ARIS –
моделирование бизнес-процессов

Третье издание

ARIS_Title.indd 1 27.08.2008 16:58:36

August-Wilhelm Scheer

ARIS –
Business Process Modelling

Third Edition

Springer

ARIS_Title.indd 2 27.08.2008 16:58:36

А.-В. Шеер

ARIS –
моделирование бизнес-процессов

Третье издание

Москва • Санкт-Петербург • Киев
2009

ARIS_Title.indd 3 27.08.2008 16:58:36

ББК (У)65.050.2
Ш38
УДК 658.51

Издательский дом "Вильямс"
Зав. редакцией *А.В. Назаренко*
Перевод с англ. и редакция докт. физ.-мат наук *А.А. Рыбянец*

По общим вопросам обращайтесь в Издательский дом "Вильямс" по адресу:
info@williamspublishing.com, http://www.williamspublishing.com

Шеер, Август-Вильгельм.

Ш38 ARIS — моделирование бизнес-процессов, 3-е издание. : Пер. с англ. — М. : ООО "И.Д. Вильямс", 2009 — 224 с. : ил. — Парал. тит. англ.

ISBN 978-5-8459-1449-1 (рус.)

Система ARIS занимает лидирующее положение на рынке средств анализа и моделирования бизнес-процессов. В данной книге рассматривается широкий круг методов моделирования в рамках единой концепции. Соответствующие метамодели составляют информационную модель ARIS.

Особое внимание уделяется применению моделей ARIS для создания бизнес-приложений. Обсуждение концепций дополняется прикладными главами, посвященными внедрению стандартных бизнес-приложений с использованием моделей ARIS, применению программной системы ARIS Framework для разработки приложений, а также созданию объектно-ориентированных систем с использованием универсального языка моделирования UML.

Потенциальные читатели этой книги — это менеджеры предприятий, разработчики организационных и информационных систем, эксперты, консультанты, преподаватели и студенты в области компьютерных наук и их применения в бизнесе.

ББК (У)65.050.2

Все названия программных продуктов являются зарегистрированными торговыми марками соответствующих фирм.

Никакая часть настоящего издания ни в каких целях не может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами, будь то электронные или механические, включая фотокопирование и запись на магнитный носитель, если на это нет письменного разрешения издательства Springer-Verlag.

All rights reserved. This Publication is protected by Copyright and permission should be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise.

Russian language edition published by Williams Publishing House according to the Agreement with R&I Enterprises International, Copyright © 2009.

Authorized translation from English language edition published by Springer-Verlag., Copyright © 2000.

ISBN 978-5-8459-1449-1 (рус.)
ISBN 3-540-65835-1 (англ.)

© Издательский дом "Вильямс", 2009
© Springer-Verlag, 2000

Оглавление

A. ARIS — МОДЕЛИРОВАНИЕ БИЗНЕС-ПРОЦЕССОВ **15**
A.I. Стратегический анализ бизнес-процессов 22
A.II. Моделирование отдельных аспектов ARIS 36
A.III. Моделирование взаимосвязей между аспектами (управление) 108

B. ПРОЦЕДУРАЛЬНЫЕ МОДЕЛИ ARIS И ИЗБРАННЫЕ ПРИЛОЖЕНИЯ **179**
B.I. Внедрение стандартных бизнес-приложений с использованием моделей ARIS 180
B.II. Внедрение систем управления потоком с использованием моделей ARIS 187
B.III. Использование ARIS Framework для разработки приложений 192
B.IV. Разработка объектно-ориентированных систем с использованием UML 199
Расшифровка аббревиатур 205
Ссылки 208
Предметный указатель 220

Содержание

Предисловие к третьему изданию 9
Предисловие ко второму изданию 10
О данной книге 11
Классификация содержания 12

A. ARIS — МОДЕЛИРОВАНИЕ БИЗНЕС-ПРОЦЕССОВ 15

A.I. Стратегический анализ бизнес-процессов 22

A.I.1 Стратегическое моделирование бизнес-процессов 22
A.I.2 Метод PROMET 33
A.I.3 Другие методы стратегического моделирования бизнес-процессов 34

A.II. Моделирование отдельных аспектов ARIS 36

A.II.1 Моделирование функционального аспекта 36
A.II.1.1 Описание требований функционального аспекта 37
A.II.1.2 Конфигурирование функций 53
A.II.1.3 Спецификация дизайна функционального аспекта 54
A.II.1.4 Внедрение функционального аспекта 63
A.II.2 Моделирование организационного аспекта 65
A.II.2.1 Описание требований организационного аспекта 65
A.II.2.2 Конфигурирование организаций 70
A.II.2.3 Спецификация дизайна организационного аспекта 71
A.II.2.4 Внедрение организационного аспекта 74
A.II.3 Моделирование данных 77
A.II.3.1 Описание требований аспекта данных 77
A.II.3.2 Конфигурация данных 85
A.II.3.3 Спецификация дизайна данных 88
A.II.3.4 Внедрение модели данных 96
A.II.4 Моделирование результата 100
A.II.4.1 Описание требований результата 101
A.II.4.2 Конфигурация результата 105

A.III. Моделирование взаимосвязей между аспектами (управление) 108

A.III.1 Взаимосвязи функций и структуры организации 108
A.III.1.1 Описание требований 109
A.III.1.2 Конфигурация 113
A.III.1.3 Спецификация дизайна 115
A.III.2 Взаимосвязи функций и данных 117
A.III.2.1 Моделирование описания требований 118

A.III.2.2 Конфигурация 139
A.III.2.3 Спецификация дизайна 141
A.III.2.4 Описание внедрения 150
A.III.3 Взаимосвязи функций и результата 151
A.III.3.1 Моделирование описания требований 151
A.III.3.2 Конфигурация 155
A.III.4 Взаимосвязи организации и данных 156
A.III.4.1 Моделирование описания требований 156
A.III.4.2 Конфигурация 158
A.III.4.3 Спецификация дизайна 159
A.III.5 Взаимосвязи организации и результата 165
A.III.5.1 Моделирование описания требований 165
A.III.5.2 Конфигурация 167
A.III.6 Взаимосвязи данных и результата 169
A.III.6.1 Моделирование описания требований 169
A.III.6.2 Конфигурация 171
A.III.7 Уточненные модели всех аспектов ARIS 173
A.III.7.1 Моделирование описания требований 173
A.III.7.2 Конфигурация 175
A.III.7.3 Спецификация дизайна 177

B. Процедуральные модели ARIS и избранные приложения 179

B.I. Внедрение стандартных бизнес-приложений с использованием моделей ARIS 180

B.I.1 Решение критических вопросов, связанных с внедрением стандартного приложения 180
B.I.2 ARIS Quickstep for R/3 181
B.I.3 ARIS Quickstep for R/3: фазы внедрения SAP 182
Предложение (заказ) 182
Прототип концептуального дизайна 183
Внедрение 185
Непрерывное усовершенствование 185
B.I.4 Выводы 185

B.II. Внедрение систем управления потоком с использованием моделей ARIS 187

B.II.1 Факторы успешного внедрения систем управления потоком 187
B.II.2 Процедуральные модели ARIS для внедрения систем управления потоком 188

B.III. Использование ARIS Framework для разработки приложений 192

B.III.1 Общая процедуральная модель 192
B.III.2 Процедуральная модель для моделирования целевых концепций 194

Создание модели процесса 194
Определение бизнес-объектов 195
Создание архитектуры приложения из EPC 197

B.IV. Разработка объектно-ориентированных систем с использованием UML 199

B.IV.1 Проектирование и описание процедуральных моделей 199
B.IV.2 Этапы проектирования процедуральной модели 201
Объектно-ориентированный анализ 201
Объектно-ориентированное проектирование 202
Объектно-ориентированная сборка 202
B.IV.3 Выводы 203

Расшифровка аббревиатур 205

Ссылки 208

Предметный указатель 220

Предисловие к третьему изданию

Мы пришли к выводу о необходимости нового издания книги “ARIS — Business Process Modeling” всего через год после выхода в свет полностью переработанного второго издания, которое стало необычайно популярным.

Так как с момента предыдущего издания прошел короткий период времени, потребовалось лишь несколько формальных поправок. Основные цели и содержание второго издания не изменились.

Я выражаю благодарность дипломированному специалисту по торговле (Dipl.-Kff.) Урсуле Маркус за помощь при внесении поправок.

Саарбрюкен, Германия, октябрь 1999
Август-Вильям Шеер

Предисловие ко второму изданию

Со времени первой публикации в 1992 году книги “Architecture of Integrated Information Systems” методология ARIS становится все более популярной. Многие убедились в том, насколько удобно и эффективно документировать стандартное программное обеспечение с помощью бизнес-моделей. Программная система ARIS Toolset, разработанная *IDS Prof. Scheer GmbH* на основе концепции ARIS, сегодня признана мировым лидером на рынке инструментов моделирования бизнес-процессов. ARIS Toolset используется в Америке, Европе, Южной Африке, Бразилии и Тихоокеанской Азии, обеспечивая ученых и практиков в сфере организационных и информационных технологий современной методологией проектирования бизнес-процессов и управления ими.

Бурное развитие информационных технологий в период после выхода в свет первого издания этой книги привело к необходимости полностью переработать ее и разбить на две книги: “ARIS — Business Process Framework” и “ARIS — Business Process Modeling”.

Мы считаем, что у каждой из этих книг есть свой круг читателей. Если первая адресована тем, кто интересуется возможностями стандартного программного обеспечения, то вторая посвящена детальному изучению методов моделирования и информационного обеспечения.

