МИНИСТЕРСТВО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
САМАРСКИЙ ГОСУДАРСТВЕННЫЙ АЭРОКОСМИЧЕСКИЙ
УНИВЕРСИТЕТ имени академика С.П. КОРОЛЕВА

ПРОЕКТИРОВАНИЕ АВТОМАТИЗИРОВАННЫХ ИНФОРМАЦИОННЫХ СИСТЕМ ПО МЕТОДОЛОГИИ SADT

Методические указания к лабораторным работам

САМАРА 2004

Составители: В.П. Дерябкин, А.В. Иващенко

УДК 681.3

Проектирование автоматизированных информационных систем по методологии SADT: Метод. указания к лаб. работам; Самар. гос. аэрокосм. ун-т; Сост. В.П. Дерябкин, А.В. Иващенко – Самара, 2004. 49 с.

Методические указания содержат сведения, необходимые для проведения лабораторного практикума по проектированию автоматизированных информационных систем (АИС). Лабораторный практикум построен как набор методологически связанных работ, посвященных изучению процесса концептуального и логического автоматизированного проектирования систем по методологии SADT с использованием CASE-средств Bpwin (AllFusion Process Modeler) и Erwin (AllFusion Erwin Data Modeler) фирмы Computer Associates, а также с использованием инструментального средства АСПЕКТ, разработанного на кафедре информационных систем и технологий СГАУ. Особое внимание уделяется процессу документирования на всех этапах проектирования.

Предназначены для использования студентами, обучающимися по специальности «Автоматизированные системы обработки информации и управления», специализация «Автоматизированные информационные системы», при изучении дисциплины «Проектирование АИС», а также для специалистов в области разработки информационных систем.

Печатается по решению редакционно-издательского совета Самарского государственного аэрокосмического университета.

Рецензент: д.т.н., проф. А.А. Калентьев

Общие сведения

Проектирование автоматизированных информационных систем (АИС) является сложной задачей, в процессе решения которой приходится рассматривать широкий круг вопросов, связанных с моделированием предметной области, анализом информационных запросов, разработкой схем баз данных и алгоритмов сбора и обработки информации, выбором комплекса технических и программных средств, документированием проекта. В настоящее время проектирование АИС ведется коллективами разработчиков с использованием специальных инструментальных программных систем – CASE-средств (Computer-Aided Software/System Engineering) [1-4, 6, 7]. В качестве теоретического базиса проектирования большинство CASE-технологий используют методы структурного системного анализа, наиболее популярными из которых являются:

- Методология Гейна-Сарсона [6-7], иногда еще называемая DFD-технологией (DFD – Data Flow Diagram, диаграмма потоков данных);
- Методология SADT (Росса) [1], зафиксированная международным стандартом IDEF0;
- Методология UML (Буча, Рамбо и Якобсона) [8], зафиксированная международным стандартом UML 1.4.

Целью данного лабораторного практикума, являющегося естественным продолжением и развитием лабораторного практикума по курсу «Проектирование АСОИУ», является освоение методологии SADT и соответствующих CASE- средств, поддерживающих указанную методологию.

Структурный системный подход к разработке информационных систем, особенно систем с централизованным управлением данными, заключается, в основном, в моделировании и всестороннем анализе требований к системе. При необходимости модели процессов (диаграммы потоков данных, концептуальные и логические модели данных, описания логики процессов) могут быть созданы обычными графическими и текстовыми редакторами для документирования проекта. На разных стадиях разработки используются различные уровни детализации этих моделей.

Настоящие указания представляют собой описание лабораторных работ по курсу «Проектирование АИС», изучаемому студентами специализации 220203 АИС специальности 220200 АСОИУ в Самарском государственном аэрокосмическом университете. При разработке этого практикума ставились две задачи: с одной стороны, познакомить студентов с CASE-средствами, архитектурой и основными концепциями проектирования, научить студентов работать с интерфейсом пользователя, с другой – привить навыки системного анализа, моделирования данных, проектирования АИС в целом на логическом уровне. В связи с этим в методические указания включены лабораторные работы, содержащие построение учебных моделей по заданному описанию предметной области, позволяющие осуществить полный цикл работ по логическому проектированию АИС, достаточному для последующей физической реализации системы.

Выполнение каждой работы рассчитано на 4 академических учебных часа. Полный семестровый цикл работ включает 8 лабораторных работ. Оформление итогового семестрового отчета и реализация проекта производится студентами в основном во внеаудиторное время, отведенное для самостоятельной работы по изучению дисциплины. На первом занятии студенты получают задание на создание учебного проекта АИС в виде описания предметной области. Предполагается, что студенты работают над проектом либо индивидуально, либо небольшими бригадами (не более 3 человек). В последнем случае описание предметной области может быть общим для бригады, однако каждый студент должен иметь не менее трех индивидуальных информационных запросов.

Уровень проработки проекта в лабораторном практикуме по каждой методологии должен быть достаточен для последующего выбора комплекса технических средств и разработки программного и информационного обеспечения.

Методические указания могут быть использованы при проведении лабораторных работ, а также и при самостоятельном изучении дисциплины студентами заочной формы обучения.

Лабораторная работа № 1

Построение контекстной диаграммы

Цель работы: Определение первого этапа моделирования. Изучение особенностей построения диаграммы верхнего уровня. Изучение основных понятий методологии SADT и языка IDEF0, CASE-средства BPWin.

Основные теоретические сведения

Согласно ГОСТ 34.003-90 “АС. Термины и определения” автоматизированная система (automated system, АС) – система, состоящая из персонала и комплекса средств автоматизации его деятельности, реализующая информационную технологию выполнения установленных функций. Таким образом, АС относится к классу человеко-машинных систем, при проектировании которых следует учитывать эргономические требования, т.е. учитывать физические и психологические особенности людей, участвующих в работе системы.

Под моделированием понимается процесс создания достаточно точного описания вновь создаваемой или существующей системы, а также интерпретация полученного описания для определения оценочных значений некоторых характеристик системы.

Общая методология SADT (Structured Analysis and Design Technique, методология Росса) – методология структурного анализа и проектирования - состоит из трех частных методологий моделирования, основанных на графическом представлении системы разными языковыми средствами:

- IDEF0 используется для создания функциональной модели, отображающей структуру и функции системы, а также потоки информации и материальных объектов, связывающих между собой эти функции.
- IDEF1 применяется для построения информационной модели, отображающей структуру и содержание информационных потоков, необходимых для поддержки функций системы.
- IDEF2 позволяет построить динамическую модель изменения во времени функций, информационных потоков и ресурсов системы.

Далее излагается концепция моделирования в терминах языка IDEF0 [2].

Модель (model) – искусственный объект, представляющий собой отображение (образ) системы и ее компонентов. Модель IDEF0 – графическое описание системы, разработанное с определенной целью и с выбранной точки зрения.

Диаграмма (diagram) – часть модели, описывающая декомпозицию функционального блока. *Контекстная диаграмма (context diagram)* представляет контекст модели, самый верхний уровень моделирования. Пример контекстной диаграммы приведен на рис.6.

Каждая модель должна иметь контекстную диаграмму верхнего уровня, на которой объект моделирования представлен единственным блоком с граничными стрелками. Эта диаграмма называется A-0. Стрелки на этой диаграмме отображают связи объекта моделирования с окружающей средой.

Проектирование производится посредством *декомпозиции (decomposition)* – деления моделируемой функции на функции-компоненты. Пример диаграммы декомпозиции приведен на рис.9.

Функция (function) – деятельность, процесс или преобразование, идентифицируемое глаголом или глагольной формой, которая описывает, что должно быть выполнено. Функции на диаграммах отображаются в виде *блоков (box)* – прямоугольников, содержащих имя и номер. Блоки диаграмм связываются стрелками.

Стрелка (arrow) – направленная линия, состоящая из одного или нескольких сегментов, которая моделирует канал, передающий данные от источника (начальная точка стрелки) к потребителю (конечная точка с «наконечником»). Имеется 4 класса стрелок:

- *входная стрелка (input arrow)* – класс стрелок, которые отображают вход функционального блока, то есть данные, которые преобразуются функцией. Входные стрелки связываются с левой стороной блока;
- *выходная стрелка (output arrow)* – класс стрелок, которые отображают выход функционального блока, то есть данные, произведенные функцией. Выходные стрелки связываются с правой стороной блока;

- *управляющая стрелка (control arrow)* – класс стрелок, отображающих управление, то есть условия, при которых выход блока будет правильным. Управляющие стрелки связываются с верхней стороной блока;
- *стрелка механизма (mechanism arrow)* – класс стрелок, которые отображают средства, используемые для выполнения функции. Стрелки механизма связываются с нижней стороной блока.

Контекстная диаграмма A-0 должна содержать краткие утверждения, определяющие точку зрения на модель и цель моделирования.

SADT модель дает полное, точное и адекватное описание системы, имеющей конкретное назначение. Это назначение называется целью модели (purpose) и вытекает из ее формального определения. Формулировка цели выражает причину создания модели, то есть содержит перечень вопросов, на которые должна отвечать модель, что в значительной мере определяет ее структуру.

Точка зрения – указание на должностное лицо, с позиции которого разрабатывается модель. У модели может быть только одна точка зрения. Изменение точки зрения приводит к рассмотрению других аспектов объекта. Аспекты, важные с одной точки зрения, могут не появиться в модели, разрабатываемой с другой точки зрения на этот объект.

Наиболее важные свойства объекта обычно выявляются на верхних уровнях иерархии, по мере декомпозиции функции верхнего уровня и разбиения ее на подфункции эти свойства уточняются. Каждая подфункция декомпозируется на элементы следующего уровня. Декомпозиция происходит до тех пор, пока не будет получена релевантная структура, позволяющая ответить на вопросы, сформулированные в цели моделирования.

Проектирование автоматизированных информационных систем может быть произведено с помощью CASE средства BPwin, поддерживающего нотации IDEF0 (функциональная модель), IDEF3 (WorkFlow Diagram) и DFD (DataFlow Diagram). Функциональная модель предназначена как для описания существующих процессов (AS-IS), так и вновь проектируемых (TO-BE).

Инструментальная среда BPwin (рис. 1) содержит область построения, меню, инструментальную панель и навигатор (левое окно). Пункты контекстного меню Font Editor и Color Editor вызывают соответствующие диалоги для установки шрифта и цвета объекта. Шрифт можно установить по умолчанию для объектов определенного типа на диаграммах и в отчетах, а также для всей модели.

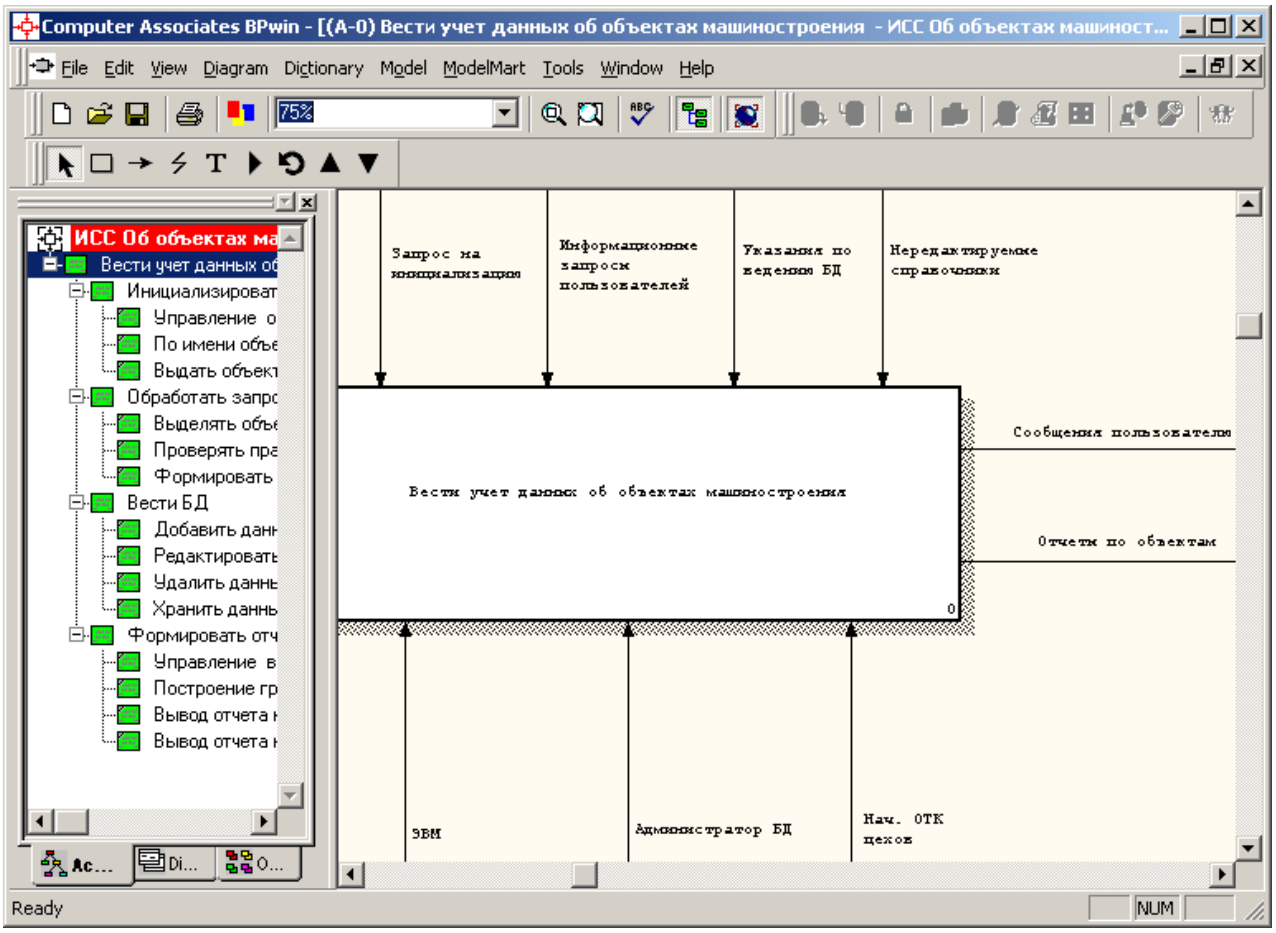

Рис. 1 – Интегрированная среда разработки BPwin

Цель моделирования и точка зрения (рис. 2) задаются с помощью пункта меню Model/Model Properties (закладка Purpose). В закладке Status можно описать статус модели, время создания и последнего редактирования. В закладке Source описываются источники информации для построения модели, закладка General служит для внесения имени проекта, модели, имени и инициалов автора, закладка Page Setup предназначена для задания параметров отображения проекта.

