различные зависимости между объектами. *Декомпозиция проблемы на объекты* – творческий, плохо формализуемый процесс.

В объектной модели должны быть отражены те понятия и объекты

реального мира, которые важны для разрабатываемой системы. В объектной модели отражается прежде всего прагматика разрабатываемой системы, что выражается в использовании терминологии прикладной области, связанной с использованием разрабатываемой системы.

Объектная модель имеет четыре главных элемента: абстрагирование, инкапсуляция, модульность, иерархия.

Эти элементы являются *главными* в том смысле, что без любого из них модель не будет объектно-ориентированной. Кроме главных, имеются еще три дополнительных элемента: типизация, параллелизм, сохраняемость.

Абстракция выделяет существенные характеристики некоторого объекта, отличающие его от всех других видов объектов и, таким образом, четко определяет его концептуальные границы с точки зрения наблюдателя. Инкапсуляция – это процесс отделения друг от друга элементов объекта, определяющих его устройство и поведение; инкапсуляция служит для того, чтобы изолировать контрактные обязательства абстракции от их реализации. Модульность – это свойство системы, которая была разложена на внутренне связные, но слабо связанные между собой модули. Иерархия – это упорядочение абстракций, расположение их по уровням. Типизация – это способ защититься от использования объектов одного класса вместо другого, или по крайней мере управлять таким использованием. Параллелизм – это свойство, отличающее активные объекты от пассивных.

Сохраняемость – способность объекта существовать во времени, переживая породивший его процесс, и (или) в пространстве, перемещаясь из своего первоначального адресного пространства.

Лекция 3. Исходные данные для проектирования, стандарты.

Методы, которые используются для изучения предметной области и технологии получения от экспертов сведений о системе, подлежащей описанию, называются сбором данных, а в информатике она больше известна как опрос (интервьюирование) или извлечение знаний. Но как бы она не называлась, способность собрать необходимую информацию, основанную на знаниях экспертов, весьма существенна для построения точной и полезной модели. Поэтому технология сбора информации составляет важную часть структурной методологии SADT.

Опрос – это сбор сведений. Первый опрос служит точкой отсчета в процессе моделирования. Чтобы провести опрос, аналитик вначале выбирает наилучший источник информации (документ или конкретного человека), а затем организует его "опрос". Цель опроса – получение порции информации, необходимой для начала либо для продолжения построения определенной части модели. После первого опроса SADT – модель используется для определения той информации, которую необходимо получить в ходе следующего опроса. В соответствии с иерархией модели может быть проведена последовательность опросов для выяснения все более конкретных деталей рассматриваемой области.

Обычно источниками информации служат эксперты. Часто именно они являются наилучшими источниками, потому что им знакомы текущие нюансы и недокументированные аспекты системы. Самое важное – это то, что экспертам известны факты, которые не отражены в документах или которые трудно объяснить. Их можно получить только путем опроса экспертов. Чтобы подготовиться к такому опросу, нужно исследовать другие источники информации, например документы. Существует множество различных стратегий для извлечения информации из этих источников: чтение документов, наблюдение за выполняемыми операциями, анкетирование, использование собственных знаний, составление описания.

Документы – хороший источник информации, потому что они чаще всего доступны и их можно "опрашивать" в удобном для себя темпе. Чтение документов – прекрасный способ получить первоначальное представление о системе и сформулировать вопросы к экспертам.

Наблюдение за работой моделируемой системы – хорошая стратегия получения информации. Оно должно проводиться всегда, когда есть такая возможность. Через наблюдение, а возможно, и участие аналитики получают информацию о происходящих день за днем операциях из первых рук. Во время наблюдения за работой системы часто возникают вопросы, которые никогда бы не появились, если бы аналитик только читал документы или разговаривал с экспертами. Однако следует соблюдать осторожность. Слишком долгие наблюдения могут привести к избыточному привыканию к текущему состоянию дел. Из-за потери объективности можно не увидеть альтернативные пути описания функций системы.

Анкетирование проводится для того, чтобы опросить большие группы экспертов в сжатые сроки. Его можно использовать, например, когда необходимо быстро получить сведения о работе какой-либо определенной части системы с разных позиций. Анкетирование при опросе экспертов позволяет выявить, какие части системы более всего нуждаются в улучшении. На практике, однако, информация, полученная от экспертов с помощью анкет, оказывается недостаточно достоверной.

Еще одна полезная стратегия – придумать описание и дать его экспертам для корректировки. Придуманные описания могут дать альтернативные схемы функционирования системы – схемы, о которых эксперты никогда не думали. Однако для реализации таких схем нужна поддержка. Предварительно нужно изучить предметную область и найти хотя бы одну доброжелательно настроенную группу экспертов, прежде чем пытаться придумать описание. В этом. случае описание будет иметь шансы отразить реальность. Придуманные описания могут, однако, потерпеть неудачу, если эксперты не готовы воспринимать новые возможности. В идеале, прежде чем прибегать к этой стратегии, автор должен установить надежные контакты с несколькими экспертами.

Создание и внедрение автоматизированных систем различных классов и назначений ведется во многих отраслях промышленности по нормативно-технической документации, устанавливающей разнообразные организацион-

но-методические и технические нормы, правила и положения, затрудняющие интеграцию систем и эффективное их совместное функционирование.

В период принятия Госстандартом РФ решения о совершенствовании межотраслевых комплексов стандартов действовали следующие комплексы и системы стандартов, устанавливающие требования к различным видам АСОИУ:

1) единая система стандартов автоматизированных систем управления, распространяющаяся на АСУ, АСУП, АСУ ТП и другие организационно-экономические системы;

2) комплекс стандартов, распространяющихся на системы автоматизированного проектирования;

3) четвертая группа стандартов, распространяющиеся на автоматизированные системы технологической подготовки производства.

Практика применения стандартов на АСУ, САПР. АСУ ТП, АСТПП показала, что в них применяется одинаковый понятийный аппарат, имеется много общих объектов стандартизации, однако требования стандартов не согласованы между собой, имеются различия по составу и содержанию работ, различия по обозначению, составу, содержанию и оформлению документов и пр.

На фоне отсутствия единой технической политики в области создания АСОИУ многообразие стандартов не обеспечивало широкой совместимости АСОИУ при их взаимодействии, не позволяло тиражировать системы, тормозило развитие перспективных направлений использования средств вычислительной техники.

В настоящее время осуществляется переход к созданию сложных автоматизированные системы (за рубежом системы CAD – CAM), включающих в свой состав АСУ технологическими процессами и производствами. САПР – конструктора, САПР – технолога, АСНИ и др. системы. Использование противоречивых правил при создании таких систем приводит к снижению качества, увеличению стоимости работ, затягиванию сроков ввода в действие.

Единый комплекс стандартов и руководящих документов должен распространяться на автоматизированные системы различного назначения: АСНИ, САПР, ОАСУ, АСУП, АСУТП, АСУГПС, АСК, АСТПП, включая их интеграцию.

При разработке межотраслевых документов следует учитывать следующие особенности автоматизированных систем как объектов стандартизации:

1) техническое задание является основным документом в соответствии с которым проводят создание АСОИУ и приемку его заказчиком,

2) АСОИУ, как правило, создают проектным путем с комплектацией изделиями серийного и единичного производства и проведением строительных, монтажных, наладочных и пусковых работ, необходимых для ввода в действие системы;

3) в общем случае АСОИУ состоит из программно-технических (ПТК), программно-методических комплексов (ПМК) и компонентов технического, программного и информационного обеспечения. Компоненты этих видов

обеспечения, а также ПМК и ПТК должны изготовляться, и поставляется как продукция производственно-технического назначения. Компоненты могут входить в систему в качестве самостоятельных частей или могут быть объединены в комплексы;

4) создание АСОИУ в организациях (предприятиях) требует специальной подготовки пользователей и обслуживающего персонала системы;

5) функционирование АСОИУ и комплексов обеспечивается совокупностью организационно-методических документов, рассматриваемых в процессе создания как компоненты правового, методического, лингвистического, математического, организационного и др. видов обеспечения. Отдельные решения, получаемые в процессе разработки этого обеспечения, могут реализовываться в виде компонентов технического, программного или информационного обеспечения;

6) совместное функционирование и взаимодействие различных систем и комплексов осуществляется на базе локальных сетей ЭВМ.

Спецификации и соглашения, принятые для локальных сетей ЭВМ, обязательны для обеспечения совместимости систем, комплексов и компонентов.

Стандартизация в области АСОИУ является составной частью работ по обобщенной проблеме «Информационная технология».

Единый комплекс стандартов руководящих документов на автоматизированные системы совместно с другими системами и комплексами стандартов должен образовывать полное нормативно-техническое обеспечение процессов создания и функционирования систем.

ЕКС АСОИУ должен охватывать специфические для автоматизированных систем направления стандартизации и распространять традиционные направления стандартизации на программно-технические, программно-методические комплексы и автоматизированные системы в целом.

Направления и задачи стандартизации при нормативно-техническом обеспечении процессов создания и функционирования АСОИУ группируют следующим образом:

1) установление технических требований к продукции;

2) регламентация методов испытаний и правил аттестации и сертификации продукции;

3) регламентация правил и порядка разработки;

4) установление правил документирования;

5) обеспечение совместимости,

6) регламентация организационно-методических вопросов функционирования систем.

Направления 1-4 являются традиционными при разработке, изготовлении и поставке продукции. Направления 5, 6 являются специфичными и вытекают из особенностей, присущих АСОИУ.

Обеспеченность АСОИУ в целом и их составных частей нормативно-технической документацией в рамках принятых направлений и задач стандартизации различна.

Компоненты технического, программного и информационного обеспечения, как продукцию производственно-технического назначения, рассматривают, соответственно, как конструкторские, программные и информационные изделия. На эти изделия распространяются действующие комплексы стандартов ЕСКД, СРПП, ЕСПД, СГИП, УСД, классификаторы и кодификаторы технико-экономической информации, комплексы стандартов вида «ОТТ», «Методы испытаний», «ТУ», а также ОТТ заказчика.

Весь жизненный цикл конструкторских изделий полностью обеспечен нормативно-технической документацией, действующей в машиностроении и приборостроении.

Программные изделия обеспечены НТД, входящей в ЕСПД и ОТТ заказчика. Однако область распространения этих НТД должна быть расширена с целью отражения вопросов, связанных с разработкой, созданием, распространением и эксплуатацией программных изделий.

Информационные изделия в настоящее время не обеспечены НТД, хотя отдельные вопросы проработаны в рамках УСД, классификаторах и кодификаторах технико-экономической информации.

Программно-технические и программно-методические комплексы рассматриваются как сложные изделия, не имеющие аналогов в машиностроении. Учитывая статус ПТК и ПМК как продукции производственно-технического назначения, правила и порядок их разработки должен быть аналогичен требованиям, установленным стандартами системы разработки и постановки продукции на производство (СРПП).

Лекция 4. Проектирование пользовательского интерфейса.

Проектирования интерфейса пользователя – сложная многофакторная и многовариантная задача, требующая системного подхода [5]. В настоящее время считается доказанным, что решение данной задачи заключается не в том, чтобы рационально «вписать» человека в контур управления, а в том, чтобы, исходя из задач управления объектом, разработать систему взаимодействия двух равноправных партнеров (человек-оператор и аппаратно-программный комплекс АСОИУ).

Цель создания эффективного эргономичного пользовательского интерфейса состоит в том, чтобы отобразить информацию настолько эффективно насколько это возможно для человеческого восприятия и структурировать отображение на дисплее таким образом, чтобы привлечь внимание к наиболее важным единицам информации.

Пользователь должен иметь возможность манипулировать объектами в среде приложения. Неплохо, если они (графические элементы) будут ему понятны и станут нести в себе информацию о том, что это такое и что произойдет, если выбрать или произвести действие над каким-то объектом. Иллюстрация этой идеи – панель быстрого доступа Word'a. Что еще, кроме как сохранение файла, можно ожидать от кнопки с изображением дискеты? Необходимо, чтобы пользователь имел наглядное средство отображения ин-

формации на различных этапах решения задач, он должен видеть, как его действия влияют на объекты, расположенные на экране.

Создание эффективного интерфейса заключается в быстром, насколько это возможно, развитии у пользователей простой концептуальной модели интерфейса. Концепция согласованности состоит в том, что при работе с компьютером у пользователя формируется система ожидания одинаковых реакций на одинаковые действия, что постоянно подкрепляет пользовательскую модель интерфейса.

Приложение должно проектироваться таким образом, чтобы пользователь в любой момент и на любом этапе работы мог получить помощь, контекстную справку или подсказку.

Безусловно, пользователю нужно дать возможность экспериментировать в приложении (нажатие любых кнопок, изменение настроек и т. д.). Но при этом необходимо избавить его от тупиковых ситуаций: все последствия экспериментов должны быть исправимы, а в лучшем случае еще и обратимы.

Интерфейс должен предоставлять информацию о том, что происходит в данный момент на компьютере. Нельзя допускать, чтобы пользователь долгое время ожидал реакции приложения на некоторое свое действие.

Цветовая гамма, компоновка элементов, пиктограммы, звуки, анимация – все должно помогать пользователю при выполнении задачи. Но здесь важно не переборщить, т.к. в этом случае внимание человека начнет рассеиваться, у него появится раздражение и, как следствие снизится эффективность работы.

Начальным этапом разработки пользовательского интерфейса являются создание его ассоциативной модели, после чего осуществляется проработка концептуального дизайна. Здесь необходимо разработать необходимый набор интерфейсных элементов, каждый из которых должен обладать определенным цветом, формой, надписью и т. п., и все вместе они должны составлять единую систему, вызывающую стойкую систему ассоциаций у пользователей.

Цвет – мощный визуальный инструмент, который необходимо использовать очень осторожно, чтобы не вызвать неадекватной реакции пользователя.

Целесообразно ограничить число цветов до 4 на экране и до 7 для последовательности экранов. Для неактивных элементов рекомендуется использовать бледные цвета. Необходимо использовать цвета согласно представлениям пользователя. Например, для картографа зеленый цвет – лес, желтый – пустыня, синий – вода. Если цвет используется для кодировки информации, необходимо удостовериться, что код адекватно воспринимается пользователем: красный – опасность/стоп, зеленый – нормально/продолжение работы, желтый – предостережение. Для привлечения внимания наиболее эффективны белый, желтый и красный цвета.

Меню необходимый элемент любой автоматизированной системы, позволяющий пользователю выполнять задачи внутри приложения и управлять процессом решения. Достоинство меню в том, что пользователи не должны

помнить название элемента или действия, которое они хотят выполнить – они должны только распознать его среди пунктов меню.

Сообщения необходимы для направления пользователя в нужную сторону, подсказок и предупреждений для выполнения необходимых действий на пути решения задачи. Они также включают подтверждения действий со стороны пользователя и подтверждения, что задачи были выполнены системой успешно либо по каким-то причинам не выполнены. Сообщения могут быть обеспечены в форме диалога, экранных заставок и т.п.

Лекция 5. Задачи информационного обеспечения, структура информационно-логической модели АСОИУ.

Логическая модель описывает понятия предметной области, их взаимосвязь, а также ограничения на данные, налагаемые предметной областью. Примеры понятий –"сотрудник", "отдел", "проект", "зарплата". Примеры взаимосвязей между понятиями –"сотрудник числится ровно в одном отделе", "сотрудник может выполнять несколько проектов", "над одним проектом может работать несколько сотрудников". Примеры ограничений – "возраст сотрудника не менее 16 и не более 60 лет".

Логическая модель данных является начальным прототипом будущей базы данных. Логическая модель строится в терминах информационных единиц, но *без привязки к конкретной СУБД.* Более того, логическая модель данных необязательно должна быть выражена средствами именно *реляционной* модели данных. Основным средством разработки логической модели данных в настоящий момент являются различные варианты *ER-диаграмм (Entity-Relationship, диаграммы сущность-связь)*. Одну и ту же ER-модель можно преобразовать как в реляционную модель данных, так и в модель данных для иерархических и сетевых СУБД, или в постреляционную модель данных.

Решения, принятые на предыдущем уровне, при разработке модели предметной области, определяют некоторые границы, в пределах которых можно развивать логическую модель данных, в пределах же этих границ можно принимать различные решения. Например, модель предметной области складского учета содержит понятия "склад", "накладная", "товар". При разработке соответствующей реляционной модели эти термины обязательно должны быть использованы, но различных способов реализации тут много – можно создать одно отношение, в котором будут присутствовать в качестве атрибутов "склад", "накладная", "товар", а можно создать три отдельных отношения, по одному на каждое понятие.

При разработке логической модели данных возникают вопросы: хорошо ли спроектированы отношения? Правильно ли они отражают модель предметной области, а следовательно и саму предметную область?

Для того чтобы оценить качество принимаемых решений на уровне логической модели данных, необходимо сформулировать некоторые критерии качества в терминах физической модели и конкретной реализации и посмотреть, как различные решения, принятые в процессе *логического* моделирова-

ния, влияют на качество *физической* модели и на скорость работы базы данных.

Таких критериев может быть очень много и выбор их произволен. Некоторые из таких критериев являются важными с точки зрения получения качественной базы данных: адекватность базы данных предметной области, легкость разработки и сопровождения базы данных, скорость выполнения операций обновления данных (вставка, обновление, удаление кортежей), скорость выполнения операций выборки данных.

База данных должна адекватно отражать предметную область. Это означает, что должны выполняться следующие условия.

1. Состояние базы данных в каждый момент времени должно соответствовать состоянию предметной области.

2. Изменение состояния предметной области должно приводить к соответствующему изменению состояния базы данных.

3. Ограничения предметной области, отраженные в модели предметной области, должны некоторым образом отражаться и учитываться базе данных.

Практически любая база данных, за исключением совершенно элементарных, содержит некоторое количество программного кода в виде триггеров и хранимых процедур.

Хранимые процедуры – это процедуры и функции, хранящиеся непосредственно в базе данных в откомпилированном виде и которые могут запускаться пользователями или приложениями, работающими с базой данных. Основное назначение хранимых процедур –реализация бизнес-процессов предметной области.

Триггеры – это хранимые процедуры, связанные с некоторыми событиями, происходящими во время работы базы данных. В качестве таких событий выступают операции вставки, обновления и удаления строк таблиц. Если в базе данных определен некоторый триггер, то он запускается *автоматически* всегда при возникновении события, с которым этот триггер связан. Триггер срабатывает независимо от того, кто из пользователей и каким способом инициировал событие, вызвавшее запуск триггера. Таким образом, основное назначение триггеров – автоматическая поддержка целостности базы данных.

Очевидно, что чем больше программного кода в виде триггеров и хранимых процедур содержит база данных, тем сложнее ее разработка и дальнейшее сопровождение.

На уровне логического моделирования определяются реляционные отношения и атрибуты этих отношений. На этом уровне можно определять какие-либо физические структуры хранения (индексы, хеширование и т.п.). Единственное, чем можно управлять – это распределение атрибутов по различным отношениям. Можно описать немного отношений с большим количеством атрибутов, или сформировать большое количество отношений, каждое из которых содержит мало атрибутов. Таким образом, необходимо попытаться ответить на вопрос – влияет ли количество отношений и количество атрибутов в отношениях на скорость выполнения операций обновления данных. Такая постановка не является достаточно корректной, т.к. скорость вы-

полнения операций с базой данных зависит от физической реализации базы данных. Тем не менее, целесообразно *качественно* оценить это влияние при *одинаковых подходах к физическому моделированию.*

Основными операциями, изменяющими состояние базы данных, являются операции вставки, обновления и удаления записей. В базах данных, требующих постоянных изменений (складской учет, системы продаж билетов и т.п.) производительность определяется скоростью выполнения большого количества небольших операций вставки, обновления и удаления.

Обычно, вставка записи производится в одну из свободных страниц памяти, выделенной для данной таблицы. СУБД постоянно хранит информацию о наличии и расположении свободных страниц. Если для таблицы не созданы индексы, то операция вставки выполняется фактически с одинаковой скоростью независимо от размера таблицы и от количества атрибутов в таблице. Если в таблице имеются индексы, то при выполнении операции вставки записи индексы должны быть перестроены. Таким образом, скорость выполнения операции вставки *уменьшается при увеличении количества индексов* у таблицы и *мало зависит от числа строк* в таблице.

Для операции обновления и удаления записей из таблицы, прежде, чем обновить или удалить запись, ее необходимо найти. Если таблица не индексирована, то единственным способом поиска является последовательное сканирование таблицы в поиске нужной записи. В этом случае, скорость операций обновления и удаления существенно увеличивается с увеличением количества записей в таблице и не зависит от количества атрибутов. Но на самом деле неиндексированные таблицы практически никогда не используются. Для каждой таблицы обычно объявляется один или несколько индексов, соответствующий потенциальным ключам. При помощи этих индексов поиск записи производится очень быстро и практически не зависит от количества строк и атрибутов в таблице (хотя, конечно, некоторая зависимость имеется). Если для таблицы объявлено несколько индексов, то при выполнении операций обновления и удаления эти индексы должны быть перестроены, на что тратится дополнительное время. Таким образом, скорость выполнения операций обновления и удаления также *уменьшается при увеличении количества индексов у* таблицы и *мало зависит от числа строк* в таблице.

Одно из назначений базы данных – предоставление информации пользователям. Информация извлекается из реляционной базы данных при помощи оператора SQL –SELECT. Одной из наиболее дорогостоящих операций при выполнении оператора SELECT является операция соединение таблиц. Таким образом, чем больше взаимосвязанных отношений было создано в ходе логического моделирования, тем больше вероятность того, что при выполнении запросов эти отношения будут соединяться, и, следовательно, тем медленнее будут выполняться запросы. Таким образом, увеличение количества отношений приводит к замедлению выполнения операций выборки данных, особенно, если запросы заранее неизвестны.

Лекция 6. Определения общих ограничений целостности, триггеров и хранимых процедур.

Следующая стадия проектирования состоит в дополнении реляционных схем разделов распределенной базы данных определениями общих ограничений целостности, триггеров и хранимых процедур. На каждом из этих компонентов схемы следует остановиться отдельно. Начнем с общих ограничений целостности.

Язык SQL-92 позволяет определить ограничения целостности, относящиеся к общему состоянию базы данных и включающие ссылки на произвольное число таблиц. Семантика таких ограничений целостности может быть существенно шире, чем ограничения, задаваемые связями *1* к *n* и даже *n* к *m*. Поэтому, часто ограничения общего вида не выводятся автоматически из концептуальной схемы базы данных, и их приходится добавлять к реляционной схеме вручную. Для того чтобы понять, какие ограничения общего вида должны быть включены в реляционные схемы разделов, приходится возвращаться к документу, содержащему анализ требований корпорации. Задача проектировщика состоит в том, чтобы, с одной стороны, выявить все необходимые ограничения целостности и, с другой стороны, не перегрузить базу данных необязательными ограничениями, любое дополнительное ограничение целостности вызывает дополнительные проверки на стороне сервера при выполнении операций изменения базы данных; проверки для ограничений общего вида могут быть весьма громоздкими. Если предполагается использование распределенной базы данных, то придется учитывать возможности сервера по части ссылок на объекты "чужих" разделов. Это может повлиять на выбор ограничений целостности и/или повлечь создание новой декомпозиции общей базы данных на разделы.

Триггер - это, фактически, хранимая процедура без параметров, содержащая оператор или операторы изменения базы данных и вызываемая сервером баз данных автоматически при совершении некоторого события, обычно под событием понимается выполнение определенного рода операторов изменения базы данных. При определении триггера указывается вид события, возбуждающего триггер, условие его срабатывания и набор операторов, которые должны быть выполнены при выполнении условия. Имеется несколько практических областей применения механизма триггеров. Триггеры позволяют поддерживать логическую целостность базы данных в тех случаях, когда пользовательская транзакция не может не нарушить эту целостность. Вторая область применения триггеров – автоматический мониторинг действий конечных пользователей.

Однако в распространенном на сегодня стандарте SQL-92 механизм триггеров не специфицирован. В СУБД, которые поддерживают этот механизм, соответствующие языковые средства и их семантика различаются. Например, в Oracle V.7 для определения триггеров можно использовать процедурное расширение языка SQL PL/SQL.

Лекция 7. Встраивание операторов языка SQL в программу.

В базе данных могут храниться хранимые процедуры. В стандарте SQL-92 вообще не встречается термин "хранимая процедура". В стандарте специфицированы два способа взаимодействия прикладной программы с сервером баз данных. Первый, наиболее часто используемый способ состоит во встраивании операторов языка SQL в программу, написанную на одном из традиционных языков программирования. В самом стандарте определены правила встраивания SQL в программы, которые написаны на языках Си, Паскаль, Фортран, Ада и т.д. Второй способ основан на специфицированном в стандарте "языке модулей SQL". С использованием этого языка можно определить модуль, содержащий несколько процедур, каждая из которых соответствует некоторому параметризованному оператору SQL. В прикладной программе содержатся не операторы SQL, а лишь вызовы процедур, с указанием фактических параметров, из модуля SQL, с которым эта прикладная программа связана, правила связывания в стандарте не определены. Однако стандарт не обязывает следовать каким-то конкретным правилам при реализации встроенного SQL или языка модулей.

Что касается встроенного SQL, то во всех реализациях прикладная программа, содержащая операторы SQL, подвергается предкомпиляции с помощью соответствующего программного продукта, который входит в состав программного обеспечения сервера баз данных. В результате предкомпиляции всегда формируется текст прикладной программы на используемом языке программирования, уже не содержащий напрямую текста на языке SQL. Каждое вхождение оператора SQL в исходный текст программы, заменяется на вызов некоторой процедуры или функции в синтаксисе включающего языка программирования. Основная разница между разными реализациями состоит в том, что из себя представляет эта процедура или функция. В большинстве систем, Oracle, Informix, Sybase, CA-OpenIngres, такая процедура представляет собой обращение к клиентской части СУБД с передачей в качестве параметра текста оператора на языке SQL. Вся обработка оператора, включая его грамматический разбор и выработку процедурного плана, производится сервером во время выполнения прикладной программы, при повторном выполнении оператора сервер использует уже подготовленный план. Однако имеется и другой подход, исторически используемый компанией IBM в ее серии продуктов DB2. В DB2 предкомпилятор, выбрав текст очередного оператора SQL, сам обращается к серверу с запросом на компиляцию оператора. Сервер выполняет грамматический разбор и строит процедурный план выполнения оператора, сохраняя этот план в базе данных и привязывая к соответствующей прикладной программе. В качестве ответа от сервера предкомпилятор получает идентификатор, указание которого серверу повлечет выполнение соответствующего плана, имя удаленной процедуры.

Лекция 8. Распределенная обработка данных.

Распределенная обработка данных – методика выполнения прикладных программ группой систем.

Сущность ее заключается в том, что пользователь получает возможность работать с сетевыми службами и прикладными процессами, расположенными в нескольких взаимосвязанных абонентских системах. При этом возможны несколько видов работ, которые он может выполнять: удаленный запрос, например, команда, позволяющая посылать одиночную заявку на выполнение обработки данных; удаленная трансакция, осуществляющая направление группы запросов прикладному процессу; распределенная трансакция, дающая возможность использования нескольких серверов и прикладных процессов, выполняемых в группе абонентских систем.

Для распределенной обработки осуществляется сегментация прикладных программ. Передача данных происходит при помощи удаленного вызова процедур либо электронной почты. Первая технология характеризуется высоким быстродействием, а вторая – низкой стоимостью. Известны также программные средства Системы Управления Распределенной Базой Данных (СУРБД), содержатся инструментальные средства распределенной среды обработки данных.

Распределенная среда обработки данных – представляет собой технологию распределенной обработки данных.

Эта среда обычно - набор сетевых служб, предназначенный для выполнения прикладных процессов, рассредоточенных по группе абонентских систем гетерогенной сети. Основные ее компоненты показаны в табл. 1.

Таблица 1. Основные компоненты распределенной обработки данных.

№ п/п	Служба	Выполняемые функции
1.	Имена	База Данных имен пользователей и средств, предназначенных для доступа пользователей к сетевым службам.
2.	Удаленный доступ	Технология, обеспечивающая взаимодействие двух прикладных программ, расположенных в различных абонентских системах.
3.	Защита данных	Программное Обеспечение разрешения на доступ к ресурсам системы или сети.
4.	Многопоточностъ	Программы, обеспечивающие одновременное выполнение нескольких задач.

Системы, имеющие программы распределенной среды, соответственно, являются серверами и клиентами. Серверы связаны (рис. 1) друг с другом ло-

гическими каналами, по которым передают друг другу файлы. Каждый сервер имеет свою группу клиентов.

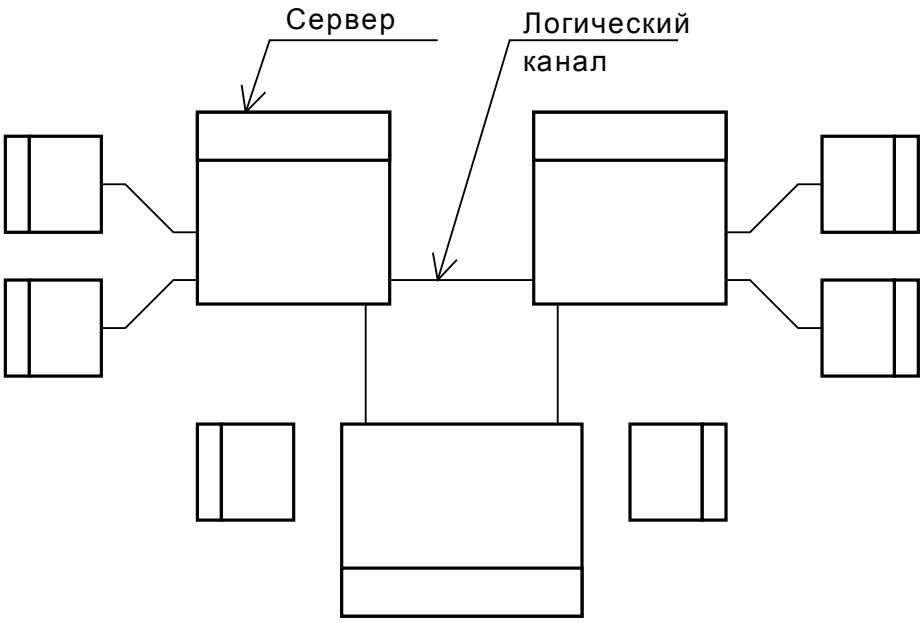


Рис. 1. Связь серверов.

Среда чаще всего имеет трехступенчатую архитектуру: прикладная программа – база данных – клиент. Функции, выполняемые средой, включают прикладные службы:

- каталогов, позволяющую клиентам находить нужные им серверы;
- интерфейса многопоточной обработки;
- удаленного вызова процедур;
- обслуживания файлов;
- безопасности данных;
- времени, синхронизирующей часы в абонентских системах.

Программное Обеспечение среды погружается, как правило, в Сетевую Операционную Систему. Серверы могут иметь свои, различные операционные системы. В роли сервера может, также, выступать главный компьютер со своей операционной системой.

Функционирование распределенной среды требует выполнения ряда административных задач. К ним, в первую очередь, относятся средства:

- регистрации и контроля за лицензиями пользователей на работу с прикладными программами;
- унифицированных интерфейсов прикладных программ;
- обеспечения безопасности данных;
- инвентаризации программного и технического обеспечения абонентских систем, работающих в сети.

С точки зрения логического управления среда обработки данных делится на ячейки распределенной среды обработки . В каждую из них может включаться от нескольких единиц до тысяч абонентских систем. Размеры ячеек территориально не ограничены. Входящие в одну и ту же ячейку системы могут быть расположены даже на разных континентах.

Лекция 9. Структура программных модулей, разработка алгоритмов, логический анализ структур АСОИУ.

Современные технологии разработки программного обеспечения информационных систем предполагают большое разнообразие концепций, подходов и методик. Наиболее популярные из них, это структурный и объектный подходы.

Структурный метод, о котором шла речь при разработке информационного обеспечения, при создании программного обеспечения базируется на использовании технологии моделирования потоков данных.

Диаграммы потоков данных (DFD) являются основным средством моделирования функциональных требований к проектируемой информационной системе. С их помощью эти требования . представляются в виде иерархии функциональных компонентов (процессов), связанных , потоками данных. Главная цель такого представления – продемонстрировать, как каждый процесс преобразует свои входные данные в выходные, а также выявить от-

ношения между этими процессами. Диаграммы потоков данных известны очень давно.

В основе методологии (Gane/Sarson) лежит построение модели анализируемой системы – проектируемой или реально существующей. В соответствии с методологией модель системы определяется как иерархия диаграмм потоков данных (ДПД или DFD), описывающих асинхронный процесс преобразования информации от ее ввода в систему до выдачи пользователю. Диаграммы верхних уровней иерархии (контекстные диаграммы) определяют основные процессы или подсистемы системы с внешними входами и выходами. Они детализируются при помощи диаграмм нижнего уровня. Такая декомпозиция продолжается, создавая многоуровневую иерархию диаграмм, до тех пор, пока не будет достигнут такой уровень декомпозиции, на котором процесс становятся элементарными и детализировать их далее невозможно.

Источники информации (внешние сущности) порождают информационные потоки (потоки данных), переносящие информацию к подсистемам или процессам. Те в свою очередь преобразуют информацию и порождают новые потоки, которые переносят информацию к другим процессам или подсистемам, накопителям данных или внешним сущностям – потребителям информации. Таким образом, основными компонентами диаграмм потоков данных являются:

- внешние сущности;
- системы/подсистемы;
- процессы;
- накопители данных;
- потоки данных.

Внешняя сущность представляет собой материальный предмет или физическое лицо, представляющее собой источник или приемник информации, например, заказчики, персонал, поставщики, клиенты. Определение некоторого объекта или системы в качестве внешней сущности указывает на то, что она находится за пределами границ анализируемой системы. В процессе анализа некоторые внешние сущности могут быть перенесены внутрь диаграммы анализируемой системы, если это необходимо, или, наоборот, часть процессов может быть вынесена за пределы диаграммы и представлена как внешняя сущность.

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 т.д.

Номер процесса служит для его идентификации. В поле имени вводится наименование процесса в виде предложения с активным недвусмысленным глаголом в неопределенной форме (вычислить, рассчитать, проверить,

определить, создать, получить), за которым следуют существительные в винительном падеже, например:

- "Ввести сведения о клиентах";
- "Выдать информацию о текущих расходах";
- "Проверить кредитоспособность клиента".

Использование таких глаголов, как "обработать", "модернизировать" или "отредактировать" означает, как правило, недостаточно глубокое понимание данного процесса и требует дальнейшего анализа.

Информация в поле физической реализации показывает, какое подразделение организации, программа или аппаратное устройство выполняет данный процесс.

Накопитель данных представляет собой абстрактное устройство для хранения информации, которую можно в любой момент поместить в накопитель и через некоторое время извлечь, причем способы помещения и извлечения могут быть любыми.

Накопитель данных в общем случае является прообразом будущей базы данных и описание хранящихся в нем данных должно быть увязано с информационной моделью.

Поток данных на диаграмме изображается линией, оканчивающейся стрелкой, которая показывает направление потока. Каждый поток данных имеет имя, отражающее его содержание.

Первым шагом при построении иерархии ДПД является построение контекстных диаграмм. Обычно при проектировании относительно простых систем строится единственная контекстная диаграмма со звездообразной топологией, в центре которой находится так называемый главный процесс, соединенный с приемниками и источниками информации, посредством которых с системой взаимодействуют пользователи и другие внешние системы.

Если же для сложной системы ограничиться единственной контекстной диаграммой, то она будет содержать слишком большое количество источников и приемников информации, которые трудно расположить на листе бумаги нормального формата, и кроме того, единственный главный процесс не раскрывает структуры распределенной системы. Признаками сложности (в смысле контекста) могут быть:

- наличие большого количества внешних сущностей (десять и более);
- распределенная природа системы;
- многофункциональность системы с уже сложившейся или выявленной группировкой функций в отдельные подсистемы.