О данной книге

В данной книге предлагаются методы моделирования бизнес-процессов, Метаструктуры методов моделирования описываются и объединяются в детализированные информационные модели ARIS.

В данном издании рассматривается намного более широкий круг методов моделирования по сравнению с первым изданием. В частности, детальнее анализируются моделирование стратегических бизнес-процессов, объектно-ориентированное моделирование и возможности универсального языка моделирования UML.

Согласно стандартизации языка UML метамодели ARIS представлены в виде диаграмм классов в нотации UML, хотя при анализе содержания процесса используются модели "сущность–связь", как это было и в первом издании.

Мы стремились сделать все для того, чтобы эти модели применялись для создания бизнес-приложений. Поэтому мы уделили особое внимание обсуждению бизнес-приложений, бизнес-объектов, различных возможностей создания программного обеспечения для планирования бизес-процессов и управления ими в соответствии с концепцией ARIS.

Обсуждение концепций мы дополнили прикладными главами, посвященными внедрению стандартных бизнес-приложений и систем управления потоком с использованием моделей ARIS, применению ARIS Framework для разработки приложений, а также созданию объектно-ориентированных систем с использованием UML.

Потенциальные читатели этой книги — это менеджеры по информационным технологиям, исследователи, преподаватели и студенты в области компьютерных наук и их применения в бизнесе.

Я выражаю благодарность дипломированному специалисту по торговле (Dipl.-Kff.) Урсуле Маркус за перевод с немецкого языка на английский, дипломированному специалисту по торговле (Dipl.-Kfm.) Франку Габерману за тщательное редактирование рукописи на немецком языке, кандидата экономических наук (cand. rer. oec.) Штефану Андресу и кандидата технических наук (cand. rer. inform.) Йогану Кунцу за подготовку англоязычных иллюстраций. Весомую техническую помощь предоставили дипломированный специалист по информатике (Dipl.-Inform.) Томас Фельд, дипломированный специалист по торговле (Dipl.-Kfm.) Дженс Хагемейер, дипломированный преподаватель коммерческих наук (Dipl.-Hdl.) Майкл Гоффман и доктор Маркус Нуттгенс.

Я также признателен г-ну Кристиану Тьюс за скрупулезный перевод текста на английский язык.

Саарбрюкен, Германия, октябрь 1998
Август-Вильям Шеер

Классификация содержания

Книги данного автора классифицируются по определенному принципу (рис. I).

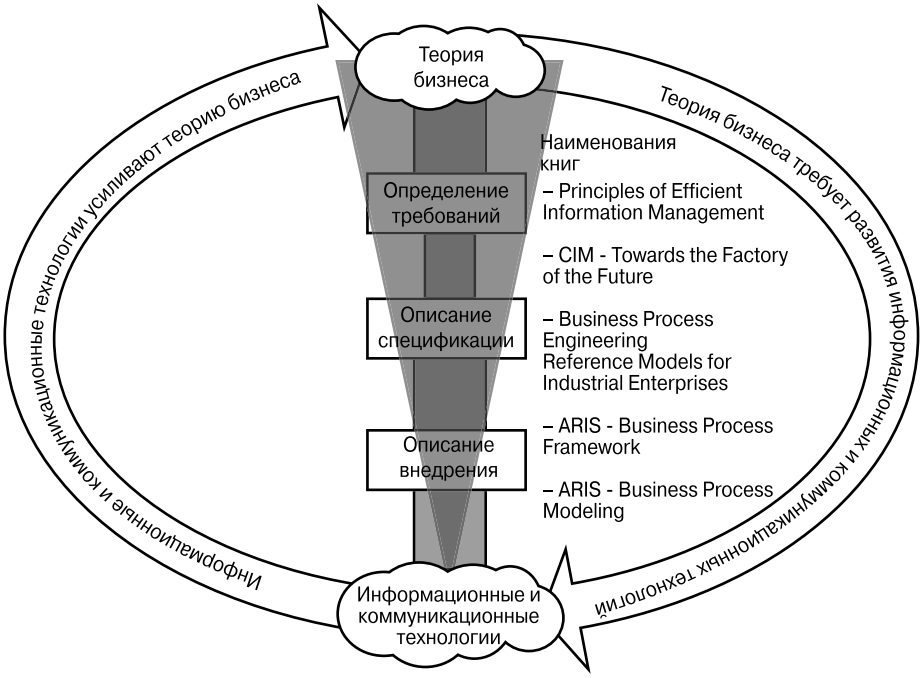


Рис. I. Классификация содержания книг данного автора

Компьютерные науки, ориентированные на применение в бизнесе, осуществляют двустороннюю взаимосвязь между теорией бизнеса и информационными и коммуникационными технологиями. С одной стороны, информационные и коммуникационные технологии способствуют появлению новых концепций теории бизнеса. Это направление взаимосвязи иллюстрирует левая стрелка на рис. I. В компьютерной науке не обязательно знать информационную технологию в полном объеме, достаточно использовать сегмент технологии, ответственный за изменение концепций бизнес-приложения.

Правая стрелка на рис. I иллюстрирует влияние требований теории бизнеса на развитие информационных и коммуникационных технологий.

Оба направления взаимосвязи обсуждаются в книге “Principles of Efficient Information Management”, второе издание которой опубликовано в 1991 году.

Ключевое влияние информационных технологий на бизнес-процессы анализируется в книге “CIM (Computer Oriented Manufacturing) — Towards the Factory of the Future”, которая вышла третьим изданием в 1994 году.

Названные книги создают прочный фундамент для проектирования и внедрения конкретных информационных систем, представляющих собой мост между бизнес-приложениями и информационными технологиями.

Книга “Architecture of Integrated Information Systems — ARIS” посвящена углубленному анализу информационных систем. Первое издание книги было опубликовано в 1992 году. Переработанная версия этой концепции опубликована в двух книгах: “ARIS — Business Process Framework” и “ARIS — Business Process Modeling”.

В книге “Business Process Engineering — Reference Models for Industrial Enterprises”, второе издание которой опубликовано в 1994 году, промышленным предприятиям предложена интегрированная информационная система с использованием моделей организации, функций, данных и процессов согласно концепции ARIS.

Стоимость описания информационных систем уменьшается по мере прогресса технической реализации. В то же время стабильность концепций также уменьшается, так как бурное развитие информационных технологий влияет на техническую реализацию информационных систем. Во всех своих книгах автор принимает во внимание этот факт в той степени, в которой рассматриваются соответствующие вопросы. Это иллюстрирует затененный треугольник на рис. I.

Все книги автора доступны на немецком языке. Книга “Business Process Engineering” переведена на китайский, а “CIM” — на португальский языки. Остальные книги готовятся к изданию на разных языках.

A

ARIS — МОДЕЛИРОВАНИЕ БИЗНЕС-ПРОЦЕССОВ

Разработчики системы ARIS (Architecture of Integrated Information Systems) подчеркивают четыре важнейших характеристики этой системы.

1. Концепция ARIS ("домик ARIS") — это *архитектурное средство для описания* бизнес-процессов.
2. Концепция ARIS предусматривает методы моделирования, метаструктуры которых воплощаются в *информационных моделях*.
3. Концепция ARIS *лежит в основе программной системы* ARIS Toolset — эффективного средства моделирования, разработанного *IDS Prof. Scheer GmbH*.
4. Концепция ARIS применительно к бизнес-инжинирингу (ARIS house of business engineering; HOBE) позволяет обеспечить *эффективное управление бизнес-процессами* с использованием современных информационных технологий.

Концепция ARIS обсуждается в книге *ARIS — Business Process Frameworks 1998*. В этой книге предложен подход, позволяющий уменьшить сложность описания бизнес-процессов за счет выделения *аспектов описания* и различных *фаз жизненного цикла* модели. Концепция ARIS (часто упоминаемая как "домик ARIS") изображена на рис. 1,*а*.

Методы моделирования классифицируются в разрезе аспектов и уровней домика ARIS. Метаструктуры методов моделирования описываются и воплощаются в детализированных информационных моделях ARIS. Именно этому посвящена данная книга.

Концепция ARIS лежит в основе программной системы ARIS Toolset, разработанной *IDS Prof. Scheer GmbH* в 1992 году для создания и поддержки компьютерных моделей. На рис. 1,*б* изображен интерфейс пользователя в среде ARIS-Easy Design.