Результат описания модели можно получить в модуле Tools/Reports/Model Report. (рис. 3)

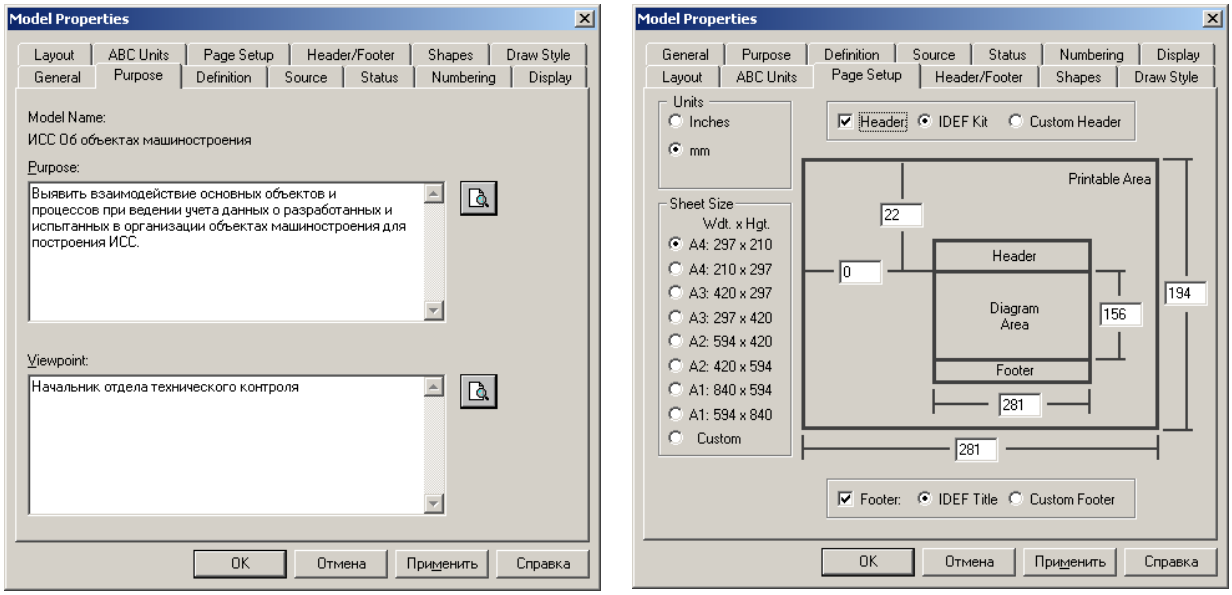

Рис. 2 – Диалог задания свойств модели

Модель может содержать 4 типа диаграмм: контекстную диаграмму, диаграммы декомпозиции, диаграммы дерева узлов, диаграммы только для экспозиции (For Exposition Only, FEO). Выбор элемента диаграммы (блока, стрелки, текста и т.д.) осуществляется с помощью кнопки ↖. Для создания диаграммы

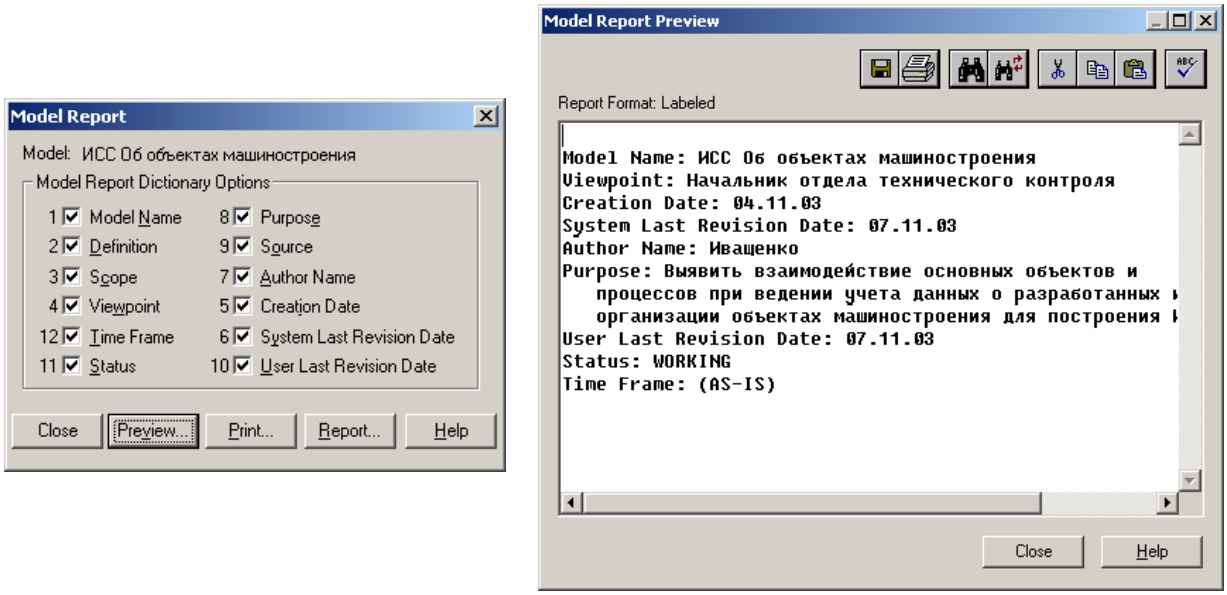

Рис. 3 – Отчет по модели

декомпозиции служит кнопка ▼. Кнопки ▼, ▲ (переход к родительской диаграмме), ↩ (переход к родственной диаграмме) и ▶ (вызов менеджера диаграмм) можно использовать для навигации по диаграммам. При создании новой диаграммы указывается нотация и количество работ в ней. Новый блок может быть добавлен на диаграмму кнопкой ❑.

Стрелки вносятся с помощью кнопки →. При этом после нажатия этой кнопки нужно щелкнуть по работе-источнику стрелки, затем – по работе-приемнику. Для создания граничных стрелок нужно связывать соответствующие стороны блока с границей области построения (при поднесении курсора появляется черная полоса). Для связывания стрелки с меткой используется зигзаг (⚡). Текстовый комментарий можно добавить на диаграмму с помощью кнопки **T**.

Изменение свойств функциональных блоков производится с помощью редактора Model/ Diagram Objects Editor, стрелок – Model/ Arrow Editor (рис. 4).

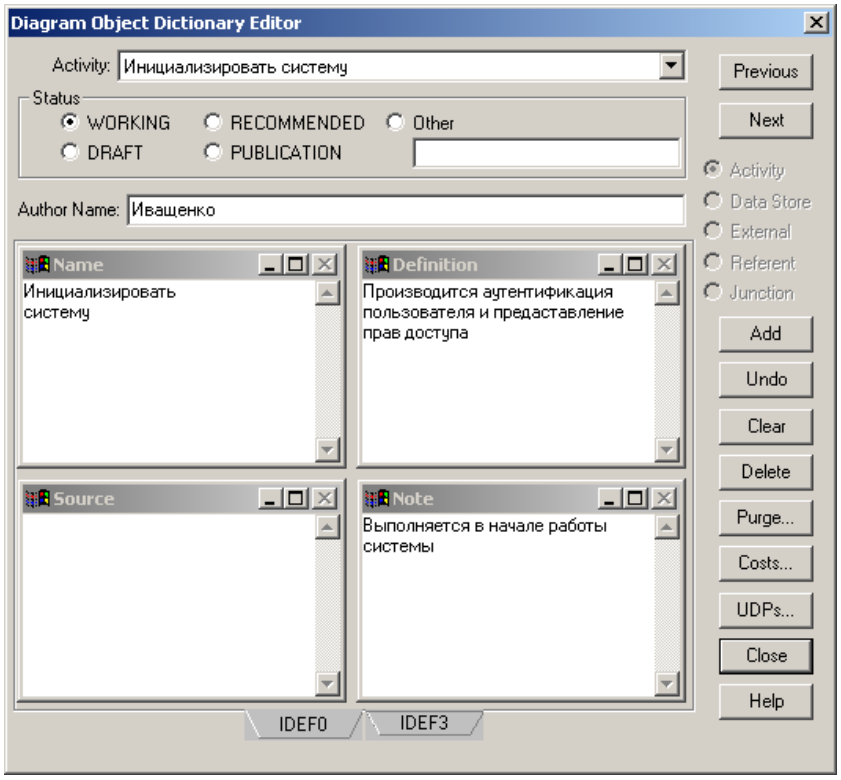

Рис. 4 – Описание функций

Порядок выполнения

1. Получить задание у преподавателя, проанализировать описание предметной области и информационные запросы пользователей.
2. Составить список всех основных объектов (данных), являющихся частью системы.
3. Перечислить все функции, выполняемые системой.
4. Составить множество вопросов, на которые должна отвечать модель. Задать степень точности ответа на эти вопросы.

5. Сформулировать, как будет использоваться модель. Определить цель моделирования.
6. По вопросам и цели определить точку зрения на модель.
7. По составленному списку данных определить дуги, представляющие взаимодействие системы с внешними объектами. Определить выходную информацию, перерабатываемую информацию, управления и механизмы.

Контрольные вопросы

1. В каких случаях в качестве механизма выступают физические лица ?
2. Перечислить правила связи блоков стрелками.
3. Определить, как влияют ошибки в определении точки зрения и цели моделирования на дальнейшее проектирование.
4. Какими соображениями следует руководствоваться при выборе точки зрения?
5. Перечислить средства повышения наглядности диаграмм.

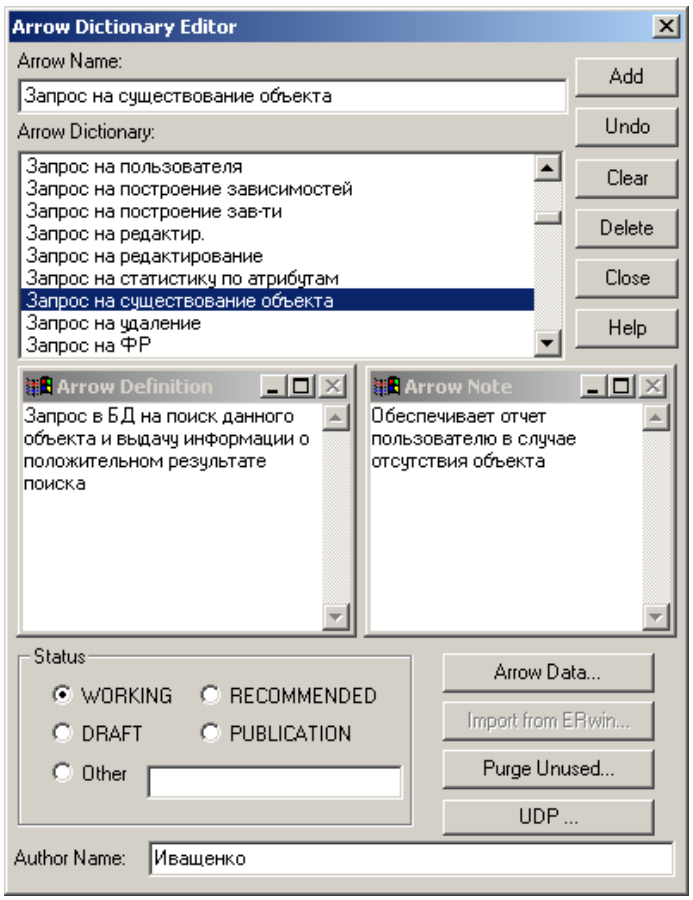

Рис. 5 – Описание стрелок

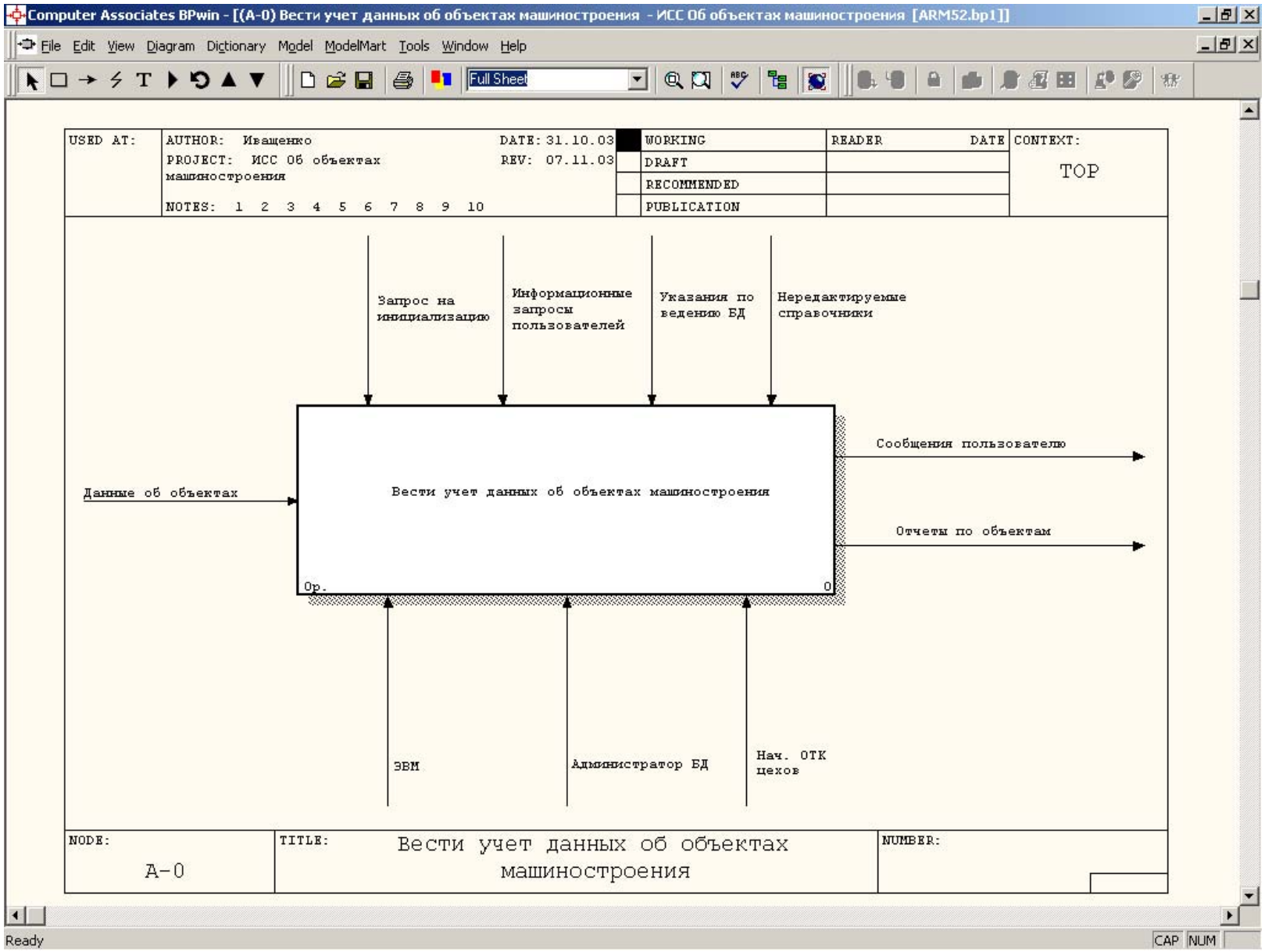

Рис. 6 – Контекстная диаграмма

Лабораторная работа № 2

Построение диаграмм декомпозиции

Цель работы: Построение диаграммы декомпозиции А0. Изучение особенностей разветвления дуг (стрелок). Корректировка диаграмм в связи с их взаимным влиянием. Описание основных процессов информационной системы в методологии SADT.