Для сложных систем строится иерархия контекстных диаграмм. При этом контекстная диаграмма верхнего уровня содержит не единственный главный процесс, а набор подсистем, соединенных потоками данных. Контекстные диаграммы следующего уровня детализируют контекст и структуру подсистем.

Разработка контекстных диаграмм решает проблему строгого определения функциональной структуры сиси на самой ранней стадии ее проекти-

рования, что особенно важно для сложных многофункциональных систем, в разработке которых участвуют разные организации и коллективы разработчиков.

Для каждой подсистемы, присутствующей на контекстных диаграммах, выполняется ее детализация при помощи ДПД. Каждый процесс на ДПД, в свою очередь, может быть детализирован при помощи ДПД или миниспецификации.

Миниспецификация (описание логики процесса) должна формулировать его основные функции таким образом, чтобы в дальнейшем специалист, выполняющий реализацию проекта, смог выполнить их или разработать соответствующую программу.

При построении иерархии ДПД переходить к детализации процессов следует только после определения содержания всех потоков и накопителей данных, которое описывается при помощи структур данных. Структуры данных конструируются из элементов данных и могут содержать альтернативы, условные вхождения и итерации. Условное вхождение означает, что данный компонент может отсутствовать в структуре. Альтернатива означает, что в структуру может входить один из перечисленных элементов. Итерация означает вхождение любого числа элементов в указанном диапазоне, Для каждого элемента данных может указываться его тип (непрерывные или дискретные данные). Для непрерывных данных может указываться единица измерения (кг, см и т.п.), диапазон значений, точность представления и форма физического кодирования. Для дискретных данных может указываться таблица допустимых значений.

Лекция 10. Разработка моделей защиты данных.

Большое внимание в настоящее время уделяется вопросам формирования принципов построения механизмов защиты информации (ЗИ) и системы требований к ним. На основе имеющегося опыта можно сформулировать следующие фундаментальные принципы организации защиты информации:

- системность;
- специализированность;
- неформальность.

Основные требования принципа *системности* сводятся к тому, что для обеспечения надежной защиты информации в современных АСОИУ должна быть обеспечена надежная и согласованная защита во всех структурных элементах, на всех технологических участках автоматизированной обработки информации и во все время функционирования АСОИУ.

Специализированность как принцип организации защиты предполагает два аспекта:

1) ввиду специфических особенностей рассматриваемой проблемы надежный механизм защиты может быть спроектирован и организован лишь профессиональными специалистами по защите информации,

2) для обеспечения эффективного функционирования механизма защиты в составе АСОИУ должны функционировать специалисты по защите информации.

В соответствии с данным принципом значительное распространение за рубежом получает специализация по различным аспектам ЗИ. В США, например, на вопросах ЗИ специализируется свыше 100 фирм.

Принцип *неформальности* означает, что методология проектирования механизма защиты и обеспечения его функционирования в основе своей является неформальной. Эта неформальность интерпретируется в том смысле, что в настоящее время не существует инженерной методики проектирования механизма защиты в традиционном понимании этого термина.

Общие требования к механизму защиты следующие:

1) адекватность, т.е. обеспечение требуемого уровня защиты (определяется степенью секретности подлежащей обработке информации) при минимальных издержках на создание механизма защиты и обеспечение его функционирования;
2) удобство для пользователей, основу чего составляет требование, чтобы механизм защиты не создавал для пользователей дополнительных трудностей, требующих значительных усилий для их преодоления; минимизация привилегий в доступе, предоставляемых пользователям, т.е. каждому пользователю должны предоставляться только действительно необходимые ему права по обращению к ресурсам системы и данным;
3) полнота контроля, т.е. обязательный контроль всех обращений к защищаемым данным; наказуемость нарушений, причем наиболее распространенной мерой наказания является отказ в доступе к системе;
4) экономичность механизма, т.е. обеспечение минимальности расходов на создание и эксплуатацию механизма;
5) несекретность проектирования, т.е. механизм защиты должен функционировать достаточно эффективно даже в том случае, если его структура и содержание известны злоумышленнику.

В автоматизированных банках данных должно быть предусмотрено наличие в них средств идентификации пользователей и ресурсов системы с периодической сменой идентифицирующей информации, многоаспектного разграничения доступа к элементам баз данных (по элементам, по разрешенным процедурам, по условиям операций и др.), криптографического закрытия данных, регистрации обращений к защищаемым данным, контроля за использованием защищаемых данных и т.д.

Основные положения по разработке систем ЗИ могут быть сформулированы так:

1) защита информации является не разовым мероприятием и даже не совокупностью мероприятий, а непрерывным процессом, который должен протекать (осуществляться) во все время и на всех этапах жизненного цикла АСОИУ;

2) осуществление непрерывного процесса защиты информации возможно лишь на базе промышленного производства средств защиты;

3) создание эффективных механизмов защиты может быть осуществлено высококвалифицированными специалистами-профессионалами в области защиты информации;

4) поддержание и обеспечение надежного функционирования механизмов защиты информации в АСОИУ сопряжено с решением специфических задач и поэтому может осуществляться лишь профессионально подготовленными специалистами.

Сохранность информации может быть нарушена в двух основных случаях: при получении несанкционированного доступа к информации и нарушении функционирования ЭВМ. Система защиты от этих угроз включает следующие основные элементы: защиту АСОИУ и ее аппаратуры, организационные мероприятия по обеспечению сохранности информации, защиту операционной системы, файлов, терминалов и каналов связи. Следует при этом иметь в виду, что все типы защиты взаимосвязаны и при выполнении своих функций хотя бы одной из них сводит на нет усилия других. Предлагаемые и реализованные схемы защиты информации в СОД очень разнообразны, что вызвано в основном выбором наиболее удобного и легко осуществимого метода контроля доступа, т.е. изменением функциональных свойств системы.

В качестве классификационного признака для схем защиты можно выбрать их функциональные свойства. На основе и этого признака выделяются системы: без схем защиты; с полной защитой; с единой схемой защиты.

Лекция 11. Автоматизация проектирования. CASE - средства.

Под термином "CASE-средства" (Computer Aided Software Engineering) понимаются программные средства, поддерживающие процессы создания и сопровождения АСОИУ, включая анализ и формулировку требований, проектирование прикладного ПО и баз данных, генерацию кода, тестирование, документирование, обеспечение качества, конфигурационное управление и управление проектом, а также другие процессы. CASE-средства вместе с системным ПО и техническими средствами образуют полную среду разработки АСОИУ.

Появлению CASE-технологии и CASE-средств предшествовали исследования в области методологии программирования. Программирование обрело черты системного подхода с разработкой и внедрением языков высокого уровня, методов структурного и модульного программирования.

CASE-технология представляет собой методологию проектирования АСОИУ, а также набор инструментальных средств, позволяющих в наглядной форме моделировать предметную область, анализировать эту модель на всех этапах разработки и сопровождения АСОИУ и разрабатывать приложения в соответствии с информационными потребностями пользователей. Большинство существующих CASE-средств основано на методологиях

структурного (в основном) или объектно-ориентированного анализа и проектирования, использующих спецификации в виде диаграмм или текстов для описания внешних требований, связей между моделями системы, динамики поведения системы и архитектуры программных средств .

Наиболее трудоемкими этапами разработки АСОИУ являются этапы анализа и проектирования, в процессе которых CASE-средства обеспечивают качество принимаемых технических решений и подготовку проектной документации. Графические средства моделирования предметной области позволяют разработчикам в наглядном виде изучать существующую АСОИУ, перестраивать ее в соответствии с поставленными целями и имеющимися ограничениями.

CASE-средства обладают следующими основными особенностями :

а) имеют мощные графические средства для описания и документирования АСОИУ, обеспечивающие удобный интерфейс с разработчиком и развивающие его творческие возможности;

б) осуществляют интеграцию отдельных компонент CASE-средств, обеспечивающую управляемость процессом разработки систем;

в) используют специальным образом организованное хранилище проектных метаданных (репозитория).

Интегрированное CASE-средство должно содержать следующие компоненты:

а) репозиторий, являющийся основой CASE-средства. Он должен обеспечивать хранение версий проекта и его отдельных компонентов, синхронизацию поступления информации от различных разработчиков при групповой разработке, контроль метаданных на полноту и непротиворечивость;

б) графические средства анализа и проектирования, обеспечивающие создание и редактирование иерархически связанных диаграмм (DFD, ERD и др.), образующих модели АСОИУ;

в) средства разработки приложений, включая языки 4GL и генераторы кодов;

г) средства конфигурационного управления;

д) средства документирования;

е) средства тестирования;

ж) средства управления проектом;

з) средства реинжиниринга.

Современный рынок программных средств насчитывает около 300 различных CASE-средств, наиболее мощные из которых используются практически всеми ведущими западными фирмами.

Все современные CASE-средства могут быть классифицированы в основном по типам и категориям. Классификация по типам отражает функциональную их ориентацию на те или иные процессы ЖЦ.

Классификация по категориям определяет степень интегрированности по выполняемым функциям и включает следующее :

а) отдельные локальные средства, решающие небольшие автономные задачи (tools);

б) набор частично интегрированных средств, охватывающих большинство этапов жизненного цикла систем (toolkit);

в) полностью интегрированные средства, поддерживающие весь ЖЦ систем и связанные общим репозиторием.

Помимо этого CASE-средства можно классифицировать по следующим признакам:

а) применяемым методологиям и моделям систем и БД;

б) степени интегрированности с СУБД;

в) доступным платформам.

Классификация по типам в основном совпадает с компонентным составом CASE-средств.

На сегодняшний день российский рынок программного обеспечения располагает следующими наиболее развитыми CASE-средствами: Vantage Team Builder (Westmount I-CASE), Designer/2000, Silverrun, ERwin+Bpwin, S-Designor, CASE-Аналитик, CASE /4/0, PRO-IV, System Architect, Visible Analyst Workbench, EasyCASE.

Лекция 12. Методологии и технологии проектирования АСОИУ.

Методологии, технологии и инструментальные средства проектирования (CASE-средства) составляют основу проекта любой ИС. Методология реализуется через конкретные технологии и поддерживающие их стандарты, методики и инструментальные средства, которые обеспечивают выполнение процессов ЖЦ.

Технология проектирования определяется как совокупность трех составляющих:

- пошаговой процедуры, определяющей последовательность технологических операций проектирования;
- критериев и правил, используемых для оценки результатов выполнения технологических операций;
- нотаций (графических и текстовых средств), используемых для описания проектируемой системы.

Технологические инструкции, составляющие основное содержание технологии, должны состоять из описания последовательности технологических операций, условий, в зависимости от которых выполняется та или иная операция, и описаний самих операций.

Технология проектирования, разработки и сопровождения ИС должна удовлетворять следующим общим требованиям:

- технология должна поддерживать полный ЖЦ ПО;
- технология должна обеспечивать гарантированное достижение целей разработки ИС с заданным качеством и в установленное время;
- технология должна обеспечивать возможность выполнения крупных проектов в виде подсистем (т.е. возможность декомпозиции проекта на составные части, разрабатываемые группами исполнителей ограниченной численности с последующей интеграцией составных ча-

стей). Опыт разработки крупных ИС показывает, что для повышения эффективности работ необходимо разбить проект на отдельные слабо связанные по данным и функциям подсистемы. Реализация подсистем должна выполняться отдельными группами специалистов. При этом необходимо обеспечить координацию ведения общего проекта и исключить дублирование результатов работ каждой проектной группы, которое может возникнуть в силу наличия общих данных и функций;

- технология должна обеспечивать возможность ведения работ по проектированию отдельных подсистем небольшими группами (3-7 человек). Это обусловлено принципами управляемости коллектива и повышения производительности за счет минимизации числа внешних связей;

технология должна обеспечивать минимальное время получения работоспособной ИС. Речь идет не о сроках готовности всей ИС, а о сроках реализации отдельных подсистем. Реализация ИС в целом в короткие сроки может потребовать привлечения большого числа разработчиков.

Лекция 13. Сущность структурного подхода.

Сущность структурного подхода к разработке ИС заключается в ее декомпозиции, разбиении, на автоматизируемые функции: система разбивается на функциональные подсистемы, которые в свою очередь делятся на подфункции, подразделяемые на задачи и так далее. Процесс разбиения продолжается вплоть до конкретных процедур. При этом автоматизируемая система сохраняет целостное представление, в котором все составляющие компоненты взаимоувязаны. При разработке системы "снизу-вверх" от отдельных задач ко всей системе целостность теряется, возникают проблемы при информационной стыковке отдельных компонентов.

Все наиболее распространенные методологии структурного подхода базируются на ряде общих принципов. В качестве двух базовых принципов используются следующие:

- принцип "разделяй и властвуй" - принцип решения сложных проблем путем их разбиения на множество меньших независимых задач, легких для понимания и решения;
- принцип иерархического упорядочивания - принцип организации составных частей проблемы в иерархические древовидные структуры с добавлением новых деталей на каждом уровне.

Выделение двух базовых принципов не означает, что остальные принципы являются второстепенными, поскольку игнорирование любого из них может привести к непредсказуемым последствиям (в том числе и к провалу всего проекта). Основными из этих принципов являются следующие:

- принцип абстрагирования - заключается в выделении существенных аспектов системы и отвлечения от несущественных;

- принцип формализации - заключается в необходимости строгого методического подхода к решению проблемы;
- принцип непротиворечивости - заключается в обоснованности и согласованности элементов;
- принцип структурирования данных - заключается в том, что данные должны быть структурированы и иерархически организованы.

В структурном анализе используются в основном две группы средств, иллюстрирующих функции, выполняемые системой и отношения между данными. Каждой группе средств соответствуют определенные виды моделей (диаграмм), наиболее распространенными среди которых являются следующие:

- SADT (Structured Analysis and Design Technique) модели и соответствующие функциональные диаграммы;
- DFD (Data Flow Diagrams) диаграммы потоков данных;
- ERD (Entity-Relationship Diagrams) диаграммы "сущность-связь".

На стадии проектирования ИС модели расширяются, уточняются и дополняются диаграммами.

Лекция 14. Методология функционального моделирования SADT.

Методология SADT разработана Дугласом Россом и получила дальнейшее развитие в работе. На ее основе разработана, в частности, известная методология IDEF0 (Icam DEFinition), которая является основной частью программы ICAM (Интеграция компьютерных и промышленных технологий), проводимой по инициативе ВВС США.

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

Основные элементы этой методологии основываются на следующих концепциях:

- графическое представление блочного моделирования. Графика блоков и дуг SADT-диаграммы отображает функцию в виде блока, а интерфейсы входа/выхода представляются дугами, соответственно входящими в блок и выходящими из него. Взаимодействие блоков друг с другом описываются посредством интерфейсных дуг, выражающих "ограничения", которые в свою очередь определяют, когда и каким образом функции выполняются и управляются;
- строгость и точность.

Выполнение правил SADT требует достаточной строгости и точности, не накладывая в то же время чрезмерных ограничений на действия аналитика.

Правила SADT включают:

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

Методология SADT может использоваться для моделирования широкого круга систем и определения требований и функций, а затем для разработки системы, которая удовлетворяет этим требованиям и реализует эти функции. Для уже существующих систем SADT может быть использована для анализа функций, выполняемых системой, а также для указания механизмов, посредством которых они осуществляются.

Лекция 15. Состав функциональной модели.

Результатом применения методологии SADT является модель, которая состоит из диаграмм, фрагментов текстов и глоссария, имеющих ссылки друг на друга. Диаграммы - главные компоненты модели, все функции ИС и интерфейсы на них представлены как блоки и дуги. Место соединения дуги с блоком определяет тип интерфейса. Управляющая информация входит в блок сверху, в то время как информация, которая подвергается обработке, показана с левой стороны блока, а результаты выхода показаны с правой стороны. Механизм (человек или автоматизированная система), который осуществляет операцию, представляется дугой, входящей в блок снизу.

Одной из наиболее важных особенностей методологии SADT является постепенное введение все больших уровней детализации по мере создания диаграмм, отображающих модель.

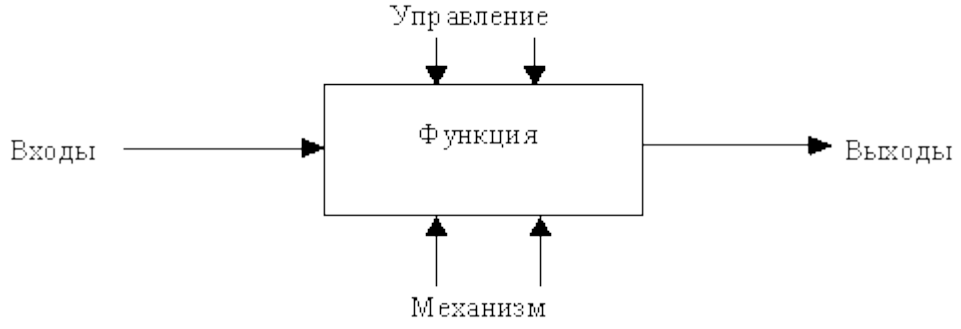


Рис.2. Функциональный блок и интерфейсные дуги.

На рисунке, где приведены четыре диаграммы и их взаимосвязи, показана структура SADT-модели. Каждый компонент модели может быть декомпозирован на другой диаграмме. Каждая диаграмма иллюстрирует "внутреннее строение" блока на родительской диаграмме.

Построение SADT-модели начинается с представления всей системы в виде простейшей компоненты - одного блока и дуг, изображающих интерфейсы с функциями вне системы. Поскольку единственный блок представляет всю систему как единое целое, имя, указанное в блоке, является общим. Это верно и для интерфейсных дуг - они также представляют полный набор внешних интерфейсов системы в целом.

Затем блок, который представляет систему в качестве единого модуля, детализируется на другой диаграмме с помощью нескольких блоков, соединенных интерфейсными дугами. Эти блоки представляют основные подфункции исходной функции. Данная декомпозиция выявляет полный набор подфункций, каждая из которых представлена как блок, границы которого определены интерфейсными дугами. Каждая из этих подфункций может быть декомпозирована подобным образом для более детального представления.

Лекция 16. Типы связей между функциями.

Одним из важных моментов при проектировании ИС с помощью методологии SADT является точная согласованность типов связей между функциями. Различают, по крайней мере, семь типов связывания (табл.).

Таблица

Тип связи	Относительная значимость
Случайная	0
Логическая	1
Временная	2
Процедурная	3
Коммуникационная	4
Последовательная	5
Функциональная	6

Ниже каждый тип связи кратко определен и проиллюстрирован с помощью типичного примера из SADT.

(0) Тип случайной связности: наименее желательный.

Случайная связность возникает, когда конкретная связь между функциями мала или полностью отсутствует. Это относится к ситуации, когда имена данных на SADT-дугах в одной диаграмме имеют малую связь друг с другом.

(1) Тип логической связности. Логическое связывание происходит тогда, когда данные и функции собираются вместе вследствие того, что они попадают в общий класс или набор элементов, но необходимых функциональных отношений между ними не обнаруживается.

(2) Тип временной связности. Связанные по времени элементы возникают вследствие того, что они представляют функции, связанные во времени, когда данные используются одновременно или функции включаются параллельно, а не последовательно.

(3) Тип процедурной связности. Процедурно-связанные элементы появляются сгруппированными вместе вследствие того, что они выполняются в течение одной и той же части цикла или процесса.

(4) Тип коммуникационной связности. Диаграммы демонстрируют коммуникационные связи, когда блоки группируются вследствие того, что они используют одни и те же входные данные и/или производят одни и те же выходные данные (рис. 3.10).

(5) Тип последовательной связности. На диаграммах, имеющих последовательные связи, выход одной функции служит входными данными для следующей функции. Связь между элементами на диаграмме является более тесной, чем на рассмотренных выше уровнях связок, поскольку моделируются причинно-следственные зависимости.

Лекция 17. Моделирование потоков данных.

В основе данной методологии (методологии Gane/Sarson [10]) лежит построение модели анализируемой ИС - проектируемой или реально существующей. В соответствии с методологией модель системы определяется как иерархия диаграмм потоков данных (ДПД или DFD), описывающих асинхронный процесс преобразования информации от ее ввода в систему до выдачи пользователю. Диаграммы верхних уровней иерархии (контекстные диаграммы) определяют основные процессы или подсистемы ИС с внешними входами и выходами. Они детализируются при помощи диаграмм нижнего уровня. Такая декомпозиция продолжается, создавая многоуровневую иерархию диаграмм, до тех пор, пока не будет достигнут такой уровень декомпозиции, на котором процесс становятся элементарными и детализировать их далее невозможно.

Источники информации (внешние сущности) порождают информационные потоки (потоки данных), переносящие информацию к подсистемам или процессам. Те в свою очередь преобразуют информацию и порождают новые потоки, которые переносят информацию к другим процессам или подсистемам, накопителям данных или внешним сущностям - потребителям информации. Таким образом, основными компонентами диаграмм потоков данных являются:

- внешние сущности;
- системы/подсистемы;
- процессы;
- накопители данных;
- потоки данных.

Внешние сущности.

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.

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.

Лекция 18. Построение иерархии диаграмм потоков данных.

Первым шагом при построении иерархии ДПД является построение контекстных диаграмм. Обычно при проектировании относительно простых ИС строится единственная контекстная диаграмма со звездообразной топологией, в центре которой находится так называемый главный процесс, соединенный с приемниками и источниками информации, посредством которых с системой взаимодействуют пользователи и другие внешние системы.

Если же для сложной системы ограничиться единственной контекстной диаграммой, то она будет содержать слишком большое количество источников и приемников информации, которые трудно расположить на листе бумаги нормального формата, и кроме того, единственный главный процесс не раскрывает структуры распределенной системы. Признаками сложности (в смысле контекста) могут быть:

- наличие большого количества внешних сущностей (десять и более); распределенная природа системы;
- многофункциональность системы с уже сложившейся или выявленной группировкой функций в отдельные подсистемы.

Для сложных ИС строится иерархия контекстных диаграмм. При этом контекстная диаграмма верхнего уровня содержит не единственный главный процесс, а набор подсистем, соединенных потоками данных. Контекстные диаграммы следующего уровня детализируют контекст и структуру подсистем.

Иерархия контекстных диаграмм определяет взаимодействие основных функциональных подсистем проектируемой ИС как между собой, так и с внешними входными и выходными потоками данных и внешними объектами (источниками и приемниками информации), с которыми взаимодействует ИС.

Разработка контекстных диаграмм решает проблему строгого определения функциональной структуры ИС на самой ранней стадии ее проектирования, что особенно важно для сложных многофункциональных систем, в раз-

работке которых участвуют разные организации и коллективы разработчиков.

После построения контекстных диаграмм полученную модель следует проверить на полноту исходных данных об объектах системы и изолированность объектов (отсутствие информационных связей с другими объектами).
Для каждой подсистемы, присутствующей на контекстных диаграммах, выполняется ее детализация при помощи ДПД. Каждый процесс на ДПД, в свою очередь, может быть детализирован при помощи ДПД или миниспецификации.

Миниспецификация (описание логики процесса) должна формулировать его основные функции таким образом, чтобы в дальнейшем специалист, выполняющий реализацию проекта, смог выполнить их или разработать соответствующую программу.

Лекция 19. Моделирование данных.

Цель моделирования данных состоит в обеспечении разработчика ИС концептуальной схемой базы данных в форме одной модели или нескольких локальных моделей, которые относительно легко могут быть отображены в любую систему баз данных.

Наиболее распространенным средством моделирования данных являются диаграммы "сущность-связь" (ERD). С их помощью определяются важные для предметной области объекты (сущности), их свойства (атрибуты) и отношения друг с другом (связи). ERD непосредственно используются для проектирования реляционных баз данных.

Нотация ERD была впервые введена П. Ченом (Chen) и получила дальнейшее развитие в работах Баркера [11]. Метод Баркера будет излагаться на примере моделирования деятельности компании по торговле автомобилями. Ниже приведены выдержки из интервью, проведенного с персоналом компании.

Главный менеджер: одна из основных обязанностей - содержание автомобильного имущества. Он должен знать, сколько заплачено за машины и каковы накладные расходы. Обладая этой информацией, он может установить нижнюю цену, за которую мог бы продать данный экземпляр. Кроме того, он несет ответственность за продавцов и ему нужно знать, кто что продает и сколько машин продал каждый из них.

Продавец: ему нужно знать, какую цену запрашивать и какова нижняя цена, за которую можно совершить сделку. Кроме того, ему нужна основная информация о машинах: год выпуска, марка, модель и т.п.

Администратор: его задача сводится к составлению контрактов, для чего нужна информация о покупателе, автомашине и продавце, поскольку именно контракты приносят продавцам вознаграждения за продажи.

Первый шаг моделирования - извлечение информации из интервью и выделение сущностей.

Сущность (Entity) - реальный либо воображаемый объект, имеющий существенное значение для рассматриваемой предметной области, информация о котором подлежит хранению.

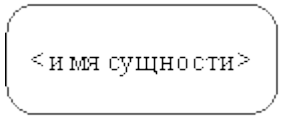


Рис.3. Графическое изображение сущности.

Каждая сущность должна обладать уникальным идентификатором. Каждый экземпляр сущности должен однозначно идентифицироваться и отличаться от всех других экземпляров данного типа сущности.

Лекция 20. Методология IDEF1.

Метод IDEF1, разработанный Т.Рэмей (T.Ramey), также основан на подходе П.Чена и позволяет построить модель данных, эквивалентную реляционной модели в третьей нормальной форме. В настоящее время на основе совершенствования методологии IDEF1 создана ее новая версия - методология IDEF1X. IDEF1X разработана с учетом таких требований, как простота изучения и возможность автоматизации. IDEF1X-диаграммы используются рядом распространенных CASE-средств (в частности, ERwin, Design/IDEF).

Сущность в методологии IDEF1X является независимой от идентификаторов или просто независимой, если каждый экземпляр сущности может быть однозначно идентифицирован без определения его отношений с другими сущностями. Сущность называется зависимой от идентификаторов или просто зависимой, если однозначная идентификация экземпляра сущности зависит от его отношения к другой сущности (рис.).

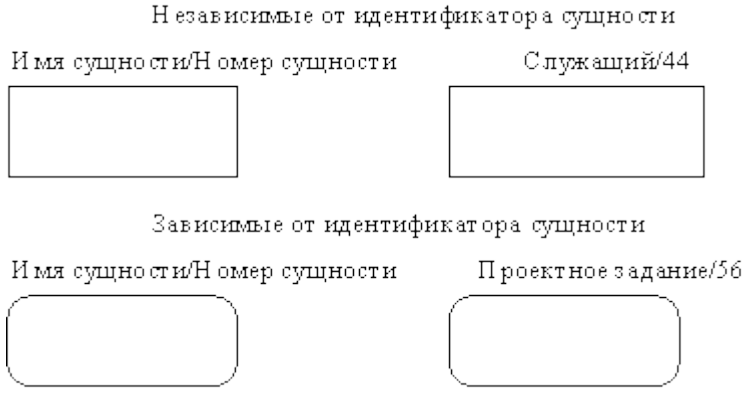


Рис.4. Сущности.

Каждой сущности присваивается уникальное имя и номер, разделяемые косой чертой "/" и помещаемые над блоком.

Связь может дополнительно определяться с помощью указания степени или мощности (количества экземпляров сущности-потомка, которое может

существовать для каждого экземпляра сущности-родителя). В IDEF1X могут быть выражены следующие мощности связей:

- каждый экземпляр сущности-родителя может иметь ноль, один или более связанных с ним экземпляров сущности-потомка;
- каждый экземпляр сущности-родителя должен иметь не менее одного связанного с ним экземпляра сущности-потомка;
- каждый экземпляр сущности-родителя должен иметь не более одного связанного с ним экземпляра сущности-потомка;
- каждый экземпляр сущности-родителя связан с некоторым фиксированным числом экземпляров сущности-потомка.

Лекция 21. Общая классификация архитектур информационных приложений.

Организация информационных систем на основе использования выделенных файл-серверов является наиболее распространенной в связи с наличием большого количества персональных компьютеров разного уровня развитости и сравнительной дешевизны связывания PC в локальные сети. Чем привлекает такая организация? Скорее всего, тем, что при опоре на файл-серверные архитектуры сохраняется автономность прикладного и большей части системного программного обеспечения, работающего на каждой PC сети. Фактически, компоненты информационной системы, выполняемые на разных PC, взаимодействуют только за счет наличия общего хранилища файлов, которое хранится на файл-сервере. В классическом случае в каждой PC дублируются не только прикладные программы, но и средства управления базами данных. Файл-сервер представляет собой разделяемое всеми PC комплекса расширение дисковой памяти.

Кратко перечислим основные достоинства и недостатки файл-серверных архитектур.

Основным достоинством является простота организации. Проектировщики и разработчики информационной системы находятся в привычных и комфортных условиях IBM PC в среде MS-DOS, Windows или какого-либо облегченного варианта Windows NT. Имеются удобные и развитые средства разработки графического пользовательского интерфейса, простые в использовании средства разработки систем баз данных и/или СУБД. Но во многом эта простота является кажущейся.

Во-первых, информационной системе предстоит работать с базой данных. Следовательно, эта база данных должна быть спроектирована. Почему-то часто разработчики файл-серверных приложений считают, что по причине простоты средств управления базами данных проблемой проектирования базы данных можно пренебречь. Это неправильно. База данных есть база данных. Чем качественнее она спроектирована, тем больше шансов впоследствии эффективно использовать информационную систему. Сложность проектирования базы данных определяется объективной сложностью моделируемой предметной области.

Во-вторых, необходимыми требованиями к базе данных информационной системы являются поддержание ее целостного состояния и гарантированная надежность хранения информации. Минимальными условиями, при соблюдении которых можно удовлетворить эти требования, являются:

1. наличие транзакционного управления,
2. хранение избыточных данных (например, с применением методов журнализации),

возможность формулировать ограничения целостности и проверять их соблюдение.

Лекция 22. Клиент-серверные приложения.

Под клиент-серверным приложением будем понимать информационную систему, основанную на использовании серверов баз данных. Общее представление информационной системы в архитектуре "клиент-сервер" показано на рисунке.

На стороне клиента выполняется код приложения, в который обязательно входят компоненты, поддерживающие интерфейс с конечным пользователем, производящие отчеты, выполняющие другие специфичные для приложения функции.

Клиентская часть приложения взаимодействует с клиентской частью программного обеспечения управления базами данных, которая, фактически, является индивидуальным представителем СУБД для приложения.

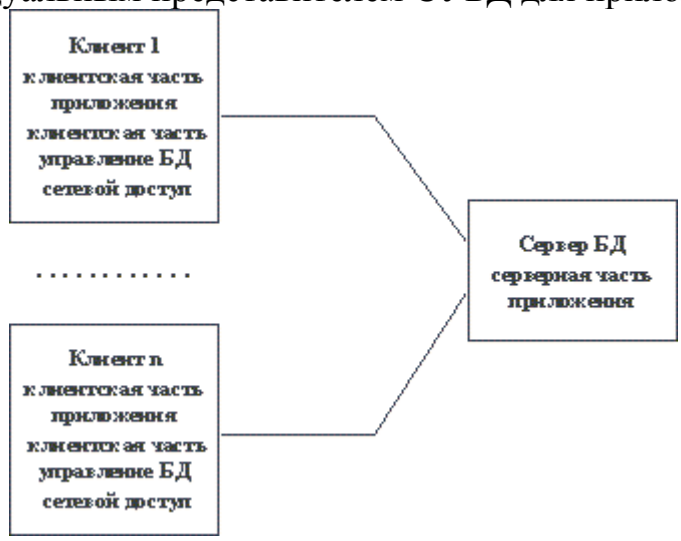


Рис.5. Общее представление информационной системы в архитектуре "клиент-сервер".

Интерфейс между клиентской частью приложения и клиентской частью сервера баз данных, как правило, основан на использовании языка SQL. Поэтому такие функции, как, например, предварительная обработка форм, предназначенных для запросов к базе данных, или формирование результирующих отчетов выполняются в коде приложения. Клиентская часть сервера

баз данных, используя средства сетевого доступа, обращается к серверу баз данных, передавая ему текст оператора языка SQL.

Обычно компании, производящие развитые серверы баз данных, стремятся к тому, чтобы обеспечить возможность использования своих продуктов не только в стандартных на сегодняшний день TCP/IP-ориентированных сетях, но в сетях, основанных на других протоколах, например, SNA или IPX/SPX. Поэтому, при организации сетевых взаимодействий между клиентской и серверной частями СУБД, часто используются не стандартные средства высокого уровня.

Лекция 23. Intranet-приложения.

Возникновение и внедрение в широкую практику высокоуровневых служб Всемирной Сети Сетей Internet (e-mail, ftp, telnet, Gopher, WWW и т.д.) естественным образом повлияли на технологию создания корпоративных информационных систем, породив направление, известное теперь под названием Intranet. Информационная Intranet-система - это корпоративная система, в которой используются методы и средства Internet. Такая система может быть локальной, изолированной от остального мира Internet, или опираться на виртуальную корпоративную подсеть Internet. В последнем случае особенно важны средства защиты информации от несанкционированного доступа.

В общем случае в Intranet-системе могут использоваться все возможные службы Internet, наибольшее внимание привлекает гипермедийная служба WWW (World Wide Web - Всемирная Паутина). Для этого имеются две основные причины. Во-первых, с использованием языка гипермедийной разметки документов HTML можно сравнительно просто разработать удобную для использования информационную структуру, которая в дальнейшем будет обслуживаться одним из готовых Web-серверов. Во-вторых, наличие нескольких готовых к использованию клиентских частей - браузеров, или "обходчиков" избавляет от необходимости создавать собственные интерфейсы с пользователями, предоставляя им удобные и развитые механизмы доступа к информации. В ряде случаев такая организация корпоративной информационной системы (рис) оказывается достаточной для удовлетворения потребностей компании.

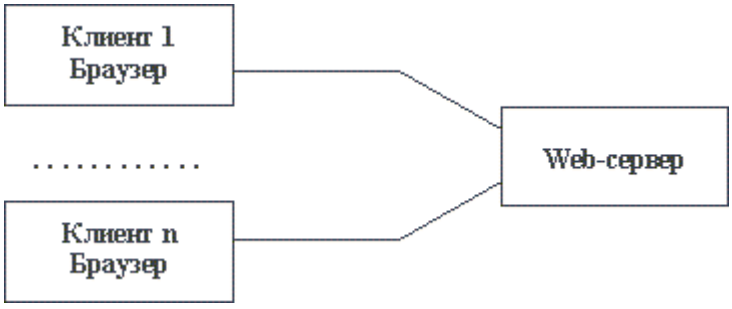


Рис.6. Простая организация Intranet-системы с использованием средств WWW.

Однако, при всех своих преимуществах, простота организации, удобство использования, стандартность интерфейсов, эта схема обладает сильными ограничениями. Прежде всего, как видно из рисунка, в информационной системе отсутствует прикладная обработка данных. Все, что может пользователь, это только просмотреть информацию, поддерживаемую Web-сервером. Гипертекстовые структуры трудно модифицируются.

Лекция 24. Склады данных и системы оперативной аналитической обработки данных.

У аналитических отделов организации и у высшего звена управляющего состава имеются следующие задачи: проанализировав поведение корпорации на рынке с учетом сопутствующих внешних факторов и спрогнозировав хотя бы ближайшее будущее, выработать тактику, а возможно, и стратегию организации. Для решения таких задач требуются данные и прикладные программы, отличные от тех, которые используются в оперативных информационных системах. В последние время все более популярным становится подход, основанный на концепциях склада данных и системы оперативной аналитической обработки данных. Трудно производить долговременные прогнозы бизнес-деятельности, но анализ прошлого и краткосрочные прогнозы будущего могут оказаться очень полезными.

Поскольку термины, связанные со складами данных не так давно появились и на английском языке, и смысл их постоянно уточняется, трудно найти правильные русскоязычные эквиваленты. На сегодняшний день "datawarehouse" разными авторами переводится на русский язык как "хранилище данных", "информационное хранилище", "склад данных". Поскольку термин "хранилище" явно перегружен (он соответствует и английским терминам "storage" и "repository"), будем использовать термин "склад данных". Термин "data mart" по смыслу толкования - термин "рынок данных. Постепенно терминология будет согласована, но это произойдет только тогда, когда склады данных будут активно использоваться в России.