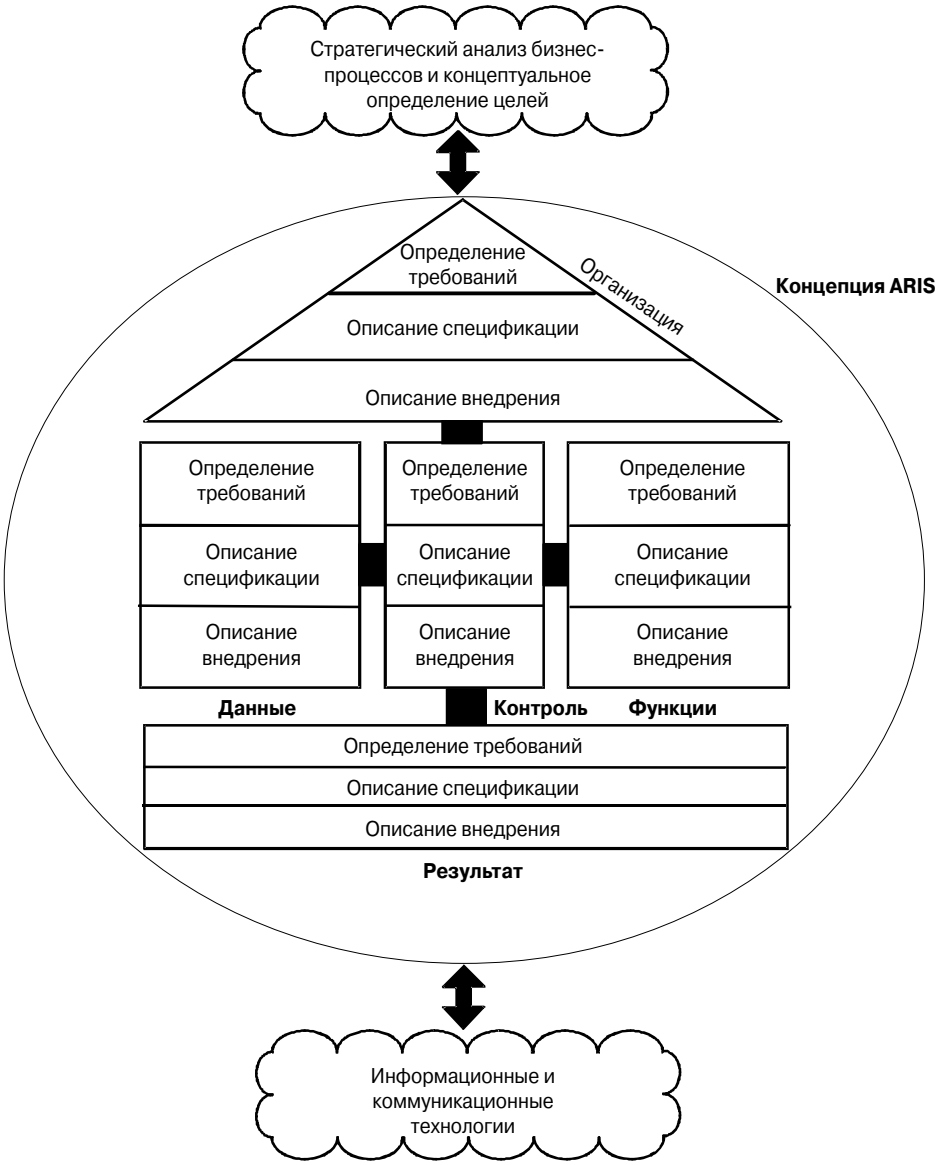


Рис. 1,а. Концепция ARIS ("домик ARIS")
(Scheer, ARIS — Business Process Frameworks 1998, Fig. 17)

ARIS Toolset пользуется неизменной популярностью и занимает первенство в различных рейтингах среди средств моделирования. Практически это общепризнанный мировой лидер в сфере моделирования. Несколько странно, что первые две характеристики системы ARIS, упомянутые в самом начале этой книги (*архитектурное средство* и *информационные модели*), иногда упускаются из виду при упоминании ARIS, под которой сразу подразумевается компьютерная программа ARIS Toolset. Однако мы полностью поддерживаем точку

зрения ряда авторов *(см. Bach/Brecht/Hess/Österle, Enabling Systematic Business Change 1996, p. 28)*, которые при обсуждении проектов моделирования утверждают, что методы моделирования всегда выбираются раньше, чем программные средства (“Сначала методы!”). Более того, архитектура должна обсуждаться перед выбором методов моделирования, при оценке пригодности и полноты методов. Следовательно, лозунг “Сначала методы!” мы дополняем лозунгом “Архитектура еще раньше!”.

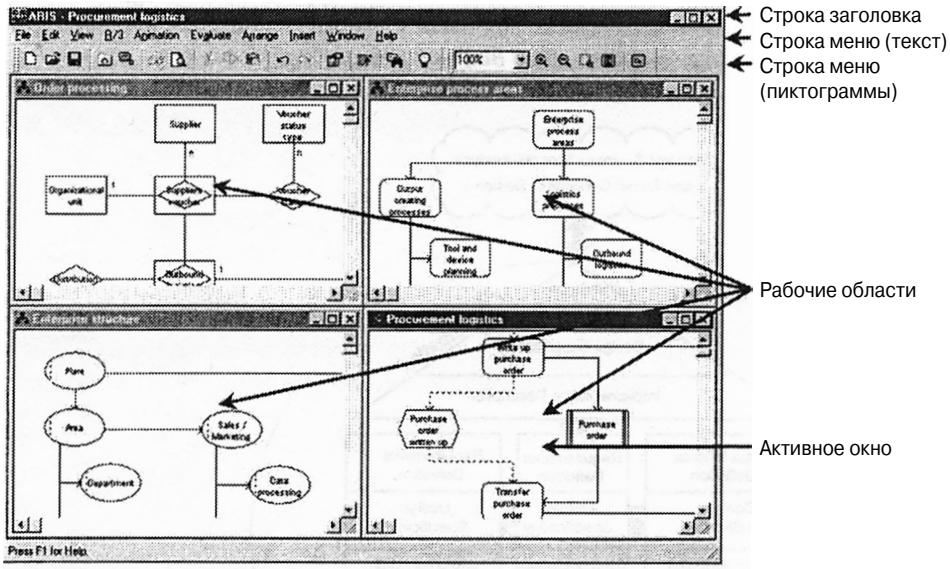


Рис. 1,б. Интерфейс пользователя в среде ARIS-Easy Design

Концепция ARIS применительно к бизнес-инжинирингу (рис. 1,*в*) показывает, как взаимосвязаны петли обратной связи — от создания бизнес-процессов до планирования и контроля, а затем до их внедрения посредством систем документооборота и функциональных компонент. В частности, взаимосвязь уровней I и II обеспечивает непрерывное усовершенствование процессов (continuous process improvement; CPI). Данная концепция детально обсуждается в книге *Scheer, ARIS — Business Process Frameworks 1998*.

Итак, вначале в книге обсуждаются методы стратегического моделирования бизнес-процессов, остальной материал структурирован в соответствии с концепцией ARIS — в разрезе аспектов и уровней домика ARIS. В пределах аспектов описания материал структурирован в разрезе фаз жизненного цикла модели: от определения требований до описания внедрения. Сначала мы детально рассмотрим методы определения требований, после чего усилим уровни конфигурации концепции HOBE от II до IV, исходя из моделей определения требований. Это иллюстрируют левые стрелки на рис. 2.

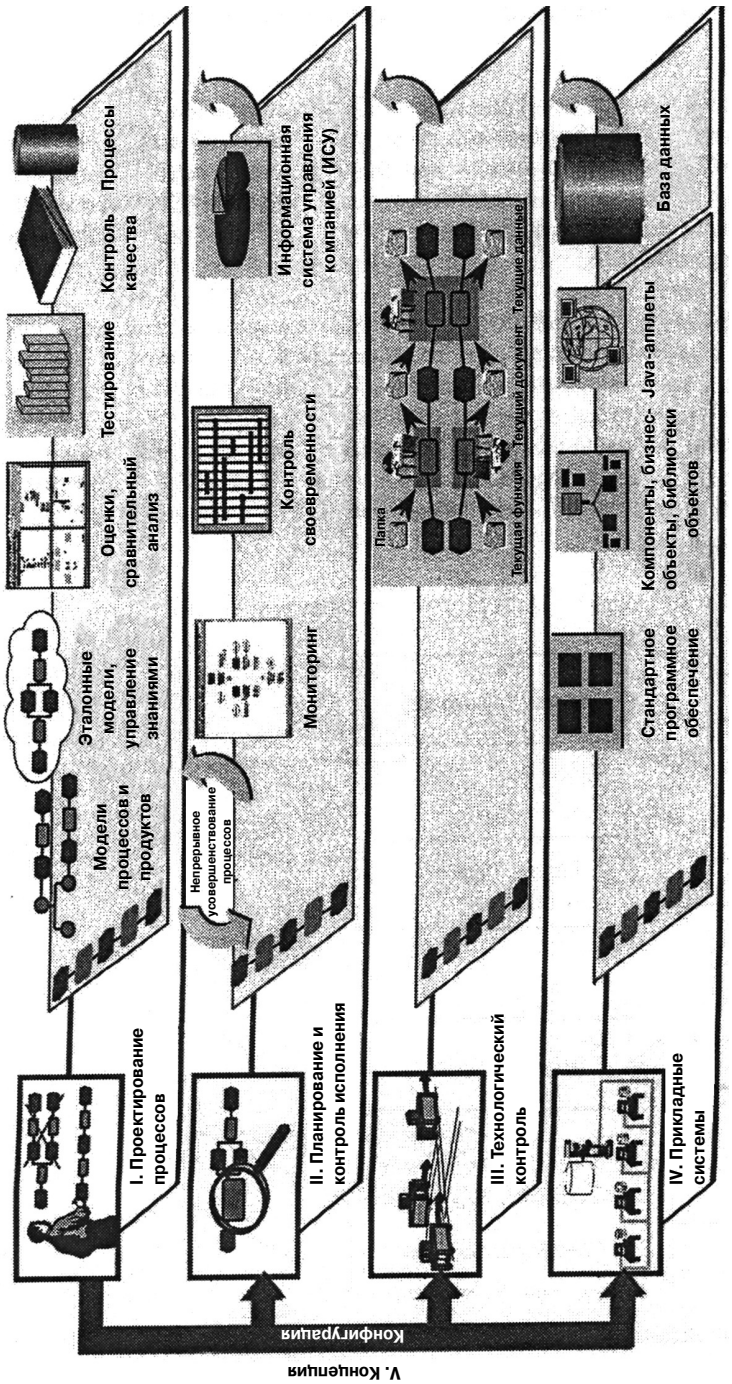


Рис. 1в. Управление процессами в соответствии с концепцией ARIS применительно к бизнес-инжинирингу
(Scheer, ARIS — Business Process Frameworks 1998, Fig. 24)