Основные теоретические сведения

Дочерняя диаграмма (child diagram) – диаграмма, детализирующая родительский (порождающий) функциональный блок.

Каждая SADT диаграмма содержит блоки и дуги. Блоки изображают функции моделируемой системы, дуги (стрелки) отображают взаимодействия и взаимосвязи между ними. Блок представляет функцию или активную часть системы, название блока содержит глагол или глагольный оборот. На каждой диаграмме отображается от трех до шести блоков. Блоки имеют доминирование – они размещаются на диаграмме по степени важности. Самым доминирующим блоком диаграммы может быть либо первый из требуемой последовательности, либо планирующий или управляющий. Наиболее доминирующий блок располагается в левом верхнем углу диаграммы, наименее – в правом нижнем. Функциональные блоки нумеруются с учетом их доминирования.

Между функциями возможны четыре вида отношения: вход, управление, выход, механизм. В методологии SADT требуется только пять типов взаимосвязей между блоками с помощью стрелок для описания их отношений: управление, вход, обратная связь по управлению, обратная связь по входу, выход-механизм (рис. 7).

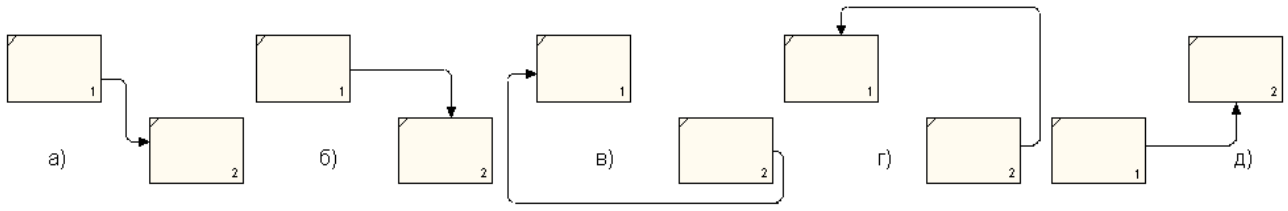

Рис. 7 – Типы взаимосвязи между блоками

Обратная связь по потоку данных между двумя функциями возникает, когда выход одной функции становится входом другой. Обратная связь по управлению возникает, когда выходы двух функций воздействуют друг на друга.

Дуга (стрелка) обычно представляет набор объектов. Поэтому они могут разветвляться и соединяться различными способами. При разветвлении или слиянии все дуги помечаются. Дуги не помечаются, если структуры дуг до и после разветвления (слияния) совпадают.

Различие между входными дугами и дугами управления заключается в том, что SADT различает объекты, преобразуемые системой (входы), и объекты, управляющие преобразованием системы (управления). В ряде случаев входные дуги могут быть обозначены как дуги управления (обратное неверно). Однако акцент на преобразовании входной информации с помощью функции чаще всего необходим.

Сегменты дуг (стрелок), за исключением стрелок вызова, помечаются существительным или оборотом существительного. Чтобы связать стрелку с меткой, следует использовать зигзаг .

Идентификация граничных стрелок осуществляется с помощью ICOM-кодов. Коды ICOM содержат префикс, соответствующий типу стрелки (Input, Control, Output, Mechanism). BP-Win вносит ICOM-коды автоматически. Для их отображения необходимо включить опцию "ICOM codes" на закладке Display диалога Model/ Model Properties.

Расположение блоков и стрелок на диаграмме должно быть таким, чтобы обеспечить хорошую читаемость диаграммы и исключить двойственность интерпретации. Диаграммы описывают модель, поэтому должны быть не только полными, но и понятными. В связи с этим допускается не указывать обозначение сегментов дуг (стрелок) в случае, когда структура данных не меняется, и использовать тоннели.

Для избежания неоднозначных трактовок терминов предметной области в системе присутствуют словари функций (Activity), стрелок (Arrow), данных (Data Store), внешних ссылок (External Reference) и проч. Они могут редактироваться с помощью пунктов меню Dictionary (рис. 8).

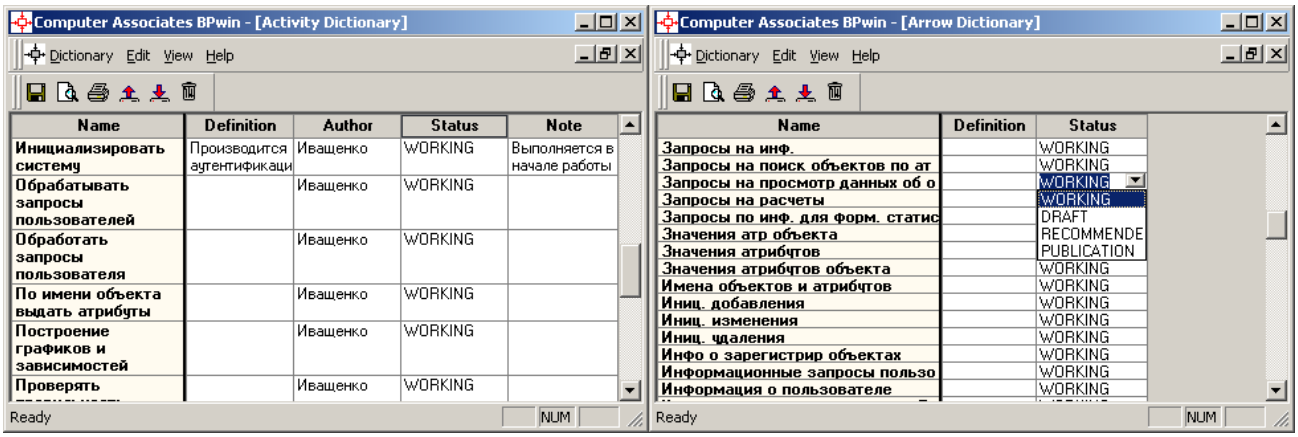

Рис. 8 – Словари функций и стрелок

Проектирование автоматизированной информационной системы имеет ряд особенностей. Они связаны с наличием в большинстве информационных систем базы данных, системы аутентификации (идентификации пользователя), системы взаимодействия с пользователем посредством интерфейсов или сообщений. Кроме этого, подавляющее большинство информационных систем предназначено для формирования отчетов по информационным запросам пользователей.

Поэтому рекомендуется включать в модель автоматизированной информационной системы функциональные блоки «Идентифицировать пользователя», «Вести БД» и «Формировать отчеты».

Функциональный блок «Идентифицировать пользователя» является доминирующим, так как определяет действие остальных функций и выполняется в первую очередь.

По информационным запросам пользователя (управление), поступающим в блок "Обработать запросы пользователя", формируется параметрический запрос в БД (управление), поступающий в блок "Вести БД". Сформированные данные для отчета поступают на вход блока "Формировать отчеты", а информация о зарегистрированных объектах на управление блока "Обработать запросы пользователя", на основе которой формируется указание по формированию отчетов для блока "Формировать отчеты". Отчеты по объектам и сообщения о результате запроса являются сообщениями пользователю.

Взаимодействие с базой данных обеспечивается следующим образом. При допуске пользователя (управление) на основе указаний по ведению БД (управление) происходит обработка данных об объектах (вход) в функциональном блоке «Вести БД». При этом формируется сообщение о результате проведенной операции.

Последовательность приведенных выше описаний проследите на рис.9 и используйте при решении своей задачи. Данные, поступающие в систему, должны перерабатываться под указанным управлением и выдаваться в качестве выходных данных. При этом действие любых функций должно приводить к какому-либо результату.

Порядок выполнения

1. Из списка функций получить посредством объединения 3-6 основных функций и расположить их в порядке выполнения с учетом доминирования (степени важности для выполнения данной функции) .
2. Используя составленный список данных, нарисовать и пометить внутренние дуги.
3. Изобразить основной поток данных, прокладывая путь от блока к блоку. Модель должна быть хорошей иллюстрацией, наглядной презентацией объекта. Дуги механизмов можно определить в самом конце построения диаграммы.
4. Учесть все исправления и привести в соответствие диаграммы А-0 и А0.
5. Описать в отчете все потоки данных, присутствующие в модели.

Контрольные вопросы

1. Пояснить и проиллюстрировать пять типов взаимосвязи между блоками.
2. Дать определение ICOM-кодам.
3. Перечислить правила обозначения функциональных блоков и информационных дуг (стрелок). Определить случаи использования «Тоннелей».
4. Какое отношение между блоками определяется без использования стрелок?
5. Перечислить правила обозначения диаграмм.

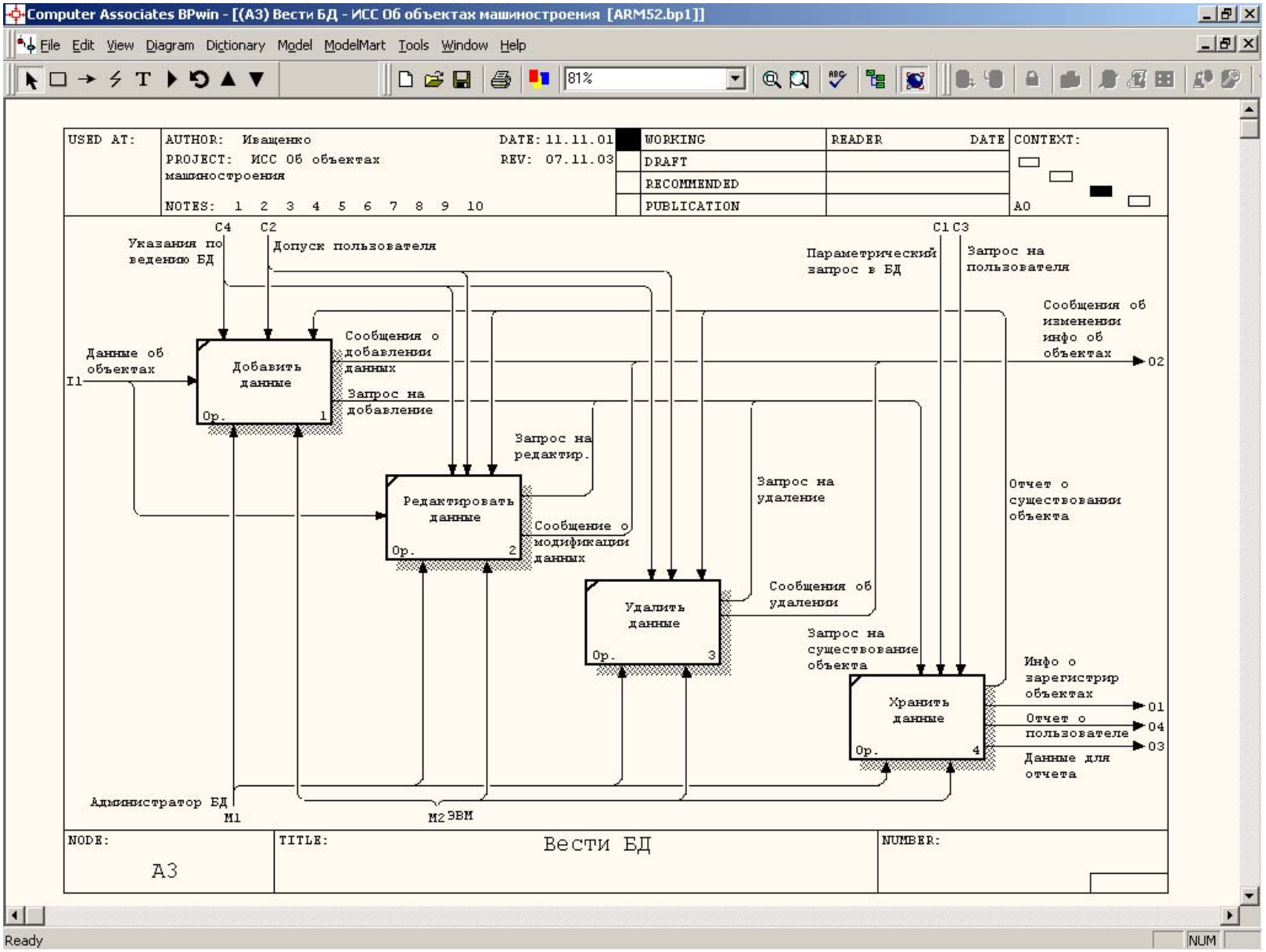

Рис. 9 – Диаграмма декомпозиции

Лабораторная работа № 3

Декомпозиция функциональных блоков

Цель работы: Изучение основных подходов к проектированию автоматизированных информационных систем. Изучение методики разработки функциональных моделей в среде IDEF0.