Прежде чем перейти к техническим аспектам, рассмотрим соответствующие вопросы на концептуальном уровне. Начнем с того, что главным образом различает оперативные и аналитические информационные приложения с точки зрения обеспечения требуемых данных, речь идет о так называемых OLAP-системах (от On-Line Analitical Processing), т.е. аналитических системах, помогающих принимать бизнес-решения за счет динамически производимых анализа, моделирования и/или прогнозирования данных.

Основным источником информации, поступающей в оперативную базу данных, является деятельность организации. Для проведения анализа данных требуется привлечение внешних источников информации, например, статистических отчетов, тем самым, склад данных должен включать как внутренние корпоративные данные, так и внешние данные, характеризующие рынок в целом.

Если для оперативной обработки, как правило, требуются свежие данные, обычно в оперативных базах данных информация сохраняется не более нескольких месяцев, то в складе данных нужно поддерживать хранение информации о деятельности корпорации и состоянии рынка на протяжении нескольких лет, для проведения достоверных анализа и прогнозирования. Как следствие, аналитические базы данных имеют объем как минимум на порядок больший, чем оперативные.

Лекция 25. Интегрированные распределенные приложения.

Не возникает никаких проблем, если с самого начала информационное приложение проектируется и разрабатывается в духе подхода открытых систем: все компоненты являются мобильными и интероперабельными, общее функционирование системы не зависит от конкретного местоположения компонентов, система обладает хорошими возможностями сопровождаемости и развития. На практике этот идеал является трудно достижимым. По разным причинам возникают потребности в интеграции независимо и по-разному организованных информационно-вычислительных ресурсов. Ни в одной действительно серьезной распределенной информационной системе не удастся обойтись без применения некоторой технологии интеграции. Существует путь решения этой проблемы, который сам лежит в русле открытых систем, - подход, предложенный крупнейшим международным консорциумом OMG (Object Management Group).

Остановимся на некоторых факторах, стимулирующих использование методов интеграции разнородных информационных ресурсов [12].

Неоднородность, распределенность и автономность информационных ресурсов системы. Неоднородность ресурсов может быть синтаксической, при их представлении используются, например, разные модели данных, и/или семантической, используются разные виды семантических правил, детализируются и/или агрегируются разные аспекты предметной области. Возможна и чисто реализационная неоднородность информационных ресурсов, обусловленная использованием разных компьютерных платформ, операционных систем, систем управления базами данных, систем программирования и т.д.

Потребности в интеграционном комплексировании компонентов информационной системы. Наиболее естественным способом организации сложной информационной системы является ее иерархически-вложенное построение. Более сложные функционально-ориентированные компоненты строятся на основе более простых компонентов, которые могли проектироваться и разрабатываться независимо.

Реинжинерия системы. После создания начального варианта информационной системы неизбежно последует процесс ее непрерывных переделок, реинжинерии, обусловленный развитием и изменением соответствующих бизнес-процессов корпорации. Реконструкция системы не должна быть

революционной. Все компоненты, не затрагиваемые процессом реинжиниринга, должны сохранять работоспособность.

Решение проблемы унаследованных систем. Любая компьютерная система со временем становится бременем корпорации. Постоянно, и чем раньше, тем лучше, приходится решать задачу встраивания устаревших информационных компонентов в систему, основанную на новой технологии. Нужно, чтобы эта задача была разрешимой, т.е. чтобы компоненты унаследованных систем сохраняли интероперабельность.

• 7. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ КУРСОВОГО ПРОЕКТА

Настоящие методические указания содержат основные требования и рекомендации по организации, выполнению и защите курсовых проектов (работ) по направлению 654600 - “Информатика и вычислительная техника“ по специальности 230102 -“Автоматизированные системы обработки информации и управления”.

В основу этих указаний положены “Инструкция по подготовке курсовых проектов (работ) в высших учебных заведениях” и квалификационная характеристика дипломированного специалиста по направлению 654600 “Информатика и вычислительная техника“ по специальности 230102 - “ Автоматизированные системы обработки информации и управления ”.

Общие положения.

Курсовой проект выполняется при обучении в девятом семестре в соответствии с установленным учебным планом.

Целью выполнения курсового проекта является формирование у студентов навыков:

• самостоятельной научно-исследовательской деятельности;
• практической деятельности;
• грамотного оформления полученных результатов в печатном виде;
• представления результатов своей работы в виде научного доклада;
• защиты полученных результатов в дискуссии.

Выбор темы работы.

Тематика курсовых проектов определяется преподавателями кафедр, осуществляющих руководство научной работой студентов. Перечень предлагаемых тем (названий) работ с указанием научного руководителя доводится до сведения студентов в течение первых трех недель текущего учебного года.

Студент самостоятельно выбирает научного руководителя и тему работы в соответствии со своими интересами, о чем лично сообщает выбранному им научному руководителю. В ходе предварительного обсуждения выбранной темы с научным руководителем и в процессе выполнения работы тема

может быть изменена по согласованию между научным руководителем и студентом.

Выбор темы должен быть сделан студентом в течение четвертой недели текущего учебного года.

Проект выполняется в течение семестра и может быть продолжением ранее начатого исследования или развитием результатов, полученных студентом в течение предшествующих лет обучения.

Содержание курсовой работы.

Курсовой проект, как правило, представляет собой:

1. Исследование актуальной задачи по специальности;

2. Разработку информационной системы (ИС) или иного программного продукта;

Исследование задачи включает следующие разделы:

1. Описание (постановка) задачи.

2. Обоснование актуальности задачи.

3. Обзор информации, содержащейся в открытых источниках, посвященных данной задаче или области исследований.

4. Исследование задачи:

а) классификация задачи, т. е. отнесение ее к некоторому известному классу задач;

б) описание известных методов решения задач этого класса;

в) описание особенностей исследуемой задачи, ее отличительных черт, которые не позволяют применять существующие методы в стандартном виде;

г) предложения по модификации существующих методов для решения задачи (близких задач) или по модификации самой задачи для применения существующих методов;

д) описание предлагаемых методов решения или подходов к решению с обоснованием их применимости к данной задаче;

е) описание возникающих в процессе решения проблем или других побочных, вспомогательных или параллельных задач.

Разработка информационной (программной) системы включает следующие разделы:

1. Постановка задачи:

а) описание предметной области, например, бизнес-процессов, протекающих в предметной области;

б) описание информационных потоков;

в) описание процессов обработки информации, управления и т. п., требующих автоматизации.

2. Обзор существующих программных продуктов, выполняющих аналогичные функции. Их достоинства и недостатки. Сравнение, классификация.

3. Обоснование необходимости разработки.

4. Описание математической задачи:

а) постановка задачи;

б) исследование задачи;

в) математическая модель;

г) метод решения возникающей математической задачи;

д) алгоритм, реализующий метод решения.

5. Проект АСОИУ:

а) архитектура, аппаратная платформа, ОС, СУБД, состав других программных продуктов;

б) описание структуры БД: таблицы, индексы, процедуры, триггеры, события, запросы;

в) описание логики программ и интерфейсов;

г) алгоритмы обработки данных.

6. Реализация АСОИУ:

а) назначение и функции программы, режимы работы программы;

б) описание категорий пользователей программы, разграничения прав пользователей;

в) описание последовательности пользовательских интерфейсов, реализующих каждую функцию АИС;

г) описание входных данных;

д) описание выходных данных;

е) описание методов защиты данных в ИС и информационных систем в целом;

ж) технические характеристики ИС (тип и минимально необходимые аппаратные ресурсы вычислительной системы, требуемое программное обеспечение и т. п.).

7. Анализ:

а) внедрение;

б) области применения АИС;

в) достоинства и недостатки по сравнению с перечисленными ранее в разделе 2 аналогами;

г) описание возникших в процессе разработки проблем, их причины, предложения по решению;

д) направления дальнейшего развития АИС.

Реферативная работа включает следующие разделы:

1. Обоснование актуальности выбранной тематики и описание целей выполнения работы.

2. Систематизация и анализ найденных в научной печати, в сети Интернет и других источниках материалов.

3. Выводы.

4. Предложения по использованию результатов работы в конкретных областях и возможные направления дальнейших исследований.

Порядок выполнения курсовой работы.

Курсовой проект выполняется в течение девятого семестра.

По итогам работы, до начала сессии, студент в установленный кафедрой срок представляет подготовленный в печатном виде материал и по решению кафедры делает сообщение по теме курсового проекта на семинаре. Руководитель на титульном листе выставляет оценку, заверяя ее своей подписью.

На титульном листе фиксируется срок представления отчета. В случае несоблюдения сроков представления или низкого качества отчета, его оценка снижается, как минимум, на один балл.

Студенты, не представившие в срок курсовые проекты, не допускаются к публичной защите. В этом случае защита переносится на начало сессии.

Оценка за курсовую работу складывается из следующих оценок:

- оценки руководителя;
- оценки публичной защиты;
- оценки оформления.

При выставлении оценки учитываются также сроки представления печатного варианта курсового проекта.

Выступление с докладом по промежуточным результатам курсового проекта на студенческой научной конференции может быть засчитано с выставлением заслуженной оценки (отлично, хорошо, удовлетворительно).

В случае отсутствия научного руководителя в период представления курсового проекта (командировка, болезнь и т. п.), курсовой проект сдается на кафедру в установленные сроки. Оценка руководителя выставляется позднее.

Структура курсового проекта.

Работа начинается с титульного листа стандартной формы, за которым следует лист задания, оглавление работы, введение, нескольких разделов, заключения, списка использованных научных источников, приложений.

Введение содержит общий обзор работы, позволяющий составить общее представление об исследуемой проблеме и полученных результатах. Во введении может быть предложена краткая аннотация отдельных разделов работы. Первый раздел должен содержать достаточно подробное описание проблемы, поставленной перед исполнителем с обоснованием ее актуальности и анализ современного состояния исследований и разработок в данной области.

В последующих разделах, число которых произвольно, описываются результаты, полученные по отдельным аспектам исследуемой проблемы. Каждый раздел может разбиваться на подразделы.

Заключение содержит перечень основных результатов, полученных в работе, и сделанных выводов. В него могут включаться рекомендации относительно перспектив продолжения данной работы.

В списке использованных источников указываются использованные автором работы научные публикации, а также другие источники, в том числе электронные, по проблемам разработки аналогичных систем, по средствам разработки, по методам решения математических задач. На все перечисленные в списке литературы источники в соответствующих местах работы должны быть сделаны ссылки (номер источника заключается в квадратные скобки).

Список использованных источников должен содержать не менее 10 печатных изданий и любого количества непечатных изданий.

Приложения могут содержать дополнительную информацию: графики, таблицы, тексты программ и т. п.

Оформление курсового проекта.

Курсовой проект представляется на кафедру в полностью готовом виде (сшитом, в переплете или в обложке).

Дополнительно к проекту прилагаются специальные (магнитные или иные) носители информации, содержащие программы (тексты и исполняемые файлы), данные или объемные приложения, включение которых в текст работы является нецелесообразным.

Текст курсовой работы оформляется в принятом для научных работ виде.

На странице располагается 30 строк., в строке 60 знаков, включая пробелы.

Следует соблюдать следующие размеры полей: левое – не менее 30мм, правое – не менее 10 мм, верхнее – не менее 15 мм, нижнее – не менее 20 мм.

Нумерация страниц выполняется арабскими цифрами, начиная с титульного листа. На титульном листе, а также на первой странице оглавления номер не ставится. На следующих страницах номер ставят в правом верхнем углу.

Работа должна быть отпечатана. Рукописные работы не принимаются.

Графический материал.

Графический материал проекта должен состоять из чертежей, выполненных в соответствии с требованиями ЕСКД, ЕСПД и других действующих ГОСТов. Чертежи должны быть выполнены или на фолиях или в мультимедийном исполнении и снабжены штампом. Каждый чертеж должен быть снабжен основной надписью, графы которой должны быть заполнены в соответствии с образцом.

Макеты чертежей перед окончательным оформлением необходимо показать руководителю проекта.

К основным ГОСТам, которые необходимо использовать при оформлении курсового проекта, приведенные в работе относятся:

ГОСТ 34.201-89 Информационная технология. Комплекс стандартов на автоматизированные системы. Виды, комплектность и обозначение документов при создании автоматизированных систем.

ГОСТ 34.602-89 Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы. с. 106-116.

РД 50-682-89 Методические указания. Информационная технология Комплекс стандартов и руководящих документов на автоматизированные системы. Общие положения. с. 157-161.

РД 50-680-88 Методические указания. Автоматизированные системы. Основные положения. с. 152-156.

ГОСТ 34.601-90 Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания. с. 100-105.

ГОСТ 34.401-90 Информационная технология. Комплекс стандартов на автоматизированные системы. Средства технические периферийные автоматизированных систем. Типы и технические требования.

РД 50-34.698-90 Методические указания. Информационная технология. Комплекс стандартов и руководящих документов на автоматизированные системы. Автоматизированные системы. Требования к содержанию документов. с. 127-151.

ГОСТ 34.003-90 Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Термины и определения.

Р50-34.119-90 Рекомендации. Информационная технология. Комплекс стандартов на автоматизированные системы. Архитектура локальных вычислительных сетей в системах промышленной автоматизации. Общие положения.

ГОСТ 24.301-80* Система технической документации на АСУ. Общие требования к выполнению текстовых документов.

ГОСТ 24.302-80 Система технической документации на АСУ. Общие требования к выполнению схем. с. 22-24.

ГОСТ 24.303-80 Система технической документации на АСУ. Обозначения условные графические технических средств. с.25-31.

ГОСТ 24.304-80 Система технической документации на АСУ. Требования к выполнению чертежей с.32-34.

ГОСТ 19.401-78* Единая система программной документации. Текст программы. Требования к содержанию и оформлению.

ГОСТ 19.402-78* Единая система программной документации Описание программы.

Кроме того, графический материал может быть представлен в виде фолий и мультимедийных разработок.

Защита курсового проекта.

Курсовой проект, подписанный исполнителем, представляется на проверку руководителю за 3-4 дня до назначенного срока защиты.

Защита проекта проводится индивидуально каждым студентом на заседании комиссии, состав которой утверждается кафедрой, с обязательным участием руководителя курсового проекта.

Студент, выполнивший курсовой проект, делает доклад (5-7 минут) и отвечает па вопросы комиссии.

В докладе студенту необходимо изложить важнейшие этапы, особенности и результаты работы, не углубляясь в тонкости конкретных технических решений, четко сформулировать конечные результаты.

Вопросы, задаваемые студенту, могут касаться деталей выполненного проекта, а также разделов курсов, по которым выполнялся проект.

Результаты защиты курсового проекта определяются оценками "отлично", "хорошо", "удовлетворительно", "неудовлетворительно"**,** которые проставляются на титульном листе пояснительной записки. При оценке работы учитывается качество выполнения и оформления курсового проекта, уровень защиты проекта и ответов на вопросы, мнение руководителя.

- 8. МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ПРОВЕДЕНИЮ ЛАБОРАТОРНЫХ РАБОТ

Лабораторная работа № 1

Теоретическое введение в предметную область

Цель работы:

Ознакомиться с системой «Служба занятости в рамках вуза»

1. Описание системы

Система состоит из четырех подсистем:

•контроля успеваемости студентов;

•профессиональных и психологических тестов;

•обработки запросов, определения категорий полномочии пользователей;

•экспертных оценок.

1.1. Подсистема контроля успеваемости студентов

Эта подсистема является частью системы «Служба занятости в рамках вуза», которая отвечает за статистическую отчетность по успеваемости с дельного студента, группы пли целого факультета, а также за хранен и правильность ее ввода.

Входными данными подсистемы являются: оценки, даты сдачи экзаменов, имена студентов, номера групп, факультет. На выходе подсистема выдает обработанные данные: средний балл по студенту, группе или фа-

культету, процентное соотношение оценок у студента в группе или на факультете, имена и количество стипендиатов в группе или на факультете. Подсистема «Контроль успеваемости студентов» может функционировать отдельно от всей системы, что дает возможность установить и использоватьее независимо**,** если это необходимо.

Подсистема «Контроль успеваемости студентов» включает следуют функции:

- ввод, вывод и редактирование информации по информационным объектам подсистемы;
- сохранение информации, поступившей от подсистемы контроля успеваемости студентов;
- расчет процентного соотношения оценок у студента в группе и на факультете и вывод его в виде таблиц, графиков и диаграмм;
- расчет среднего балла по студенту, группе или факультету;
- формирование данных по студенту, группе или факультету;
- выявление сильнейших и слабейших студентов в группе или на факультете;
- расчет количества стипендиатов в группе или на факультете;

проверку правильности ввода данных.

1.2. Подсистема профессиональных и психологических тестов. Модульная структура.

1. Ф.И.О., новые тесты, название теста.
2. Ответы на тесты.
3. Вопросы, результаты теста, Ф.И.О.
4. Ответы, Ф.И.О., название теста.
5. Новый тест.
6. Ф.И.О., название теста.
7. Результаты теста, Ф.И.О.
8. Ф.И.О., специальность.
9. Ф.И.О., набор тестов.
10. Ф.И.О., профессионально-психологический тест, психологический тест, специальный тест, ответы.
11. Ф.И.О., информация о пройденных тестах.
12. Ф.И.О., набор тестов для клиента.

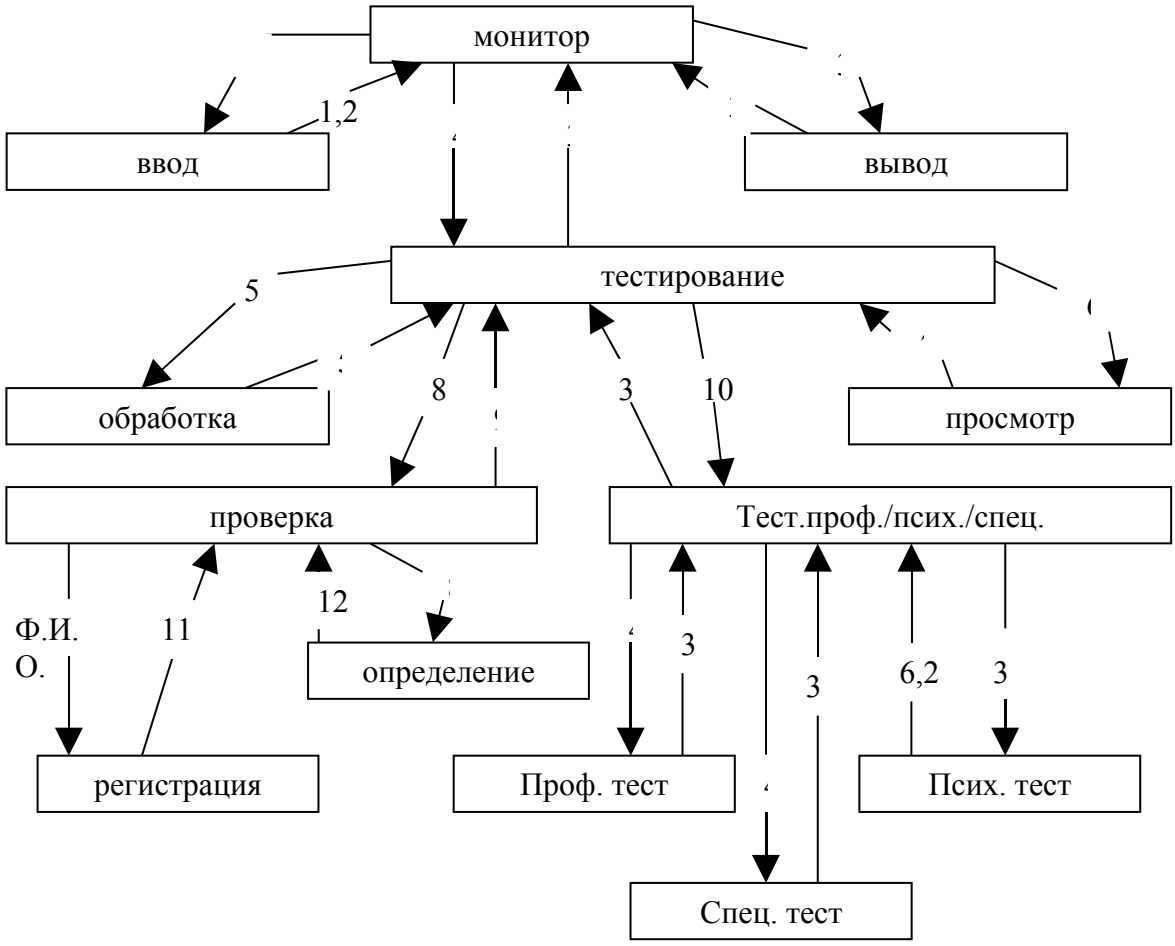


Рис. 1.1. Подсистема профессиональных и психологических тестов.

Внешние сущности

Фирма	Тип производственного объединения предприятия. В данной ситуации фирма выступает как работодатель. Она отправляет запрос на специалиста.
Клиент	Человек, поступивший учиться в институт и пользующийся услугами СЛУЖБЫ ЗАНЯТОСТИ для нахождения работы. Клиент получает запрос на проведение профессиональ-

	ного, психологического и специального тестов и вопросы тестов. Затем КЛИЕНТ отправляет входные данные, ответы на вопросы тестов, запрос на результаты тестов.
Архив	Хранилище, где хранятся все данные о клиенте, начиная с момента пользования услугами СЛУЖБЫ ЗАНЯТОСТИ.
Дополнительные источники	Источники, из которых поступают новые, более современные тесты. Это INTERNET, журналы, какие-либо специализированные центры.

Работа модулей подсистемы

Монитор - вызывает модуль ввода и получает от него данные для выполнения дальнейших задач; обращается к модулю тестирования, передавая ему исходные данные; после выполнения своей задачи тестирование передает ему свои результаты; вызывает модуль вывода и получает от него конечный результат.

Ввод - получает запрос от монитора, передает ему запрашиваемые данные.

Вывод - вызывается монитором и получает от него результаты, выводя их на дисплей.

Тестирование - вызывается монитором, получает от него исходные данные и в зависимости от них обращается к тому или иному модулю. Выполнив свои функции, модули посылают тестированию результаты действий, а тестирование уже передает полученные результаты монитору.

Проверка - получает исходные данные от тестирования и передает их своим подмодулям: регистрация тестируемых, определение тестов. Подмодули передают полученные результаты модулю проверки. Тестирование (профессиональное, психологическое, специальное) получает исходные данные от модуля тестирования и передает их своим подмодулям: профессиональное, психологическое, специальное тестирование. Подмодули предают полученные результаты.

Обработка - получает данные от тестирования, выполняет свои задачи и возвращает полученные результаты модулю тестирования.

Просмотр - получает данные от модуля тестирования, выполняет свои задачи и посылает полученные результаты модулю тестирования.

1.3. Подсистема обработки запросов, определения категорий пользователей

Данная подсистема предназначена для определения категории, полномочий и обработки запросов пользователей службы занятости. В частности, она выполняет следующие функции:

- регистрацию новых фирм;
- регистрацию новых студентов;
- определение прав доступа зарегистрированного пользователя;
- обработку запросов;
- прием регистрационных данных от фирм;
- прием регистрационных данных от студентов;
- прием регистрационных данных от обслуживающего персонала;
- составление резюме;
- запись данных в БД студентов;
- запись данных в БД фирм;
- запись данных в БД зарегистрированных пользователей.

В соответствии с выполняемыми функциями система работает со следующими данными:

- регистрационными данными студентов;
- регистрационными данными фирм;
- личными данными студентов;
- информацией о студентах (получаемой фирмами);
- информацией о фирмах (получаемой студентами);
- идентификационными данными пользователей;
- информацией о системе;
- запросом;
- служебной информацией (для обслуживающего персонала);
- результатом психологического теста;
- результатом профессионального теста;
- экспертными оценками.

Модульная структура.

Определение категории - модуль, определяющий категорию пользователя.

Определение полномочий - модуль, определяющий полномочия пользователя.

Обработка запроса - модуль, предназначенный для обработки запросов пользователя.

Выполнение запроса - модуль, предназначенный для выполнения запросов пользователя.

Запись в БД зарегистрированных пользователей - модуль, предназначенный для работы с базой данных зарегистрированных пользователей.

Запись в БД студентов - модуль, предназначенный для работы- с БД студентов.

Запись в БД фирм - модуль, предназначенный для работы с БД фирм.

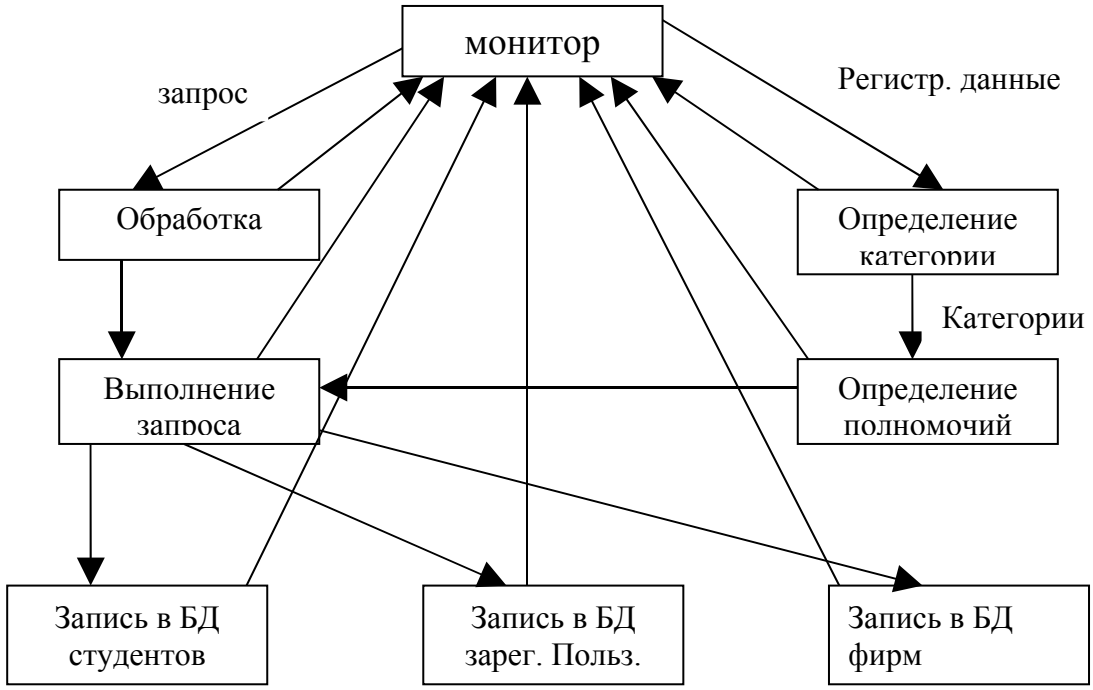


Рис. 1.2. Подсистема обработки запросов, определения категорий пользователей.

1.4. Подсистема экспертных оценок.

Эта подсистема предназначена для установки и просмотра экспертной оценки. Она дает краткую информацию преподавателю о студенте или группе. Студент может с помощью се ориентироваться в учебе.

Кроме того, подсистема может дать представителю фирмы некоторое представление о студенте для рассмотрения его в качестве новых кадров. Подсистема является как информационной средой, так и средой установки экспертной оценки.

Определение статуса клиента

Клиент входит в подсистему с идентификационным номером. По номеру присваивается уровень доступа; Подсистема выдает меню работы, соответствующее уровню доступа.

Обработка отчетного задания

•Эксперт: посылает запрос на предоставление информации о студенте (группе), результатах тестирования. Из общей БД поступает успеваемость студента (группы), результаты тестирования. Делает запрос

на критерии. Из хранилища критериев поступают критерии для установки экспертной оценки. Вводит экспертную оценку. Делает запрос на просмотр экспертной оценки. Из общей БД поступает экспертная оценка.

•Студент: посылает запрос на предоставление информации об успеваемости, результатах тестирования. Из общей БД поступает информация об успеваемости студента, результаты тестирования. Делает запрос на просмотр, экспертной оценки, которая поступает из общей БД.

•Представитель фирмы: делает запрос на просмотр экспертной оценки, которая поступает общей БД.

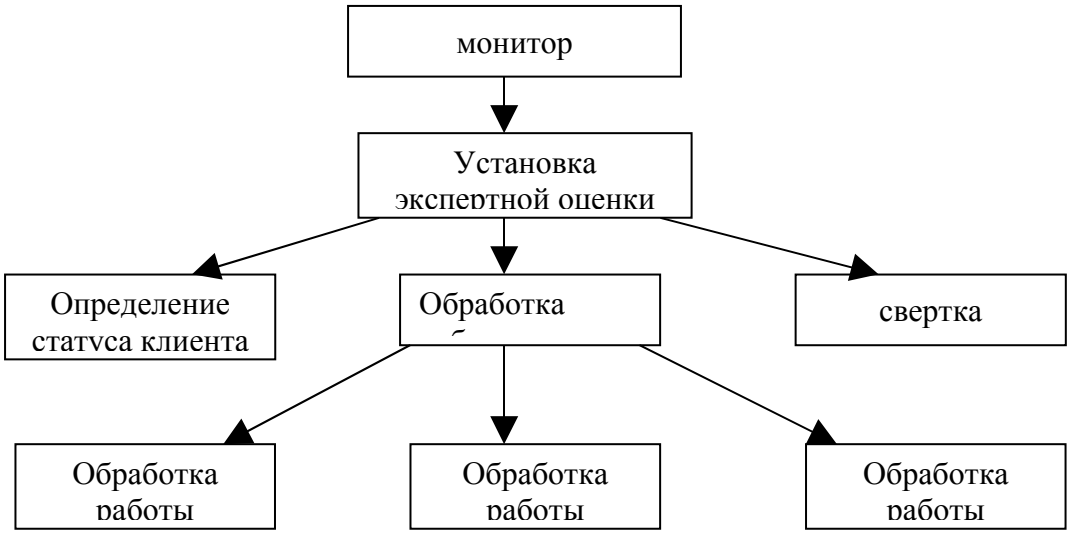


Рис. 1.3. Подсистема экспертных оценок.

Свертка: модуль, получающий из временной БД экспертные оценки, проводит их обработку и передает обработанные данные в общую БД.

Общая база данных содержит в себе:

•экспертные оценки (предоставляются подсистемой экспертных оценок),

•успеваемость студента (предоставляется подсистемой контроля студенческой успеваемости),

•успеваемость группы (предоставляется подсистемой контроля успеваемости),

•результаты профессионального теста (предоставляются подсистемой профессиональных и психологических тестов),

•результаты психологического теста (предоставляются подсистемой профессиональных и психологических тестов).

Временная база данных содержит в себе экспертные оценки (поставленные экспертом).

Критерии экспертных оценок

•По 5-балльной системе оценить студента в начале, середине, конце семестра.

•По 5-балльной системе оценить степень запоминания курса студентом в начале, середине, конце семестра.

•По 5-балльной системе оценить степень применения знаний студентом в начале, середине, конце семестра.

2. Модульная структура системы.

1.Запрос, имя, уровень доступа.
2.Диаграммы, графики, списки, ведомости, количество стипендиатов, отчеты об успеваемости.
3.Имя, личные данные студента, запросы на экспертные оценки.
4.Данные студента.
5.Данные студента, данные фирмы, запросы, оценки, пароль, имя, уровень доступа.
6.Отчеты, резюме, графика, вопросы тестов.
7.Уровень доступа.
8.Имя.
9.Пароль.
10.Полномочия.
11.Резюме.
12.Результаты тестов.
13.Запрос.
14.Найденные записи.
15.Экспертные оценки.

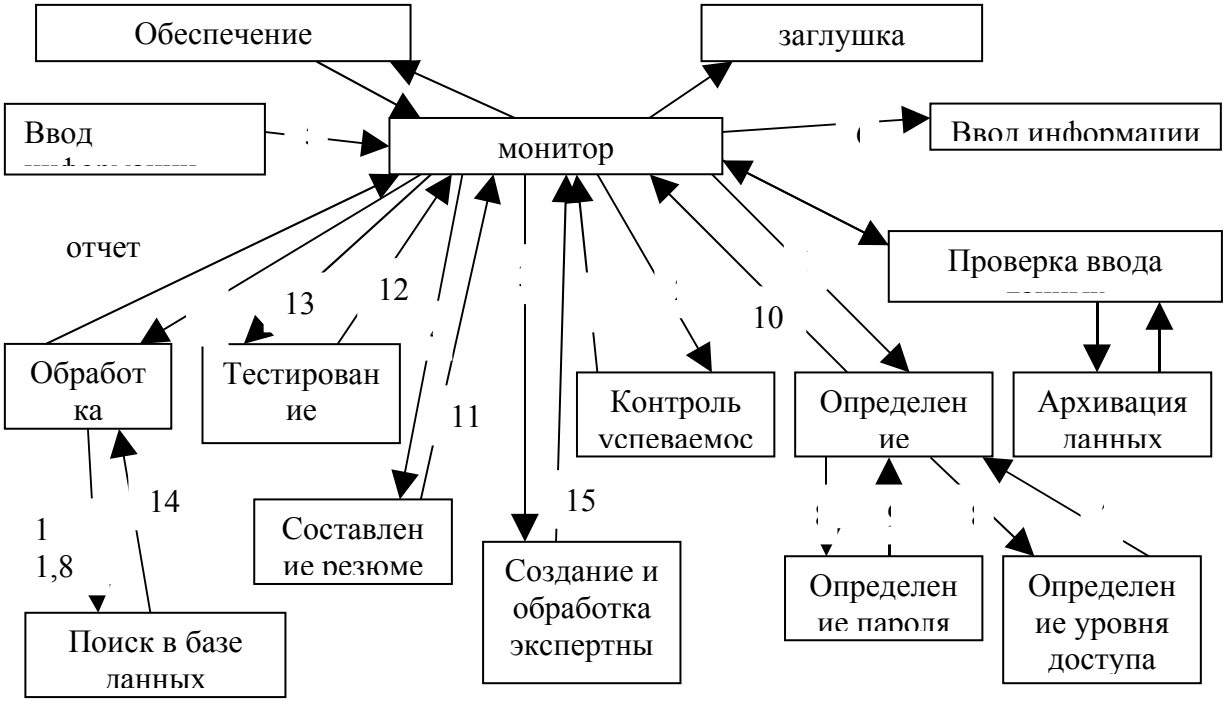


Рис. 1.4. Модульная структура системы.

3. Информационные объекты системы

Под информационным объектом хранения (информационным элементом) понимается логически однородная единица информации, для хранения которой достаточно одной записи таблицы.

Информационные объекты хранения для БД системы:

А. Общего назначения:

1.Факультет.
2.Учебная специальность.
3.Группа студентов.
4.Студент.
5.Преподаватель. -
6.Учебный предмет (дисциплина).
7.Тип отчетного задания (курсовой проект, реферат, контрольная работа, зачет, экзамен и т. д.).
8.Отчетное задание по конкретному изучаемому предмету - описание сути отчетного задания и необходимых требований для его успешного выполнения.
9.Отчетное задание, сданное конкретным студентом, - экземпляр от четного задания.
10.Экспертная оценка успеваемости студента - кумулятивная оценка успеваемости студента по предмету (например, за семестр).
11.Экспертная оценка успеваемости группы студентов - аналогично предыдущему, но по целой группе.
12.Контрольный вопрос для контроля студентов по изучаемым дисциплинам.
13.. Ответ студента на контрольный вопрос (правильный или нет).
14.Правильный ответ контрольного вопроса.
15.Вопрос профессионального теста.
16.Ответ студента на вопрос профессионального теста (правильный или нет).
17.Правильный ответ профессионального теста.
18.Вопрос психологического теста.
19.Вариант ответа на вопрос психологического теста.
20.Правильный вариант ответа.
21.Ответ студента на вопрос психологического теста.
22.Экспертная оценка по результату профессионального теста.
23.Экспертная оценка по результату психологического теста.
24.Почтовое сообщение - сообщение по электронной почте. Сообщение, например, может уведомить студента о принятии его на работу.
25.Объявление.
26.Резюме (характеристика) студента.