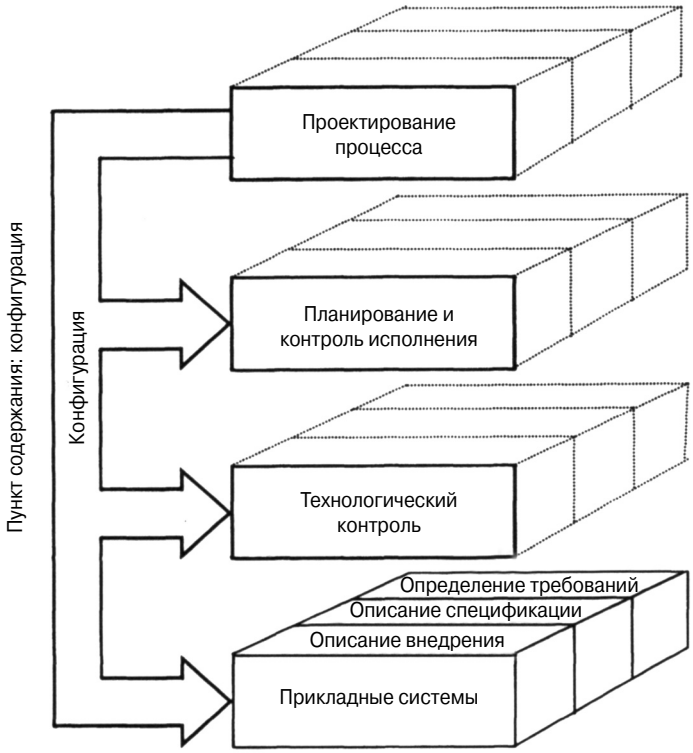


Рис. 2. Структура данной книги в соответствии с концепцией ARIS применительно к бизнес-инжинирингу

Таким образом, программное обеспечение на уровнях конфигурации от II до IV представляет собой некую оболочку. Для привязки к конкретным приложениям эту оболочку необходимо заполнить содержанием моделей бизнес-процессов, определяемых на уровне I.

Например, информационная система планирования пропускной способности хирургической больницы на уровне II сначала должна быть увязана с моделью лечебного процесса, прежде чем ее можно будет использовать для управления деятельностью операционной. Точно так же система планирования пропускной способности автомобильного парка должна быть согласована с моделью процесса автомобильных перевозок, прежде чем ее можно будет использовать для диспетчирования. Следовательно, принятые на первом уровне модели бизнес-процесса содержательные наименования ресурсов, таких как “хирург, аппарат искусственного дыхания и кровать” или “курьер, автомобиль и место на складе”, определяют наименования ресурсов системы планирования пропускной способности.

Таким же образом модели процессов используются для конфигурации систем управления потоком на уровне III и прикладных систем на уровне IV, что говорит о необходимости для этих систем соответствующего интерфейса. Современные стандартные приложения, такие как SAP R/3 или BAAN IV, снаб-

жены собственным интерфейсом, ориентированным на внутренние инструменты конфигурации и средства диалога с пользователем. В системах управления потоком используются специальные языки конфигурации, такие как FDL (flow defining language) в системе FlowMark фирмы IBM.

Учитывая тот факт, что конфигурация рассматривается на уровне модели бизнес-процесса, мы считаем конфигурацию вершиной описания требований. Поэтому мы обсудим вопросы конфигурации после анализа требований.

Затем мы рассмотрим применение бизнес-моделей для спецификации дизайна и описания внедрения, соответственно.

Если информационные системы применяются на всех четырех уровнях концепции HOBE, переход от анализа требований к описанию внедрения в общем случае требует вмешательства в программное обеспечение на каждом уровне. Эта процедура не зависит от типа используемой программной системы. На рис. 2 показаны фазы определения требований, спецификации дизайна и описания внедрения как последовательные этапы на уровне IV. Это же касается программного обеспечения на остальных уровнях. Следуя концепции автора о компьютерных науках, применяемых в бизнесе, в данной книге мы будем только поверхностно рассматривать вопросы внедрения.

Диаграммы классов в нотации унифицированного языка моделирования UML (Unified Modeling Language) используются для описания каждой метамодели *(см. UML Notation Guide 1997)*. Диаграммы классов похожи на иллюстрации в рамках модели "сущность–связь" (Entity Relationship Model; ERM) Чена, используемые в предыдущих изданиях данной книги *(см. Chen, Entity Relationship Model 1976)*.

На рис. 3 диаграммы классов в нотации языка UML, используемые в данном издании книги, сравниваются с иллюстрациями в рамках модели "сущность–связь". Диаграммы классов в нотации UML рассматриваются более детально в разделе A.III.2.1.1.1.

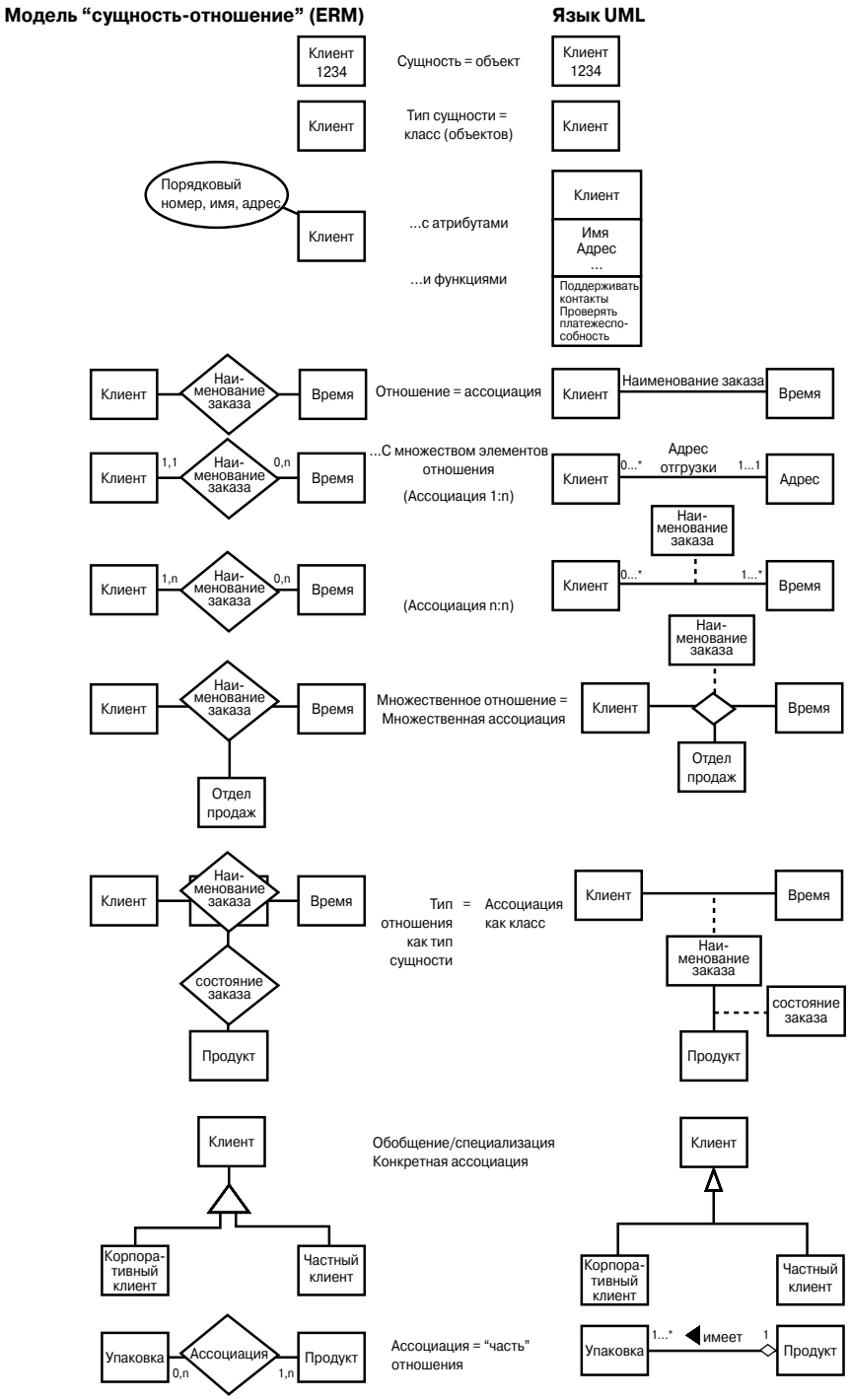


Рис. 3. Сравнение иллюстраций в рамках модели "сущность–связь" и диаграмм классов в нотации UML

A.I

Стратегический анализ бизнес-процессов

Организация, основанная на бизнес-процессах, — это новая парадигма в теории и практике управления предприятием. Стратегический анализ бизнес-процессов выполняется раньше построения домика ARIS (см. рис. 1а). Тэйлоризм (научный менеджмент) фокусировался на функциональной структуре, в основе которой лежал объем затраченных усилий, что, в свою очередь, вело к специализации работников. Организация бизнес-процессов, напротив, предполагает тщательный анализ синергических взаимосвязей функций в рамках единой процедуры. В этом случае целью является достижение наивысших стандартов квалификации работников, способных слаженно и эффективно выполнять бизнес-процессы.