Основные теоретические сведения

Разработанная модель IDEF0 со всеми уровнями структурной декомпозиции может быть представлена в виде дерева узлов. Это дерево не обязательно сбалансировано.

Каждому блоку на диаграмме присваивается номер, помещаемый в нижнем правом внутреннем углу блока. Узловой номер базируется на положении блока в иерархии модели. Обычно узловой номер формируется добавлением номера блока к номеру диаграммы, на которой он появляется.

Все функциональные блоки на диаграмме являются преобразующими. Преобразующий блок – блок диаграммы, преобразующий входы в выходы под действием управлений при помощи механизмов. Преобразование – цель и результат работы любого блока на диаграмме любого уровня декомпозиции.

Все функции преобразующих блоков можно классифицировать по следующим видам:

1. *Деятельность* – совокупность процессов, выполняемых (протекающих) последовательно или/и параллельно, преобразующих множество информационных потоков с изменением их свойств. Деятельность осуществляется в соответствии с заранее определенной целью с потреблением различных ресурсов при выполнении ограничений со стороны внешней среды.
2. *Процесс (бизнес-процесс)* – совокупность последовательно или/и параллельно выполняемых операций, преобразующих информационный поток с изменением его свойств. Процесс протекает в соответствии с управляющими директивами, вырабатываемыми на основе целей деятельности.
3. *Операция* – совокупность последовательно или/и параллельно выполняемых действий, преобразующих объекты с изменением их свойств.

4. *Действие* – преобразование одного свойства информационного объекта в другое. Действие выполняется в соответствии с командой, которая является частью директивы на выполнение операции.
5. *Субдеятельность* – совокупность нескольких процессов в составе деятельности, объединенных некоторой частной целью.
6. *Подпроцесс* – группа операций в составе процесса, объединенных технологически или организационно.

Первые четыре функции находятся между собой в отношении иерархической подчиненности по принципу «сверху вниз». Последние два вида функций являются дополнительными. Каждая функция выполняется посредством механизма, в качестве которого служит организационно-техническая структура. Несмотря на то что одним из концептуальных принципов методологии SADT является отделение организации от функций, между иерархией функций и механизмов существует соответствие:

1. Организационно-техническая система – организационная структура, персонал и комплекс технических средств (оборудования), необходимые для осуществления деятельности.
2. Организационно-техническая подсистема – часть организационно-технической системы, обеспечивающая протекание процесса (субдеятельности).
3. Организационно-технический комплекс (модуль) – часть организационно-технической подсистемы, предназначенная для выполнения операции.
4. Организационно-технический блок – часть организационно-технического комплекса, обеспечивающая выполнение действия.

Таким образом, организационно-техническая структура становится результатом функционального моделирования.

Механизм любого уровня обеспечивает выполнение деятельности (процесса, операции, действия), потребляя ресурсы.

В данной лабораторной работе подробно рассмотрена функция, наиболее часто встречающаяся в автоматизированных информационных системах – «Вести БД» (см. рис. 10). Реализующий ее блок не является обязательным, од-

нако достаточно хорошо иллюстрируют методику разработки функциональных моделей.

Кроме этого в информационной системе необходимо обязательно предусмотреть вывод информации (на экран, печать, в файл, для управления и т.д.). Ошибкой при этом является «угасание» системы, когда какие-либо операции производятся, но не приводят к отчетам или сообщениям. Другой ошибкой является отсутствие управлений для функционального блока, когда переработка информации не сопровождается необходимыми указаниями. Очевидно, что при этом не обеспечивается выполнение функции (каждый блок должен иметь какое-либо управление).

Моделирование целесообразно производить, связывая функции стрелками для решения обозначенных в цели моделирования задач. Проверить это можно посредством «чтения» диаграммы от входа к выходу каждого функционального блока.

Порядок выполнения

1. Разработать диаграмму декомпозиции функционального блока «Идентифицировать пользователя».
2. Предусмотреть формирование параметризованного запроса.
3. Разработать диаграмму декомпозиции функционального блока «Вести БД».
4. Скорректировать диаграмму А0 в соответствии с введенными изменениями.
5. Провести анализ механизмов, необходимых для выполнения указанных функций.
6. Описать в отчете особенности разработанных блоков в вашем варианте.

Контрольные вопросы

1. Пояснить методику декомпозиции функций.
2. Пояснить и проиллюстрировать методику применения механизмов.
3. Определить и проиллюстрировать функциональные блоки, наиболее часто встречающиеся при проектировании автоматизированных информационных систем.

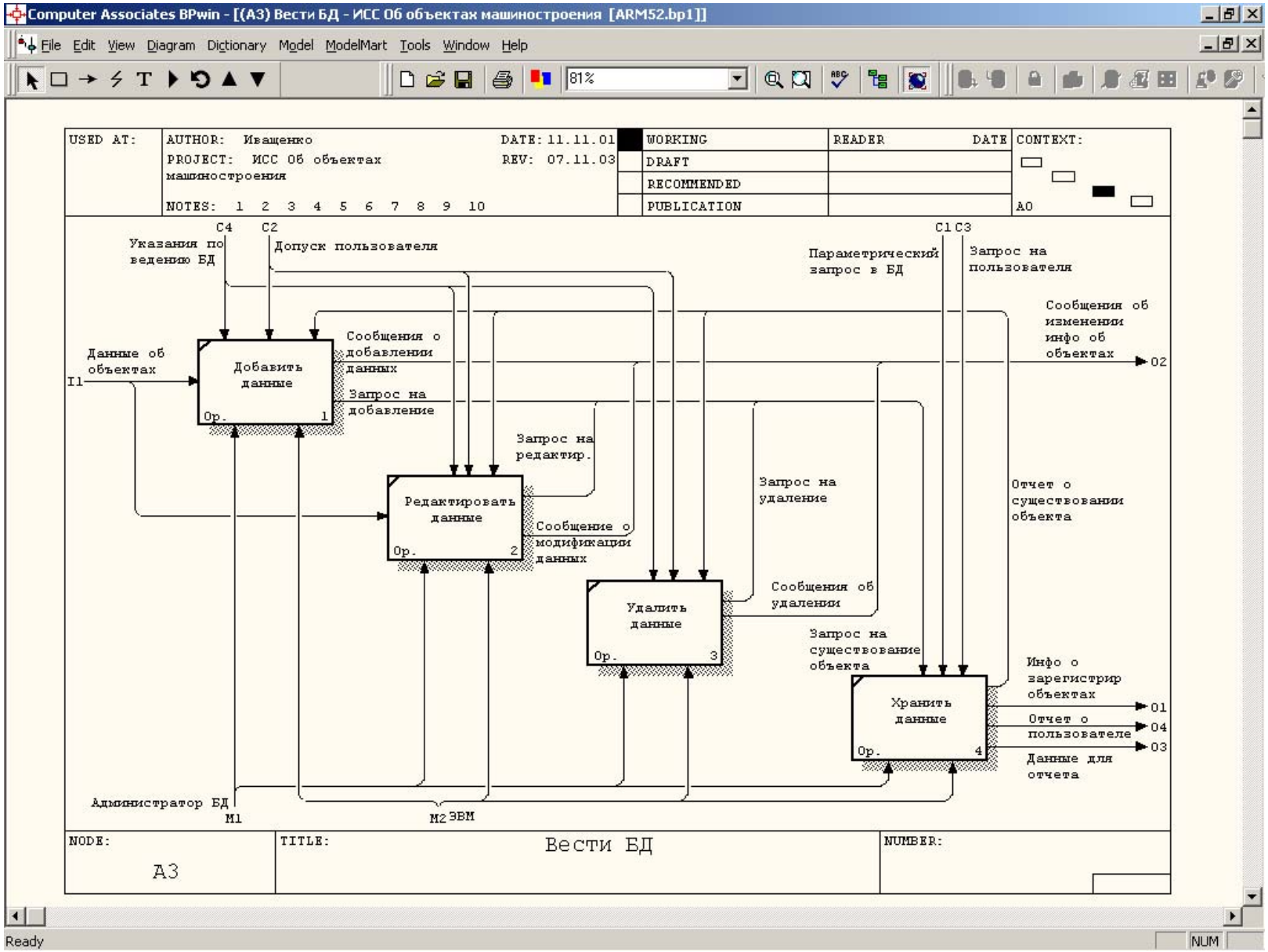

Рис. 10 – Функция ведения БД

Лабораторная работа № 4

Проектирование базы данных

Цель работы: Изучение методов построения структуры базы данных в виде ER-модели данных с помощью средства Erwin при проектировании автоматизированных информационных систем.

Основные теоретические сведения

Большинство автоматизированных информационных систем должны обеспечивать обработку данных. Для этого в их состав вводится функциональный блок ведения базы данных, рассмотренный в предыдущей работе. Проектирование структуры базы данных должно быть произведено с учетом требований по обеспечению логической и физической целостности данных, защите информации, возможности восстановления и расширения, совместимости с другими системами. Модель данных может быть иерархической, сетевой, хронологической или реляционной, однако учитывая распространенность современных реляционных СУБД (SQL-Server, InterBase, Oracle, Sybase и т.д.), в данной работе выбрана реляционная модель логического представления данных.

Отметим, что созданная база данных считается доступной из любого функционального блока автоматизированной информационной системы.

Проектирование базы данных может быть разделено на этапы концептуального, логического и физического проектирования.

ERwin – средство моделирования баз данных, в котором используется методология информационного моделирования на основе ER-диаграмм (диаграмм сущность-связь) IDEF1X (см. рис. 11). ERwin реализует проектирование схемы баз данных и генерацию её описания на языке целевой СУБД. ERwin имеет два уровня представления данных – логический и физический.

ERwin имеет несколько уровней отображения диаграммы: сущностей, атрибутов, определений, первичных ключей и иконок. Переключиться между первыми тремя уровнями можно с использованием кнопок панели инструментов. Переключиться на другие уровни отображения можно при помощи контек-

стного меню, в котором следует выбрать пункт Display Level и затем необходимый уровень отображения.

В терминологии ERwin различают для логической модели дополнительно три уровня представления, отличающихся степенью детализации данных:

- *диаграмма сущность-связь* - модель данных верхнего уровня. Она включает сущности и взаимосвязи, отражающие основные бизнес-правила предметной области, может включать связи многие-ко-многим и не включать описания ключей;
- *модель данных, основанная на ключах* – более подробное представление данных. Она включает описание всех сущностей и первичных ключей и предназначена для представления структуры данных и ключей, которые соответствуют предметной области;
- *полная атрибутивная модель* – наиболее детальное представление структуры данных: представляет данные в третьей нормальной форме и включает все сущности, атрибуты и связи.

Основные компоненты диаграммы ERwin – это сущности, атрибуты и связи.

Сущность определяется как объект, событие или концепция, информация о котором должна сохраняться. Сущности должны иметь наименование с четким смысловым значением, именоваться существительными в единственном числе, не носить «технических» наименований и быть достаточно важными, чтобы их моделировать. Каждый экземпляр сущности должен быть уникальным. Добавление сущности производится с помощью кнопки  .

Атрибут хранит информацию об определенном свойстве сущности. Атрибут или группа атрибутов, которые однозначно идентифицируют сущность, называются первичным ключом (Primary Key). Для описания атрибутов следует в контекстном меню выбрать пункт Attribute Editor. Атрибуты должны именоваться в единственном числе и иметь четкое смысловое значение.

Связь является логическим соотношением между сущностями. Каждая связь должна именоваться глаголом или глагольной фразой (Relationship Verb Phrases). По умолчанию имя связи на диаграмме не показывается. Для отобра-

жения имени связи следует в пункте Display Options/Relationship контекстного меню диаграммы включить опцию Verb Phrase.

На логическом уровне можно установить идентифицирующую связь один-ко-многим, связь многие-ко-многим и неидентифицирующую связь один-ко-многим. Связь сущности с другими сущностями автоматически определяет ее тип (зависимая и независимая).

Идентифицирующая связь устанавливается между независимой (родительский конец связи) и зависимой (дочерний конец связи) сущностями. Зависимая сущность не может существовать самостоятельно и изображается прямоугольником со скругленными углами. При установлении идентифицирующей связи атрибуты первичного ключа родительской сущности автоматически переносятся в состав первичного ключа дочерней сущности. Эта операция дополнения атрибутов дочерней сущности при создании связи называется миграцией атрибутов. В дочерней сущности новые атрибуты помечаются как внешний ключ – (FK).

Неидентифицирующая связь служит для связывания независимых сущностей. При этом атрибуты первичного ключа родительской сущности мигрируют в состав неключевых компонентов дочерней сущности.

Редактирование свойств связи осуществляется с помощью пункта Relationship Editor контекстного меню диаграммы. Мощность связи (Cardinality) служит для обозначения отношения числа экземпляров родительской сущности к числу экземпляров дочерней. На вкладке Definition можно дать полное определение связи. На вкладке Rolename/RI Actions можно задать имя роли и правила ссылочной целостности. Имя роли (функциональное имя) – это синоним атрибута внешнего ключа, который показывает, какую роль играет атрибут в дочерней сущности. Имя роли используется, когда несколько атрибутов одной сущности имеют одинаковую область значений, но разный смысл или в случае рекурсивных связей (fish hook). Рекурсия может быть иерархической или сетевой.

Правила ссылочной целостности (Referential Integrity) – логические конструкции (условия, предикаты), которые выражают бизнес-правила использо-

вания данных и представляют собой правила правильного выполнения операций добавления, замены и удаления данных. При генерации схемы БД на основе логической модели будут сгенерированы правила ссылочной целостности, которые должны быть предписаны для каждой связи, и триггеры (программные реализации правил), обеспечивающие ссылочную целостность.