27.Архивная запись по группе студентов.
28.Архивная запись по студенту.
29.Архивная запись отчетного задания, сданного студентом.
30.Архивная запись экспертных оценок. Каждая экспертная оценка ответствует отчетному заданию.
31.Архивная запись результатов, профессионального, теста.
32.Архивная запись результатов психологического теста.
33.График успеваемости студента (зависимость средней оценки от временного промежутка)
34.График успеваемости группы студентов.
35.Архивная запись графиков успеваемости группы студентов.

Б. Служебные:
1.Запись об активности соединений.
2.Запись о пользовательской транзакции.
3.Уровень доступа в систему.
4.Учетная запись полномочий пользователя (для организации защиты от несанкционированного доступа).

4. Функциональные характеристики системы.
1. Первоначальный ввод информации в БД.
2. Изменение содержания БД:
- ввод новых данных,
- изменение существующих данных,
- архивация данных.

3. Осуществление поиска в БД по запросу пользователя.
4.Удаленный доступ к системе по протоколу TCP/IP.
5.Обеспечение защиты и безопасности данных, в частности:
•разграничение прав доступа к ресурсам сервера (владелец, группа и т.д.),
•контроль вводимой информации,
•обеспечение целостности БД.
6. Вывод найденной информации.

5. Цели и задачи системы.
• Система будет обеспечивать хранение, выдачу и обновление информации системы дистанционного обучения студентов и системы «Служба занятости в рамках вуза», а именно:
•представлять и получать накопленную информацию по конкретным объектам;
•представлять и получать информацию от подсистемы обработки запросов пользователей;
•представлять и получать информацию от подсистемы контроля студентов по изучаемым на кафедре дисциплинам;
•обеспечивать разграничение доступа к информационным ресурсам

системы;

•обеспечивать мониторинг активных и пассивных пользователей и системных событий;

•обеспечивать пользователей возможностью информационного обмена

•обеспечивать связь между фирмами и службой (институтом);

•обеспечивать регулярное прохождение студентами профессиональных и психологических тестов;

•обеспечивать поиск данных по запросам.

6. Категории пользователей

При работе с системой на стадиях заполнения эксплуатации БД необходимо участие следующих категорий пользователей:

•администратора БД,

•группы экспертов.

Администратор системы осуществляет заполнение БД информацией, подготовленной учебной частью, деканатом или группой экспертов. Внесение изменений в БД системы осуществляется лишь администратором системы под руководством группы экспертов. Преподаватели и студенты являются внешними пользователями, работающими с системой в соответствии с ролями доступа в информационно-поисковом режиме.

Предоставляемые возможности пользователям системы:

студенту:

- ввод личных анкетных данных;
- просмотр экспертных оценок по отчетным заданиям и результатам;
- прохождение психологических и профессиональных .тестов;
- просмотр сводных таблиц и графиков;
- получение и сдача контрольных заданий;
- доступ к справочным материалам (данные из службы удаленного обучения, а именно методическое обеспечение);
- просмотр сообщений и внесение изменений в сообщения доски объявлений;
- поиск вакансии в БД по запросу, эксперту **(**преподавателю**)**;
- предоставление экспертной оценки, а также изменение ее;
- просмотр других оценок;
- просмотр программы курса и внесение изменений в нее;
- ввод контрольных заданий и назначение их студенту;
- контроль ответов на задания;
- доступ к интеллектуальным ресурсам;
- составление резюме (характеристик);
- деканату (декану, зам. декана и т. д.);
- просмотр программы курса;
- просмотр динамики успеваемости курса, группы, отдельного студента;
- просмотр сводных таблиц и графиков;

- просмотр экспертных оценок и характеристик преподавателей;

администратору:

- определение прав доступа;
- ввод и корректировка системных данных;
- контроль работы системы;
- осуществление контроля защиты системы от несанкционированного доступа;
- изменение физической модели данных системы;

оператору:

- составление сводных таблиц и графиков;
- заполнение нолей БД системы (ввод информации);

фирме:

- поиск в БД данных о студенте по запросу,
- просмотр резюме студентов.

Лабораторная работа № 2

Методология IDEFO

Цель работы:

•изучение основных принципов методологии EDEF. 0,-,
•создание нового проекта в BPWin,
•формирование контекстной диаграммы,
•проведение связей.

Описание системы с помощью IDEF0 называется функциональной моделью. Функциональная модель предназначена для описания существующих бизнес-процессов, в котором используются как естественный, так и графический языки. Для передачи информации о конкретной системе источником графического языка является сама методология IDEF0.

Методология IDEF0 предписывает построение иерархической системы диаграмм – единичных описаний фрагментов системы. Сначала проводится описание системы в целом и ее взаимодействия с окружающим, миром .(контекстная диаграмма), после чего проводится функциональная декомпозиция - система разбивается на подсистемы и каждая подсистема описывается отдельно (диаграммы декомпозиции). Затем каждая. подсистема 'разбивается на более мелкие и так далее до достижения нужной степени подробности.

Каждая IDEF0-диаграмма содержит блоки и дуги. Блоки изображают функции моделируемой системы. Дуги связывают блоки вместе и отображают взаимодействия и взаимосвязи между ними.

Функциональные блоки (работы) на диаграммах изображаются прямоугольниками, означающими поименованные процессы, функции или задали, ко-

торые происходят в течение определенного времени и имеют распознаваемые результаты. Имя работы должно быть выражено отглагольным существительным, обозначающим действие.

IDEF0 требует, чтобы в диаграмме было не менее трех и не более шести блоков. Эти ограничения поддерживают сложность диаграмм и модели *I* на уровне, доступном для чтения, понимания и использования. Каждая сторона блока имеет особое, вполне определенное назначение. Левая сторона блока предназначена для входов, верхняя для управления, управление ограничивает или предписывает условия выполнения преобразований, механизмы показывают, что и как выполняет функция.

Блоки в IDEF0 размещаются по степени важности, как ее понимает автор диаграммы. Этот относительный порядок называется доминированием. Доминирование понимается как влияние, которое один блок оказывает на другие блоки диаграммы. Например, самым доминирующим блоком диаграммы может быть либо первый из требуемой последовательности функций, либо планирующая или контролирующая функция, влияющая на все другие.

Наиболее доминирующий блок обычно размещается в верхнем левом углу диаграммы, а наименее доминирующий - в правом углу.

Расположение блоков на странице отражает авторское определение доминирования. Таким образом, топология диаграммы показывает, какие функции оказывают большее влияние на остальные. Чтобы подчеркнуть это, аналитик может перенумеровать блоки в соответствии с порядком их доминирования Порядок доминирования может обозначаться цифрой, размещенной в правом нижнем углу каждого прямоугольника: 1 - будет указывать на наибольшее доминирование, 2 - на следующее и т. д.

Взаимодействие работ с внешним миром и между собой описывается в виде стрелок, изображаемых одинарными линиями со стрелками на концах. Стрелки представляют собой некую информацию и именуются существительными.

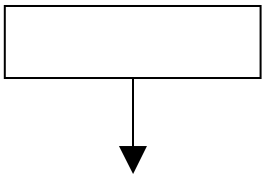
В IDEF0 различают пять типов стрелок.

Вход - объекты, используемые и преобразуемые работой для получения результата (выхода). Допускается, что работа может не иметь ни одной стрелки входа. Стрелка входа рисуется как входящая в левую грань работы.

Управление - информация, управляющая действиями работы, Обычно управляющие стрелки несут информацию, которая указывает, что должна выполнять работа. Каждая работа должна иметь хотя бы одну стрелку управления, которая изображается как входящая в верхнюю грань работы. Выход - объекты, в которые преобразуются входы. Каждая работа должна иметь хотя бы одну стрелку выхода, которая рисуется как исходящая из правой грани работы.

Механизм - ресурсы, выполняющие работу. Стрелка механизма рисуется как входящая в нижнюю грань работы. По усмотрению аналитика стрелки механизма могут не изображаться на модели.

Вызов - специальная стрелка, указывающая на другую модель работы.



В методологии IDEF0 требуется только пять типов взаимодействий между блоками для описании их отношений: управление, вход, обратная связь по управлению, обратная связь по входу, выход механизм. Связи по управлению и входу являются простейшими, поскольку они отражают прямые прямые воздействии, которые интуитивно понятны и очень просты.

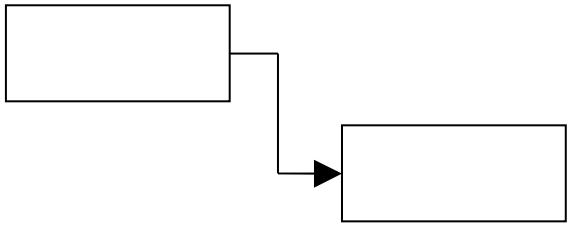


Рис.2.2. Связь по входу

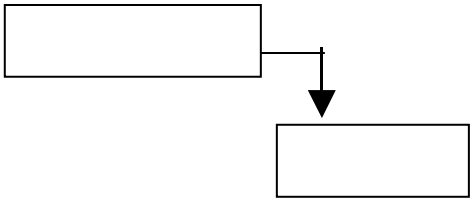


Рис. 2.3. Связь по управлению.

Отношение управления возникает тогда, когда выход одного блока непосредственно влияет на блок с меньшим доминированием.

Обратная связь по управлению и обратная связь по входу являются более сложными, поскольку представляют собой итерацию или рекурсию. А именно выходы из одной работы влияют на будущее выполнение других работ, что впоследствии повлияет на исходную работу.

Обратная связь по управлению возникает тогда, когда выход некоторого блока влияет на блок с большим формированием.

Связи с «выход-механизм» встречаются нечасто. Они отражают ситуацию, при которой выход одной функции является средством достижения для другой.

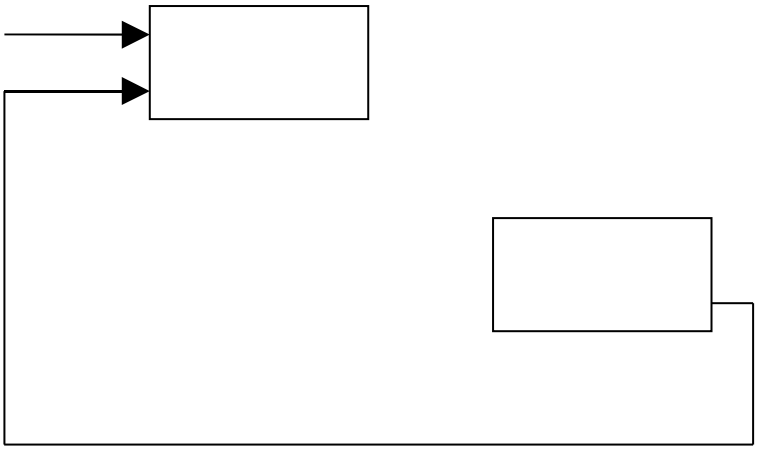


Рис . 2.4. Обратная связь по входу.

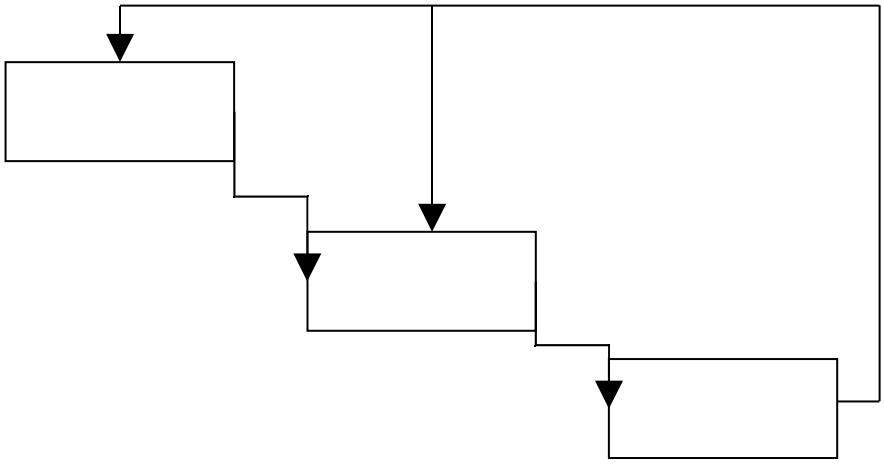


Рис. 2.5. Обратная связь по управлению.

Связи «выход-механизм» характерны при распределении источников ресурсов (например, требуемые инструменты, обученный персонал, физическое пространство, оборудование, финансирование, материалы).

В IDEF0 дуга редко изображает один объект. Обычно она символизирует набор объектов. Так как дуги представляют наборы объектов, они могут иметь множество начальных точек (источников) и конечных точек (назначений). Поэтому дуги могут разветвляться и соединяться различными способами. Вся дуга или ее часть может, выходить из одного или нескольких блоков и заканчиваться в одном или нескольких блоках.

Разветвление дуг, изображаемое в виде расходящихся линий, означает, что все содержимое дуг или его часть может появиться в каждом ответвлении. Дуга всегда помечается до разветвления, чтобы дать название всему

набору. Кроме того, каждая ветвь дуги может быть помечена или не помечена в соответствии со следующими правилами:

•непомеченные ветви содержат все объекты, указанные в метке дуги перед разветвлением;

•ветви, помеченные после точки разветвления, содержат вес объекты или их часть, указанные в метке дуги перед разветвлением.

Слияния дуг в IDEF0, изображаемое как сходящиеся вместе линии, указывает, что содержимое каждой ветви идет на формирование метки для дуги, являющейся результатом слияния исходных дуг. После слияния результирующая дуга всегда помечается для указания нового набора объектов, возникшего после объединения. Кроме того, каждая ветвь перед слиянием может, помечаться или не помечаться в соответствии со следующими правилами:

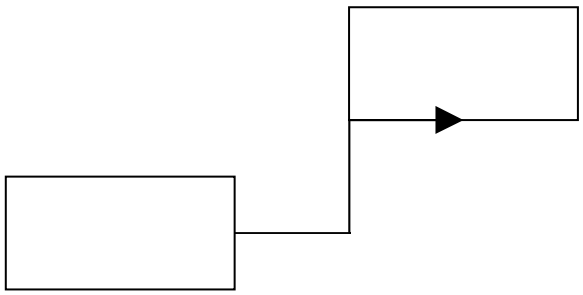


Рис. 2.6. Связь выход-механизм.

•непомеченные ветви содержат все объекты, указанные в общей метке дуги после слияния;

•помеченные перед слиянием ветви содержат все или некоторые объекты из перечисленных в общей метке после слияния.

1. Количественный анализ диаграмм.

Для приведении количественном, анализа диаграмм перечислим показатели модели:

- количество блоков, на диаграмме – *N;*
- уровень декомпозиции диаграммы *L;*
- сбалансированность диаграммы *B;*
- число стрелок, соединяющихся с блоком *A*.

Данный набор факторов относится к каждой диаграмме модели. Далее будут перечислены рекомендации по желательным значениям факторов диаграммы.

Необходимо стремимся к тому, чтобы количество блоков на родительских диаграммах нижних уровней было бы ниже количества на родительских диаграммах, т. е. с увеличением уровня декомпозиции убывал бы коэффициент $\frac{N}{L}$. Таким образом, убывание этого коэффициента говорит о том, что по мере декомпозиции модели функции должны упрощаться, следовательно, количество блоков должно убывать.

Диаграммы должны быть сбалансированы. Это означает, что в рамках одной диаграммы не должно проходить ситуации, изображенной на рис.2.7 у работы 1 входящих стрелок и стрелок управления значительно больше чем выходящих. Следует отметить, что данная рекомендация может не выполняться в моделях описывающих производственные процессы. Например, при описании процедуры в блок может входить множество стрелок, описывающих компоненты изделия, а может выходить одна стрелка готовое изделие.

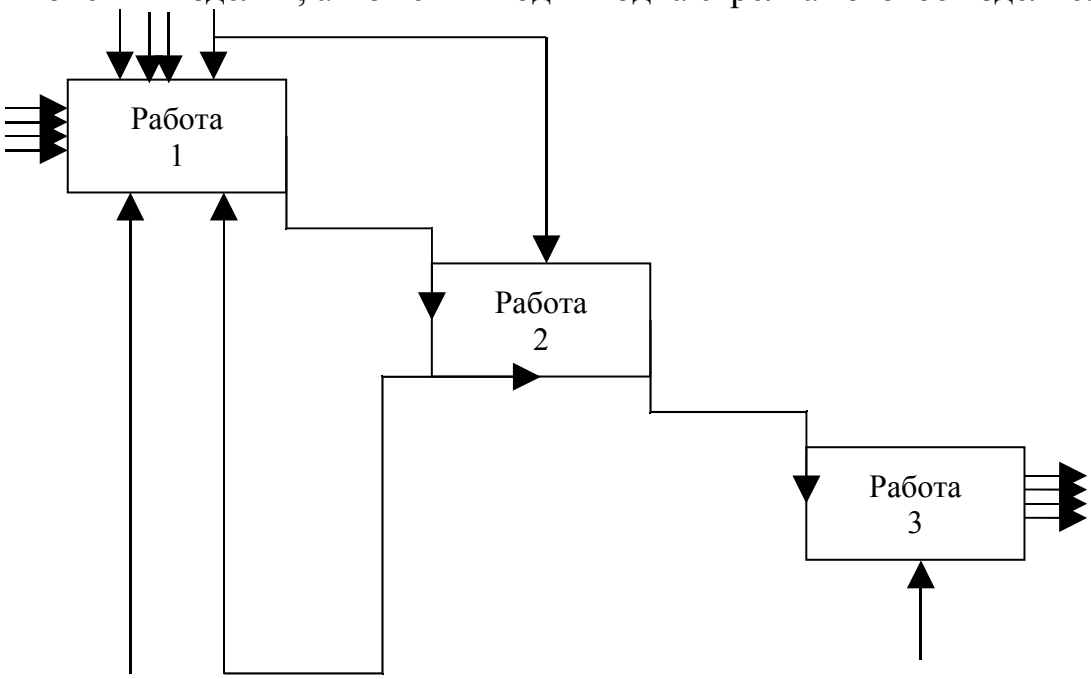


Рис. 2.7. Пример несбалансированной диаграммы

Введем коэффициент сбалансированности диаграммы:

$$K_b = \left| \frac{\sum_{i=1}^{N} A_i}{N} - \max_{i=1}^{N}(A_i) \right|.$$

Необходимо стремиться, чтобы, был минимален для диаграммы. Помимо анализа графических элементов диаграммы необходимо рассматривать наименования блоков. Для оценки имен составляется словарь элементарных (тривиальных) функций моделируемой системы. Фактически в данный словарь должны попасть функции нижнего, уровня декомпозиции диаграмм. Например, для модели БД элементарными могут являться функции «найти запись», «добавить запись в БД», в то время как функция «регистрация пользователя» требует дальнейшего описания.

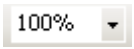
После формирования словаря и составления пакета диаграмм системы необходимо рассмотреть нижний уровень модели. Если на нем обнаружатся совпадения названий блоков диаграмм и слов из словаря, то это говорит, что достаточный уровень декомпозиции достигнут. Коэффициент, количественно отражающий данный критерий, можно записать как *L*C* -произведение уровня модели на число совпадений имен блоков со словами из словаря. Чем ниже уровень модели (больше *L),* тем ценнее совпадения.

2. Инструментарий BPWin

При запуске BPWin по умолчанию появляется основная панель инструментов, палитра инструментом и Model Explorer.

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

Таблица 2.1 Описание элементов управления панели инструментов BPWin 2.5

Элемент управления	Описание	Соответствующий пункт в меню
	Создать новую модель	File>New
	Открыть модель	File> Open
	Сохранить модель	File>Save
	Напечатать модель	File>Print
	Выбор масштаба	View>Zoom
	Проверка правописания	Tools>Spelling

При создании новой модели возникает диалог, в котором следует указать, будет ли создана моде заново или она будет открыта из репозитория ModelMart, внести имя модели и выбрать методологию, в которой будет построена модель (рис. 2.8).

BPW in поддерживает три методологии: IDEFO, IDF3 и DFD. В BRWin возможно построение смешанных моделей, т.е., модель может содержать одновременно как диаграммы IDFO, так и IDEF3 и DFD. Состав палитры инструментов изменяется автоматически, когда происходит переключение из одной нотации в другую.

Модель в BPWin рассматривается как совокупность работ, каждая из которых оперирует с некоторым набором данных. Если щелкнуть по любому объекту модели левой кнопкой мыши появиться в всплывающее меню каждый пункт которого соответствует редактору какого-либо свойства объекта.

Лабораторная работа № 3

Создание логической модели

Цель работы:

- ознакомиться с технологией построения логической модели в. ERWin,
- изучить методы определения ключевых атрибутов сущностей**,**
- освоить метод проверки адекватности логической модели,
- изучить типы связей между сущностями.

Первым шагом при создании логической модели БД является построение диаграммы ERD (Entity Relationship Diagram). ERD-диаграммы состоят из трех частей: сущностей, атрибутов и взаимосвязей. Сущностями являются существительные, атрибуты - прилагательными или модификаторами, взаимосвязи - глаголами.

ERD-диаграмма позволяет рассмотреть систему целиком и выяснить требования, необходимые для ее разработки, касающиеся хранения информации.

ERD-диаграммы можно подразделить на отдельные куски, соответствующие отдельным задачам, решаемым проектируемой системой. Это позволяет рассматривать систему с точки зрения функциональных возможностей делая процесс проектирования управляемым.

1. ERD-диаграммы

Как известно основным компонентом реляционных БД является таблица. Таблица используется для структуризации и хранения информации. В реляционных БД каждая ячейка таблицы содержит одно значение. Кроме того, внутри одной БД существуют взаимосвязи между таблицами, каждая из которых задает совместное пользование данными таблицы.

ERD-диаграмма графически представляет структуру данных проектируемой информационной системы. Сущности отображаются при помощи прямоугольников, содержащих имя. Имена принято выражать существительными в единственном числе, взаимосвязи - при помощи линий, соединяющих отдельные сущности. Взаимосвязь показывает, что данные одной сущности ссылаются или связаны с данными другой.

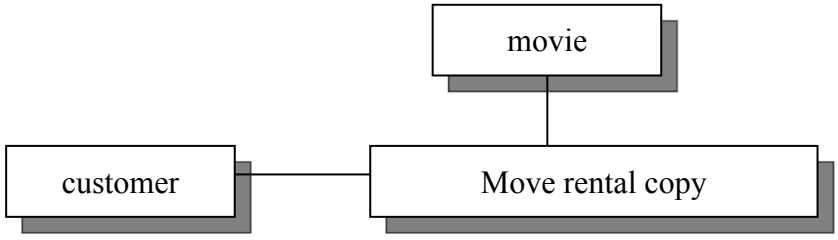


Рис. 6.1. Пример ERD-диаграммы.

1.1. Определение сущностей и атрибутов

Сущность - это субъект, место, вещь, событие или понятие, содержащие информацию. Точнее, сущность - это набор (объединение) объектов, называемых экземплярами; В приведенном на рис. 6.1 примере сущность CUSTOMER (клиент) представляет всех возможных клиентов. Каждый экземпляр сущности обладает набором характеристик. Так, каждый клиент может иметь имя, адрес, телефон и т. д. В логической модели все эти характеристики называются атрибутами сущности.

1.2. Логические взаимосвязи

Логические взаимосвязи представляют собой связи между сущностями. Они определяются глаголами, показывающими, как одна сущность относится к другой.

Некоторые примеры взаимосвязей:

- команда включает много игроков,
- самолет перевозит много пассажиров,
- продавец продает много продуктов.

Во всех этих случаях взаимосвязи отражают взаимодействие между двумя сущностями, называемое «один-ко-многим». Это означает, что один экземпляр первой сущности взаимодействует с несколькими экземплярами другой сущности. Взаимосвязи отображаются линиями, соединяющими две сущности с точкой на одном конце и глаголом, располагаемым над линией. Кроме взаимосвязи «один-ко-многим» существует еще один тип - это : «многие-ко-многим». Этот тип связи описывает ситуацию, при которой экземпляры сущностей могут взаимодействовать с несколькими экземплярами других сущностей. Связь «многие-ко-многим» используют на первоначальных стадиях проектирования. Этот тип взаимосвязи отображается сплошной линией с точками на обоих концах.

Связь «многие-ко-многим» может не учитывать определенные ограничения системы, поэтому может быть заменена на «один-ко-многим» при последующем пересмотре проекта.

1.3. Проверка адекватности логической модели

Если взаимосвязи между сущностями были правильно установлены, то можно составить предложения, их описывающие. Например, по модели, показанной на рис. 3.3, можно составить следующие предложения:

Самолет перевозит пассажиров.

Много пассажиров перевозятся одним самолетом.

Составление таких предложений позволяет проверить соответствие порученной модели требованиям и ограничениям создаваемой системы.

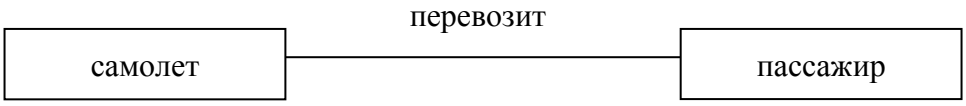


Рис. 6.3. Пример логической модели со взаимосвязью.

2. Модель данных, основанная на ключах

Каждая сущность содержит горизонтальную линию, разделяющую атр! буты на две группы. Атрибуты, расположенные над линией, называются первичным ключом. Первичный ключ предназначен для уникальной идентификации экземпляра сущности.

2.1. Выбор первичного ключа

При создании сущности необходимо выделить группу атрибутов, которые потенциально могут стать первичным ключом (потенциальные ключи*);* затем произвести отбор атрибутов для включения в состав первичного ключа, следуя следующим рекомендациям:

• Первичный ключ должен быть подобран таким образом, чтобы по значениям атрибутов, в него включенных, можно было точно идентифицировать экземпляр сущности.

• Никакой из атрибутов первичного ключа не должен иметь нулевое значение.

• Значения атрибутов первичного ключа не должны меняться. Если значение изменилось, значит, это уже другой экземпляр сущности.

При выборе первичного ключа можно внести в сущность дополнительный атрибут и сделать его ключом. Так, для определения первичного ключа часто используют уникальные номера, которые могут автоматически генерироваться системой при добавлении экземпляра сущности в БД. Применение уникальных номеров облегчает процесс индексации и поиска в БД.

Первичный ключ, выбранный при создании логической модели, может быть неудачным для осуществления эффективного доступа к БД и должен быть изменен при проектировании физической модели.

Потенциальный ключ, не ставший первичным, называется альтернативным ключом (Alternate Key). ERWin позволяет выделять атрибуты альтернативных ключей, и по умолчанию в дальнейшем при генерации схемы БД по этим атрибутам будет генерироваться уникальный индекс. При создании альтернативного ключа на диаграмме рядом с атрибутом появляются символы (АК).

Атрибуты, участвующие в неуникальных индексах, называются инверсионными входами (Inversion Entries). Инверсионные входы – это атрибут или группа атрибутов, которые не определяют экземпляр уникальным об-

разом, но часто используются для обращения к экземплярам сущности. ERWin генерирует неуникальный индекс для каждого инверсионного входа. При проведении связи между двумя сущностями в дочерней сущности автоматически образуются внешние ключи (foreign key). Связь образует ссылку на атрибуты первичного ключа в дочерней сущности, и эти атрибуты образуют внешний ключ в дочерней сущности. Атрибуты внешнего ключа обозначаются символами (FK) после своего имени.

3. Пример

Рассмотрим процесс построения логической модели на примере БД студентов, системы «Служба занятости в рамках вуза». Первым этапом является определение сущностей и атрибутов. В БД будут храниться записи о студентах, следовательно, сущностью будет: студент.

Атрибут	**Описание**
Номер	Уникальный номер для идентификации пользователя
Ф.И.О.	Фамилия, имя и отчество пользователя
Пароль	Пароль для доступа в систему
Возраст	Возраст студента
Пол	Пол студента
Характеристика	Memo-поле с общей характеристикой пользователя
E-mail	Адреса электронной почты
Телефон	Номера телефонов студента (домашний, рабочий)
Опыт работы	Специальности и опыт работы студента по каждой из них
Специальность	Специальность, получаемая студентом при окончании учебного заведения
Специализация	Направление специальности; по которому обучается студент
Иностранный язык	Список иностранных языков и уровень владения ими
Тестирование	Список тестов и отметки о их прохождении
Экспертная оценка	Список предметов с экспертными оценками по каждому из них
Оценки по экзаменам	Список сданных предметов с оценками

В полученном списке существуют атрибуты, которые нельзя определить в виде одного поля БД. Такие атрибуты требуют дополнительных Определений и

должны рассматриваться как сущности, состоящие, в свою очередь, из атрибутов. К таковым относятся: опыт работы, иностранный язык, тестирование, экспертная оценка, оценки по экзаменам. Определим их атрибуты.

Таблица 3.2. Атрибуты сущности «Опыт работы»

Атрибут	Описание
Специальность	Название специальности, по которой у студента есть опыт работы
Атрибут	Описание
Опыт	Опыт работы по данной специальности в годах
Место работы.	Наименование предприятия, где приобретался опыт

Таблица 3.3. *Атрибуты сущности «Иностранный язык»*

Атрибут	Описание
Язык	Название иностранного языка, которым владеет студент
Уровень владения	Численная оценка уровня владения иностранным языком

Таблица 3.4. *Атрибуты, сущности «Тестирование»*

Атрибут	**Описание**
Название	Название теста, который прошел студент
Описание	Содержит краткое описание теста
Оценка	Оценка, которую получил студент в результате прохождения теста

Таблица 3.5. *Атрибуты сущности « Экспертная оценка»*

Атрибут	Описание
Дисциплина	Наименование дисциплины, по которой оценивался студент
Ф.И.О. преподавателя	Ф.И.О. преподавателя, который оценивал студента
Оценка	Экспертная оценка преподавателя

Таблица 3.6. *Атрибуты сущности «Оценки по экзаменам»*

Атрибут	Описание
Предмет	Название предмета, по которому сдавался экзамен.
Оценка	Полученная оценка.

Составим ERD-диаграмму, определяя типы атрибутов и проставляя связи между сущностями (рис. 3.4). Все сущности будут зависимыми от сущности «Студент». Связи будут типа «один-ко-многим».

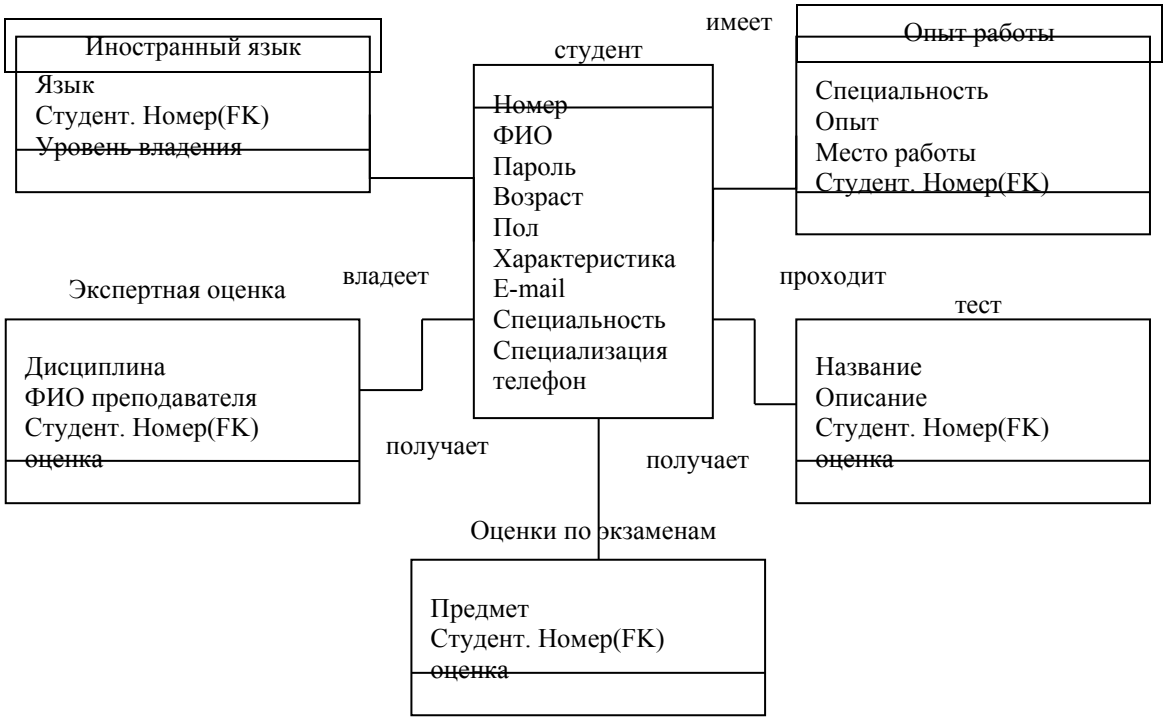


Рис. 3.4. ERD-диаграмма БД студентов

На полученной диаграмме рядом со связью отражается ее имя, показывающее соотношение между сущностями. При проведении связи между сущностями первичный ключ мигрирует в дочернюю сущность. Следующим этапом при построении логической модели является определение *ключевых* атрибутов и типов атрибутов.

Таблица 3.7. Типы атрибутов

Атрибут	Тип
Номер	Number
Ф.И.О.	String
Пароль	String
Возраст	Number

Атрибут	Тип
Пол	String
Характеристика	String
E-mail	**String**
Специальность	**String**
Специализация	**String**
Опыт	**Number**
Место работы	**String**
Язык	**String**
Уровень владения	**Number**
Название	**String**
Описание	**String**
Оценка	**Number**
Дисциплина	**String**
Ф.И.О. преподавателя	**String**
Предмет	**String**

Выберем для каждой сущности ключевые атрибуты, однозначно определяющие сущность. Для сущности «Студент» это будет уникальный номер, для сущности «Опыт работы»'все поля являются ключевыми, так как по разным специальностям студент может иметь разный опыт работы в разных фирмах. Сущность «Тест» определяется названием, так как студент по одному тесту может иметь только одну оценку. Оценка по экзамену определяется только названием предмета, экспертная оценка зависит от преподавателя, который ее составил, поэтому в качестве ключевых атрибутов выберем «Дисциплину» и «Ф.И.О. преподавателя». У сущности «Иностранный язык» уровень владения зависит только от наименования языка, следовательно, это и будет являться ключевым атрибутом.

Получим новую диаграмму, изображенную на рис. 3.5, где все ключевые атрибуты будут находиться над горизонтальной чертой внутри рамки, изображающей сущность.

Рис. 3.5. ERD-диаграмма БД студентов с ключевыми атрибутами.

9. ПЕРЕЧЕНЬ ПРОГРАММНЫХ ПРОДУКТОВ, ИСПОЛЬЗУЕМЫХ В ПРАКТИКЕ ВЫПУСКНИКОВ И УЧЕБНО-МЕТОДИЧЕСКОЕ ПОСОБИЕ

Программные продукты – BRWin, ERWin.

Учебно-методическое пособие: Черемных С.В., Семенов И.О., Ручкин В.С. Моделирование и анализ систем. IDEF – технологии: практикум. – М.: Финансы и статистика, 2002. -192 с.

10. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ПРИМЕНЕНИЮ СОВРЕМЕННЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ ДЛЯ ПРЕПОДАВАНИЯ УЧЕБНОЙ ДИСЦИПЛИНЫ.