Эти два базовых подхода к организации предприятия имеют стратегический характер, так как они определяют конкурентоспособность предприятия. Это наиболее очевидно с позиций философии "структура вслед за процессом, а процесс после стратегии" *(см. Osterloh/Frost, Prozeβmanagement 1996, p. 3)*. Поэтому для моделирования ключевых процессов применяются принципы стратегического планирования. Следовательно, стратегический анализ бизнес-процессов и концептуальное определение стратегических целей обнаруживают ключевые цели, сферы деятельности, позволяют распознать новые бизнес-процессы (которые нуждаются в разработке) и даже слабые места.

Стратегический анализ бизнес-процессов также помогает решить, какие новые информационные технологии следует использовать. По сути, он определяет организацию бизнес-процессов.

A.I.1 Стратегическое моделирование бизнес-процессов

Модель организации должна соответствовать различным требованиям эффективности, таким как эффективность ресурсов, эффективность процессов и эффективность рынка *(см. Frese, Grundlagen der Organisation 1995, p. 26)*. Предприятие считается рыночно эффективным, если оно полностью реализует потенциал рынка, а внутренние подразделения предприятия, работающие с клиентами, не взаимозависимы, что уменьшает потребность в координационных усилиях. Эффективность ресурсов означает эффективное использование

ресурсов организации, особенно человеческого капитала и операционных ресурсов. Эффективность процессов требует, чтобы процессы были направлены на достижение корпоративных целей.

Критерии эффективности в общем случае противоречат друг другу. Например, Гутенберг *(см. Gutenberg, Die Production 1983, p. 216)* исследует дилемму между целями минимизации времени выполнения (эффективность процессов) и максимальной загрузки мощностей (эффективность ресурсов).

На рис. 4,а схематически изображена структура предприятия, основанного на функциональном подходе, когда внутренние подразделения предприятия созданы для выполнения определенных функций. Теоретически такая организационная структура должна обеспечивать эффективность ресурсов. Однако в связи с необходимостью координировать различные функции, бизнес-процессы (от L1 до L4), такие как выполнение заказов, скорее всего, будут неэффективными.

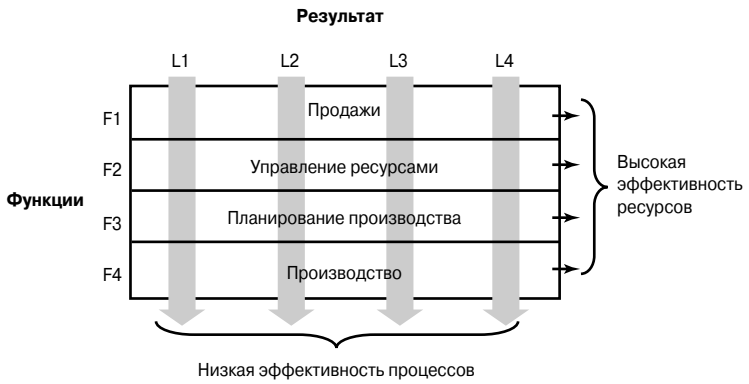


Рис. 4,а. Предприятие, основанное на функциональном подходе

Рис. 4,б иллюстрирует строение организации, основанной исключительно на бизнес-процессах, когда организационные единицы сгруппированы в соответствии с процессами. В таком случае эффективность процессов доминирует над эффективностью ресурсов. В свою очередь, на рис. 4в показана схема гибридного предприятия, которое стремится согласовать все три критерия эффективности, достигая эффективности рынка за счет отдела продаж на уровне предприятия (по принципу “one face to the customer” — “единая позиция во взаимоотношениях с клиентами”). На таком предприятии логистические процессы индивидуальных продуктовых групп имеют межфункциональный характер, что обеспечивает эффективность процессов, при этом ресурсы потенциальных факторов расходуются эффективно.

Несмотря на то что критерии эффективности иногда противоречат друг другу, ориентация на процессы является сильной стороной сегодняшнего предприятия. Одна из причин этого состоит в том, что потребители ценят эффективность корпоративных процессов (например, в логистике или разработке продуктов). Кроме того, руководство может контролировать корпоративные процессы, управляя целями — например, урезая процессы за счет определенного фактора.

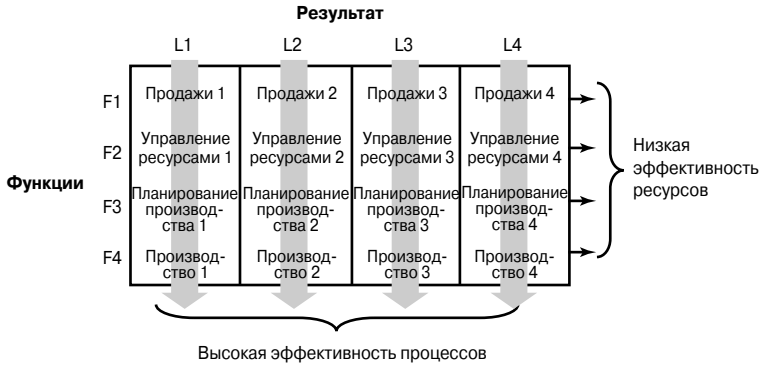


Рис. 4,б. Предприятие, основанное на бизнес-процессах

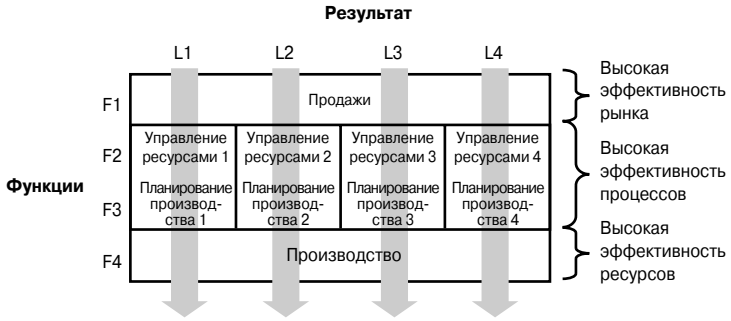


Рис. 4,в. Гибридное предприятие

Предприятия стремятся достичь рыночной эффективности, делая ударение на своей конкурентоспособности (рыночный критерий) и создавая стратегически важные бизнес-процессы (ключевые процессы). Эффективность ресурсов достигается, когда потенциальные факторы являются максимально гибкими, например, имеется возможность быстрой переориентации деятельности на другие процессы.

Ключевые процессы определяют конкурентоспособность предприятия, являются межфункциональными и взаимодействуют как с клиентами, так и с поставщиками. Две важнейшие группы ключевых процессов — это логистика заказов и создание продуктов. Процессы информационного обеспечения и координации группируются вокруг этих ключевых групп (рис. 5). Эта классификация имеет общий характер. Для конкретных целей процессы классифицируются более детально и в разных разрезах *(см. Osterloh/Frost, Prozeβmanagement 1996, p. 50)*. Ключевые процессы можно различать по степени сложности — например, при обработке заказов классифицировать заказы по срочности (срочные, приоритетные и обычные) или по группам клиентов (корпоративные или частные).

Для анализа рыночной эффективности (конкурентоспособности) можно использовать принципы стратегического планирования *(см. Krcmar, Informationsmanagement 1997, p. 203)*.

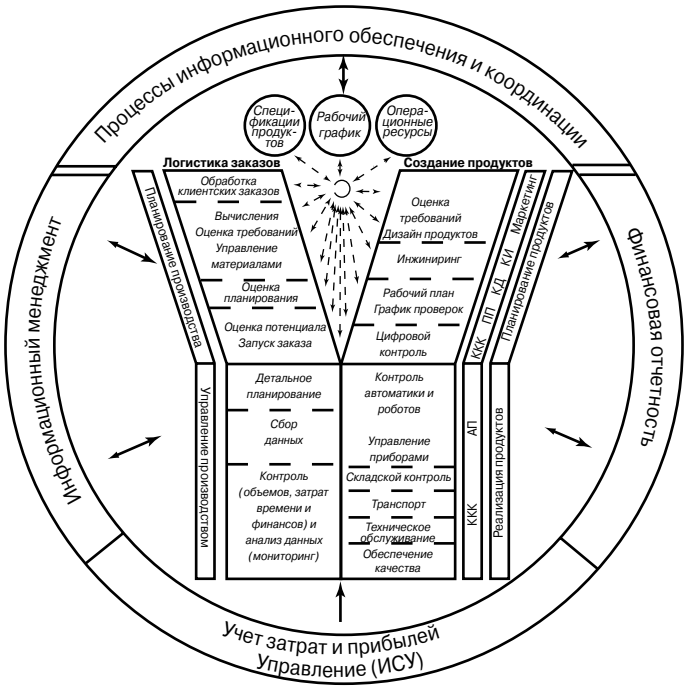


Рис. 5. Ключевые и вспомогательные процессы
(Scheer, Business Process Engineering 1994, p. 83)

Один из интересных подходов — это использование критических факторов успеха по Рокарту *(см. Rockart, Critical Success Factors 1982)*. Критические факторы успеха — это важнейшие для успеха корпорации цели, такие как высокие стандарты качества, образцовый уровень доставки, конкурентные преимущества в научно-исследовательской деятельности или высокая степень гибкости. Бизнес-процессы должны быть сфокусированы на достижении этих целей.