Иерархия наследования (категорий) – особый тип объединения сущностей, обладающих общими характеристиками (категориальных сущностей)-обобщение. Для каждой категории указывается дискриминатор – атрибут родителя, который показывает, как отличить одну категориальную сущность от другой. Для редактирования категорий нужно выбрать символ категории и выбрать в контекстном меню пункт Subtype Relationship Editor. В диалоге Subtype Relationship можно указать атрибут – дискриминатор категории (Discriminator Attribute Choice) и тип категории – неполная  /полная  (Complete/Incomplete). Неполная категория имеет неполное перечисление обобщаемых частных примеров.

Различают два уровня физической модели: трансформационная модель (Transformation Model) и модель СУБД (DBSM Model). Физическая модель содержит всю информацию, необходимую для реализации конкретной базы данных. Трансформационная модель содержит информацию для реализации отдельного проекта, который может быть частью общей информационной системы и описывать подмножество предметной области. ERwin поддерживает ведение отдельных проектов, позволяя проектировщику выделять подмножество модели в виде предметных областей (Subject Area). Модель СУБД автоматически генерируется из трансформационной модели и является точным отображением системного каталога СУБД.

Физический уровень представления модели зависит от выбранного сервера (предполагается клиент-серверная архитектура АС).

Для выбора СУБД служит редактор Target Server (меню Server/Target Server… доступно только на физическом уровне). ERwin поддерживает практически все распространенные СУБД, всего более 20 реляционных и нереляционных баз данных.

Представления (виды) – объекты базы данных, данные в которых не хранятся постоянно, как в таблице, а формируются динамически при обращении к представлению. Палитра инструментов ERwin на физическом уровне содержит кнопки внесения представлений и установления связей между таблицами и представлениями.

При генерации схемы физической базы данных ERwin автоматически создает отдельный индексный файл на основе первичного ключа каждой таблицы, а также на основе всех альтернативных ключей, внешних ключей и инверсионных входов, поскольку эти столбцы наиболее часто используются для поиска данных. Изменить характеристики существующего индекса или создать новый можно с помощью контекстного меню таблицы Index в редакторе Index Editor.

Для создания триггера служит редактор Table Trigger Editor (вызывается кнопкой Table Trigger диалога Table Trigger Viewer). Триггеры и хранимые

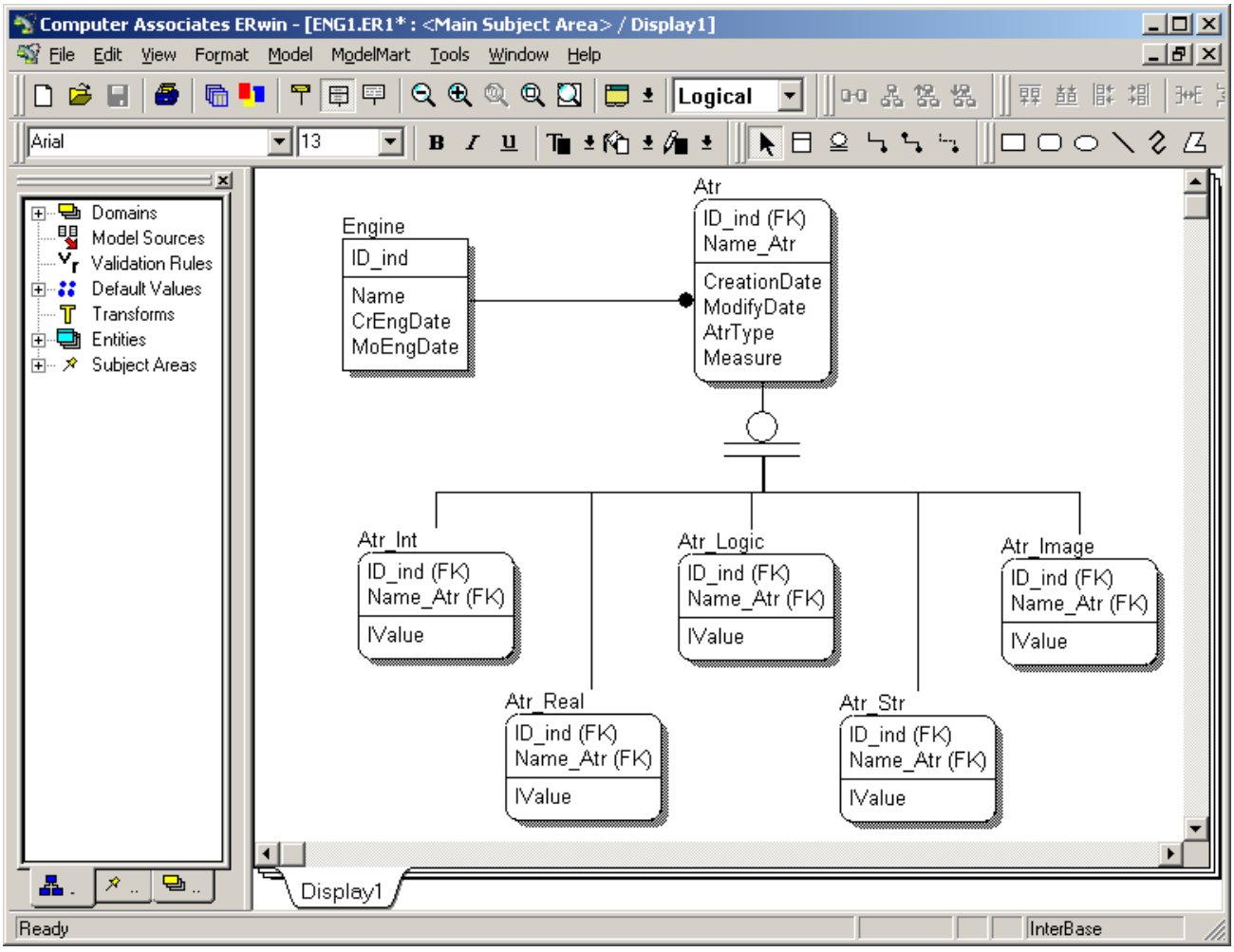

Рис. 11 – ER модель базы данных

процедуры – это именованные блоки кода SQL, которые заранее откомпилированы и хранятся на сервере для того, чтобы быстро производить выполнение запросов, проверку достоверности данных и выполнять другие часто вызываемые функции. Хранимые процедуры могут вызываться из клиентского приложения или другой хранимой процедуры. Триггер – это особый вид хранимой процедуры, выполняемый автоматически при добавлении, изменении или удалении строк в существующие таблицы. Для генерации триггеров ERwin использует механизм шаблонов – специальных скриптов, использующих макрокоманды. Шаблоны триггеров ссылочной целостности, генерируемые ERwin, по умолчанию можно изменить, кроме того, можно переопределить как триггеры для конкретной связи, так и шаблоны во всей модели в целом. Для создания или редактирования хранимой процедуры следует выбрать в контекстном меню пункт Table Editor/ Stored Procedure. ERwin позволяет связывать хранимые процедуры не только с отдельными таблицами, но и со всей моделью.

Порядок выполнения

1. Построить концептуальную, логическую и физическую модели базы данных. Предусмотреть требования целостности.
2. Осуществить переход к реляционной модели. Осуществить нормализацию полученной модели базы данных (до 3НФ).
3. Реализовать ведение базы данных в выбранной СУБД.
4. Внести необходимые исправления во все диаграммы модели.
5. Описать проведенную работу в отчете. Привести логическую, физическую и реляционную схемы базы данных.

Контрольные вопросы

1. Различия концептуального, логического и физического проектирования.
2. Определить различия идентифицирующей и неидентифицирующей связей, зависимой/независимой сущности.
3. Для чего используется имя роли? Описать виды иерархии.
4. Описать процедуру построения физической модели с помощью ERwin.

Лабораторная работа № 5

Построение иерархической модели БД

Цель работы: Изучение методов построения реляционной схемы отношений базы данных с помощью средства АСПЕКТ.

Основные теоретические сведения

В основу методологии автоматизированного синтеза учебно-исследовательской системы АСПЕКТ положен принцип проектирования схемы реляционной базы данных как единой семантической сущности. Результатом применения методологии является база данных, удовлетворяющая предположению о существовании универсальной реляционной схемы (ПУРС) рассматриваемой предметной области.

Модель аспектов реляционной модели во многом аналогична иерархической семантической модели (SHM-модели, модели Смита) [9].

Проектирование базы данных начинается с проектирования АСПЕКТ-модели на концептуальном уровне. Аспект выражает определенную точку зрения на понятия предметной области и представляет законченный фрагмент концептуальной схемы. Формально аспект представляет собой отношение между объектами и по смыслу является агрегацией. Но в силу ПУРС легко показать, что вся схема на концептуальном уровне может быть представлена в виде иерархии аспектов с добавлением таких понятий, как обобщение (is_a) и унификация (тождество).

Обобщение – форма абстракции, в которой сходные объекты связываются с обобщенным объектом более высокого уровня. Составляющие объекты могут рассматриваться как результат специализации обобщенного объекта. На уровне обобщенного объекта различия в специализации подавляются, и делается акцент на общие свойства.

Ассоциация – форма абстракции, в которой совокупность сходных объектов рассматривается как объект – множество более высокого уровня. Детали объектов – членов ассоциации подавляются, а свойства объекта-множества выделяются.

ПУРС включает однозначное определение семантики понятий. В каких бы отношениях понятие не находилось с другими понятиями, смысл его не должен изменяться. ПУРС предполагает существование единой схемы отношений для данной предметной области. Это может быть одно сложное отношение, связывающее все понятия, или иерархия отношений. Любое понятие или сущность предметной области отражается набором кортежей, т.е. строк отношений. Кортеж соответствует операции агрегации. Доказано, что если ПУРС выполняется, то иерархия кортежей всегда существует [10]. Обобщая известное в программировании понятие типа данных на любые значения объектов, введём понятие типа объектов.

Имеется две категории типов: базовые (стандартные для целевой СУБД: целые, вещественные, строковые, логические и т.д.) и схемные (типы объектов предметной области). Схемные типы явно отображаются на концептуальной схеме. С каждым схемным типом ассоциируется активный домен, состоящий из различных объектов предметной области. Над типами возможны базовые операции: выделение подмножества и суперпозиция (объединение или агрегация). В операции "обобщение" тип не меняется, а лишь подвергается некоторой специализации.

Аспект – набор кортежей, соответствующий одному отношению или иерархии отношений. На концептуальном уровне аспект отражается агрегацией или иерархией агрегаций. Под агрегацией понимается форма абстракции, в которой некое отношение между типами рассматривается как тип объектов более высокого уровня абстракции. Наблюдается связь с теорией типов: если тип – допустимое множество значений набора атрибутов, тогда кортеж – значение типа или экземпляра типа. Если имеется одна агрегация, то это будет простой схемный тип, если имеется более одной агрегации, то это составной схемный тип.

Важной семантической характеристикой аспекта является идентификатор. Идентификатором отношения в данном аспекте (идентификатором аспекта) называется совокупность имен типов, объекты которых однозначно определяют кортеж отношения.

Семантическая связь между простым и базовым типом моделируется с помощью абстракции специализации, позволяющей установить отношение «быть подмножеством» между двумя типами объектов (отношение is_a). Отношение «быть подмножеством» означает, что каждый объект подтипа является объектом супертипа, или, иными словами, подтип включается в супертип.

Наряду с абстракцией специализации используется абстракция обобщения, которая также формирует отношение is_a. В отличие от специализации она устанавливает такое соответствие между типами, при котором обобщающий тип разделяется на подтипы и объединение подтипов полностью покрывает этот тип. С помощью этой абстракции моделируется обобщение аспектов.

Семантическая идентичность аспектов устанавливается в результате операции унификации.

Многократное применение абстракций агрегации и обобщения приводит к созданию иерархических семантических структур. Многократное применение унификации образует класс синонимов, который с учетом порядка указания имен также представляется иерархической структурой.

Над аспектами определены следующие операции:

1. *Конструирование семантических структур*. Правила конструирования:
   1) Недопустимо повторное использование имен типов в пределах одной структуры.
   2) Структура, имеющая имя, совпадающее с именем одного из типов другой структуры, должна быть включена в структуру или переименована.
2. *Согласование аспектов*.
3. *Унификация аспектов* – объявление аспектов тождественными.

В процессе согласования осуществляются структурные преобразования для получения логически непротиворечивых аспектов. Аспекты, содержащие общие типы, считаются согласованными, если они обобщены, унифицированы или соединены в иерархическую структуру. Согласование аспектов производится по правилам:

1. Если совпадают идентификаторы аспектов (набор ключевых атрибутов), то необходимо образовать их обобщение, унификацию, иерархию или объединить соответствующие им структуры в одном из них.
2. Если идентификаторы частично пересекаются, то необходимо образовать иерархию аспектов.
3. Если идентификатор одного аспекта полностью входит в идентификатор другого, то необходимо образовать иерархию аспектов.
4. Если аспекты, в которых имеются общие типы, не удовлетворяют условиям 1,2,3, то их необходимо переименовать.

Если все правила выполнены, то аспект считается согласованным. Предварительно каждому схемному типу должен быть назначен базовый тип.

Согласованная концептуальная схема на следующем этапе проектирования трансформируется в схему реляционной базы данных. Нормализация – процесс преобразования согласованной схемы аспектов в реляционную структуру отношений. Нормализация заключается в последовательном обходе иерархии аспектов, выделении отдельных аспектов и замене в них составляющих аспектов идентификаторами. В ходе нормализации определяются избыточные схемы отношений. Основные правила нормализации:

1. Если в структуру данного аспекта входит другой аспект, идентификатор которого состоит из всех его компонентов, то возможно объединение аспектов в одном из них.
2. Если идентификатор данного аспекта состоит из одного аспекта и не содержит имен других типов, то возможно объединение аспектов в одном из них.

Средство АСПЕКТ включает графический редактор для построения фрагментов концептуальных схем.

Иерархии аспектов, обобщений и унификаций называются фрагментами концептуальной схемы. Фрагменты обеспечивают модульный принцип построения и просмотра схем. Произвольный фрагмент может быть вызван на экран для редактирования, но схема в целом на экран не вызывается.

Результат моделирования – схема реляционных отношений с указанием идентификатора.

Порядок выполнения

1. Построить схему реляционных отношений с помощью системы АСПЕКТ.
2. Провести нормализацию.
3. Построить реляционную модель базы данных.
4. Сравнить результаты моделирования с реляционной моделью, полученной в предыдущей лабораторной работе. Внести исправления при необходимости.
5. Отразить результаты работы в отчете.