Для проведения анализа и реорганизации бизнес-процессов Logic Works предлагает CASE - средство верхнего уровня - BPwin, поддерживающий методологии IDEF0 (функциональная модель), IDEF3 (WorkFlow Diagram) и DFD (DataFlow Diagram). Функциональная модель предназначена для описания существующих бизнес-процессов на предприятии (так называемая модель AS-IS) и идеального положения вещей - того, к чему нужно стремиться (модель TO-BE). Методология IDEF0 предписывает построение иерархической системы диаграмм - единичных описаний фрагментов системы. Сначала проводится описание системы в целом и ее взаимодействия с окружающим миром (контекстная диаграмма), после чего проводится функ-

циональная декомпозиция - система разбивается на подсистемы и каждая подсистема описывается отдельно (диаграммы декомпозиции). Затем каждая подсистема разбивается на более мелкие и так далее до достижения нужной степени подробности. После каждого сеанса декомпозиции проводится сеанс экспертизы, каждая диаграмма проверяется экспертами предметной области, представителями заказчика, людьми, непосредственно участвующими в бизнес - процессе. Такая технология создания модели позволяет построить модель адекватную предметной области на всех уровнях абстрагирования. Если в процессе моделирования нужно осветить специфические стороны технологии предприятия, BPwin позволяет переключиться на любой ветви модели на нотацию IDEF3 или DFD и создать смешанную модель. Нотация DFD включает такие понятия как внешняя ссылка и хранилище данных, что делает ее более удобной (по сравнению с IDEF0) для моделирования документооборота. Методология IDEF3 включает элемент "перекресток", что позволяет описать логику взаимодействия компонентов системы.На основе модели BPwin'a можно построить модель данных.

Для построения модели данных Logic Works предлагает мощный и удобный инструмент - ERwin. Хотя процесс преобразования модели BPwin в модель данных плохо формализуется и поэтому полностью не автоматизирован, Logic Works предлагает удобный инструмент для облегчения построения модели данных на основе функциональной модели - механизм двунаправленной связи BPwin - ERwin. ERwin имеет два уровня представления модели - логический и физический. На логическом уровне данные представляются безотносительно конкретной СУБД, поэтому могут быть наглядно представлены даже для неспециалистов. Физический уровень данных - это, по - существу, отображение системного каталога, который зависит от конкретной реализации СУБД. ERwin позволяет проводить процессы прямого и обратного проектирования БД. Это означает, что по модели данных можно сгенерировать схему БД или автоматически создать модель данных на основе информации системного каталога. Кроме того, ERwin позволяет выравнивать модель и содержимое системного каталога после редактирования того, либо другого. ERwin интегрируется с популярными средствами разработки клиентской части - PowerBuilder, SQLWindows, Visual Basic, Delphi, что позволяет автоматически генерировать код приложения, который готов к компиляции и выполнению.

Создание современных информационных систем, основанных на широком использовании распределенных вычислений, объединении традиционных и новейших информационных технологий, требует тесного взаимодействия всех участников проекта: менеджеров, бизнес и системных аналитиков, администраторов баз данных, разработчиков. Для этого использующиеся на разных этапах и разными специалистами средства моделирования и разработки должны быть объединены общей системой организации совместной работы. Фирма Logic Works разработала систему Model Mart - хранилище моделей, к которому открыт доступ для участников проекта создания информа-

ционной системы. Model Mart удовлетворяет всем требованиям, предъявляемым к средствам разработки крупных информационных систем, а именно.

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

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

Управление доступом. Для каждого участника проекта определяются права доступа, в соответствии с которыми они получают возможность работать только с определенными моделями. Права доступа могут быть определены как для групп, так и для отдельных участников проекта. Роль специалистов, участвующих в различных проектах может меняться, поэтому в Model Mart можно определять и управлять правами доступа участников проекта к библиотекам, моделям и даже к специфическим областям модели.

Архитектура Model Mart. Model Mart реализована на архитектуре клиент - сервер. В качестве платформы реализации хранилища выбраны РСУБД Sybase, Microsoft SQL Server и Oracle. Клиентскими приложениями являются ERwin 3.x и BPwin 2.0x. В следующих версиях Model Mart предполагается открыть доступ к хранилищу моделей через API, что позволит постоянно наращивать возможности интегрированной среды путем включения новых инструментов моделирования и анализа.

Как было указано выше, при разработке крупных проектов критичным становится время реализации проекта. Одним из решений проблемы может стать автоматическая генерация кода приложения (клиентской части) CASE - средствами на основе модели предметной области. Хотя ERwin решает эту задачу, код генерируется на основе модели IDEF1X, то есть фактически на основе реляционной модели данных, которая непосредственно не содержит информацию о бизнес - процессах. Как следствие этого, сгенерированный код не может полностью обеспечить функциональность приложения со слож-

ной бизнес-логикой. Существует альтернативная технология кодогенерации, которая лишена этого недостатка - объектно-ориентированное проектирование, реализованное в Rational Rose (Rational Software). Rational Rose - позволяющее строить объектные модели в различных нотациях (OMT, UML, Буч) и генерировать на основе получеенной модели приложения на языках программирования C++, Visual Basic, Power Builder, Java, Ada, Smalltalk и др. Поскольку генерация кода реализована на основе знаний предметной области, а не на основе реляционной структуры данных, полученный код более полно отражает бизнес-логику. Rational Rose поддерживает не только прямую генерацию кода, но и обратное проектирование, то есть создание объектной модели по исходному коду приложения.

Создание модели процессов в BPwin (IDEF0)

На начальных этапах создания ИС необходимо понять, как работает организация, которую мы собираемся автоматизировать. Никто в организации не знает, как она работает в той мере подробности, которая необходима для создания ИС. Руководитель хорошо знает работу в целом, но не в состоянии вникнуть в детали работы каждого рядового сотрудника. Рядовой сотрудник хорошо знает, что творится на его рабочем месте, но плохо знает, как работают коллеги. Поэтому для описания работы предприятия необходимо построить модель. BPwin как раз и предназначен для построения такой модели - функциональной модели (или модели процессов).

Обычно сначала строится модель существующей организации работы -"AS-IS" (как есть). Анализ функциональной модели позволяет понять, где находятся наиболее слабые места, в чем будут состоять преимущества новых бизнес-процессов и насколько глубоким изменениям подвергнется существующая структура организации бизнеса. Найденные в модели "AS-IS" недостатки можно исправить при создании модели "TO-BE" (как должно быть) - модели новой организации бизнес-процессов.

Наиболее удобным языком моделирования бизнес- процессов является IDEF0, предложенный более 20 лет назад Дугласом Россом (Ранее назывался SADT - Structured Analysis and Design Technique).

Под моделью в IDEF0 понимают описание системы (текстовое и графическое), которое должно дать ответ на некоторые заранее определенные вопросы. Процесс моделирования какой -либо системы в IDEF0 начинается с определения контекста, т.е. наиболее абстрактного уровня описания системы в целом. В контекст входит определение субъекта моделирования, цели и точки зрения на модель. Под субъектом понимается сама система, при этом необходимо точно установить, что входит в систему, а что лежит за ее пределами, другими словами, мы должны определить, что мы будем в дальнейшем рассматривать как компоненты системы, а что как внешнее воздействие. На определение субъекта системы будет существенно влиять позиция, с которой рассматривается система, и цель моделирования - вопросы, на которые построенная модель должна дать ответ. IDEF0-модель предполагает наличие

четко сформулированной цели, единственного субъекта моделирования и одной точки зрения.

Основу методологии IDEF0 составляет графический язык описания бизнес- процессов. Модель в нотации IDEF0 представляет собой совокупность иерархически упорядоченных и взаимосвязанных диаграмм. Вершина этой древовидной структуры, представляющая собой самое общее описание системы и ее взаимодействия с внешней средой, называется контекстной диаграммой. После описания системы в целом проводится разбиение ее на крупные фрагменты. Этот процесс называется функциональной декомпозицией, а диаграммы, которые описывают каждый фрагмент и взаимодействие фрагментов, называются диаграммами декомпозиции. После декомпозиции контекстной диаграммы проводится декомпозиция каждого большого фрагмента системы на более мелкие и так далее до достижения нужного уровня подробности описания. После каждого сеанса декомпозиции проводятся сеансы экспертизы - эксперты предметной области указывают на соответствие реальных бизнес - процессов созданным диаграммам. Найденные несоответствия исправляются и только после прохождения экспертизы без замечаний можно приступать к следующему сеансу декомпозиции. Таким образом достигается соответствие модели реальным бизнес - процессам на любом и каждом уровне модели. Синтаксис описания системы в целом и каждого ее фрагмента одинаков во всей модели. Работы (Activity), которые означают некие поименованные процессы, функции или задачи, изображаются в виде прямоугольников. Именем работы должен быть глагол или глагольная форма (например "Изготовление детали", "Прием заказа" и т.д.). Взаимодействие работ с внешним миром и между собой описывается в виде стрелок. Стрелки представляют собой некую информацию и именуются существительными (например, "Заготовка", "Изделие", "Заказ"). В IDEF0 различают пять типов стрелок:

- Вход (Input) - материал или информация, которая используется или преобразовывается работой.
- Управление (Control) - правила, стратегии, процедуры или стандарты, которыми руководствуется работа. Каждая работа должна иметь хотя бы одну стрелку управления.
- Выход (Output) - материал или информация, которая производится работой. Каждая работа должна иметь хотя бы одну стрелку выхода.
- Механизм (Mechanism)- ресурсы, которые выполняют работу, например, персонал предприятия станки, механизмы и т.д.
- Вызов - специальная стрелка, указывающая на другую модель работы.

Каждый тип стрелок подходит или выходит к определенной стороне прямоугольника, изображающего работу. К левой стороне подходят стрелки входов, к верхней - стрелки управления, к нижней - механизмов реализации

выполняемой функции, а из правой - выходят стрелки выходов. Такое соглашение предполагает, что используя управляющую информацию и реализующий ее механизм, функция преобразует свои входы в соответствующие выходы.

При создании новой модели (меню File / New) автоматически создается контекстная диаграмма с единственной работой, изображающей систему в целом. Для внесения имени работы следует кликнуть по работе правой кнопкой мыши, выбрать в меню Name Editor и в появившемся диалоге внести имя работы. Для описания других аспектов контекста служит диалог Model Definition Editor (вызывается из меню Edit/Model Definition).

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

1. кликните на символе стрелки в палитре инструментов 
2. перенесите курсор к левой стороне экрана, пока не появится начальная штриховая полоска
3. щелкните один раз по полоске (откуда выходит стрелка) и еще раз в левой части работы со стороны входа (где заканчивается стрелка)
4. вернитесь в палитру инструментов и выберите опцию редактирования стрелки  дважды щелкните на линии стрелки, во всплывающем меню выберите Name Editor и добавьте имя стрелки
5. стрелки управления, выхода, механизма и выхода изображаются аналогично.

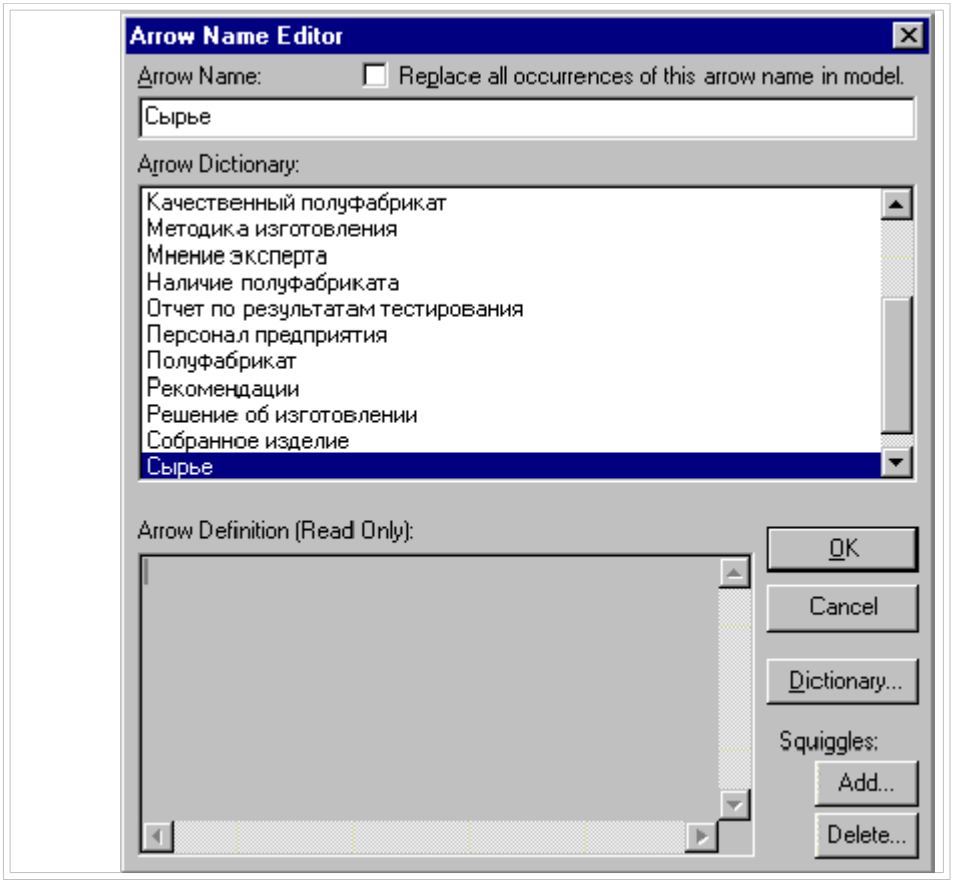


Рис.1. Задание имени.

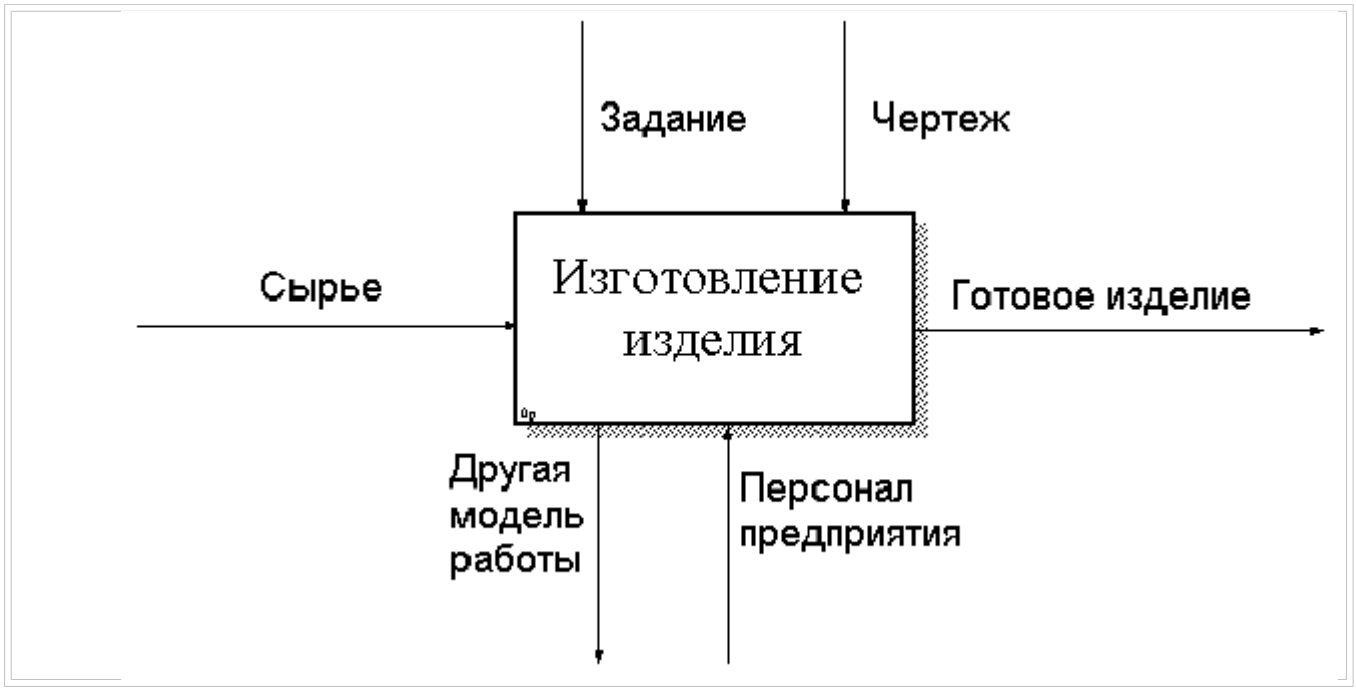


Рис.2. Контекстная диаграмма.

Имена вновь внесенных стрелок автоматически заносятся в словарь (Arrow Dictionary). Словарь стрелок редактируется при помощи специального редактора Arrow Dictionary Editor, в котором определяется стрелка и вносится относящийся к ней комментарий. Словарь стрелок можно распечатать в

виде отчета (меню Report / Arrow Report...) и получить тем самым толковый словарь терминов предметной области, использующихся в модели.

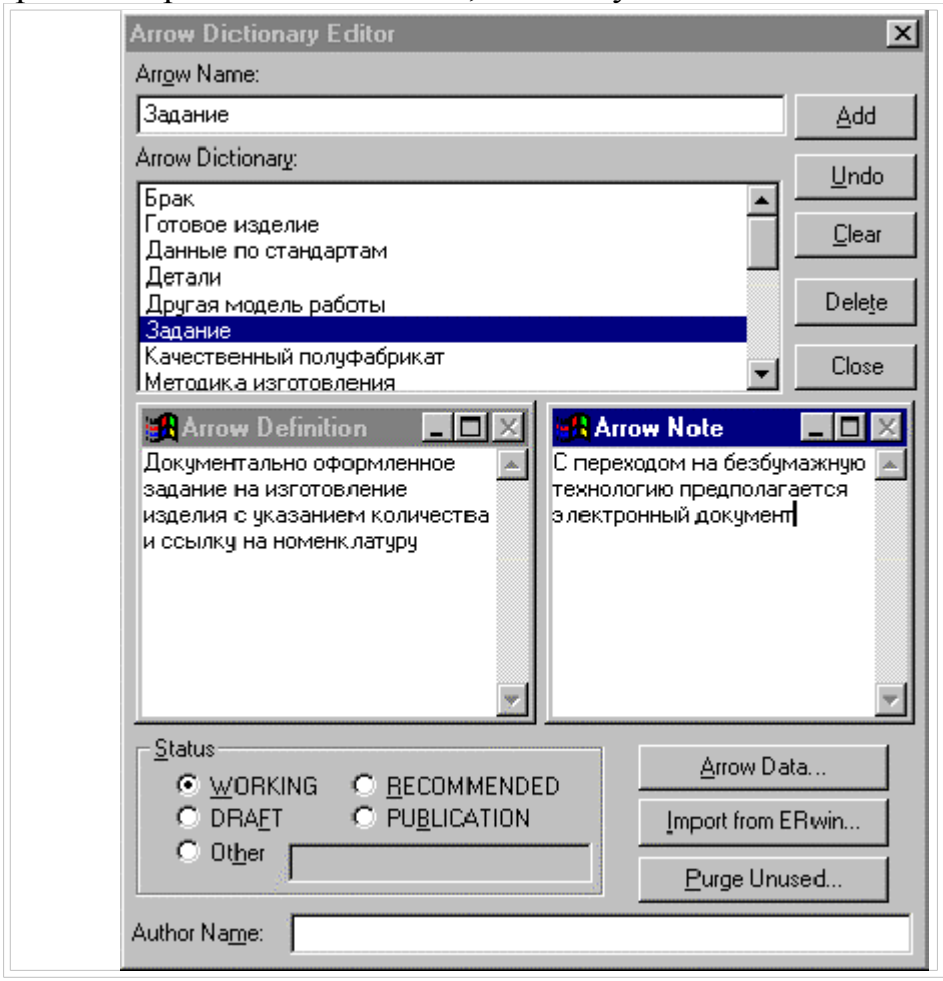


Рис.3. Словарь стрелок.

После создания контекстной диаграммы можно приступить к декомпозиции. Для этого нужно кликнуть по кнопке перехода на нижний уровень  . Появляется диалог Activity Box Count, в котором необходимо указать количество работ на диаграмме декомпозиции (в дальнейшем можно будет добавить недостающие работы или удалить лишние) и нотацию диаграммы. BPwin позволяет создавать смешанные модели - в рамках одной модели могут сосуществовать и быть связанными модели IDEF0, DFD и IDEF3. Такой подход позволяет описать интересующие нас аспекты каждой подсистемы. Для обеспечения наглядности и лучшего понимания моделируемых процессов рекомендуется использовать от 3-х до 6-ти блоков на одной диаграмме. Остановимся пока на нотации IDEF0 и кликнем на OK. Появляется диаграмма декомпозиции. Работы расположены в так называемом порядке доминирования (по степени важности или в порядке очередности выполнения), начиная с левого верхнего угла и кончая нижним правым углом, что значительно облегчает в дальнейшем чтение диаграммы. Стрелки, которые были внесены на контекстной диаграмме, показываются и на диаграмме декомпозиции (миграция стрелок), но при этом не касаются работ. Такие стрелки называются несвязанными и воспринимаются, как синтаксическая ошибка. Для

связывания стрелки необходимо перейти в режим редактирования стрелок, кликнуть по стрелке и кликнуть по соответствующему сегменту работы. Для связи работ между собой используются внутренние стрелки, т.е. стрелки, которые не касаются границы диаграммы, начинаются у одной и кончаются у другой работы.

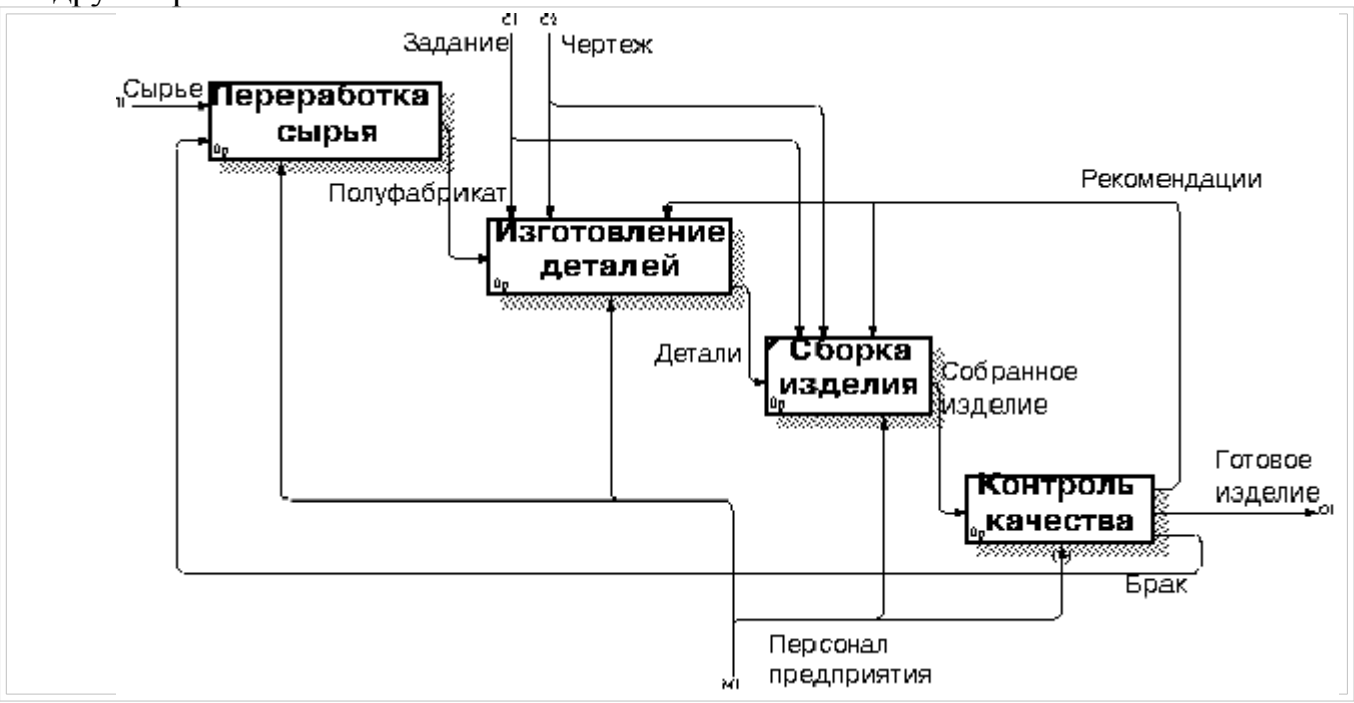


Рис. 4. Диаграмма декомпозиции.

Для рисования внутренней стрелки необходимо в режиме рисования стрелок кликнуть по сегменту (например выхода) одной работы и затем по сегменту (например входа) другой. В IDEF0 различают пять типов связей работ:

1. прямая связь по входу, когда стрелка выхода вышестоящей (далее просто выход) работы направляется на вход нижестоящей (например, на рисунке стрелка "Детали" связывает работы "Изготовление деталей" и "Сборка деталей");
2. прямая связь по управлению, когда выход вышестоящей работы направляется на управление нижестоящей;
3. обратная связь по входу, когда выход нижестоящей работы направляется на вход вышестоящей (стрелка "Брак");
4. обратная связь по управлению, когда выход нижестоящей работы направляется на управление вышестоящей (стрелка "Рекомендации");
5. связь выход-механизм, когда выход одной работы направляется на механизм другой.

Вновь внесенные граничные стрелки на диаграмме декомпозиции нижнего уровня изображаются в квадратных скобках и автоматически не появляются на диаграмме верхнего уровня. Для их "перетаскивания" наверх

нужно сначала выбрать кнопку  на палитре инструментов и кликнуть по квадратным скобкам граничной стрелки. Появляется диалог Border Arrow Editor.

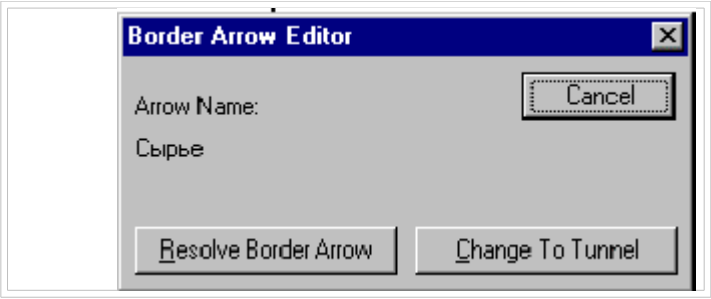


Рис. 5. Диалог Border Arrow Editor.

Если кликнуть по кнопке Resolve Border Arrow, стрелка мигрирует на диаграмму верхнего уровня, если по кнопке Change To Tunnel, стрелка будет затуннелирована и не попадет на другую диаграмму. Туннелирование может быть применено для изображения малозначимых стрелок.

Одна и та же информация может обрабатываться в нескольких работах, в то же время из нескольких работ могут выходить одинаковые данные, то есть стрелки могут разветвляться и сливаться. Для разветвления стрелки нужно в режиме редактирования стрелки кликнуть по фрагменту стрелки и по соответствующему сегменту работы.

Дополнение модели процессов диаграммами DFD и Workflow (IDEF3)

Общие принципы построения модели в методологиях DFD и IDEF3 сходны с IDEF0: модель представляет собой совокупность иерархически зависимых диаграмм, прямоугольники изображают работы или процессы, стрелки- это тоже некие данные, построение модели осуществляется сверху вниз путем проведения декомпозиции крупных работ на более мелкие. Диаграммы потоков данных (Data flow diagramming, DFD) используются для описания документооборота и обработки информации. Их можно использовать как дополнение к модели IDEF0 для более наглядного отображения текущих операций документооборота в корпоративных системах обработки информации. DFD описывают функции обработки информации (работы), документы (стрелки, arrow), объекты, сотрудников или отделы, которые участвуют в обработке информации (внешние ссылки, external references) и таблицы для хранения документов (хранилище данных, data store). В отличие от IDEF0 для стрелок нет понятия вход, выход, управление или механизм и неважно, в какую грань работы входит или из какой грани выходят стрелки. В BPwin для построения диаграмм потоков данных используется нотация Гейна-Сарсона. Для того, чтобы дополнить модель IDEF0 диаграммой DFD нужно в процессе декомпозиции в диалоге Activity Box Count кликнуть по ра-

диокнопке DFD. В палитре инструментов на новой диаграмме DFD появляются новые кнопки:


- добавить в диаграмму внешнюю ссылку (External Reference). Внешняя ссылка является источником или приемником данных извне модели ;


- добавить в диаграмму хранилище данных (Data store). Хранилище данных позволяет описать данные, которые необходимо сохранить в памяти прежде чем использовать в работах.


- ссылка на другую страницу. В отличие от IDEF0 инструмент off- page reference позволяет направить стрелку на любую диаграмму (а не только на верхний уровень).

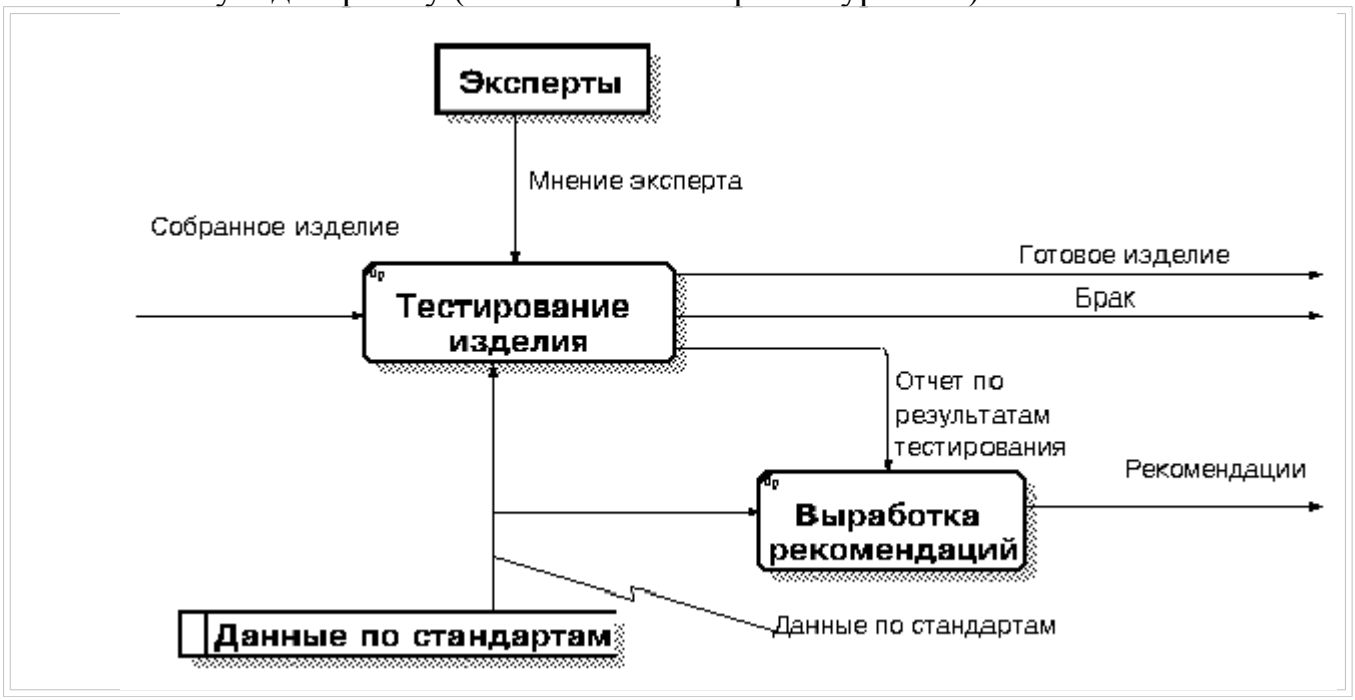


Рис. 6. Диаграмма в нотации DFD.

Наличие в диаграммах DFD элементов для описания источников, приемников и хранилищ данных позволяет более эффективно и наглядно описать процесс документооборота. Однако, для описания логики взаимодействия информационных потоков более подходит IDEF3, называемая также workflow diagramming, - методология моделирования, использующая графическое описание информационных потоков, взаимоотношений между процессами обработки информации и объектов, являющихся частью этих процессов. Диаграммы Workflow могут быть использованы в моделировании бизнес - процессов для анализа завершенности процедур обработки информации. С их помощью можно описывать сценарии действий сотрудников организации, например, последовательность обработки заказа или события, кото-

рые необходимо обработать за конечное время. Каждый сценарий сопровождается описанием процесса и может быть использован для документирования каждой функции.

Прямоугольники на диаграмме Workflow называются единицами работы (Unit of Work, UOW) и обозначают событие, процесс, решение или работу. Для редактирования диаграммы используются примерно те же диалоги, что и для IDEF0. В палитре инструментов на диаграмме Workflow имеются кнопки для новых элементов:


- добавить в диаграмму объект ссылки (Referent). Объект ссылки в IDEF3 выражает некую идею, концепцию или данные, которые нельзя связать со стрелкой, перекрестком или работой. Имя объекта ссылки задается в диалоге Referent (пункт всплывающего меню Name Editor), в качестве имени можно использовать имя какой-либо стрелки с других диаграмм или имя сущности из модели данных (о том, как использовать модель данных в BPwin будет рассказано в следующей статье). Объекты ссылки должны быть связаны с единицами работ или перекрестками пунктирными линиями. Официальная спецификация IDEF3 различает три стиля объектов ссылок - безусловные (unconditional), синхронные (synchronous) и асинхронные (asynchronous). BPwin поддерживает только безусловные. Синхронные и асинхронные, используемые в диаграммах переходов состояний объектов не поддерживаются.


- добавить в диаграмму перекресток (Junction). Перекрестки используются для отображения логики взаимодействия стрелок при слиянии и разветвлении или для отображения множества событий, которые могут или должны быть завершены перед началом следующей работы. Различают перекрестки для слияния (Fan-in Junction) и разветвления (Fan-out Junction) стрелок. Перекресток не может использоваться одновременно для слияния и для разветвления. При внесении перекрестка в диаграмму в диалоге Junction Type Editor необходимо указать тип перекрестка. Смысл каждого типа приведен в таблице.

Обозначение	Наименование	Смысл в случае слияния стрелок (Fan-in Junction)	Смысл в случае разветвления стрелок (Fan-out Junction)
	Asynchronous AND	Все предшествующие процессы должны быть завершены	Все следующие процессы должны быть запущены
	Synchronous AND	Все предшествующие	Все следующие про-

		процессы завершены одновременно	цессы запускаются одновременно
	Asynchronous OR	Один или несколько предшествующих процессов должны быть завершены	Один или несколько следующих процессов должны быть запущены
	Synchronous OR	Один или несколько предшествующих процессов завершены одновременно	Один или несколько следующих процессов запускаются одновременно
	XOR (Exclusive OR)	Только один предшествующий процесс завершен	Только один следующий процесс запускается

Все перекрестки на диаграмме нумеруются, каждый номер имеет префикс "J". Можно редактировать свойства перекрестка при помощи диалога Definition Editor. В отличие от IDEF0 и DFD, в IDEF3 стрелки могут сливаться и разветвляться только через перекрестки. Здесь различают три типа стрелок, стиль которых устанавливается через меню Edit / Arrow Style :

- Старшая (Precedence)  - сплошная линия, связывающая единицы работ (UOW). Рисуется слева направо или сверху вниз.
- Отношения (Relational Link)  - пунктирная линия, использующаяся для изображения связей между единицами работ (UOW) и между единицами работ и объектами ссылок.
- Потоки объектов (Object Flow)  - стрелка с двумя наконечниками используется для описания того факта, что объект используется в двух или более единицах работы, например, когда объект порождается в одной работе и используется в другой.