Цепочка создания стоимости по Портеру (рис. 6) демонстрирует значение основных и вспомогательных функций. Основные виды деятельности представляют собой функции, направленные непосредственно на создание и реализацию продукта корпорации. Вспомогательные виды деятельности поддерживают основные функции, обеспечивая инфраструктуру, развитие технологии или управление персоналом. Анализ взаимосвязи основных и вспомогательных видов деятельности ведет к созданию предварительного варианта структуры процесса, что можно рассматривать как первый шаг к организации бизнес-процессов.

Рис. 7 иллюстрирует реальный пример структуры бизнес-процесса. На схеме изображены подпроцессы и их взаимосвязи. Базовая разработка поддерживает все выходные поля производства готовых изделий и запасных частей. Процессы обслуживания клиентов идентичны для всех выходных полей, в то время как бизнес-процессы обработки заказов разделены на два варианта — производство запасных частей и производство готовых изделий.

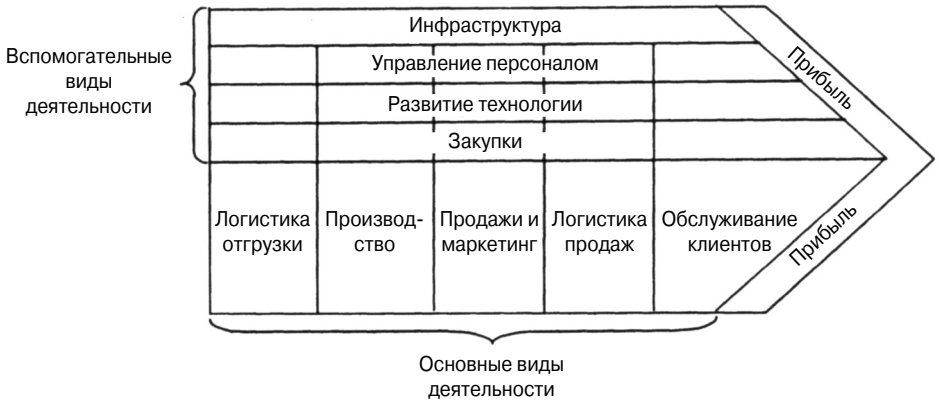


Рис. 6. Цепочка создания стоимости по Портеру
(Porter, Wettbewerbsvorteile 1992, p. 62)

На рис. 8 изображена стратегическая метамодель бизнес-процесса.

Класс ПАРАМЕТР ЭФФЕКТИВНОСТИ состоит из показателей эффективности организации (эффективности процессов, ресурсов и рынка). Иерархические сегменты, показанные на рис. 4а–4в, составляют класс ОРГАНИЗАЦИОННЫЙ СЕГМЕНТ. Этот класс разбит на два подкласса: ФУНКЦИОНАЛЬНЫЙ СЕГМЕНТ, имеющий горизонтальную структуру, и СЕГМЕНТ БИЗНЕС-ПРОЦЕССОВ, который соответствует вертикально структурированному предприятию. Следовательно, каждому организационному сегменту соответствует поле на рис. 4в. Вклад организационных сегментов в достижение эффективности предприятия оценивается в классе связей ВКЛАД В ЭФФЕКТИВНОСТЬ.

Вклад организационных сегментов в эффективность можно также изобразить в виде таблицы — или словесно, или, если возможно, численно. Таблица на рис. 9 соответствует рис. 4в.

Класс ОРГАНИЗАЦИОННЫЙ СЕГМЕНТ через подклассы ФУНКЦИОНАЛЬНЫЙ СЕГМЕНТ и СЕГМЕНТ БИЗНЕС-ПРОЦЕССОВ связан с классами ФУНКЦИЯ ДОБАВЛЕННОЙ СТОИМОСТИ и ЭЛЕМЕНТ ВЫПУСКА.

Класс ФУНКЦИЯ ДОБАВЛЕННОЙ СТОИМОСТИ включает вспомогательные функции, создающие добавленную стоимость (см. рис. 6).

Класс ЭЛЕМЕНТ ВЫПУСКА включает важнейшие взаимосвязанные элементы выпуска (рис. 10).

Функционально ориентированные организации поддерживают лишь несколько ассоциаций (по крайней мере, одну) между классами ФУНКЦИОНАЛЬНЫЙ СЕГМЕНТ и ФУНКЦИЯ ДОБАВЛЕННОЙ СТОИМОСТИ и множественные ассоциации между классами ФУНКЦИОНАЛЬНЫЙ СЕГМЕНТ и ЭЛЕМЕНТ ВЫПУСКА. Организации, ориентированные на процессы, напротив, имеют несколько ассоциаций (по крайней мере, одну) между классами СЕГМЕНТ БИЗНЕС-ПРОЦЕССОВ и ЭЛЕМЕНТ ВЫПУСКА и множественные ассоциации между классами СЕГМЕНТ БИЗНЕС-ПРОЦЕССОВ и ФУНКЦИЯ ДОБАВЛЕННОЙ СТОИМОСТИ.

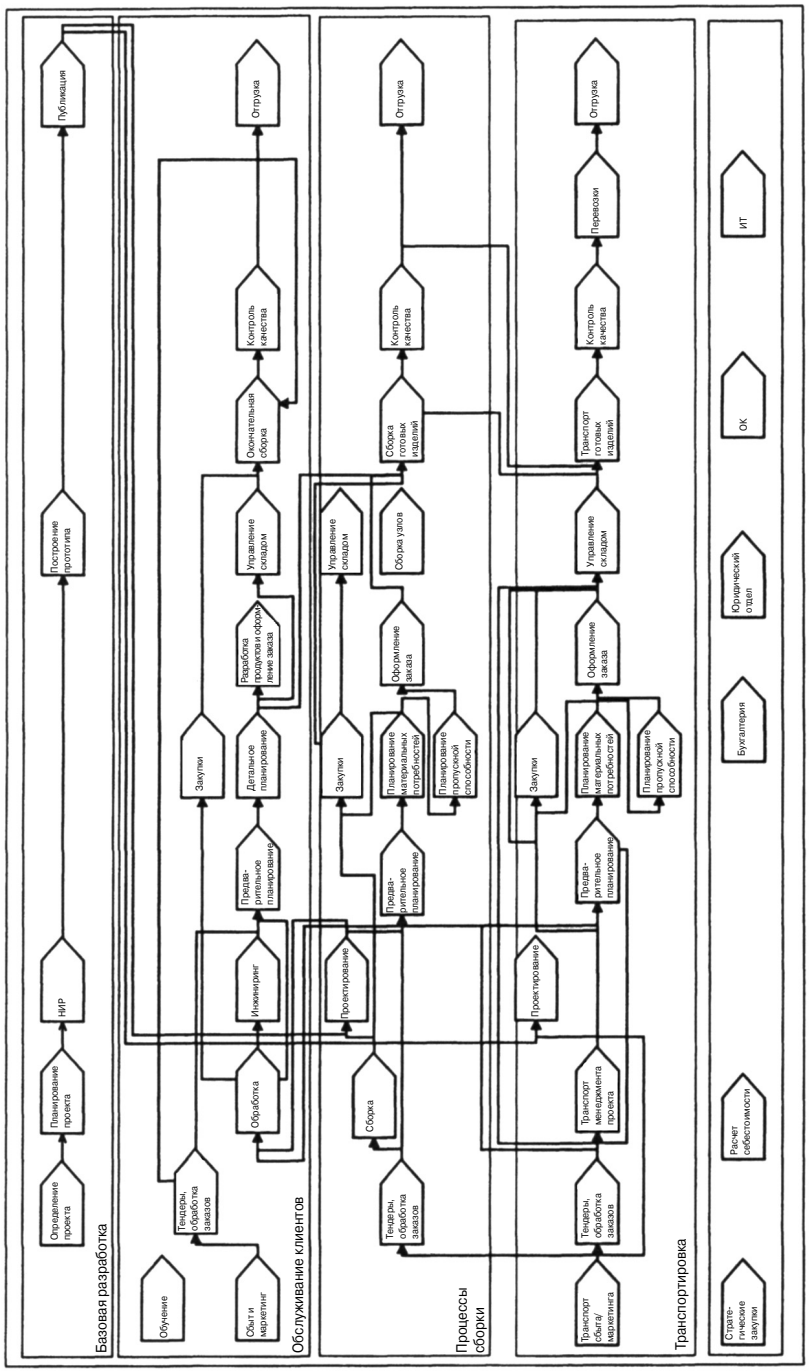


Рис. 7. Пример структуры бизнес-процесса
(Kirchmer, Definition von Geschäftsprozessen 1995, p. 271)