Контрольные вопросы

1. Определить ПУРС, понятие аспекта, фрагмента.
2. Перечислить операции над аспектами.
3. Дать понятие нормализации. Перечислить правила нормализации.
4. Перечислить виды семантических моделей.
5. Определить правила построения иерархии аспектов.

Лабораторная работа № 6

Построение отчетов и контроль модели в ERwin

Цель работы: Ознакомление с генератором отчетов Erwin, использование генератора отчетов как средства контроля качества модели и составления документации.

Основные теоретические сведения

При проектировании модели данных возможно представление информации графической диаграммы в виде отчета в текстовом формате. ERwin располагает встроенным генератором и редактором отчетов.

Доступ к генератору отчетов возможен из панели инструментов, либо из меню Tasks/ Generate Reports. В генераторе отчетов Report Browser (рис. 12) имеются следующие возможности:

- *генерация отчета на основании выбранного шаблона;*
- *изменение структуры отчета и его внешнего вида;*
- *создание нового или изменение существующего шаблона отчета;*
- *сохранение и вывод на печать результатов отчета;*
- *экспорт данных в другие приложения.*

Встроенные шаблоны ERwin поделены на группы, каждая из которых имеет свою «точку зрения» на состав модели:

- *Column Reports – отчеты о колонках таблиц физической модели и их характеристиках;*
- *Attribute Reports – отчеты по атрибутам сущностей;*
- *Entity Reports – отчеты о сущностях логической модели, их составе и характеристиках;*
- *Domain Reports – отчеты о доменах и их параметрах;*
- *Table Reports – отчеты о таблицах физической модели;*
- *Subject Area Reports – отчеты по областям модели и их составу;*
- *Model Validation Reports – отчеты для контроля качества модели;*

- *Stored Procedure Reports – отчеты о хранимых процедурах физической модели;*
- *Relationship Reports – отчеты о связях модели и их параметрах;*
- *View Reports – отчеты по представлениям физической модели;*
- *ERWin Volume Reports – отчеты по физической структуре базы данных и потребностях базы данных в ресурсах СУБД.*

Каждая группа отчетов включает несколько видов отчетов, отражающих наиболее общие задачи проверки модели. ERwin позволяет создать отчет по всей диаграмме, по части диаграммы или по части области.

Интерфейс генератора отчетов имеет несколько основных компонент: перечень шаблонов готовых отчетов (All reports), окно вывода результатов (Result Set), окно описания отчета (Report description), панель управления деревом отчетов (под Report description), строка меню, панель инструментов.

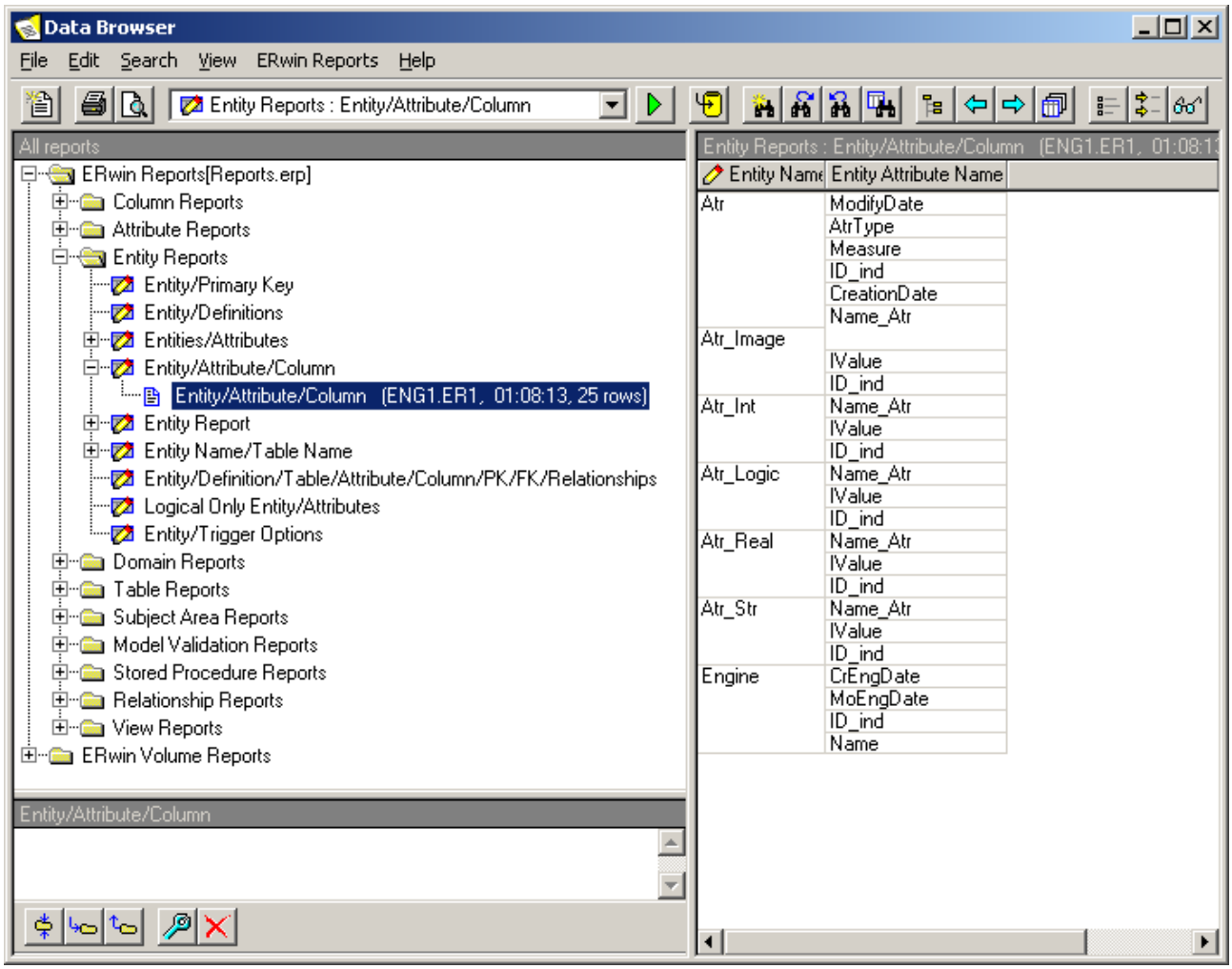

Рис. 12 – Генератор отчетов в ERwin

Для генерации какого-либо отчета необходимо сначала выбрать его шаблон из списка, а затем выполнить File/ Execute Report (либо воспользоваться контекстным меню). После этого под соответствующим шаблоном появится указатель отчета (с указанием даты и времени генерации). Созданные отчеты сохраняются вместе с моделью и могут быть просмотрены при ее повторной загрузке. Для удаления ненужного отчета используется команда Edit/ Delete. Команды меню Search позволяют искать необходимые отчеты (Available reports) или данные в отчете (Find).

После генерации отчета помимо его просмотра возможно произвольное изменение его структуры и/или порядка сортировки столбцов данных. Для этого применяется контекстное меню заголовков столбцов отчета.

Контекстное меню сгенерированного отчета используется для выполнения всех задач, связанных с отчетом, а именно его редактирование, распечатывание, экспорт, удаление и т.п.

Для изменения структуры отчета используется команда Edit/Report Format, при вызове которой открывается редактор настройки формата отчета. При помощи этого редактора можно изменять порядок колонок (move up, move down), название колонок (heading) и определять тип колонки (option). В ERwin существует четыре группы колонок: обычные (visible в option), всплывающие (popup в option), выделенные полужирным текстом (bold в option), выделенные серым цветом (grayed в option). Каждая колонка может участвовать в нескольких группах одновременно. Если колонка отключена как обычная, то она становится невидимой. При помощи кнопки Save можно сохранить текущий дизайн отчета в дереве отчетов и добавить его к активному шаблону отчета, при этом необходимо указать его название и комментарий.

Помимо использования стандартных шаблонов ERwin допускает разработку новых или модификацию имеющихся отчетов. Для этого применяется редактор отчетов, доступный по команде File/ New ERwin report. При входе в этот редактор необходимо определить класс отчета, его категорию и присвоить ему имя. Класс отчета говорит о модели (логической или физической), использующейся для генерации отчета. Категория отчета определяет разновидность отчета в своем классе.

Все элементы модели представлены в виде дерева параметров (options) с точки зрения текущей категории отчетов. С помощью группы options можно определить: отображать все элементы модели или только выбранные. В составе дерева параметров имеются два специальных пункта: Filter (способ выделения элементов, подлежащих включению) и Sort by (способ сортировки данных отчета).

С помощью вкладок Definition и Notes можно задать определение для данного типа отчета и какие-либо дополнительные сведения (примеры использования, прочие условия). Кнопки вкладки Options управляют списком приводимых параметров: кнопка Show Selected открывает все группы дерева, содержащие выделенные параметры; Collapse All сворачивает все дерево параметров; Clear All – очищает все выделенные параметры.

ERwin позволяет сохранять подборки шаблонов отчетов в отдельных файлах и подгружать их к новым моделям по мере необходимости. Все эти операции доступны из меню ERwin Reports.

Помимо операций генерации, редактирования и печати отчетов, ERwin позволяет экспортировать результаты отчета в другие форматы (в частности в Microsoft Word), что позволяет в сжатые сроки готовить качественную документацию по модели. Для экспорта данных используется команда File→Export. При нажатии этого пункта меню открывается окно настройки процесса экспорта.

ERwin позволяет экспортировать данные в несколько форматов файлов (каждый из форматов снабжается краткой аннотацией):

- *файлы гипертекста (HTML);*
- *текстовые файлы в формате CSV (значения, разделенные запятыми);*
- *файлы программы RPTwin.*

Кроме этого имеется возможность экспорта через интерфейс DDE (для экспорта в Microsoft Word, Microsoft Excel и т.п.).

Между BPwin и ERwin возможен обмен информацией о сущностях и их атрибутах для установления соответствия между информационными потоками (стрелками) в BPwin и сущностями в ERwin. Поскольку разработка сущностей может происходить как с помощью BPwin, так и ERwin, синхронизация моделей в

процессе проектирования осуществляется с помощью механизма импорта/экспорта данных в файлы с расширением .eax (ERwin to BPwin) и .bpx.

Экспорт сущностей и их атрибутов в ERwin осуществляется с помощью пункта меню File/ Export/ BPwin. При импорте (File/ Import/ ERwin) BPwin присваивает сущностям внутренние ID идентификаторы, созданные в ERwin. Для импорта сущностей и атрибутов без идентификаторов ERwin необходимо установить флажок «Import as new entities and attributes», иначе будет установлена связь с исходной моделью в ERwin. Установка флажка «BPwin Only» означает использование сущностей и атрибутов только в модели BPwin. Флажок «Update by Name» означает, что идентификаторы ERwin будут игнорироваться. Идентификаторы необходимо сохранять для того, чтобы повторный импорт из ERwin в BPwin в связи с изменением сущности в ERwin сохранил связи сущности со стрелками. При установке флажка «Update by Name» связи будут созданы не по идентификаторам, а по именам сущностей и атрибутов. При импорте словарь сущностей и атрибутов будет автоматически пополнен.

Форматы .eax и .bpx используются не только для добавления новых сущностей в модели ERwin или BPwin, но и для синхронизации процесса проектирования. С помощью этих файлов можно создать постоянную связь модели в BPwin и модели в ERwin на основе внутренних ID идентификаторов. При этом не важно, где – в BPwin или в ERwin – изначально создавались сущности. Связь устанавливается с помощью следующих действий:

- *Экспорт сущностей и атрибутов в файл .bpx. При этом BPwin автоматически ассоциирует файл .bpx с моделью bp1.*
- *Импорт файла .bpx в модель ERwin. При этом BPwin обновляет содержание файла .bpx, добавляя к сущностям внутренние идентификаторы ID, присвоенные в ERwin и исключает его повторный импорт в ERwin.*
- *Импорт файла .bpx в модель BPwin. В случае, если этот файл был открыт в ERwin, BPwin добавляет созданные идентификаторы в модель SADT.*

Установить соответствие между сущностями информационной модели и стрелками SADT модели автоматизированной информационной системы можно с помощью окна свойств стрелок (рис. 13).

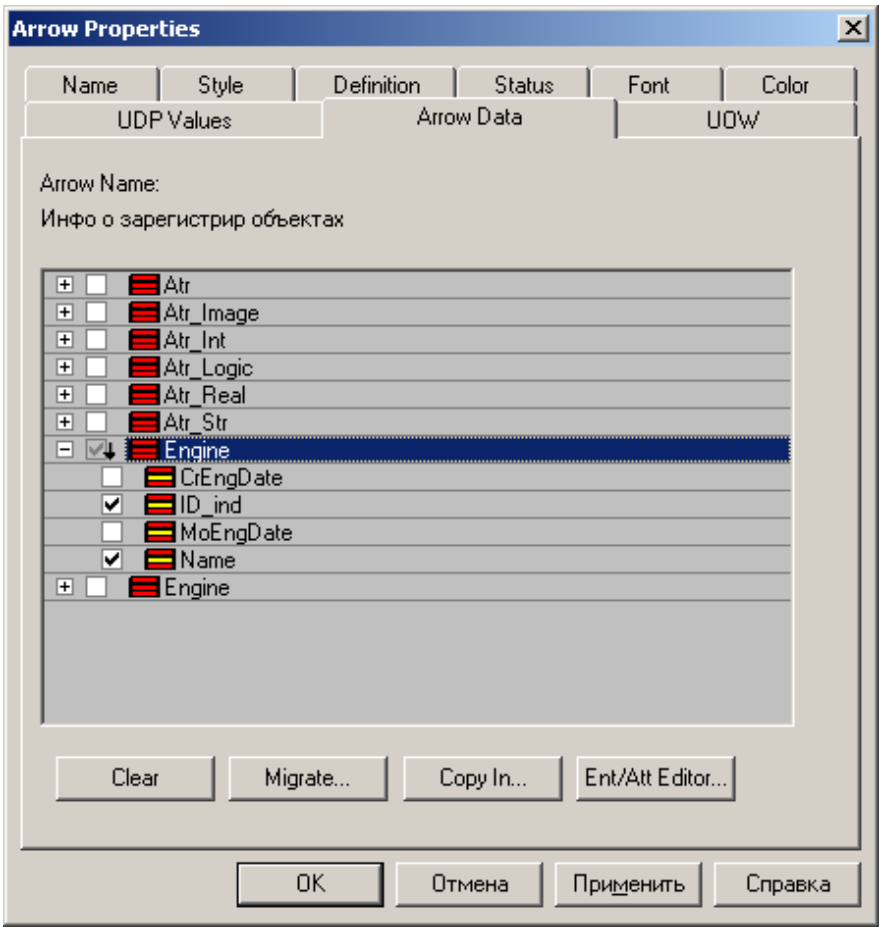

Рис. 13 – Свойства стрелок

Порядок выполнения

1. Создать отчет по модели данных в ERwin.
2. Осуществить связь созданных объектов модели данных с информационными потоками SADT модели автоматизированной информационной системы.
3. Отразить работу в отчете.