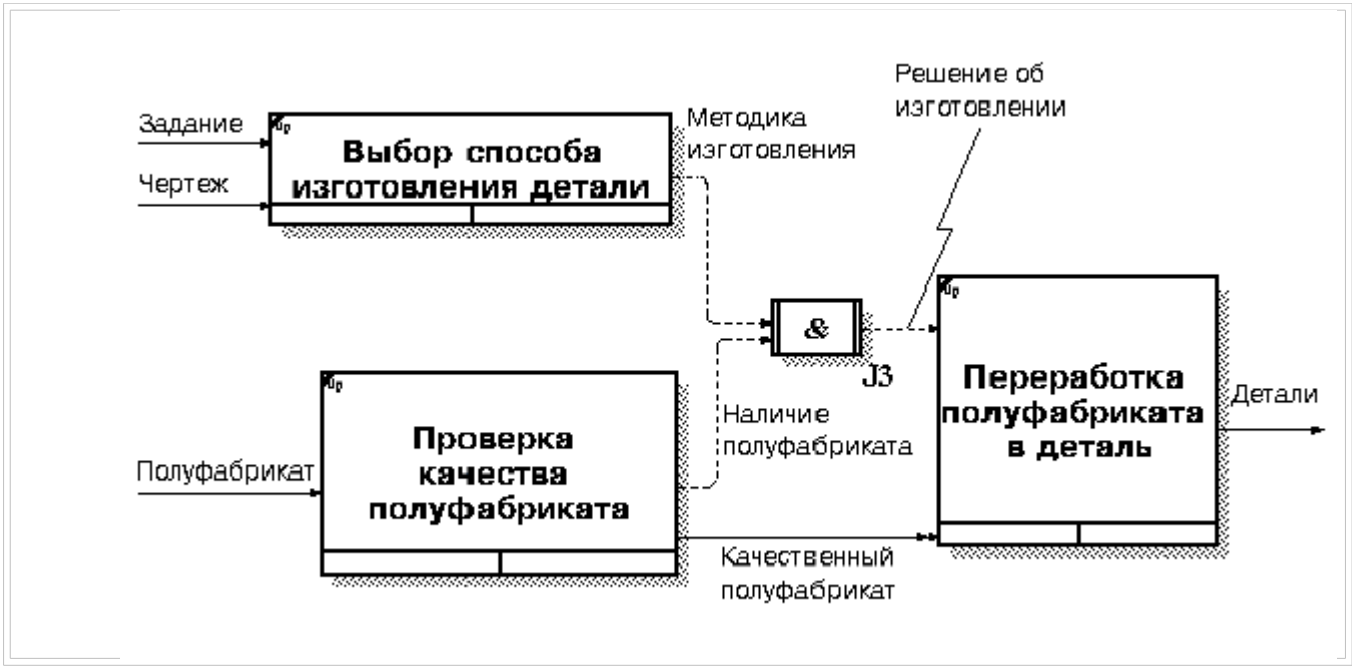


Рис. 7. Диаграмма в нотации IDEF3.

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

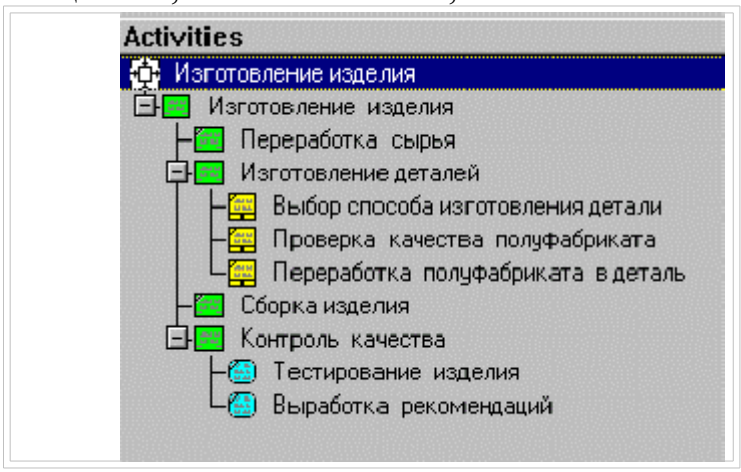


Рис. 8. Смешанная модель в окне Model Explorer.

Синтаксический анализ модели BPwin'a позволяет легко обнаружить "бесполезные" (не имеющие выхода), "неуправляемые" (не имеющие управления) и "простаивающие" работы. Более тонкий анализ позволяет выявить дублирующие, избыточные или неэффективные работы. Модель дает целостное представление о работе системы в целом и позволяет понять взаимосвязи всех составляющих системы. При этом часто выясняется, что обработка информации и использование ресурсов неэффективны, важная информация не доходит до соответствующего рабочего места и т.д. Признаком неэффективной организации работ является, например, отсутствие обратных связей по входу и управлению для многих, критически важных работ.

Невозможно построить эффективную ИС при неэффективной общей организации работы. Поэтому результатом анализа и критической оценки мо-

дели AS-IS должно быть перенаправление информационных потоков и усовершенствование бизнес- процессов в новой модели TO-BE, которая должна использоваться для реорганизации деятельности предприятия. Такой результат построения модели сам по себе самодостаточен, то есть если удается более рационально организовать бизнес-процессы на предприятии - это уже результат, оправдывающей капиталовложения. Однако при создании ИС модель процессов - это только первый шаг, за которым обычно следует построение модели данных.

Соответствие модели данных и модели процессов

Стрелки в модели процессов означают некоторую информацию, использующуюся в моделируемой системе. ERWin поддерживает два уровня представления модели данных - логический и физический. Логический уровень не зависит от конкретной реализации БД и позволяет наглядно представить данные для обсуждения с экспертами предметной области. Физический уровень является отображением системного каталога БД и зависит от конкретной реализации БД. На логическом уровне модели данных информация отображается в виде сущностей (соответствуют таблицам на физическом уровне), состоящих из атрибутов сущностей (соответствуют колонкам таблицы). Сущности состоят из совокупности отдельных записей - экземпляров сущностей (соответствуют записям в таблице). К модели данных предъявляются определенные требования (т.н. нормализация данных), которые призваны обеспечить компактность и непротиворечивость хранения данных. Основная идея нормализации данных - каждый факт должен хранится в одном месте. Это приводит к тому, что информация, которая моделируется в виде одной стрелки в модели процессов может содержаться в нескольких сущностях и атрибутах в модели данных. Кроме того, на диаграмме модели процессов могут присутствовать различные стрелки, изображающие одни и те же данные, но на разных этапах обработки (например, необработанные детали - обработанные детали - собранное изделие). Информация о таких стрелках находится в одних и тех же сущностях. Следовательно, одной и той же стрелке в модели процессов могут соответствовать несколько сущностей в модели данных и наоборот, одной сущности может соответствовать несколько стрелок.

Стрелке в модели процессов может соответствовать отдельная сущность в модели данных. Так, стрелке "Части" на рис. 9 соответствует сущность "Часть", стрелке "Конечные продукты" - сущность "Продукт".

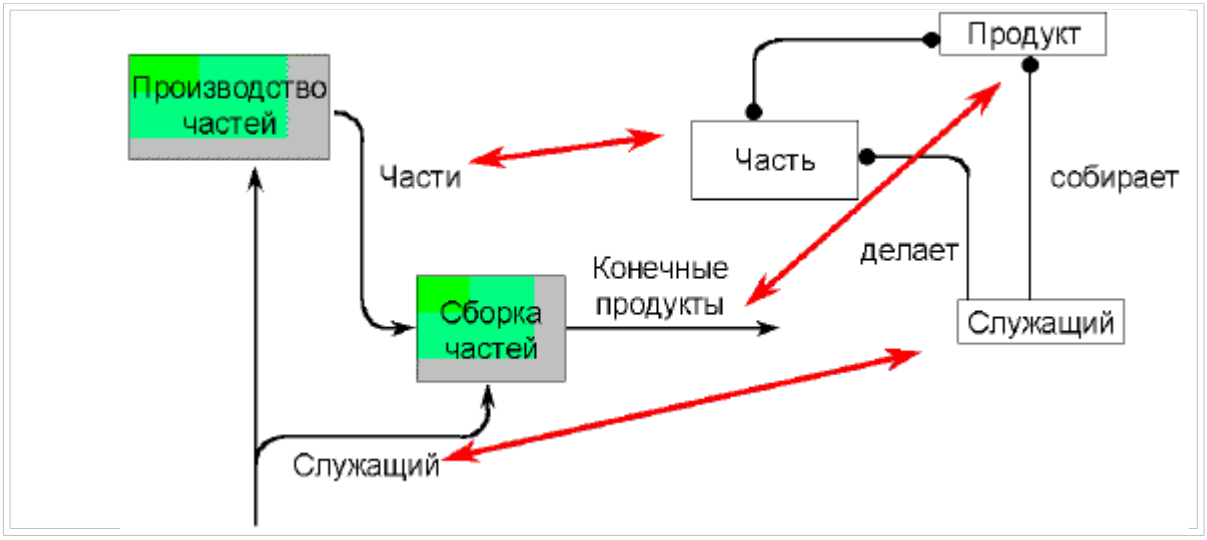


Рис. 9. Преобразование стрелки в сущность.

Информация о стрелке может содержаться только в нескольких атрибутах сущности. Разным атрибутам одной и той же сущности могут соответствовать разные стрелки. На рис. 10 стрелка "Новая часть" соответствует атрибутам "Номер части" и "Название части", стрелка "Наличное количество" - атрибутам "Количество".

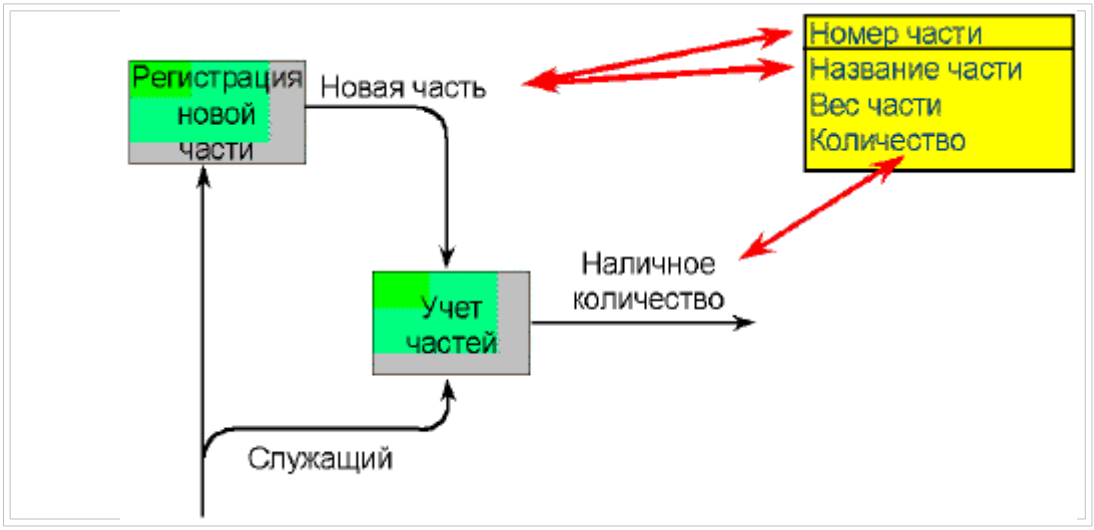


Рис. 10. Преобразование стрелки в атрибут.

Работы в модели процессов могут создавать или изменять данные, которые соответствуют входящим или выходящим стрелкам. Они могут воздействовать как целиком на сущности (создавая или модифицируя экземпляры сущности, рис. 11), так и на отдельные атрибуты сущности (рис. 12).

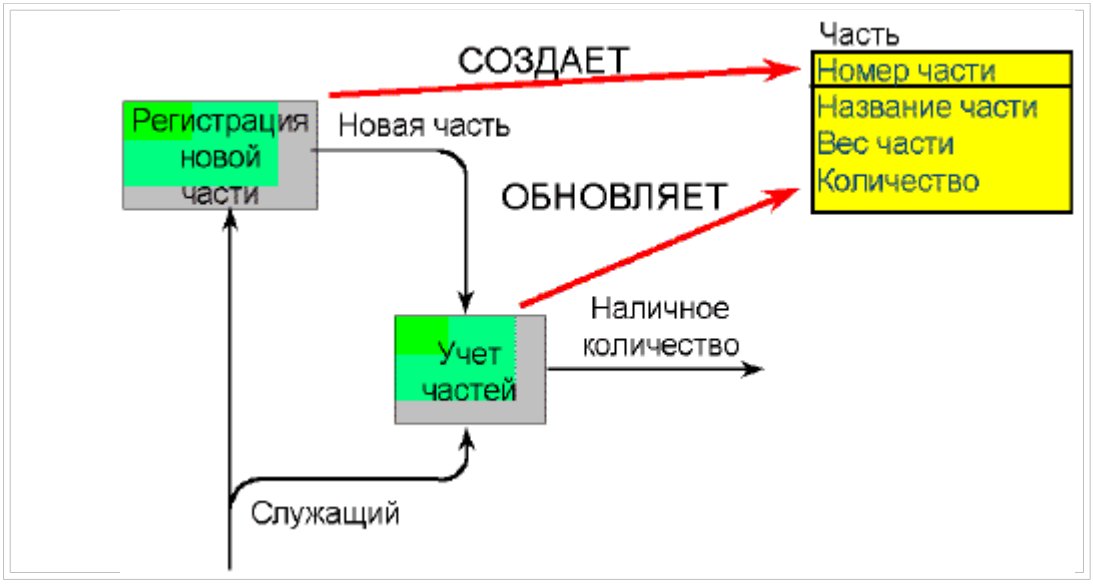


Рис. 11. Воздействие работы на сущность.

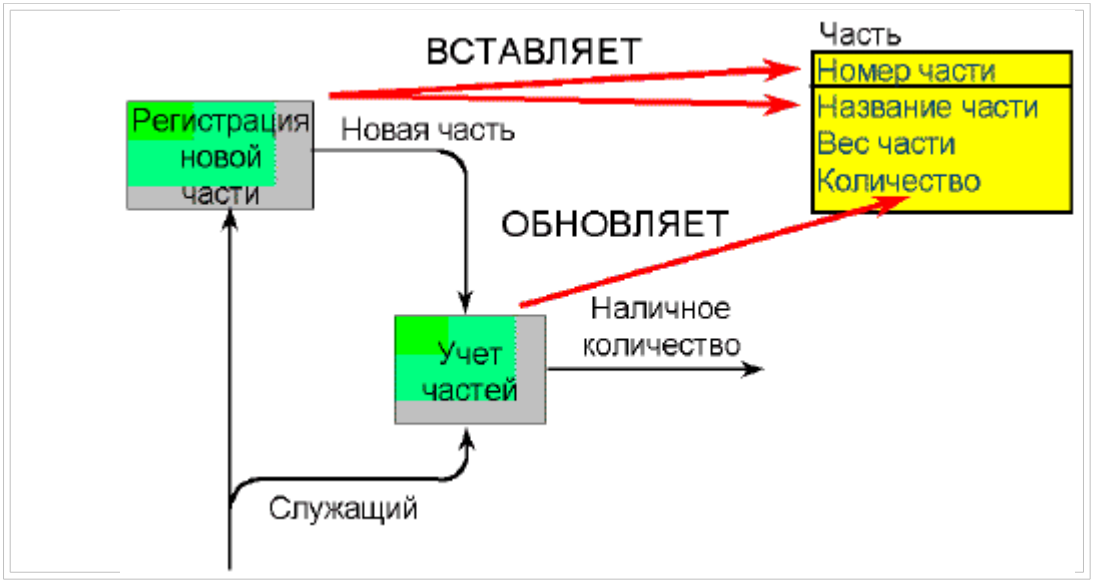


Рис. 12. Воздействие работы на атрибут.

BPWin позволяет связывать элементы модели данных, созданной с помощью ERWin, документировать влияние работ на данные и, тем самым, позволяет создать спецификации на права доступа к данным для каждого процесса .

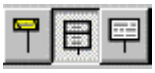
Создание модели данных с помощью ERWin

Построение модели данных предполагает определение сущностей и атрибутов, то есть 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 "техниче-

ских" наименований и быть достаточно важными для того, чтобы их моделировать.

ERwin имеет развитый иструмент для облегчения проектирования модели данных. Интерфейс выполнен в стиле Windows-приложений, достаточно прост и интуитивно понятен. В дальнейшем будет описан интерфейс версии 3.5.

Таблица 1.Кнопки панели инструментов описаны в таблице 1.

	Создание, открытие, сохранение и печать модели.
	Вызов диалога Report Browser для генерации отчетов.
	Изменение уровня просмотра модели: уровень сущностей, уровень атрибутов и уровень определений.
	Изменение масштаба просмотра модели.
	Генерация схемы БД, выравнивание схемы с моделью и выбор сервера (доступны только на уровне физической модели)
	Вызов дополнительной панели инструментов для работы с репозиторием Model Mart. (Работа с Model Mart будет рассмотрена в следующей статье).
	Переключение между областями модели - Subject Area.

Для создания моделей данных в ERwin можно использовать две нотации: IDEF1X и IE (Information Engineering). В примерах будет использоваться нотация IDEF1X. 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 Вы хотите расположить новую сущность. Кликнув правой кнопкой мыши по сущности и выбрав из всплывающего меню пункт Entity Editor... можно вызвать диалог Entity Editor, в котором определяются имя, описание и комментарии сущности.

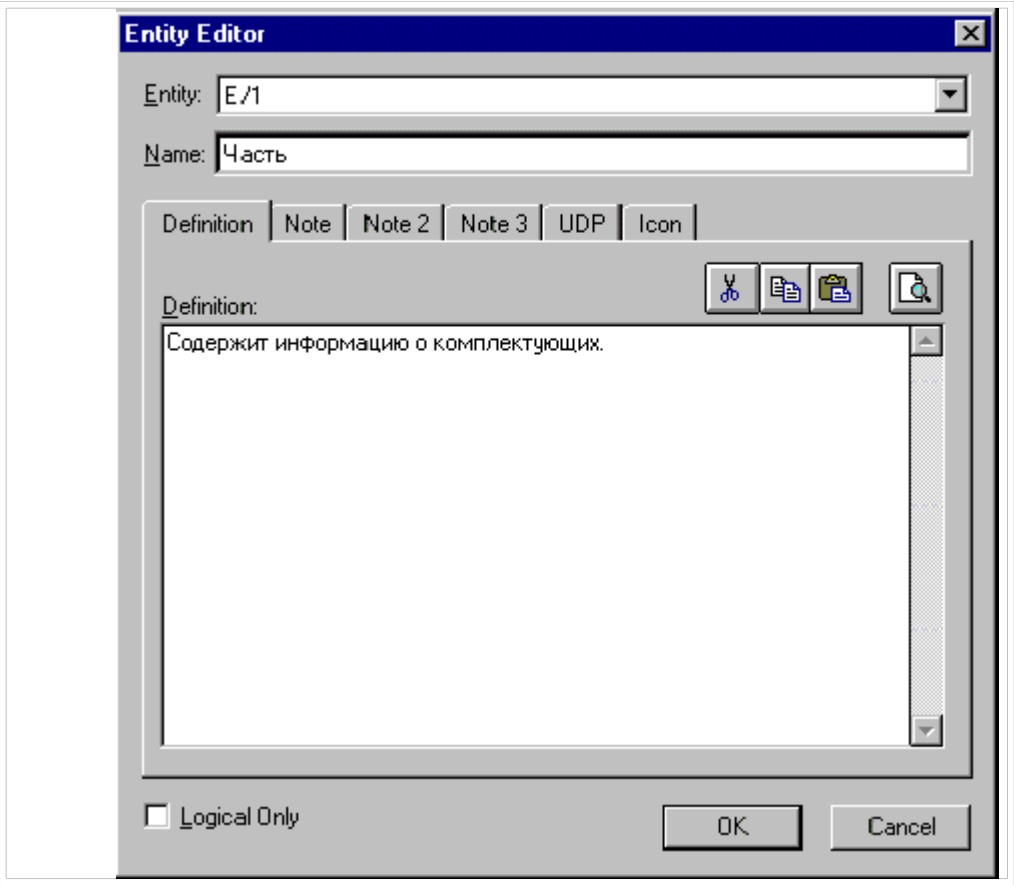


Рис. 13. Диалог Entity Editor

Каждая сущность должна быть полностью определена с помощью текстового описания в закладке Definition. Закладки Note, Note2, Note3, UDP (User Defined Properties - Свойства, Определенные Пользователем) служат для внесения дополнительных комментариев и определений сущности. В закладке Icon каждой сущности можно поставить в соответствие изображение (файл bmp), которое будет отображатся в режиме просмотра модели на уровне иконок.

Каждый атрибут хранит информацию об определенном свойстве сущности. Каждый экземпляр сущности должен быть уникальным. Атрибут или группа атрибутов, которые идентифицируют сущность, называется первичным ключом. Для описания атрибутов следует, кликнув правой кнопкой по сущности, выбрать в появившемся меню пункт Attribute Editor. Появляется диалог Attribute Editor.

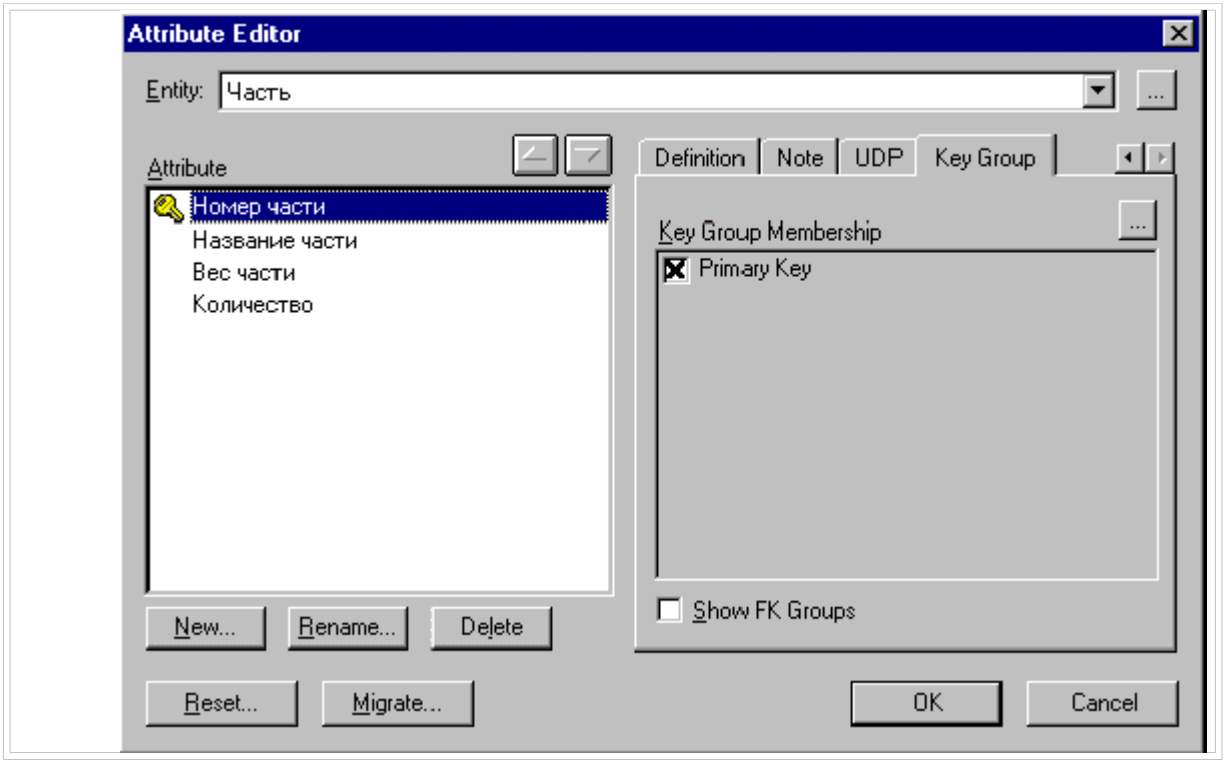


Рис. 14. Диалог Attribute Editor

Кликнув по кнопке New..., в появившемся диалоге New Attribute следует указать имя атрибута, имя соответствующей ему колонки и домен. Домен атрибута будет использоваться при определении типа колонки на уровне физической модели (подробнее о доменах см. в статье "ERwin расширяет свои возможности", Компьютер Пресс, 3, 1998). Для атрибутов первичного ключа в закладке Key Group диалога Attribute Editor необходимо сделать пометку в окне выбора Primary Key. При определении первичного ключа может быть рассмотрено несколько наборов атрибутов. Такие наборы называются потенциальными ключами. Например, если рассматривается сущность "Сотрудник", такими наборами могут быть:

- `Имя, Фамилия, Отчество, Дата рождения`
- `Номер паспорта`
- `Табельный номер`
- `Отдел`

К первичным ключам предъявляются определенные требования. Первичный ключ должен однозначно идентифицировать экземпляр сущности (этому требованию не удовлетворяет четвертый ключ, поскольку он может идентифицировать группу сотрудников, работающих в определенном отделе, но не каждого сотрудника). Первичный ключ должен быть компактен, то есть удаление любого атрибута из состава первичного ключа должно приводить к потере уникальности экземпляра сущности (если удалить Дату рождения из первого ключа, то невозможно будет идентифицировать полных тезок). Каждый атрибут из состава первичного ключа не должен принимать NULL - зна-

чений (например, если принять в качестве первичного ключа номер паспорта, необходимо быть уверенным, что все сотрудники имеют паспорта). Каждый атрибут первичного ключа не должен менять свое значение в течение всего времени существования экземпляра сущности (сотрудник может сменить фамилию и паспорт, поэтому первый и второй потенциальные ключи не могут стать первичными). Потенциальные ключи, не ставшие первичными, называются альтернативными. Атрибуты, или наборы атрибутов, использующиеся для доступа к группе экземпляров сущности, называются инверсионными ключами. Для описания альтернативных и инверсионных ключей необходимо кликнуть по кнопке ... (диалог Attribute Editor, закладка Key Group) и в появившемся диалоге закладка Key Group Editor создать новую ключевую группу (либо инверсионную, либо альтернативную) и указать, какие атрибуты входят в ту или иную группу.

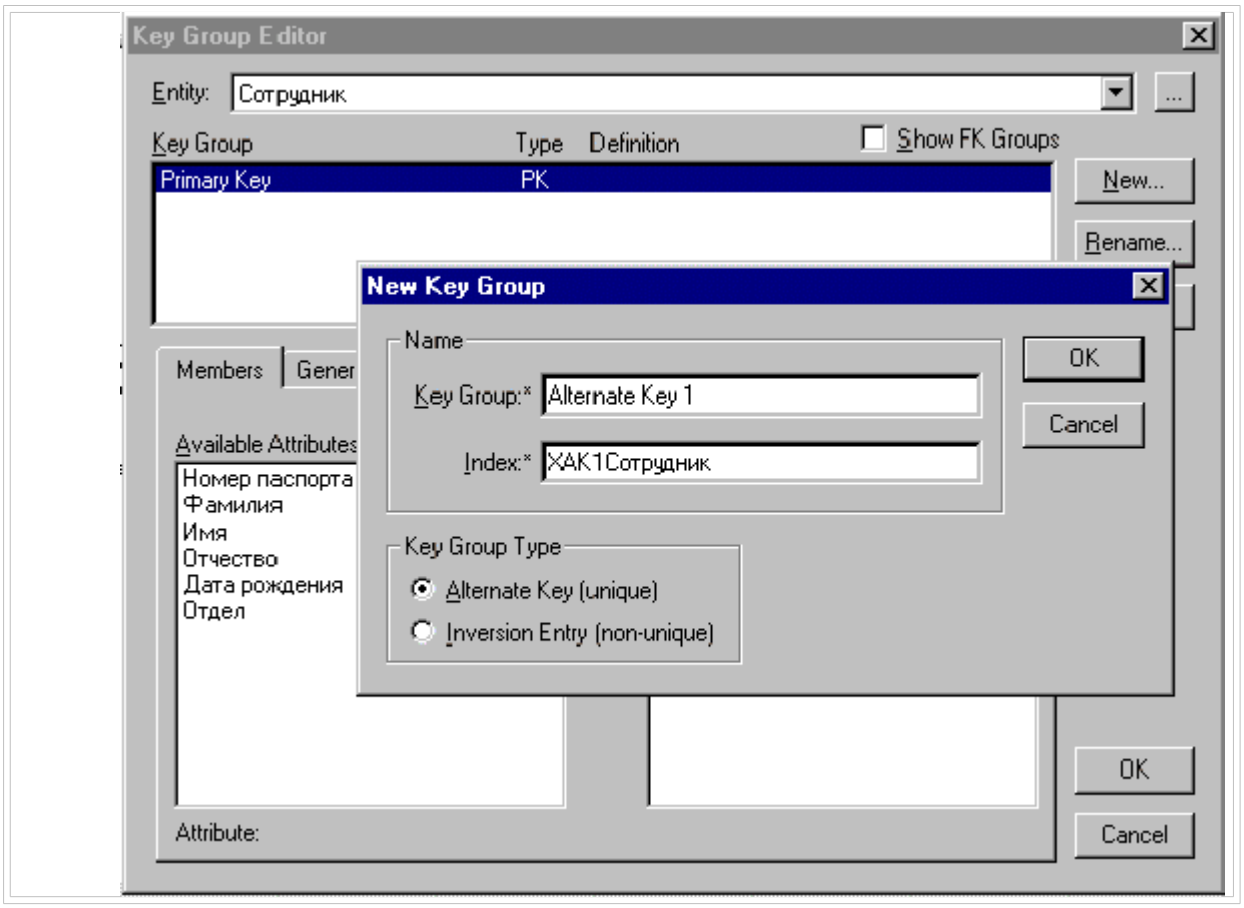


Рис. 15. Диалог Key Group Editor

ERwin имеет несколько уровней отображения диаграммы. Переключиться между ними можно кликнув по любому месту диаграммы, не занятому объектами модели и выбрав в появившемся меню пункт Display Level.

На уровне абли атрибутов атрибуты альтернативного ключа помечаются номером (AKm.n), где m - номер ключа, n - номер атрибута в ключе. Инверсионные ключи помечаются номером (IEm.n). В дальнейшем при генерации БД на атрибутах альтернативных ключей могут быть сгенерированы уникальные индексы, на атрибутах инверсионного ключа - неуникальные.

Имена индексов задаются в диалоге New Key Group (рис. 15). Атрибуты первичного ключа отображаются выше горизонтальной линии - прочие атрибуты - ниже.

К модели данных предъявляются определенные требования, называемые нормальными формами. Процесс приведения к нормальным формам называется нормализацией. Так, первая нормальная форма требует, чтобы все атрибуты были атомарными (не должно быть атрибута "Адрес" - должны быть атрибуты "Индекс", "Страна", "Область", "Город", "Улица", "Дом", "Квартира"). Вторая нормальная форма требует, чтобы каждый неключевой атрибут зависел от всего первичного ключа, не должно быть зависимости от части ключа. (В данной статье не ставится целью описание приведения к нормальным формам. Подробно вопросы нормализации освещены, например, в книге К. Дж. Дейта "Введение в системы баз данных", Киев, издательство "Диалектика", 1998). Для приведения ко второй нормальной форме необходимо создать новую сущность, перенести в нее атрибуты, зависящие от части ключа, сделать часть ключа первичным ключом новой сущности и установить идентифицирующую связь (см. ниже) от новой сущности к старой. Например, в сущности "Служащий" (слева на рис. 16)

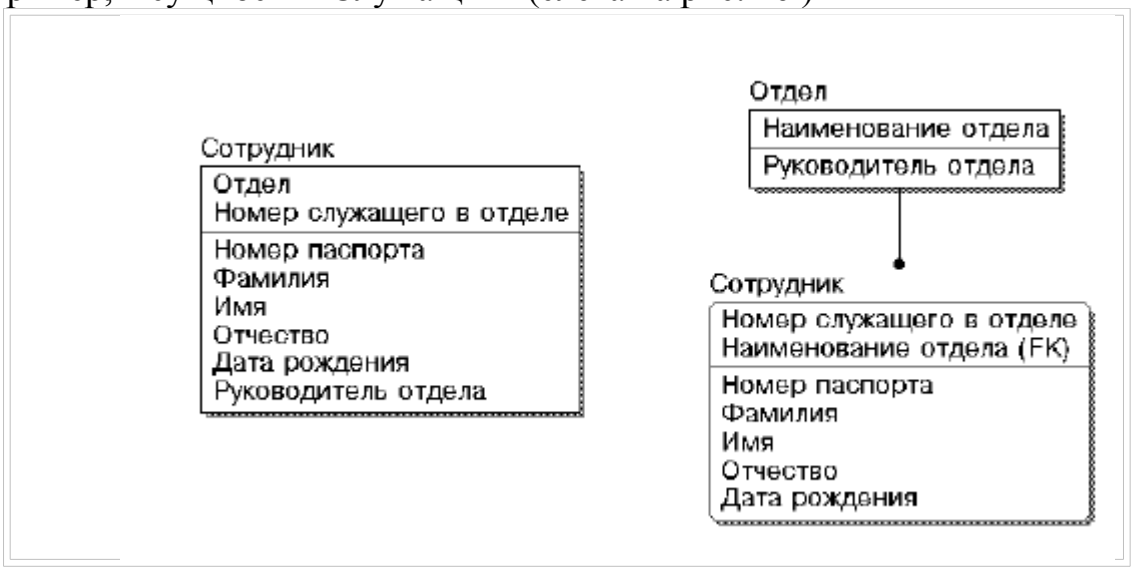
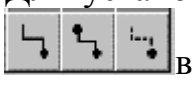


Рис. 16. Иллюстрация второй нормальной формы.

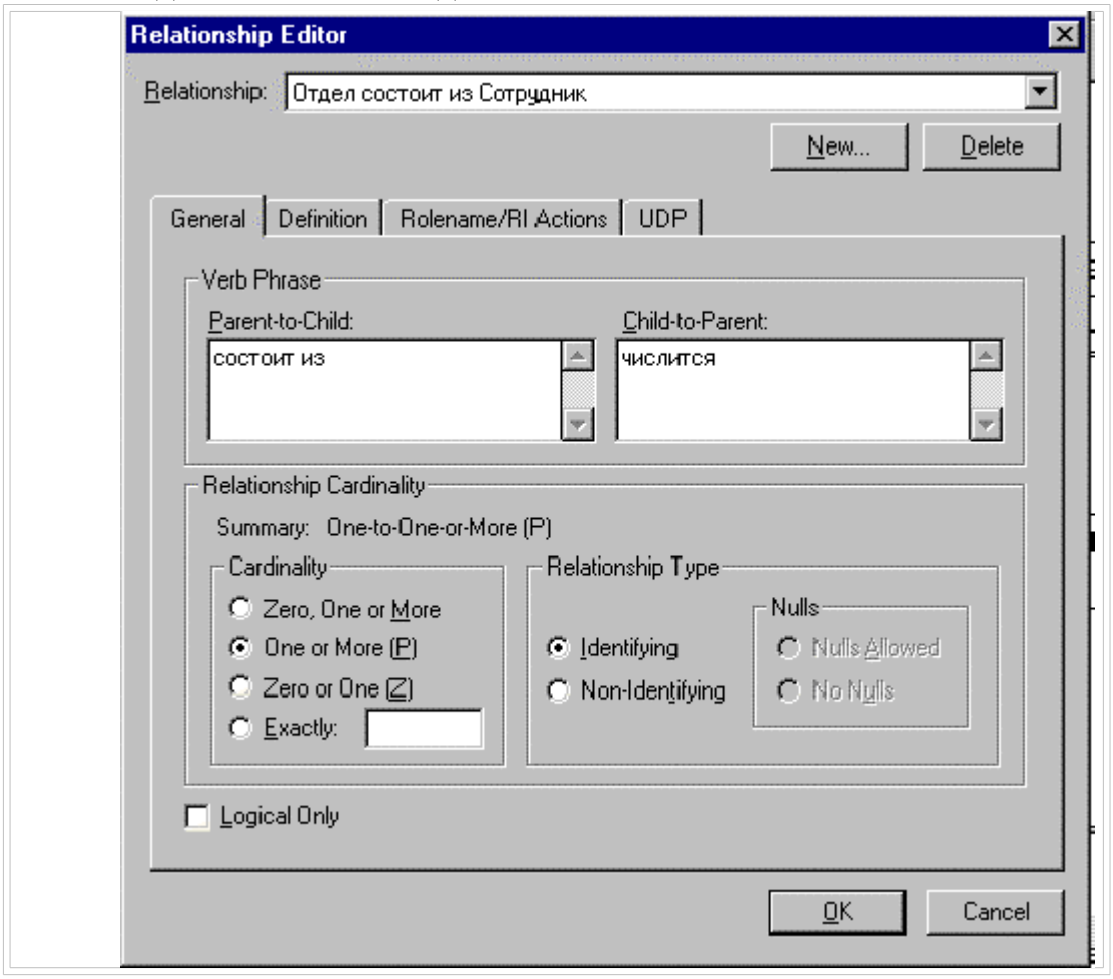
атрибут "Руководитель" отдела зависит от "Наименования отдела". Справа изображены сущности, приведенные ко второй нормальной форме. Для установки связи между сущностями нужно воспользоваться кнопками  в палитре инструментов.