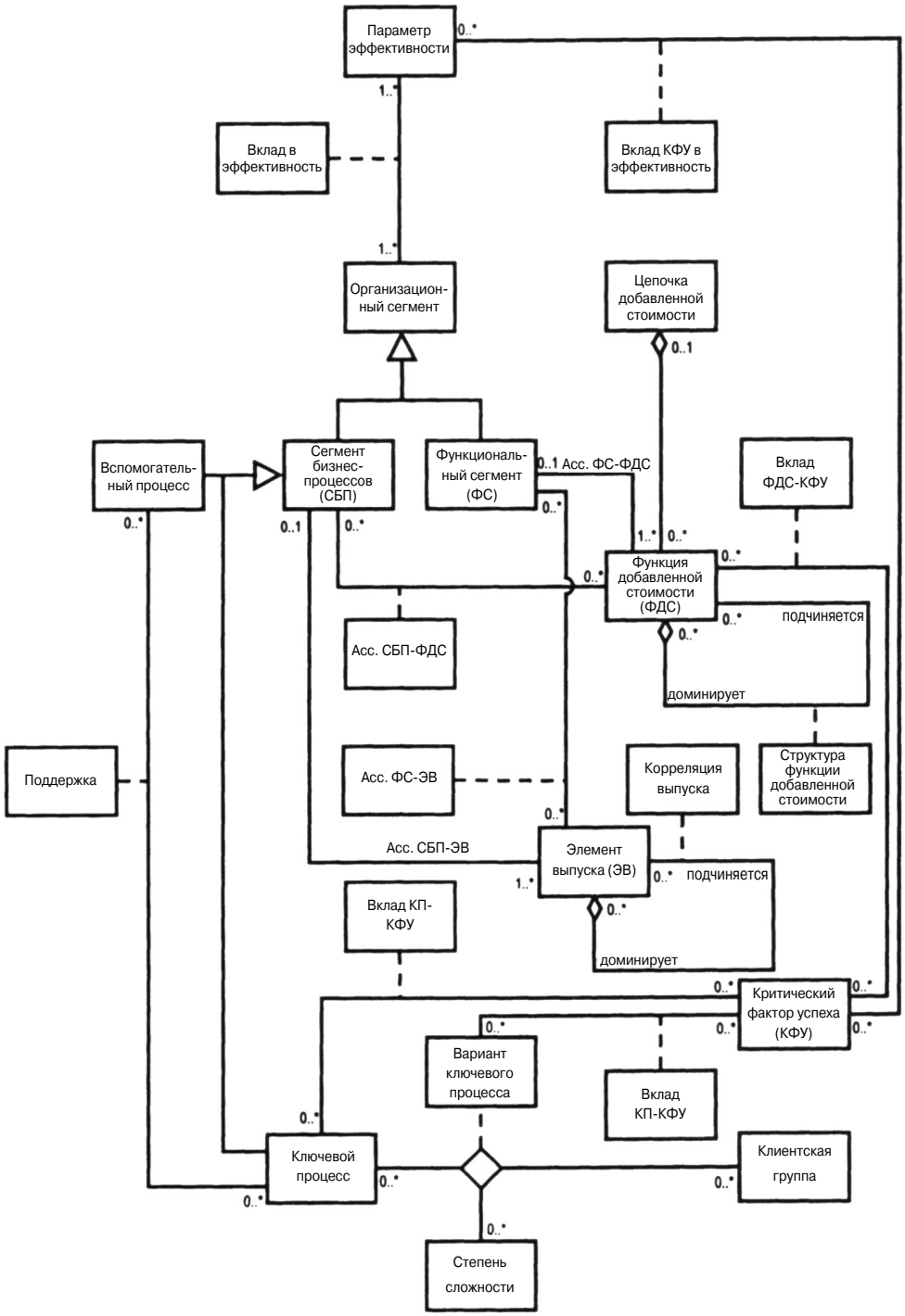


Рис. 8. Стратегическая метамодель бизнес-процесса

Параметр эффективности / Организационный сегмент	Эффективность рынка	Эффективность процесса	Эффективность ресурсов
Продажи	Высокий. Необходимы строгая координация, позиционирование на рынке по принципу “one face to the customer”	Средний. Необходима координация логистических систем	Успешное использование возможностей работников, рекламы и т.п.
Логистика заказов	Средний. Необходимы координация и централизованный сбыт	Высокий в отделе логистики, средний в смежных отделах сбыта и разработки	Средний. Ресурсы структурированы по отдельным сегментам, низкий потенциал для согласования.
Разработка продукции	Средний. Разработка направлена на все продукты, слабый учет запросов	Средний. Потиворечивые запросы мешают друг другу	Высокий. Все производственные ресурсы используются совместно

Рис. 9. Вклад организационных сегментов в эффективность предприятия

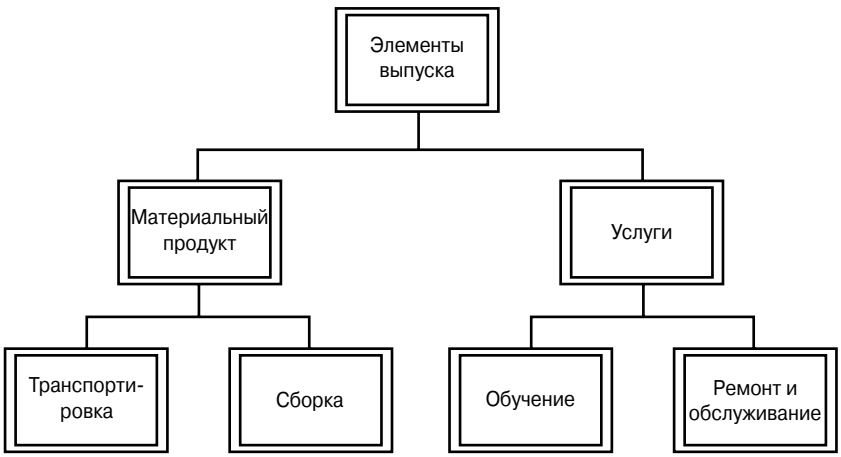


Рис. 10. Элементы выпуска

Взаимосвязь между классами ЭЛЕМЕНТ ВЫПУСКА и СЕГМЕНТ БИЗНЕС-ПРОЦЕССОВ показывает, какие бизнес-процессы необходимы, чтобы обеспечить выпуск (например, когда принимается решение о том, нужен ли

аутсорсинг). Если определенный элемент выпуска больше не создается, соответствующие процессы больше не нужны. В то же время, возникают новые процессы для обеспечения аутсорсинга. Корреляция между элементами выпуска выражается связью КОРРЕЛЯЦИЯ ВЫПУСКА. Таким образом, метамодель охватывает элементы выпуска, изображенные на рис. 10.

Взаимосвязи типа “предшественник-последователь” между функциями добавленной стоимости внутри определенных бизнес-процессов показаны связями СТРУКТУРА ФУНКЦИИ ДОБАВЛЕННОЙ СТОИМОСТИ. Таким образом обеспечивается фундамент для стратегического моделирования бизнес-процессов в метаструктуре (см. рис. 7).

В соответствии с моделью добавленной стоимости Портера бизнес-процессы классифицируются как КЛЮЧЕВОЙ ПРОЦЕСС и ВСПОМОГАТЕЛЬНЫЙ ПРОЦЕСС.

Критические факторы успеха можно отнести к ключевым процессам или к отдельным функциям добавленной стоимости, что отображается связями ВКЛАД КФУ В ЭФФЕКТИВНОСТЬ.

Варианты ключевых процессов, классифицируемые в соответствии с клиентскими группами или степенью сложности, создают связи между классами КЛЮЧЕВОЙ ПРОЦЕСС, КЛИЕНТСКАЯ ГРУППА и СТЕПЕНЬ СЛОЖНОСТИ. Различные оценки критических факторов успеха формируют класс ассоциаций ВАРИАНТ КЛЮЧЕВОГО ПРОЦЕССА, который впоследствии может стать основой для формирования целей усовершенствования процессов.

Взаимосвязи между процессами и критическими факторами успеха можно изобразить в виде таблицы. В таблице на рис. 11 представлены три варианта процесса обработки клиентских заказов. Увязывая критические факторы успеха и параметры эффективности, можно показать значение параметров эффективности в достижении успеха корпорации.

Критические факторы успеха / Варианты процесса Обработка клиентских заказов	Гарантии качества	Своевременная доставка	Экономическая эффективность
Обработка клиентских заказов Стандартные заказы	Высокое	Высокое	Очень высокое
Обработка клиентских заказов Индивидуальные заказы	Очень высокое	Очень высокое	Высокое
Обработка клиентских заказов Поставка запчастей	Исключительно высокое	Исключительно высокое	Среднее

Рис. 11. Значение критических факторов успеха для различных вариантов процессов

После идентификации варианты бизнес-процессов разделяются на отдельные операционные шаги с помощью диаграмм цепочки процесса. Иллюстрации остаются на стратегическом уровне, хотя они смешиваются с описанием требований. Иллюстрации процесса детализируются только в случае, когда нужно продемонстрировать главные стратегические недостатки текущего процесса и преимущества будущей процедуры. Бизнес-процессы изображаются в диаграммах цепочки как закрытые единицы (простые варианты диаграмм цепочки процесса были представлены автором в *Sheer, Principles of Efficient Information Management, 1991, p. 22*).

Рис. 12а и 12б иллюстрируют предварительные процедуры для ситуаций "как есть" ("as-is") и "как должно быть" ("to be") в бизнес-процессе обработки заказов. Диаграммы цепочки процесса показывают сокращенные версии ключевых взаимосвязей в ARIS модели бизнес-процесса. Представление в виде таблицы делает диаграммы событийно-управляемых процессов более читабельными по сравнению со свободным рисунком, однако в то же время это затрудняет отображение сложных структур, таких как петля.