Контрольные вопросы

1. Перечислить инструменты ERwin, используемые для составления отчетности по модели.
2. Определить возможность генерирования отчета, содержащего максимум информации о модели.
3. Разработать новый шаблон отчета на базе уже имеющегося.
4. Построить отчет, перечисляющий для каждой сущности модели все ее сущности.
5. Определить, как осуществляется обмен данными между ERwin и BPwin.

Лабораторная работа № 7

Завершение моделирования. Построение отчетов в BPwin

Цель работы: Изучение методов декомпозиции функциональных блоков. Исследование особенностей моделирования в методологии SADT при проектировании автоматизированных информационных систем. Изучение возможностей создания отчетов средствами BPwin и RPTwin.

Основные теоретические сведения

Графика SADT устраняет неоднозначность описаний, выполненных экспертом на естественном языке. Устранение неоднозначности достигается с одной стороны в результате стандартной интерпретации графических обозначений, с другой – в результате декомпозиции и уточнения диаграмм высокого уровня. Декомпозиция и уточнения производятся до тех пор, пока диаграммы нижнего уровня не станут достаточно подробными для того, чтобы обеспечить точное значение объектов и функций системы.

Важной особенностью также является согласованное изложение материала с помощью диаграмм, для которого существенным является понятие точки зрения. Определенная точка зрения выделяет одни аспекты системы и игнорирует другие. Выбор точки зрения означает также применение определенной терминологии.

Декомпозиция заключается в начальном разделении объекта на более мелкие части и последующем их соединении для более детального описания объекта. Автор производит анализ и синтез вначале системных объектов, затем – функций системы в соответствии со списком данных.

В ходе моделирования важной является ясность изложения. Поэтому выбор стратегии декомпозиции часто требует нескольких итераций и многих изменений.

Достаточно результативными являются следующие стратегии декомпозиции:

Функциональная стратегия декомпозиции базируется на функциональных взаимоотношениях действий системы. Основной акцент ставится на том, что

делает система независимо от того, как она работает. Предпочтение отдается уточнениям ограничений на функции системы, а не их последовательности. Декомпозиция в соответствии с функциями представляет общую картину, поэтому наиболее предпочтительна.

Для систем команд и управления часто эффективна декомпозиция в соответствии с уже известными стабильными подсистемами. Это приводит к созданию моделей – по одной на каждую подсистему. Затем строится объединяющая составная модель всей системы. Данная стратегия теряет смысл при нестабильности границ подсистем.

Стратегия декомпозиции, основанная на отслеживании жизненного цикла для ключевых входов системы, оказывается эффективной для систем, непрерывно преобразующих свои входы в конечный продукт. Стратегия может быть рекомендована в случаях, когда целью является улучшение одного из основных входов и можно легко определить последовательные стадии этого улучшения.

Декомпозиция по физическому процессу состоит в выделении функциональных стадий, этапов завершения или шагов выполнения. Результатом такой декомпозиции является последовательное описание системы, при которой могут быть не учтены ограничения на функции или скрыта последовательность управления.

Декомпозиция прекращается, когда диаграммы, образующие нижний уровень модели, достаточно детализированы для достижения цели модели, когда модель достаточно точна, чтобы отвечать на вопросы, соответствующие ее цели. В случае разработки автоматизированной информационной системы декомпозиция прекращается, когда модель содержит всю информацию, необходимую для ее реализации группой программистов. То есть не детализируются блоки, представляющие элементарные логические структуры языка программирования высокого уровня.

BPwin имеет мощный инструмент генерации отчетов. Отчеты вызываются из меню Tools/ Reports. Всего имеется семь типов отчетов:

1. *Model Report – общая информация о модели* – наименование модели, точку зрения, область, цель, имя автора, дату создания и др.

2. *Diagram Report – отчет по диаграмме* (функциям, стрелкам, хранилищам и т.д.) конкретной диаграммы.
3. *Diagram Object Report – отчет по объектам диаграмм* – наиболее полый отчет по модели. Может включать полный список объектов модели и свойства, определяемые пользователем.
4. *Activity Cost Report – отчет о результатах стоимостного анализа*.
5. *Arrow Report – отчет по стрелкам* с указанием информации из словаря стрелок, разветвлении и слиянии стрелок, информации о блоке-источнике и блоке-назначении.
6. *DataUsage Report – отчет о результатах связывания модели процессов и модели данных.*
7. *Model Consistency Report – отчет по синтаксическим ошибкам модели*.

С помощью модуля построения отчетов Tools/ Report Builder возможно построение отчета по модели на основе заданного шаблона (рис 14). Поддерживаются HTML, RTF, Text форматы отчета.

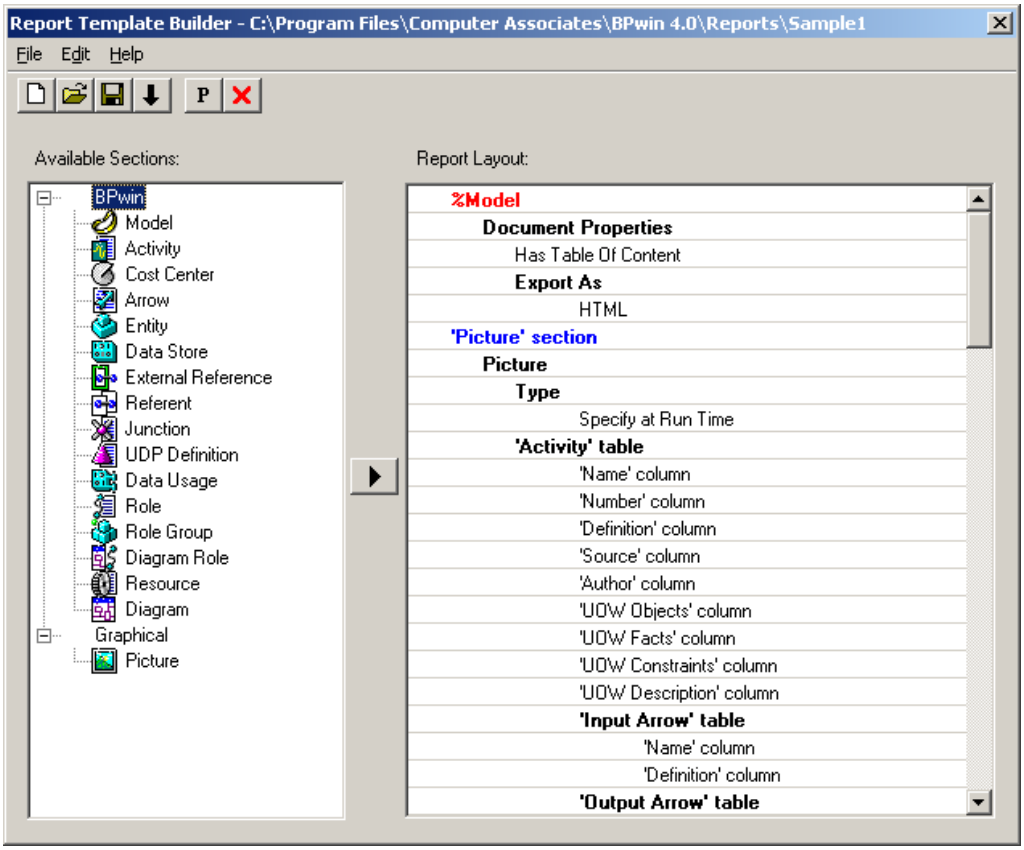

Рис. 14 – Шаблон отчета

Отдельно поставляется специализированный генератор отчетов RPTwin, с помощью которого можно создавать более качественные отчеты по моделям процессов и данных (см. рис. 15). RPTwin позволяет не только создавать отчеты презентационного качества, но и производить сложную обработку данных.

После создания отчета в BPwin и выбора RPTwin в качестве формата (Report Format) возникает диалог сохранения данных отчета, где указывается имя файла отчета (с расширением .lwd). В диалоге New Report можно выбрать тип создаваемого отчета:

1. *Quick Reports* – создание простейших отчетов;
2. *Guided Reports* – пошаговое создание отчета с сортировкой, группировкой и сложным форматированием данных.

   Шаблон отчета включает следующие секции:

- Report Header – печатается один раз в начале отчета;
- Page Header – печатается в верхней части каждой страницы;
- Group Header – печатается в начале каждой группы;
- Detail – печатается для каждой стоки набора данных;
- Group Footer – печатается в конце каждой группы;
- Page Footer – печатается в нижней части каждой страницы;
- Report Footer – печатается один раз в конце отчета.

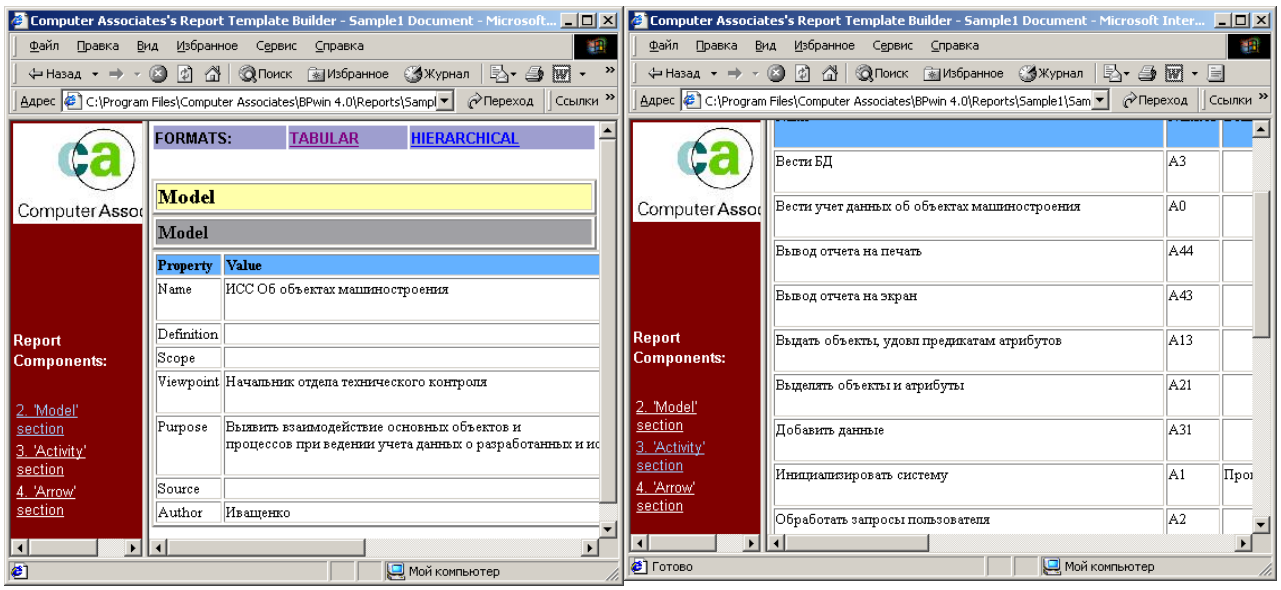

Рис. 15 – Отчет по модели в формате HTML

В секциях могут располагаться данные (Data Fields), поясняющий текст (Text Fields), вычисляемые поля (Formula Fields), OLE объекты и время, номер страницы, номер записи и т.д. (Special Fields).

Добавить тестовое поле, формулу, разрыв страницы и специальные поля можно с помощью палитры инструментов (Tool Box).

RPTwin позволяет выстроить данные отчета в определенном порядке (сортировка), либо объединить их в группы (группировка). Для установления группировки следует выбрать пункт меню Layout/Sorting and Grouping. В левом списке диалога Sorting/Grouping содержатся имена всех полей набора данных (DataSet Columns), в правом – список полей, по которым производится сортировка или группировка (Sort/Group On). Возможна установка порядка сортировки (Ascending – по возрастанию и Descending – по убыванию) и режима сортировки (Case sensitive). RPTwin позволяет установить сортировку и группировку по вычисляемому значению (Sort/Group on Calculated Value).

RPTwin позволяет преобразовать в формулу любое поле данных. Для этого в диалоге Data Field Properties следует выбрать Formula Editor. Задаваемые с помощью этого редактора формулы должны удовлетворять следующим требованиям:

Имена колонок не должны начинаться с цифры и не должны содержать специальных символов (пробелов и т.д.). Для использования имен колонок, содержащих специальные символы, их следует заключить в фигурные скобки. RPTwin поддерживает три типа операторов: арифметические, текстовый оператор конкатенации &, операторы сравнения, использующиеся в предикате конструкции If, логические операторы (is in, contains, and, or, not, is null, is not null). В арифметических выражениях могут быть использованы круглые скобки. Текстовые константы заключаются в двойные кавычки.

При выполнении действий над данными необходимо соблюдать правила соответствия типов. RPTwin различает пять типов данных: Number, Text, Date, Time, Datetime. Если возвращаемое значение формулы – строка, то в некоторых случаях при несоответствии типов RPTwin конвертирует операнды автоматически.

Функции RPTwin позволяют производить сложные вычисления и обработку данных отчета. Так же, как операторы, функции возвращают значения определенного типа. Для внесения функции в формулу нужно выбрать ее из списка Functions диалога Formula Editor.