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

Идентифицирующая связь устанавливается между независимой (родительский конец связи) и зависимой (дочерний конец связи) сущностями. Зависимая сущность изображается прямоугольником со скругленными углами

(сущность "Служащий", справа на рис. 16). Экземпляр зависимой сущности определяется только через отношение к родительской сущности, то есть в структуре на рис.16 информация о служащем не может быть внесена и не имеет смысла без информации об отделе, в котором он работает. При установлении идентифицирующей связи атрибуты первичного ключа родительской сущности переносятся в состав первичного ключа дочерней сущности (миграция атрибутов). В дочерней сущности они помечаются как внешний ключ - (FK). При установлении неидентифицирующей связи дочерняя сущность остается независимой, а атрибуты первичного ключа родительской сущности мигрируют в состав неключевых компонентов родительской сущности.

Для редактирования свойств связи следует кликнуть правой кнопкой мыши по связи и выбрать на контекстном меню пункт Relationship Editor. В появившемся диалоге можно задать:



- Мощность (cardinality) связи - служит для обозначения отношения числа экземпляров родительской сущности к числу экземпляров дочерней.
- Verb Phrase - фраза, характеризующая отношение между родительской и дочерней сущностями.
- Тип связи (идентифицирующая / не идентифицирующая).
- Описание связи.

- Правила ссылочной целостности (будут сгенерированы при генерации схемы БД).
- Имя роли. Имя роли - это синоним атрибута внешнего ключа, которое необходимо, например, при циклической связи. В этом случае нельзя иметь два атрибута с одинаковым именем внутри одной сущности. При задании имени роли атрибут мигрирует в качестве внешнего ключа в состав неключевых атрибутов с именем роли.

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

Связь многие ко многим возможна только на уровне логической модели данных. При переходе к физическому уровню ERwin автоматически преобразует связь многие ко многим, добавляя новую, ассоциативную сущность и устанавливая две новые связи один ко многим от старых к новой сущности.

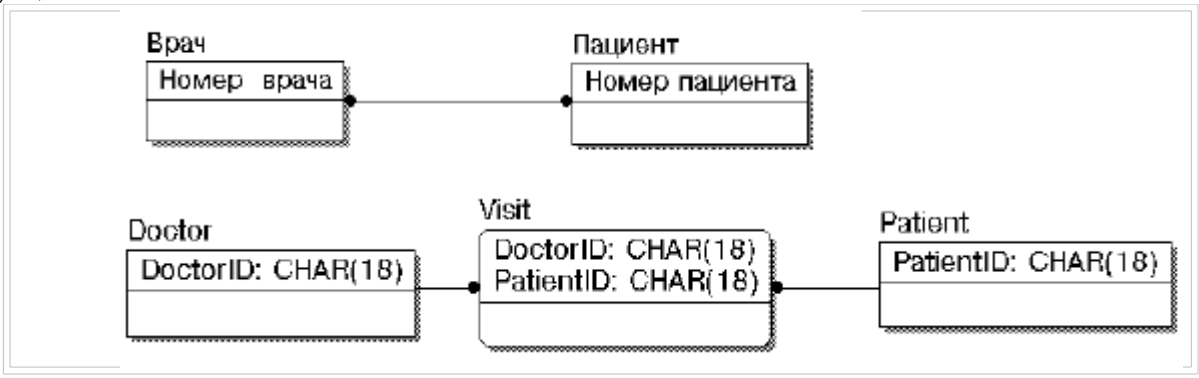


Рис. 17. Иллюстрация разрешения связи многие ко многим.

Следует заметить, что автоматического решения проблемы связи многие ко многим не всегда оказывается достаточно. В данном примере один и тот же пациент может много раз посещать врача, поэтому для того, чтобы идентифицировать визит необходимо в состав первичного ключа таблицы "Visit" добавить, например, дату-время посещения.

Иерархия категорий представляет собой особый тип объединения сущностей, которые разделяют общие характеристики. Например, в организации работают служащие, занятые полный рабочий день, совместители и консультанты. Из их общих свойств можно сформировать обобщенную сущность (родовой предок), чтобы представить информацию общую для всех типов служащих. Специфическая для каждого типа информация может быть расположена в категориальных сущностях. Для моделирования категорий служит кнопка  в палитре инструментов. Для каждой категории можно указать дискриминатор - атрибут родового предка, который показывает как отличить одну категориальную сущность от другой (Атрибут "Тип" на рис. 18).

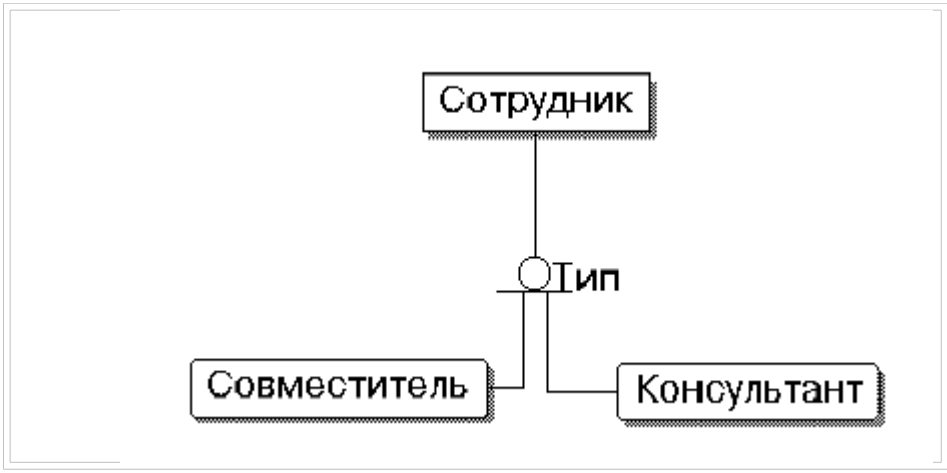


Рис. 18. Иерархия категорий.

При создании реальных моделей данных количество сущностей и атрибутов может исчисляться сотнями.

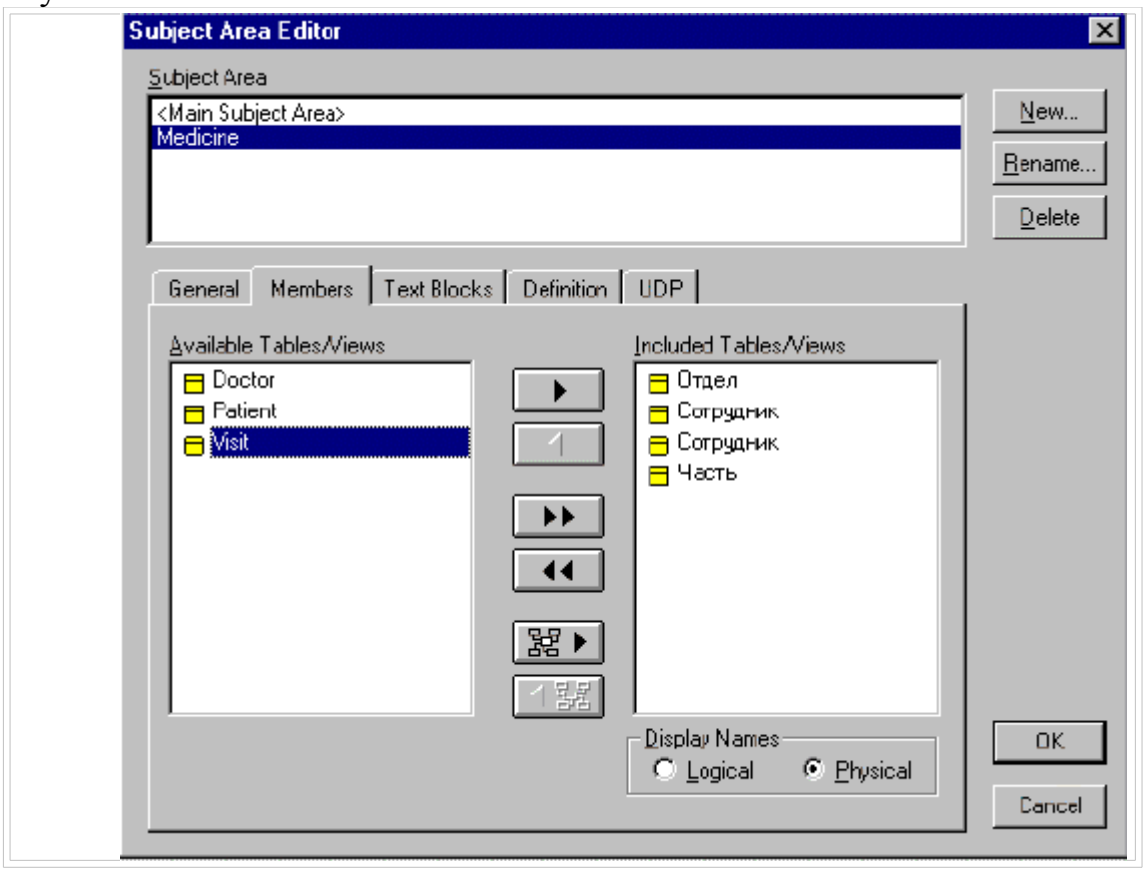


Рис. 19 . Диалог Subject Area Editor.

Для более удобной работы с большими моделями в ERwin'e предусмотрены предметные области (Subject Area), в которые можно включить тематически общие сущности. Для создания предметных областей нужно вызвать диалог Subject Area Editor (меню Edit/ Subject Area...), в котором указывается имя предметной области и входящие в нее сущности. Все изменения, сделанные в предметной области, автоматически отображаются на общей модели.

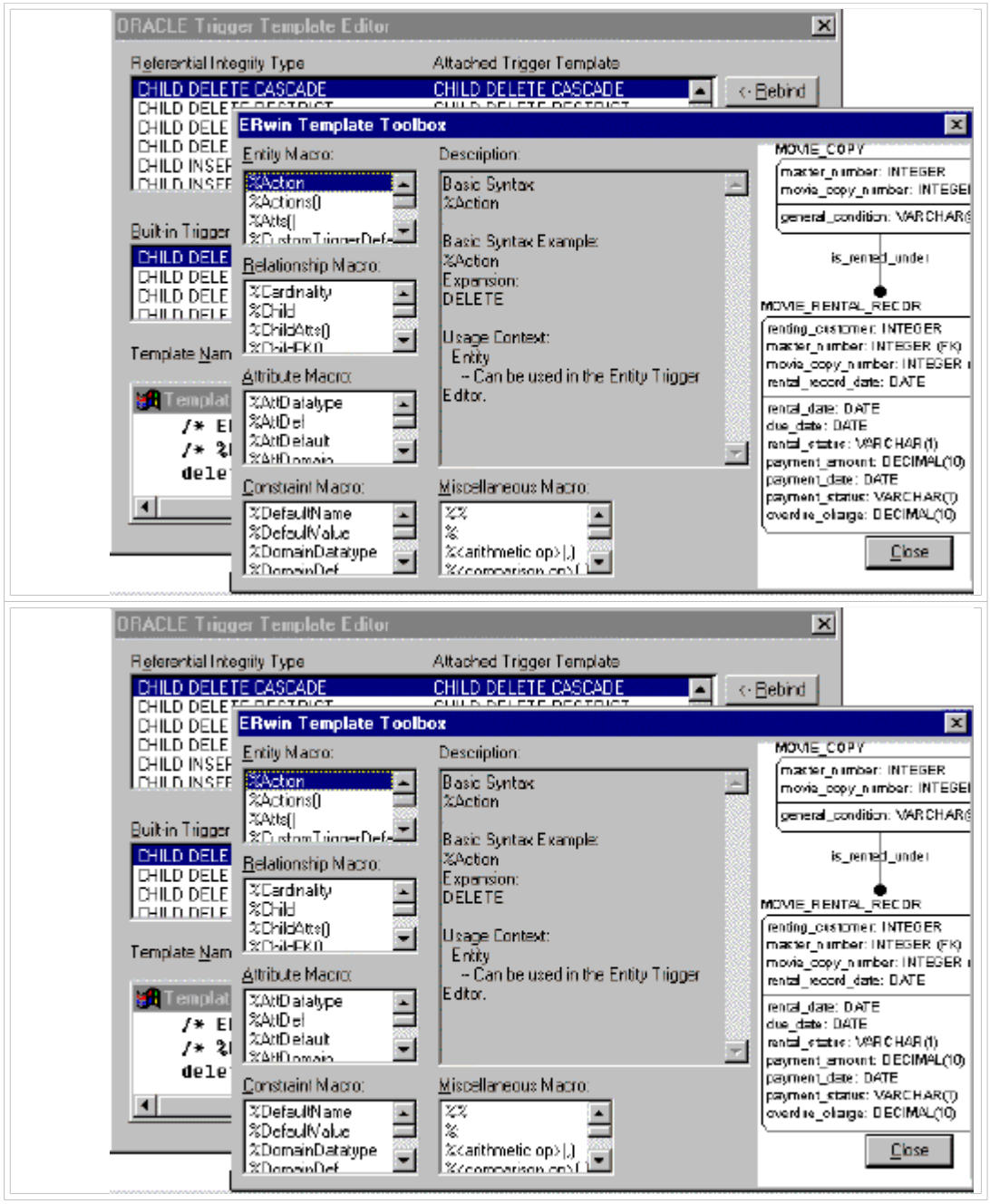


Рис. 20. Диалоги Trigger Template Editor и ERwin Template Toolbox.

Физический уровень представления модели зависит от выбранного сервера (меню Server/ Target Server..). На физическом уровне модель данных необходимо дополнить такой информацией как учет ограничений ссылочной целостности, хранимые процедуры, триггеры, индексы. Триггеры и хранимые процедуры представляют собой программный код и хранятся на сервере. ERwin обеспечивает мощный инструментарий для создания триггеров: шаблоны и библиотеки макросов. Макросы содержат наиболее часто используемые данные и конструкции. Для редактирования шаблонов триггеров используется редактор Trigger Template Editor (для его вызова следует кликнуть правой кнопкой по таблице и выбрать пункт Trigger в появившемся меню).

По умолчанию ERwin генерирует триггеры, дублирующие декларативную ссылочную целостность (опцию можно отменить).

После завершения проектирования модель может быть перенесена в среду целевой СУБД-сервера. Для этого нужно выбрать в главном меню Tasks / Forward Engineer. Можно либо сгенерировать схему БД, либо скрипт на диалекте SQL, соответствующем заранее выбранному серверу. Возможна обратная задача - по существующей схеме БД сгенерировать графическую модель данных. Возможно также выравнивание схемы БД с моделью данных. Для этого следует использовать соответствующую кнопку в панели инструментов (см. таблицу 1). В процессе выравнивания появляется диалог, в котором предлагается указать объекты БД для переноса в графическую модель и объекты модели для переноса в схему БД.

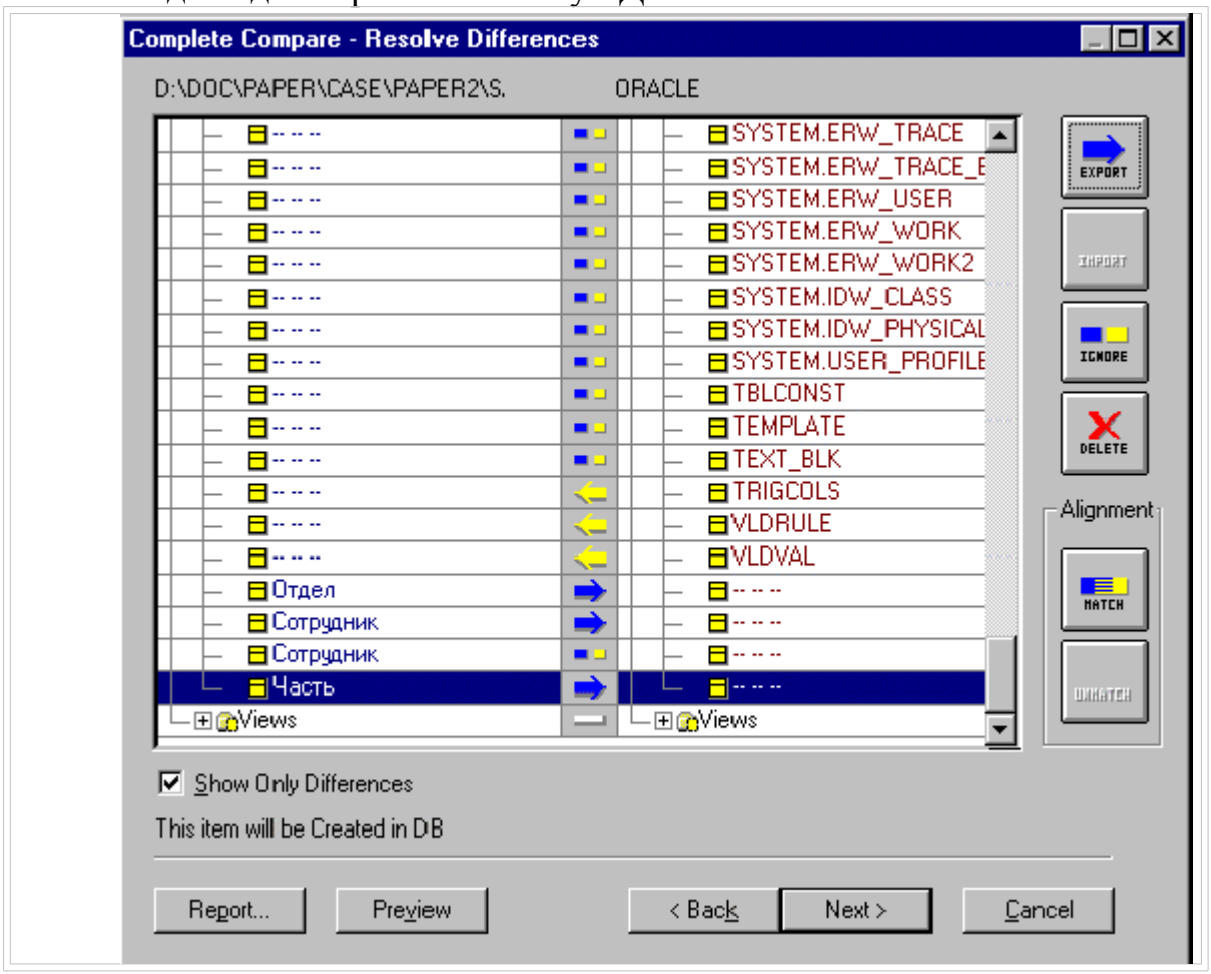


Рис. 21. Диалог Complete Compare- Resolve Differeces.

Для генерации клиентской части во-первых, необходимо выбрать язык программирования (меню Client/ Target Client..., можно выбрать Visual Basic либо Power Builder; далее в примере используется Visual Basic), затем на физическом уровне для каждой колонки необходимо указать свойства - тип визуального представления (style), код валидации и начальное значение (для задания свойств следует кликнуть по таблице правой кнопкой мыши, в контекстном меню выбрать VB Extended Att). Каждое из этих свойств можно предварительно описать в соответствующем редакторе (кнопка ...).

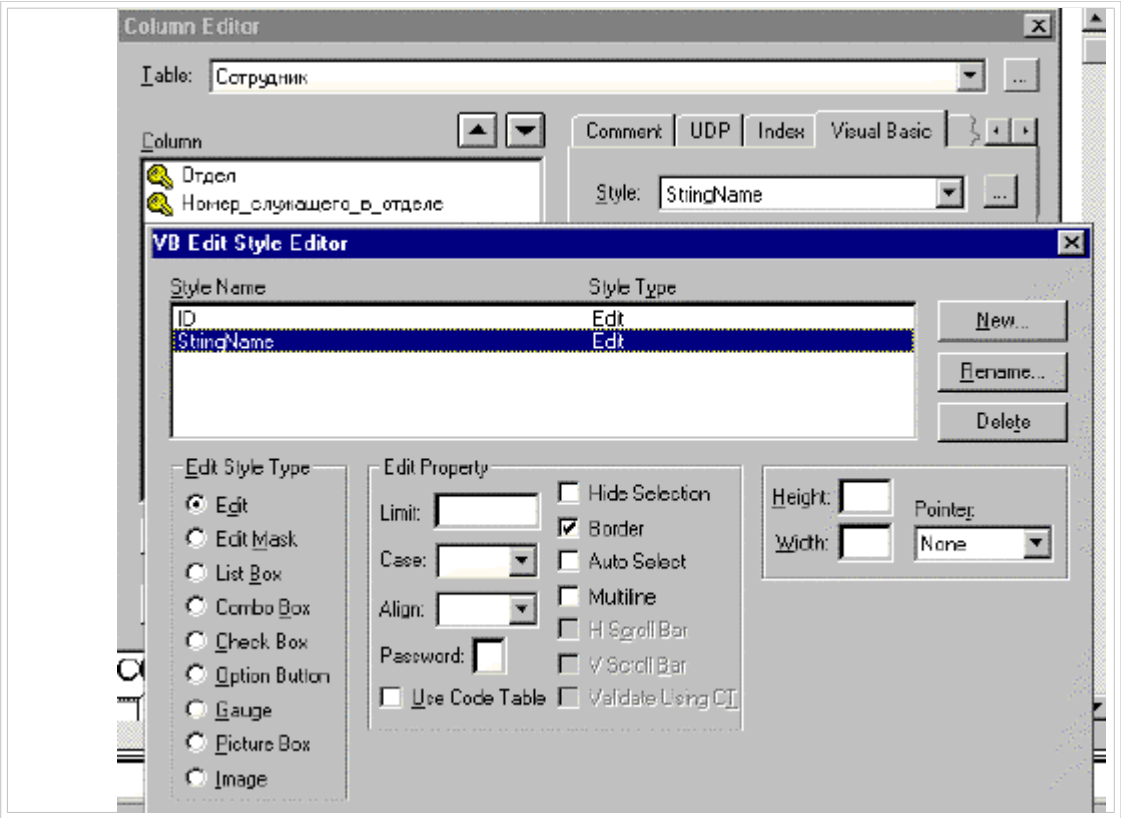
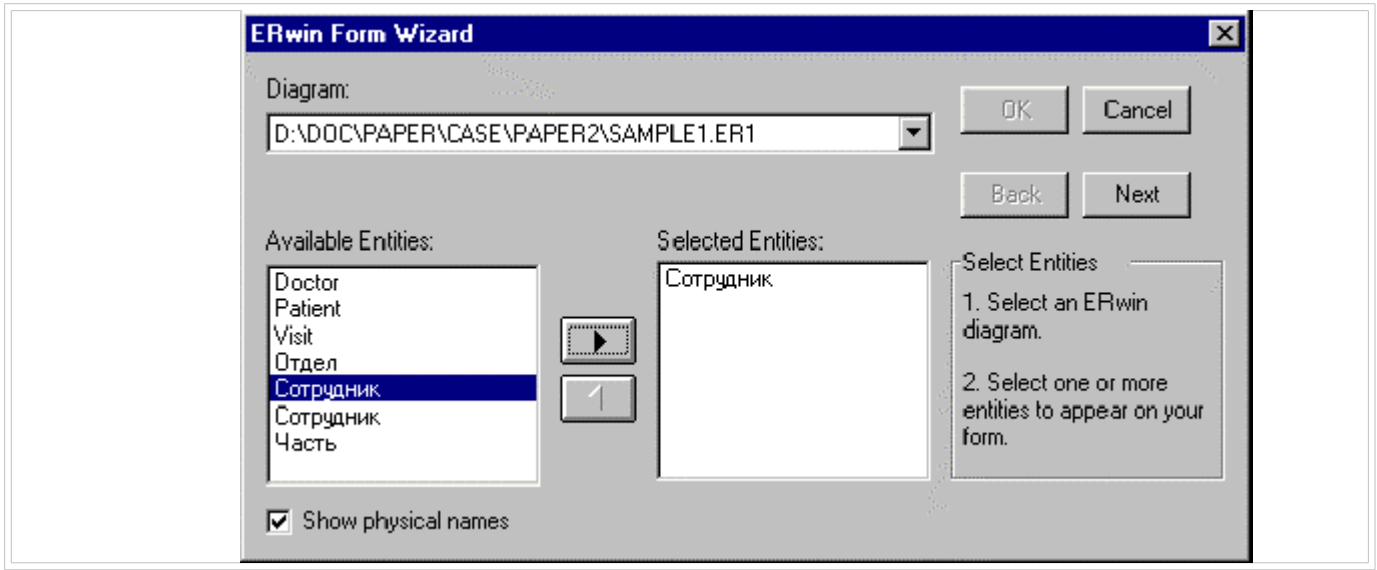
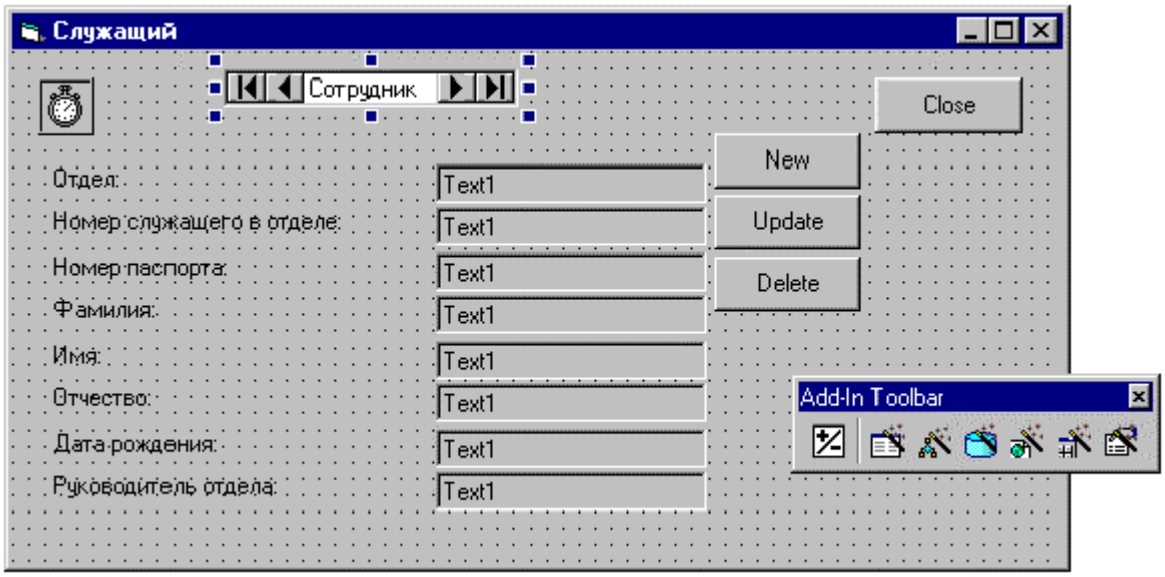


Рис. 22. Редактор стиля.

После задания свойств для каждой колонки следует сохранить модель. Техника генерации приложения зависит от конкретной среды программирования. Рассмотрим в качестве примера MS Visual Basic 5.0. В среде Visual Basic 5.0 следует выбрать в меню Add-Ins / ERwin / Form Wizard. В появившемся диалоге следует указать имя файла модели, таблицы (возможно построение формы по родительской и нескольким дочерним таблицам) и колонки, которые будут отображены в сгенерированной форме. В результате будет сгенерировано приложение, которое может быть откомпилировано и выполнено без дополнительного редактирования.





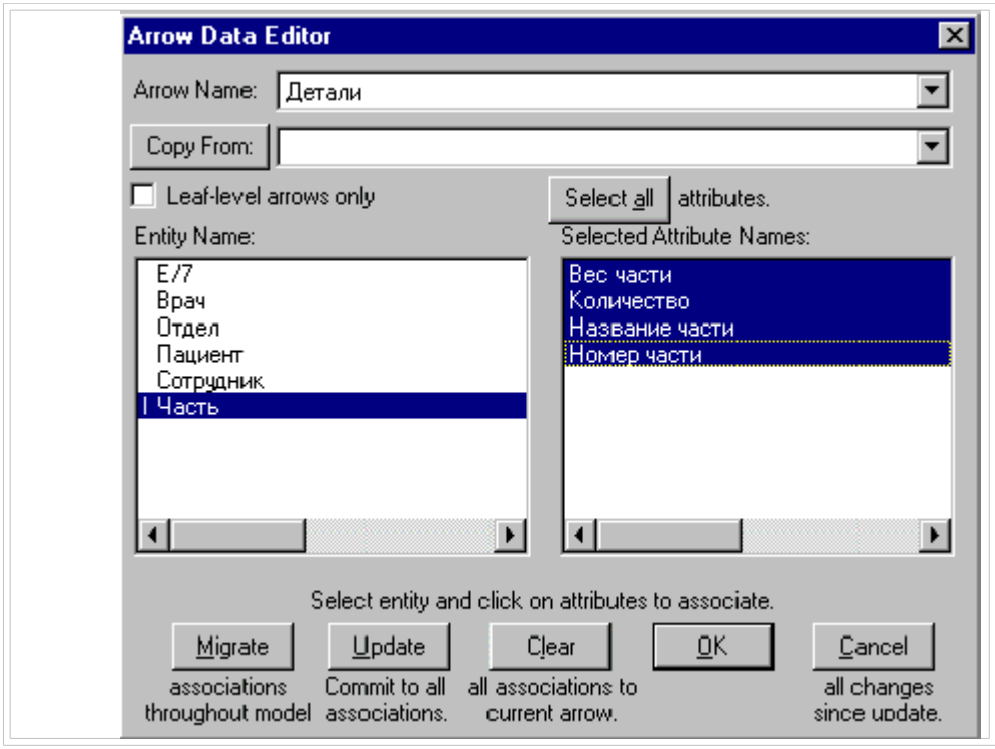
Связывание модели данных и модели процессов

После разработки модели данных ее следует связать с моделью процессов. Такая связь гарантирует завершенность анализа, гарантирует, что есть источник данных (Сущность) для всех потребностей данных (Работа) и позволяет делить данные между единицами и функциями бизнес-процессов.

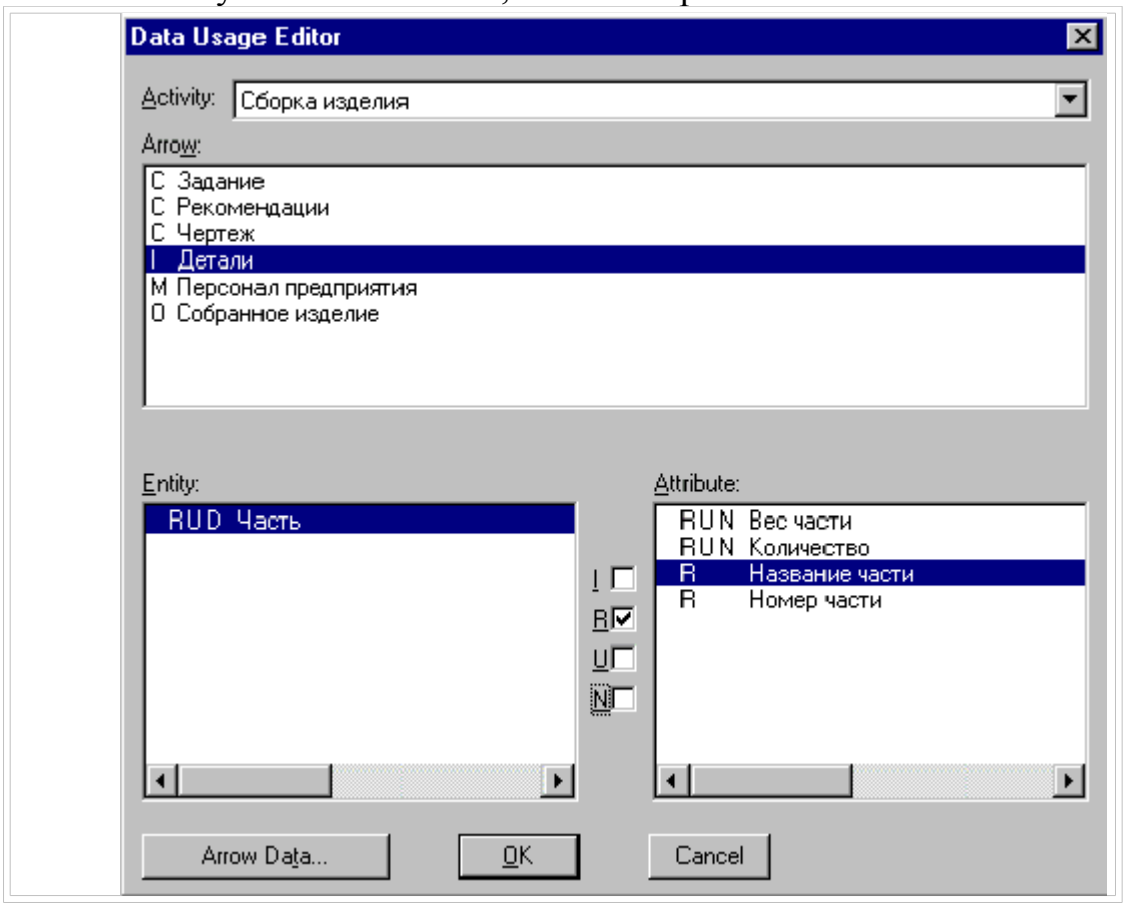
Каждая стрелка в модели процессов может быть связана с несколькими атрибутами различных сущностей. Связи объектов способствуют согласованности, корректности и завершенности анализа.

Для экспорта модели данных из ERwin'a в BPwin необходимо в ERwin'e открыть модель, войти в меню File, выбрать опцию Bpwin / Export, выбрать имя файла *.eax и нажать OK. Появится сообщение "Export Successful".

Затем в BPwin'e нужно открыть желаемую модель процесса, выбрать из меню File / Import / Erwin (EAX)..., выбрать имя файла и нажать OK. Появится протокол импорта. Нужно закрыть диалог протокола и в следующем диалоге кликнуть по кнопке Accept Changes. Теперь можно связать сущности и атрибуты со стрелками. Правой кнопкой нужно кликнуть по стрелке и выбрать в контекстном меню Arrow Data. Появляется диалог Arrow Data Editor.



В нем необходимо указать сущности и атрибут(ы), связанные со стрелкой и кликнуть по кнопке OK, чтобы сохранить изменения.



Если в процессе связывания стрелок с объектами модели данных окажется, что каких либо сущностей или атрибутов не хватает, их можно добавить (меню Edit / Entity/Attribute Dictionary), а затем экспортировать в ERwin

(в BPwin'e меню File / Export / ERwin(BPX), в ERwin'e меню BPwin / Import) . Как было указано выше, работы могут воздействовать на данные. Для документирования такого воздействия необходимо кликнуть правой кнопкой мыши по желаемой работе и выбрать Data Usage Editor.

В появившемся диалоге Data Usage Editor нужно в верхнем списке кликнуть по имени стрелки, с которой были связаны сущности и атрибуты. В нижнем левом окне появится список связанных сущностей. Если выбрать сущность, то, во-первых, в правом окне появится список соответствующих атрибутов, во- вторых, в центре открываются окна выбора CRUD (Create, Retrieve, Update, Delete). Если кликнуть по атрибуту, то значение окон выбора меняется на IRUN (Insert, Retrieve, Update, Nullify). Ассоциации CRUD и IRUN -это правила использования сущностей и атрибутов работами. Данные не могут использоваться работами произвольно. Стрелки входа представляют данные, которые работа преобразовывает в выход или потребляет. Такие данные могут быть восстановлены (Retrieve), обновлены (Update), удалены (Delete), но не могут быть созданы (Create). Стрелки контроля могут быть только восстановлены (Retrieve) и не могут быть изменены. Стрелки выхода могут быть обновлены (если им соответствуют данные стрелок входа) или созданы (Create).

Результат связывания объектов модели процессов можно отобразить в отчете Data Usage Report (меню Report / Data Usage Report).

Arrow Name	Entity Name	C_R_U_D	Attribute Name	I_R_N	U_R
Детали	Часть	R U D	Вес части	R U N	R
		R U D	Количество	R U N	R
		R U D	Название части	R U	R
		R U D	Номер части		R

Групповая разработка моделей данных и моделей процессов с помощью Logic Works Model Mart

ModelMart является системой групповой разработки крупных проектов, которая интегрирует инструментальные средства системных аналитиков и разработчиков БД. Одной из проблем, возникающей при многопользовательской работе с моделями, является разграничение прав доступа. Поскольку ModelMart является специализированным хранилищем моделей, помимо разграничения прав доступа на уровне модели возможно регулирование прав на уровне отдельных элементов модели. Для управления правами доступа в

состав ModelMart включена утилита ModelMart Security Manager. Для начала работы необходимо установить сеанс связи с ModelMart, нажав кнопку  (из дополнительной линейки инструментов ModelMart среды ERWin'a или BPWin'a) и набрать имя и пароль пользователя в появившемся диалоге. После успешного входа становится доступной кнопка  , которая вызывает диалог Security Manager.

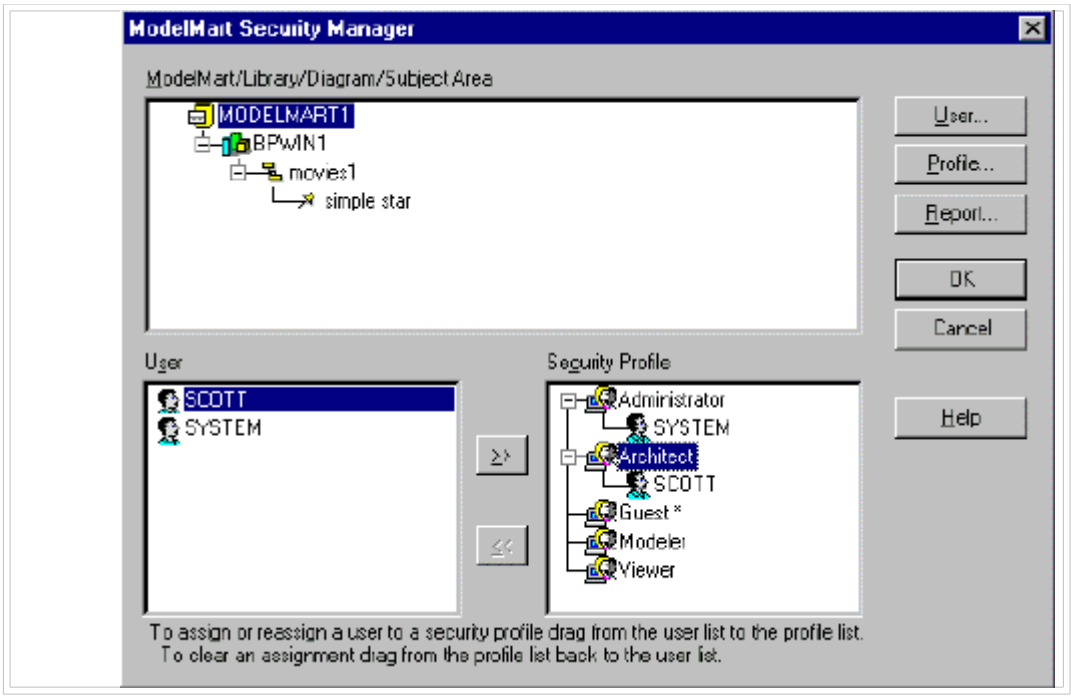


Рис.23. ModelMart Security Manager - диалог формирования групп пользователей.

В окне диалога Security Manager можно задать группу пользователей и права каждой группы (кнопка Profile...) на создание, редактирование и удаление библиотек и моделей. В случае работы с моделями данных в ERWin'e можно также задать права на создание, редактирование и удаление для предметных областей (Subject Area) и сущностей. Нажав на кнопку Users..., можно вызвать диалог внесения новых пользователей ModelMart, которых необходимо предварительно завести как пользователей БД (в примере - Oracle).

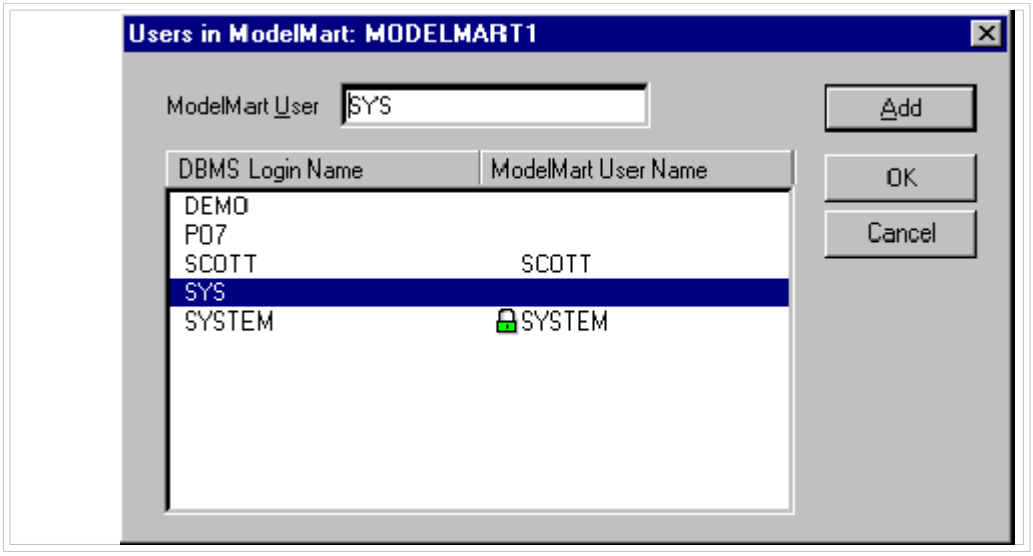


Рис. 24. Users in ModelMart- диалог внесения новых пользователей.

Не все пользователи БД могут быть пользователями ModelMart, но все пользователи ModelMart должны быть пользователями БД.

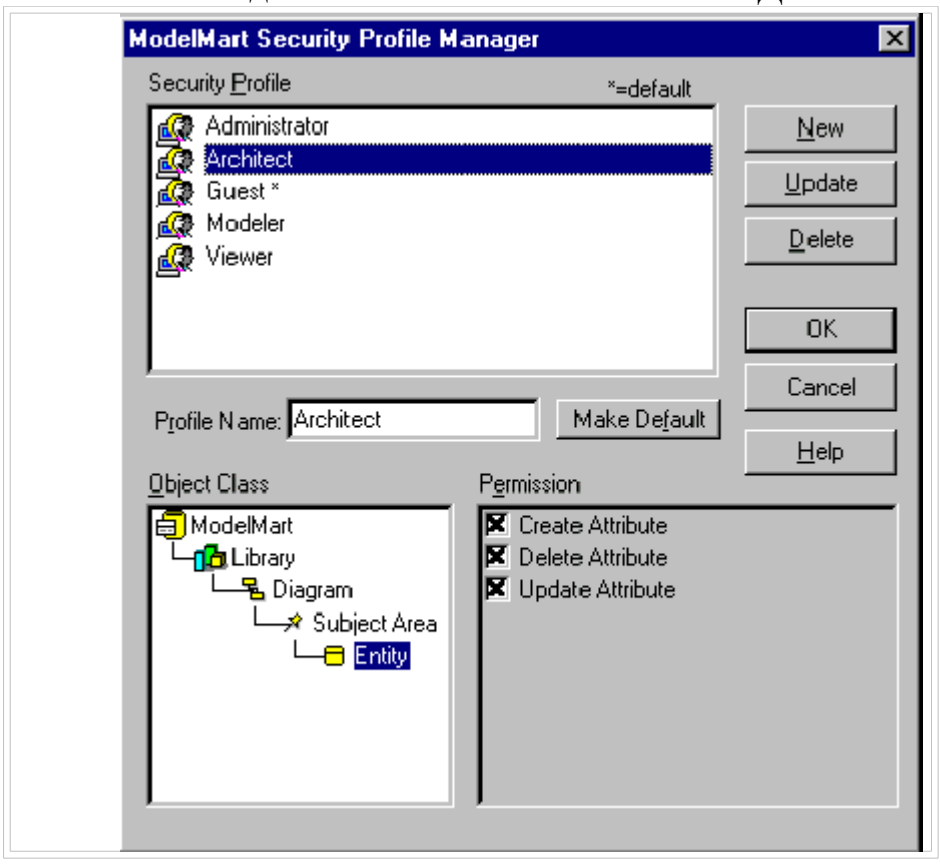


Рис.25. ModelMart Security Profile Manager - диалог задания прав группам пользователей

Затем можно внести пользователей ModelMart в ту или иную группу пользователей ModelMart. На рис.1 показано, что пользователь SYSTEM внесен в группу Administrator, пользователь SCOTT- в группу Architect. Следовательно, если в течение жизненного цикла разработки проекта роль проектировщика меняется, администратор ModelMart может соответственно ме-

нять права доступа без изменения его прав как пользователя БД, что дает возможность гибкого управления проектами.
Если пользователь имеет соответствующие привилегии, он может создать библиотеку моделей ModelMart, нажав кнопку . В состав библиотеки могут входить модели процессов BPWin'a или модели данных и отдельные предметные области моделей данных ERWin'a (диалог создания предметных областей вызывается кнопкой) . Принцип работы с библиотеками моделей является весьма полезным при работе с большим количеством моделей, поскольку можно формировать библиотеки готовых решений (как для моделей процессов, так и для моделей данных) и добавлять в новый проект модели-блоки из заранее сформированных библиотек. Для создания новой модели в ModelMart , добавления, открытия и сохранения модели служат кнопки . Создать или сохранить модель можно только в составе какой-либо библиотеки. При открытии модели возникает диалог Open ModelMart Diagramm, в котором можно указать опции блокировки модели.

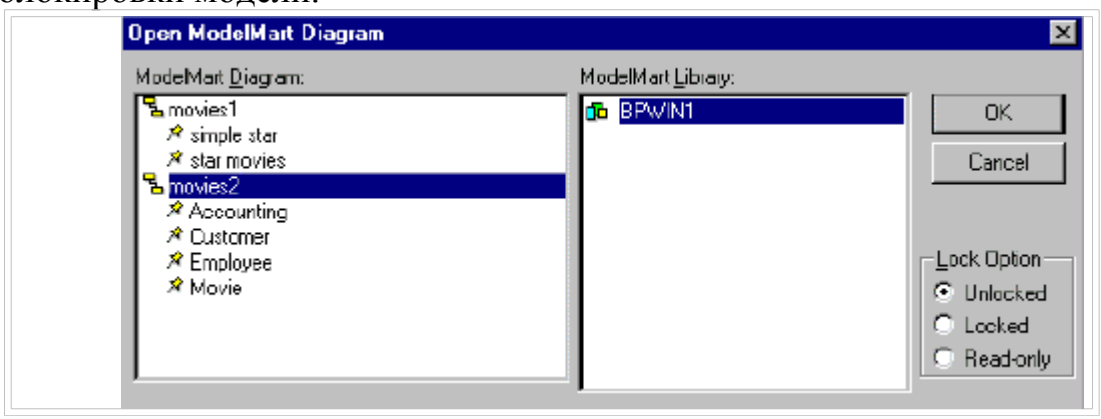


Рис.26. Диалог Open ModelMart Diagramm.

Открытие модели в режиме Read Only означает, что измененную модель нельзя будет сохранить в репозитории. В режиме Locked модель блокируется и другие пользователи не смогут изменить модель. В режиме Unlocked все пользователи могут открыть и изменить модель, при попытке сохранить модель, измененную и сохраненную другим пользователем во время сеанса работы возникает диалог - Intelligent Conflict Resolution, показывающий различия текущей и имеющейся в репозитории моделей. Открытую модель можно перевести в режим Locked, нажав кнопку . Кнопка вызывает диалог ModelMart Merge Manager, который служит для слияния моделей. В зависимости от настройки, слияние может быть проведено в одну из существующих, либо во вновь создаваемую диаграмму. Обновление загруженной диаграммы можно осуществить кликнув по кнопке

Список изменений, сделанных в поцессе работы с моделью, показывается в диалоге Review Changes (вызывается кнопкой).

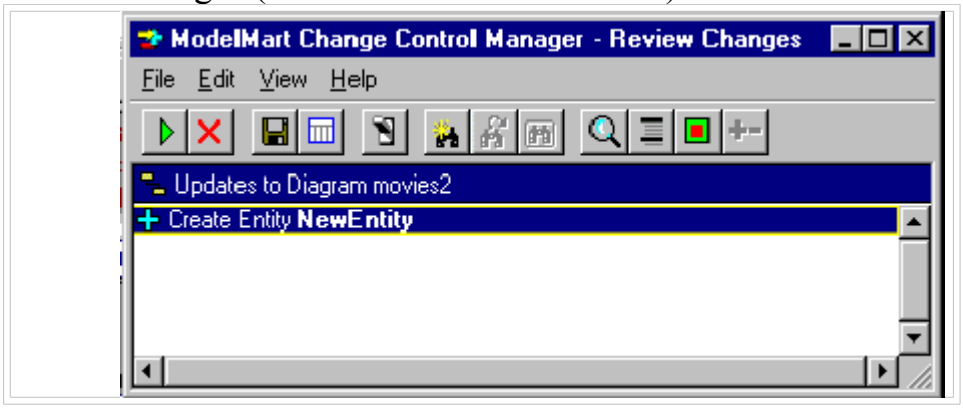


Рис. 27. Диалог Review Changes.

Хранимые в репозитории модели можно сравнивать (кнопка). В диалоге Version Manager следует выбрать сравниваемые версии и кликнуть по кнопке Diff. В появившемся диалоге Version Differences отображается список отличий версий.

В репозитории ModelMart реализована функциональность синхронизации моделей процессов и моделей данных, которая ранее была реализована в ERWin'e и BPWin'e путем экспорта и импорта через файлы .bpx -.eax (см. предыдущую статью). Для синхронизации моделей необходимо кликнуть по кнопке ..

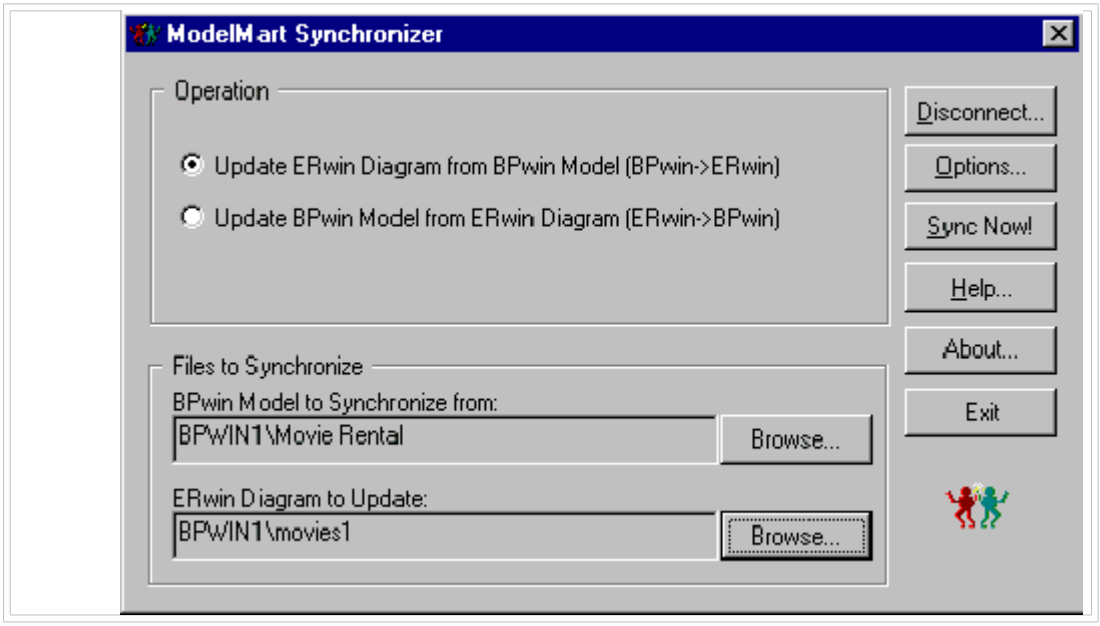


Рис.28. Диалог ModelMart Synchronizer.

В диалоге ModelMart Synchronizer следует указать хранящиеся в репозитории модели процессов и данных, указать направление синхронизации и запустить процесс синхронизации. Затем можно работать с синхронизированными моделями процессов и данных так же, как было выше.

11.МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ОРГАНИЗАЦИИ МЕЖСЕССИОННОГО И ЭКЗАМЕНАЦИОННОГО КОНТРОЛЯ ЗНАНИЙ СТУДЕНТОВ.

1. Межсессионная аттестация студентов проводится дважды в семестр на 7 и 13 неделях 9-го семестра.

2. Аттестационная оценка складывается из оценок, полученных на аттестационных занятиях по лабораторным работам и собеседованиях:

8-й семестр:

2.1. Первое аттестационное занятие. Проверка знаний студентов по основным направлениям развития проектирования Информационных систем.

2.2. Второе аттестационное занятие. Проверка знаний студентов по представлению этапов проектирования ИС, состав работ и проектной документации.

2.3. Первое собеседование. Функциональные и обеспечивающие подсистемы ИС;

2.4. Второе собеседование. Методологические основы проектирования ИС.

9-й семестр:

2.51. Первое аттестационное занятие. Проверка знаний и навыков студентов по проектированию клиент-серверных корпоративных ИС.

2.6. Второе аттестационное занятие. Проверка знаний и навыков студентов по возможностям функционально-ориентированного проектирования ИС.

2.7. Первое собеседование. Объектно-ориентированное проектирование ИС. Прототипное проектирование ИС (RAD-технология);

2.8. Второе собеседование. Типовое проектирование ИС.

3. Организация аттестации студентов, проводится в соответствии с положением АмГУ о курсовых экзаменах и зачетах*.

* 3.1. Организация аттестации студентов в университете по специальностям и направлениям высшего профессионального образования регламентируется рабочим учебным планом, расписанием учебных занятий и программами учебных дисциплин, утверждаемыми в установленном в университете порядке.

Контроль за качеством освоения образовательных программ осуществляется путем текущей внутрисеместровой аттестации, ректорской контрольной аттестации, промежуточной аттестации студентов в форме курсовых экзаменов и зачетов и итоговой аттестации выпускников.

3.2. Курсовые экзамены и зачеты проводятся по дисциплинам утвержденного учебного плана по соответствующим специальностям и направлениям высшего профессионального образования. Знания, умения и навыки обучающегося определяются оценками "отлично", " хорошо", " удовлетворительно", " неудовлетворительно", "зачтено" и "незачтено".

3.3. Студенты, обучающиеся по основным программам высшего профессионального образования, сдают в течение учебного года не более 10 экзаменов и 12 зачетов. В это число не входит аттестация по физической культуре и факультативным дисциплинам.

Студенты, обучающиеся в сокращенные сроки (по индивидуальным планам), в течение учебного года сдают не более 20 экзаменов и 24 зачетов.

3.4. Сроки проведения курсовых зачетов и экзаменов (экзаменационная сессия) и начало очередного учебного семестра устанавливаются графиком учебного процесса, утвержденным проректором по учебной работе.

Расписание экзаменов составляется в соответствии с графиком учебного процесса, утверждается проректором по учебно-научной работе и доводится до сведения преподавателей и студентов не позднее, чем за две недели до начала сессии. Расписание составляется таким образом, чтобы на подготовку к экзаменам по каждой дисциплине было отведено не менее 3 дней, исключая день предыдущего экзамена. По согласованию с деканами и заведующими соответствующих кафедр отдельные экзамены (зачеты) могут проводиться в течение семестра по завершении преподавания дисциплины.

- 12. КОМПЛЕКТЫ ЗАДАНИЙ ДЛЯ ЛАБОРАТОРНЫХ РАБОТ

Контрольные вопросы к лабораторной работе № 1.

1. Для чего предназначена система «Служба занятости» и из каких подсистем она состоит?

2. Укажите цели системы.

3. Изобразите модульную структуру всей системы.

4. Перечислите основные информационные объекты системы «Служба занятости».

5.Опишите порядок занесения студента в БД.

6. Перечислите функциональные характеристики системы.

7. Какие данные поступают на вход подсистемы контроля успеваемости студентов?

8. Перечислите внешние сущности подсистемы профессиональных и психологических тестов.

9. Назовите категории пользователей системы и предоставляемые им возможности.

10. Каково назначение подсистемы обработки запросов, определения категории пользователей?

11.Опишите критерии экспертных оценок.

12.Как формируется временная БД и из чего она состоит?

Задания и контрольные вопросы к лабораторной работе № 2.

1. Создать новый проект в BPWin.

2. Сформировать контекстную диаграмму по системе согласно методологии IDEF0.
3. Задать входы, выходы, механизмы и управление.
4. Декомпозировать контекстную диаграмму.
5. Провести связи по выходу.
6. Провести связи по управлению.
7. Провести связи по входу.
8. Сохранить проект в отдельный файл.

1. Что представляет собой модель в нотации IDEF0?
2. Что обозначают работы в IDEF0?
3. Назовите порядок наименования работ?
4. Какое количество работ должно присутствовать на одной диаграмме?
5. Что называется порядком доминирования?
6. Как располагаются работы по принципу доминирования?
7. Каково назначение сторон прямоугольников на диаграмме?
8. Перечислите типы стрелок?
9. Назовите виды взаимосвязей?
10. Что называется граничными стрелками?
11. Объясните принцип именования разветвляющихся и сливающихся стрелок?
12. Какие методологии поддерживаются BPWin?
13. Как провести связь между работами?
14. Опишите процесс декомпозиции работы?
15. Может ли модель BPWin содержать диаграммы нескольких методологий?

Задания и контрольные вопросы к лабораторной работе № 3

1. Задать атрибуты сущностей, созданных в предыдущей лабораторной работе.
2. Определить первичные ключи в сущностях.
3. Определить состав альтернативных ключей.
4. Связать сущности между собой, используя описанные типы связей.
5. После проведения связей определить состав внешних ключей.
6. Проверить модель на соответствие предметной области.
7. Сохранить полученную диаграмму.

1. Основные части ERD диаграмм?
2. Цель ERD диаграмм?
3. Что является основным компонентом реляционных БД?
4. Что называется сущностью?

5. Сформулируйте принцип именования сущностей.
6. Что показывает взаимосвязь между сущностями?
7. Назовите типы логических взаимосвязей.
8. Опишите механизм проверки адекватности логической модели.
9. Что называется инверсионным входом?
10. Что называется альтернативным ключом?
11. В каком случае образуются внешние ключи?

13. ФОНД ТЕСТОВЫХ И КОНТРОЛЬНЫХ ЗАДАНИЙ ДЛЯ ОЦЕНКИ КАЧЕСТВА ЗНАНИЙ ПО ДИСЦИПЛИНЕ.

ТЕСТЫ

по дисциплине «Проектирование АСОИУ»

1. Какой компонент совокупности элементов системы, объединенных для выполнения определенных функций или задач, является обязательным:
   a) понятия и способы отображения, хранения и передачи информации;
   b) алгоритмические и программные составляющие системы;
   c) математическое обеспечение, реализуемое в алгоритмах и программах;
   d) аппаратурная часть;
   e) системы и приборы отображения и представления информации;
   f) производственная среда;
   g) человек, оператор.

2. Какие средства описывают АСОИУ как систему:
   a) Аппаратные.
   b) Математические.
   c) Телекоммуникационные.
   d) Средства сбора информации.
   a) Средства отображения информации.
   b) Средства хранения информации.
3. Какой уровень представления информации является предметным:
   a) Концептуальный.
   b) Логический.
   c) Физический.
4. Для какой группы основные технические требования систем обусловлены спецификой объекта и специфицируются в техническом задании:
   a) Время протекания процесса совпадает со временем отображения процесса.
   b) Время протекания процесса больше времени отображения.
   c) Время протекания процесса меньше времени отображения.
   d) Время протекания процесса примерно равно времени отображения.
5. На этапе формирования требований к автоматизированной системе выполняется:

a) Оценка операционных систем, пакетов прикладных программ (СУБД и др.) и средства автоматизации разработки АС (CASE-средств).
b) Проектирование структуры базы данных АС.
c) Обследование объекта автоматизации, определение целей, идей и потребностей в новой или модифицированной АС, формализации функций АС, характеристик внешней среды.

6. Для описания диаграмм потоков данных используются символы, обозначающие:
a) Процессы, потоки данных, хранилища, внешние сущности, объединение и расщепление потоков данных.
b) Процессы, потоки данных, хранилища, сущности концептуальной схемы базы данных, объединение и расщепление потоков данных.
c) Модулей программ, данных программ, без данных, объединение и расщепление исходных данных программ.

7. Ядро СУБД дублируется на каждой рабочей станции в модели:
a) Сервера базы данных.
b) Сервера приложений.
c) Файлового сервера.

8. ТЗ на автоматизированную систему разрабатывается:
a) По ГОСТ 34.602-89.
b) По ГОСТ 19.201-78.
c) Произвольным образом.

9. С помощью логической БД описывается:
a) DDL – сценарий.
b) Дополнительные индексы.
c) Диаграмма «сущность-связь».

10. Максимальное расстояние между двумя наиболее удаленными станциями сети Ethernet на 10 Мбит/с равно 2.5 км. Это объясняется
a) Ограниченной надежностью коммуникационного оборудования.
b) Необходимостью гарантированного обнаружения коллизий.
c) Затуханием сигнала, распространяемого по сети.

11. На диаграммах “Сущность-связь” связи изображаются:
a) Не изображаются
b) Линиями
c) Прямоугольниками
d) Овалами

13. Функциональные диаграммы могут изображаться в нотации: (b)
a) DFD
b) IDEF0
c) IDEF1X
d) IDEF2

14. Диаграммы потоков данных могут изображаться в нотации:
a) DFD
b) IDEF0
c) IDEF1X

d) IDEF2

15. Диаграммы сущность-связь могут изображаться в нотации:
    a) DFD
    b) IDEF0
    c) IDEF1X
    d) IDEF2

16. Репозиторий проекта АС
    a) Обеспечивает ведение версий и взаимодействие между группами программистов.
    b) Поддерживает работу распределенной системы.
    c) Является центром выдачи сертификатов.

17. По какому из приведенных типов атрибуты ***не могут*** классифицироваться:
    a) описательные;
    b) связующие;
    c) указывающие;
    d) вспомогательные.

18. Отдельный реальный, гипотетический или абстрактный мир, населенный отчетливым набором объектов, которые ведут себя в соответствии с характерными для него правилами и линиями поведения, это
    a) Множество;
    b) Сущность;
    c) Домен;
    d) Класс.

19. Домен, который обеспечивает общие механизмы и сервисные функции, необходимые для поддержки прикладного домена, это
    a) Домен механизмов;
    b) Сервисный домен;
    c) Архитектурный домен;
    d) Домены реализации

20. Предметная область системы с точки зрения конечного пользователя системы, это:
    a) Прикладной домен;
    b) Сервисный домен;
    c) Архитектурный домен;
    d) Домены реализации

21. Домен, включающий в себя языки программирования, сети, операционные системы и общие библиотеки классов и обеспечивающий концептуальные сущности, в которых будет реализована вся система, это
    a) Домен механизмов;
    b) Сервисный домен;
    c) Архитектурный домен;
    d) Домены реализации.

22. Домен, который обеспечивает общие механизмы и структуры для управления данными и управления системой как единым целым, это:

a) Домен механизмов;
b) Сервисный домен;
c) Архитектурный домен;
d) Домены реализации

23. В ООА справедлива следующая цепочка декомпозиции задачи:
a) Задача – объект – процесс – действие;
b) Задача – процесс – объект – действие;
c) Задача – процесс – действие – объект;
d) Задача – объект – действие – процесс;

24. В ООА при формализации связи один-к-одному вспомогательные атрибуты могут быть добавлены:
a) к первому объекту
b) ко второму объекту
c) к обоим объектам вместе
d) к любому объекту (но не к обоим)

25. В ООА при формализации связи один-ко-многим вспомогательные атрибуты должны быть:
a) добавлены к объекту на стороне "один"
b) добавлены к объекту на стороне "много"
c) добавлены к обоим объектам
d) не должны добавляться

26. В диаграмме переходов в состояние переход обозначается:
a) прямоугольником
b) овалом
c) стрелкой
d) надписью

27. Что из ниже перечисленного не может включаться в диаграммы потоков данных:
a) таймер,
b) внешняя сущность,
c) процессы,
d) накопители данных

28. Определяет информацию, передаваемую через некоторое соединение от источника к приемнику (в ДПД):
a) внешняя сущность
b) процесс
c) накопитель данных
d) поток данных

29. Преобразование входных потоков в выходные в соответствии с определенным алгоритмом (в ДПД):
a) внешняя сущность
b) процесс
c) накопитель данных
d) поток данных

30. Абстрактное устройство для хранения информации (в ДПД):

a) внешняя сущность
b) процесс
c) накопитель данных
d) поток данных

31. Материальный предмет или физическое лицо, представляющие собой источник и приемник информации (в ДПД):

a) внешняя сущность
b) процесс
c) накопитель данных
d) поток данных

32. Чем характеризуется информационная переменная:

a) наименованием, значением и обозначением
b) множеством допустимых значений
c) наименованием переменной
d) перечнем ее основных характеристик

№ пп в билете	Вариант 1		Вариант 2		Вариант3		Вариант4	
	№ из списка	Правильный ответ	№ из списка	Правильный ответ	№ из списка	Правильный ответ	№ из списка	Правильный ответ
1	1	d	2	b	3	b	4	b
2	5	a	6	d	7	c	8	b
3	9	d	10	c	11	c	12	b
4	13	b	14	a	15	c	16	c
5	17	b	18	c	19	b	20	a
6	21	d	22	c	23	d	24	d
7	25	b	26	c	27	a	28	d
8	29	b	30	c	31	a	32	a

На выполнение заданий теста дается 40 минут.

В каждом вопросе за каждый полный ответ – 2 балла.

За неполный ответ – 1 балл.
Максимальное количество набранных баллов – 16.
Критерии оценок:

14 – 16 баллов – «отлично»
12 – 13 баллов – «хорошо»
9 – 11 баллов – «удовлетворительно»
0 – 8 баллов – «неудовлетворительно»

14. КОМПЛЕКТ ЭКЗАМЕНАЦИОННЫХ БИЛЕТОВ

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.
Заведующий кафедрой
Утверждаю: _____

Факультет *М и И*
Курс *5*
Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 1

1. Характеристика процесса проектирования. Базовые понятия.
2. Типы входных сигналов, которые требуют анализа при проектировании единичной нити.

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.
Заведующий кафедрой
Утверждаю: _____

Факультет *М и И*
Курс *5*
Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 2

1. Особенности автоматизации административного управления.
2. Метод большой нагрузки.

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.
Заведующий кафедрой
Утверждаю: _____

Факультет *М и И*
Курс *5*
Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 3

1. Этапы системного анализа.
2. Метод конфликтных ситуаций.

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.
Заведующий кафедрой
Утверждаю: _____

Факультет *М и И*
Курс *5*
Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 4

1. Декомпозиция системы. Цели и задачи.
2. Критерии для выбора альтернатив в условиях неопределенности.

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.
Заведующий кафедрой
Утверждаю: _____

Факультет *М и И*
Курс *5*
Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 5

1. Методологические основы построения АСОИУ.
2. Методы анализа информационных потоков.

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.
Заведующий кафедрой
Утверждаю: _____

Факультет *М и И*
Курс *5*
Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 6

1. Существующие подходы при определении функций управления, подлежащих автоматизации и их постановка.
2. Анализ информационных потоков.

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.
Заведующий кафедрой
Утверждаю: _____

Факультет *М и И*
Курс *5*
Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 7

1. Постановка задачи. Выбор задач автоматизации и их формальная постановка.

2. Общая характеристика процесса проектирования АСОИУ.

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.
Заведующий кафедрой
Утверждаю: _____

Факультет *М и И*
Курс *5*
Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 8

1. Основные принципы проектирования АСОИУ.

2. Разработка функциональной модели АСОИУ.

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.
Заведующий кафедрой
Утверждаю: _____

Факультет *М и И*
Курс *5*
Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 9

1. Уровни типизации при проектировании.
2. Исходные данные для проектирования. Стандарты..

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.
Заведующий кафедрой
Утверждаю: _____

Факультет *М и И*
Курс *5*
Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 10

1. Описание системных функций при проектировании.
2. Проектирование пользовательского интерфейса.

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.
Заведующий кафедрой
Утверждаю: _____

Факультет *М и И*
Курс *5*
Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 11

1. Техническое и программное обеспечение проектирования.
2. Задачи информационного обеспечения, структура информационно-логической модели АСОИУ.

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.
Заведующий кафедрой
Утверждаю: _____

Факультет *М и И*
Курс *5*
Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 12

1. Документация пользователя ППП.
2. Распределенная обработка данных.

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.

Заведующий кафедрой

Утверждаю: _____

Факультет *М и И*

Курс *5*

Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 13

1. Этапы и задачи создания АСОИУ.
2. Структура программных модулей, разработка алгоритмов, логический анализ структур АСОИУ.

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.

Заведующий кафедрой

Утверждаю: _____

Факультет *М и И*

Курс *5*

Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 14

1. Техническое задание на проектирование. Изучение существующей системы.
2. Разработка моделей защиты данных.

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.
Заведующий кафедрой
Утверждаю: _____

Факультет *М и И*
Курс *5*
Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 15

1. Состав и содержание технической документации.
2. Автоматизация процесса проектирования. CASE-средства.

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.
Заведующий кафедрой
Утверждаю: _____

Факультет *М и И*
Курс *5*
Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 16

1. Организация разработки АСОИУ.
2. Основные недостатки внедренных АСОИУ и причины их возникновения.

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.
Заведующий кафедрой
Утверждаю: _____

Факультет *М и И*
Курс *5*
Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 17

1. Проектирование систем на макроуровне. Неформальные этапы разработки систем.

2. Понятие технологии проектирования. Технологическая операция проектирования.

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.
Заведующий кафедрой
Утверждаю: _____

Факультет *М и И*
Курс *5*
Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 18

1. Внешнее проектирование.

2. Технологическая сеть проектирования.

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.
Заведующий кафедрой
Утверждаю: _____

Факультет *М и И*
Курс *5*
Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 19

1. Определение целей и критериев эффективности.

3. Классификация методов проектирования АСУ.

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра *И и УС*

Утверждено на заседании кафедры

“ __ “ _____ 200_ г.
Заведующий кафедрой
Утверждаю: _____

Факультет *М и И*
Курс *5*
Дисциплина *Проектирование АСОиУ*

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 20

1. Внутреннее проектирование. Метод единичной нити.

2. САПР АСОИУ.

- 15. КАРТА КАДРОВОЙ ОБЕСПЕЧЕНОСТИ ДИСЦИПЛИНЫ

Лектор – доцент, канд.техн.наук, Бушманов А.В.
Руководитель лабораторных работ – ассистент Жилиндина О.В.

Бушманов Александр Вениаминович,
к.т.н., доцент.

ПРОЕКТИРОВАНИЕ АСОИУ
Учебно-методический комплекс дисциплины

Изд-во АмГУ. Подписано к печати ???. Формат ???. Усл. печ. л. ???, уч. - изд. л. ???. Тираж 100. Заказ ???.