Агрегативные функции позволяют производить вычисления по нескольким строкам отчета. Некоторые функции (Sum, Avg, Min, Max, Count) выполняются контекстно, то есть возвращают результат в зависимости от той секции отчета, в которой находятся. Другие агрегативные функции (GroupAvg, GroupSum, GroupMin, GroupMax, GroupCount, ReportAvg, ReportCount, ReportMin, ReportMax, ReportSum) возвращают значение независимо от их расположения в отчете.

RPTwin является двухпроходным генератором отчетов. На первом этапе рассматриваются все данные и происходит вычисление функций. На втором этапе происходит непосредственно процесс печати или вывода на экран в режиме предварительного просмотра.

Порядок выполнения

1. Реализовать обработку запросов с помощью функциональных блоков. Рассмотреть необходимые механизмы и управления. (См. пример на рис. 16).
2. Провести анализ всех диаграмм с точки зрения стратегий функционального моделирования. При необходимости построить диаграммы, пользуясь другой стратегией и выбрать лучший вариант. Отразить рассуждения в отчете.
3. Изучить особенности построения отчетов с помощью RPTwin.
4. Составить отчет в соответствии с заданием.
5. Внести исправления во все диаграммы модели. Внести дополнения в отчет.

Контрольные вопросы

1. Описать возможности построения отчетов.
2. Описать существующие стратегии декомпозиции. Привести примеры.
3. Каким образом достигается согласованное изложение материала на диаграммах в методологии SADT?

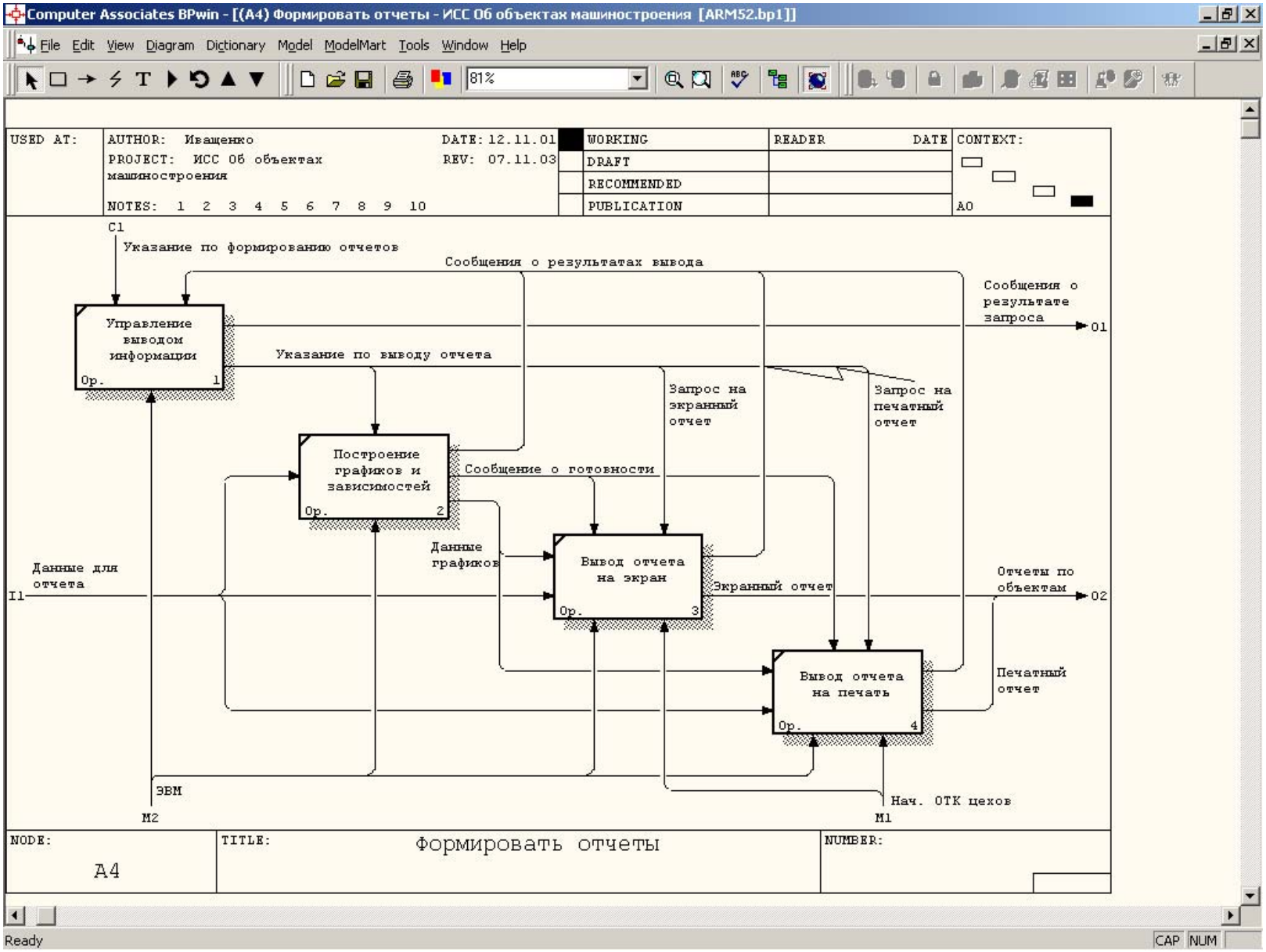

Рис. 16 – Функциональный блок формирования отчетов

Лабораторная работа № 8

Реализация проекта. Оформление пояснительной записки

Цель работы: Изучение особенностей разработки автоматизированных информационных систем, основанной на спроектированной модели.

Основные теоретические сведения

Результатом предыдущих лабораторных работ должны стать построенная SADT модель автоматизированной информационной системы с реляционной схемой базы данных. На основе разработанного проекта можно создать спецификацию функций системы, которая служит основой технического задания на разработку собственно автоматизированной информационной системы. Для этого для каждого недекомпозированного функционального блока необходимо описать действия, которые определяют, что необходимо для того, чтобы функции выполнялись правильно и каковы результаты их выполнения и свойства – численные и текстовые описания характеристик функций и данных системы.

SADT определяет действие как вид работы функции после того, как она "включается" посредством управления для преобразования некоторых входов в выходы. Каждое конкретное действие использует не все возможные управления и входы и производит не все возможные выходы. Правила действия включают номер блока, уникальные идентификатор действия, предусловия и постусловия. Каждое правило формируется в соответствии с синтаксисом:

[Модель/] блок* действие: предусловия → постусловия

Блок и действие дают правилу уникальное имя, однозначно идентифицируют одно правило действия в модели. Наименование модели пишется в том случае, когда правила действия рассматриваются для более чем одной модели. Если истинные предусловия, выполняется функция "блок" и делает истинными постусловия. Логические операторы AND, OR и NOT вместе со скобками представляют средство для записи сложных логических выражений.

Для определения всех способов действия функции в процессе работы системы возможно построение таблиц истинности, представляющих собой декартово произведение всех возможных сочетаний присутствия (True) и обяза-

тельного отсутствия (False) входных дуг, дуг управления и выходных дуг. Каждый столбец такой таблицы становится потенциальным правилом действия.

SADT проект является достаточным для описания модели автоматизированной информационной системы. Однако для построения технического задания разработчику необходимо описать модель базы данных и способы действия функций.

Таким образом, отчет по лабораторному практикуму должен включать:

1. Титульный лист.
2. Задание.
3. Список функций и объектов системы.
4. Формулировка точки зрения, цели и области моделирования.
5. Контекстная диаграмма IDEF0.
6. Диаграммы декомпозиции со способами действия недекомпозированных функций.
7. Пояснения к каждой диаграмме.
8. ER – модель базы данных. Логическая и физическая модели.
9. Реляционная модель базы данных.
10. Результаты моделирования в системе АСПЕКТ.
11. Описание модели базы данных. Результаты проверки на 3НФ и нормализации.
12. Описания базы данных в выбранной СУБД. Пояснения к выбору СУБД. Описания выбранных средств обеспечения целостности.
13. Пояснение к выбору средства реализации – языка высокого уровня.
14. Описание автоматизированной информационной системы – руководство пользователю.
15. Описание контрольного примера (см. пример на рис. 17).
16. Код основного модуля.

Отчет оформляется в соответствии со стандартом СГАУ по оформлению учебных текстовых документов [5].

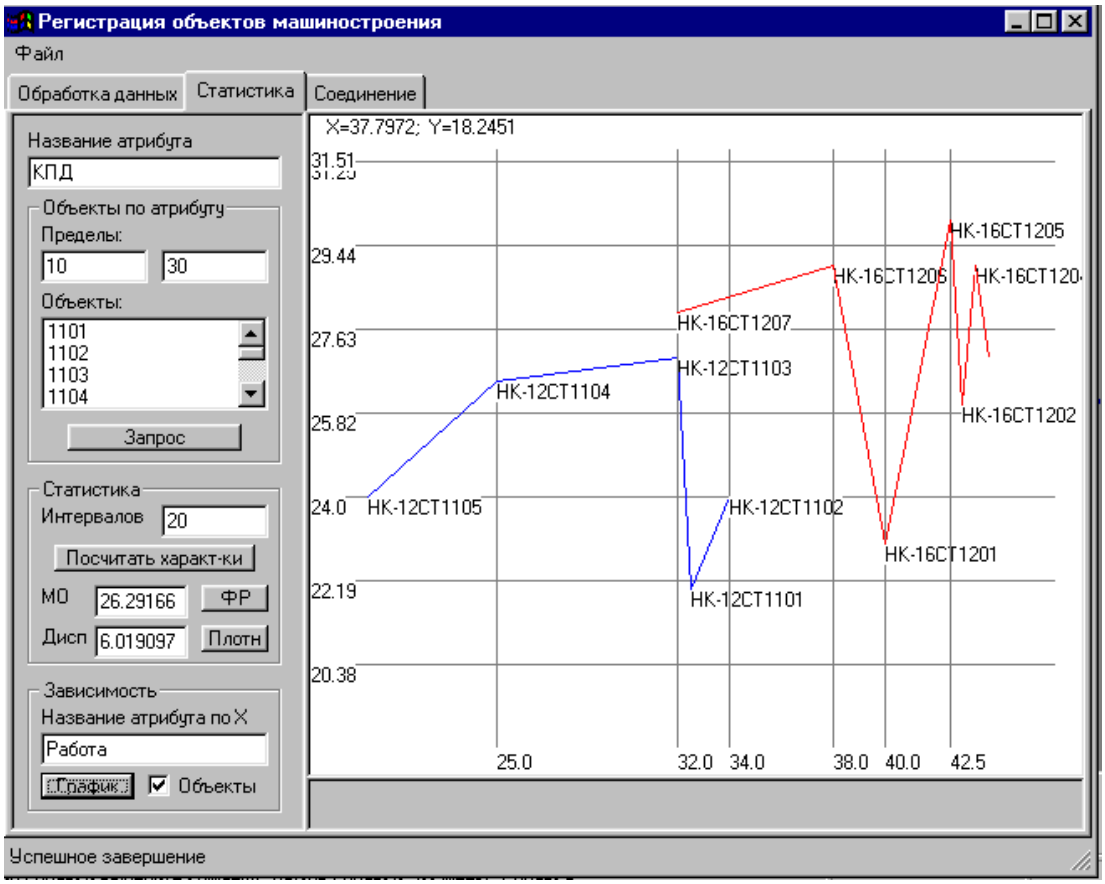

Рис. 17 – Пример экранной формы реализации проекта

Порядок выполнения

1. Дополнить недекомпозированные блоки описанием способов действия функций.
2. Выполнить реализацию основных модулей автоматизированной информационной системы.
3. Оформить отчет.

Контрольные вопросы

1. Пояснить механизм описания способов действия функций.

Библиографический список

1. Дэвид Марка, Клемент МакГоуэн. Методология структурного анализа и проектирования. / Пер. с англ. – М.: Метатехнология, 1993. – 240 с.
2. Методология функционального моделирования IDEF0. Руководящий документ. – М.: ИПК Издательство стандартов, 2000.
3. Маклаков С.В. BPwin и ERwin. CASE-средства разработки информационных систем. – М.: ДИАЛОГ-МИФИ, 2002. – 256 с.
4. Маклаков С.В. Моделирование бизнес-процессов с AllFusion Process Modeler (Bpwin 4.1). – М.: ДИАЛОГ-МИФИ, 2003. - 240 с.
5. Стандарт предприятия СТП СГАУ 6.1.4–97: Метод. указания./ Самар. гос. аэрокосм. ун-т. – Самара, 1997.
6. Гейн К., Сарсон Т. Структурный системный анализ: средства и методы. В 2 ч. / Пер. с англ.; Под ред. Козлинского А.В. – М.: Научно-техническое предприятие ЭЙТЕКС, 1993.
7. Дерябкин В.П. Проектирование автоматизированных систем обработки информации и управления: Курс лекций. Самар. гос. аэрокосм. ун-т. – Самара, 2001. – 120 с.
8. Рамбо Дж., Якобсон А., Буч Г. UML: Специальный справочник. – СПб.: Питер,2002. – 656 с.
9. Тиори Т., Фрай Дж. Проектирование структур баз данных: В 2 кн. – М.: Мир, 1985. Кн.1. – 287 с. Кн.2. –320 с.
10. Мейер Д. Теория реляционных баз данных. – М.: Мир, 1987. – 608 с.

Учебное издание

ПРОЕКТИРОВАНИЕ АВТОМАТИЗИРОВАННЫХ ИНФОРМАЦИОННЫХ СИСТЕМ ПО МЕТОДОЛОГИИ SADT

Методические указания к лабораторным работам

Составители:

Дерябкин Валентин Павлович

Иващенко Антон Владимирович

Корректор: Н.С. Куприянова

Подписано в печать 24.09.2004. Формат 60x84 ¹⁄₁₆

Бумага офсетная. Печать офсетная.

Усл. печ. л. 3,23. Усл. кр.-отт. 3,22. Уч.-изд. л. 3,25

Тираж 100 экз. Заказ Арт. С-54/2004

Самарский государственный аэрокосмический университет

443086 Самара, Московское шоссе, 34

РИО