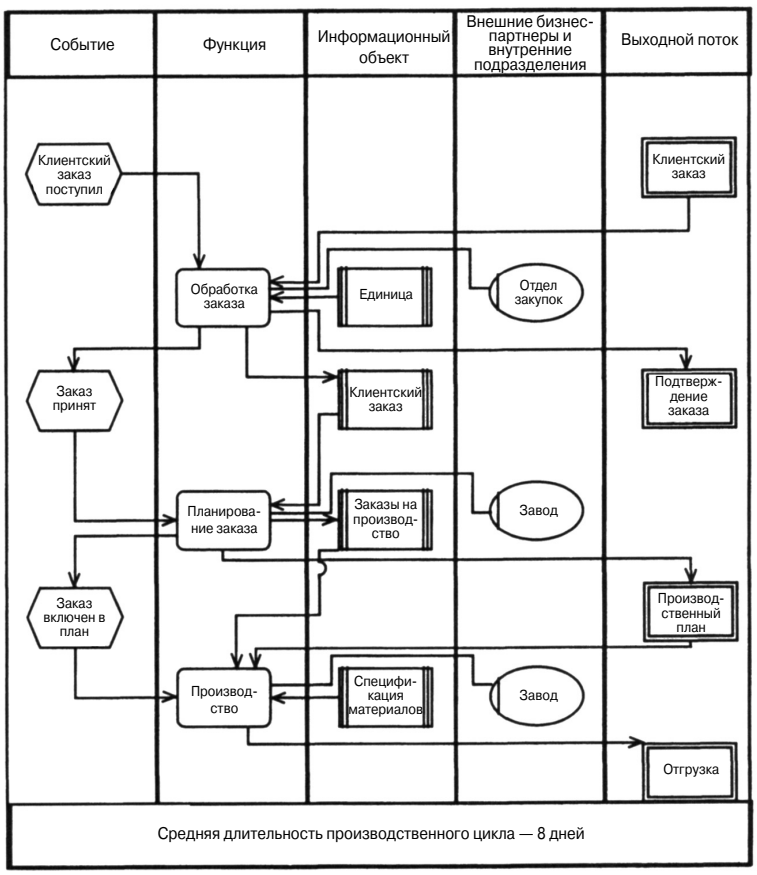


Рис. 12,а. Стратегическая диаграмма цепочки процесса в ситуации "как есть"

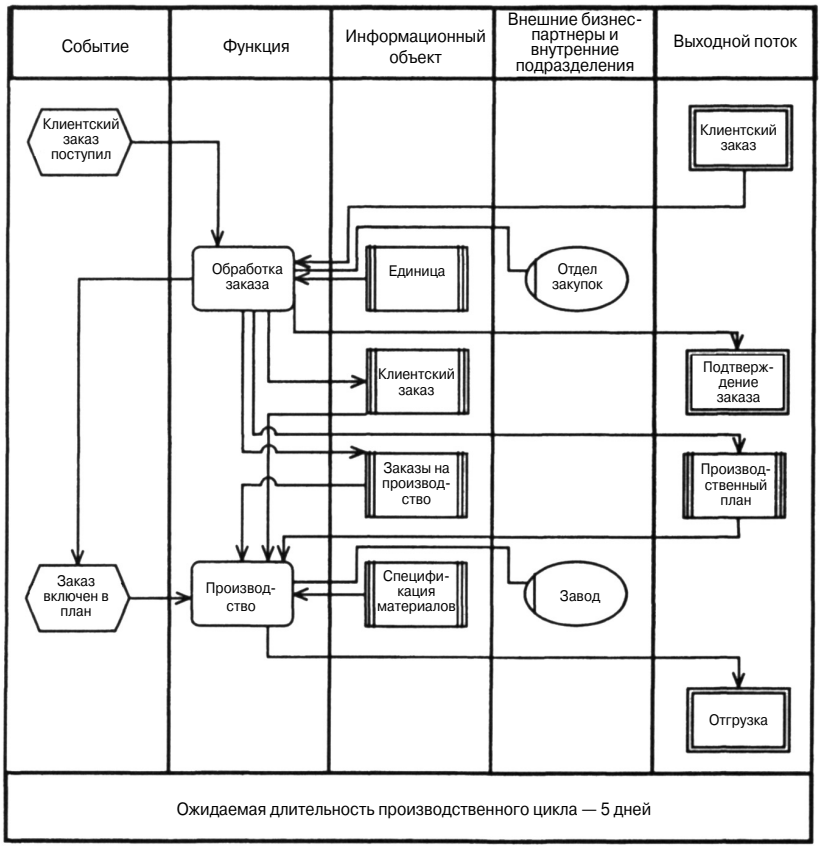


Рис. 12,б. Стратегическая диаграмма цепочки процесса в ситуации "как должно быть"

Первые две колонки иллюстрируют поток управления. Информационные объекты (третья колонка) характеризуют поток данных. Внешние бизнес-партнеры и внутренние организационные единицы, участвующие в процессе, перечислены в колонке 4. Выходной поток процесса показан в колонке 5.

Сравнивая диаграммы процессов "как есть" и "как должно быть", мы видим, что целевой процесс улучшен уже за счет того, что отделы продаж сравнивают новые клиентские заказы с существующим производственным планом. Исключение излишних внутренних коммуникаций и циркуляции данных между подразделениями позволяет улучшить ситуацию "как есть", где включение нового заказа в производственный план занимает несколько дней. При такой стратегии заметно сокращается общая длительность производственного цикла.

На этом этапе в диаграмму цепочки процесса можно включить общие понятия информационной системы, чтобы выявить серьезные противоречия или подчеркнуть цели стратегического информационного менеджмента в концепции "как должно быть".

Диаграммы цепочки процесса используются повторно при описании требований и проектировании метаструктуры.

A.I.2 Метод PROMET

Метод PROMET был разработан в университете Санкт Галлен в Швейцарии *(см. Information Management Gesellschaft, PROMET 1994; Österle, Business Engineering 1 1995; Österle/Volger, Praxis des Workflow-Managements 1996)*. PROMET включает различные сетевые диаграммы, где определенными стрелками указаны взаимосвязи различных элементов и их удельные веса. PROMET позволяет анализировать бизнесс-процессы и увязать стратегическое корпоративное планирование с информационными технологиями. Метод PROMET поддерживается средствами ARIS Toolset.

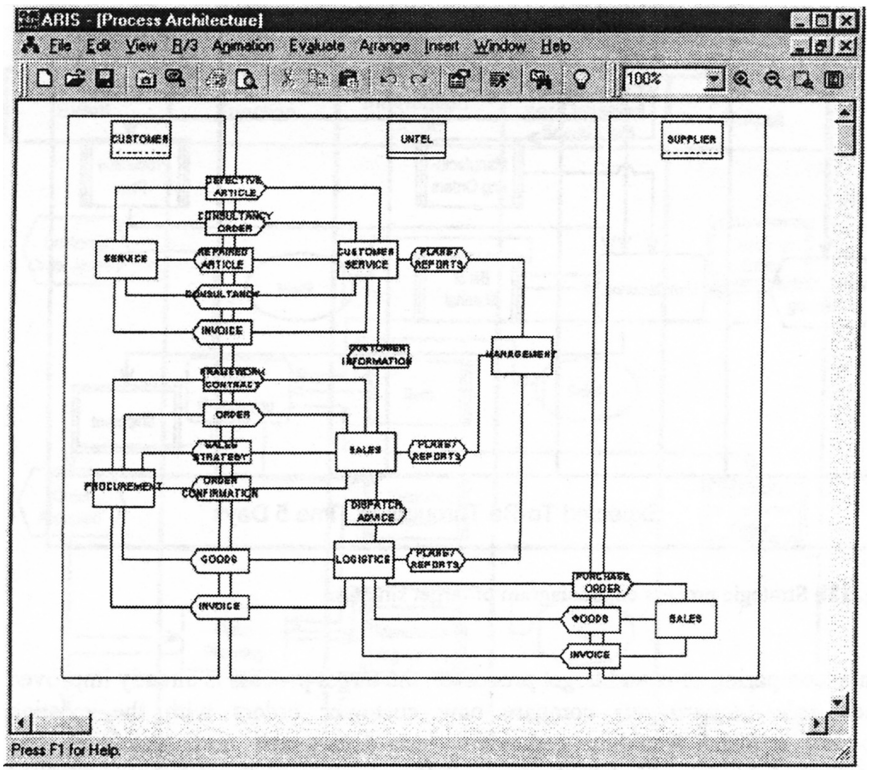


Рис. 13. Последовательность процесса согласно PROMET

(Bach/Brecht/Hess/Österle, Enabling Systematic Business Change 1996, p. 271)

Первый шаг — это определение корпоративной бизнес-стратегии, т.е. решений, касающихся альянсов, организационной структуры, сфер бизнеса и инструментов управления. Сети SWOT (Strengths, Weakness, Opportunities, Threats — сильные стороны, слабые стороны, возможности, угрозы) иллюстрируют взаимосвязи между конкурирующими подразделениями внутри предприятия. Секторные сети описывают различных участников рынка и их взаимоотношения.

Комбинации продуктов и рынков увязаны с каналами сбыта в матрице сферы бизнеса.

После того как корпоративная стратегия определена и описана, бизнес-процессы распределяются по сферам бизнеса. Затем продолжительность и качество отдельных компонентов процесса сравниваются с аналогичными характеристиками процессов конкурентов. Документ “цепочки задач” предварительно описывает последовательность процесса, где информационные технологии обеспечивают практическое внедрение (см. рис. 13). Показанную на рис. 13 схему можно сопоставить с диаграммами цепочки процессов концепции ARIS.

На рис. 14 изображена метамодель метода PROMET, которая содержит некоторые наглядные дополнения к метамодели, представленной на рис. 8.

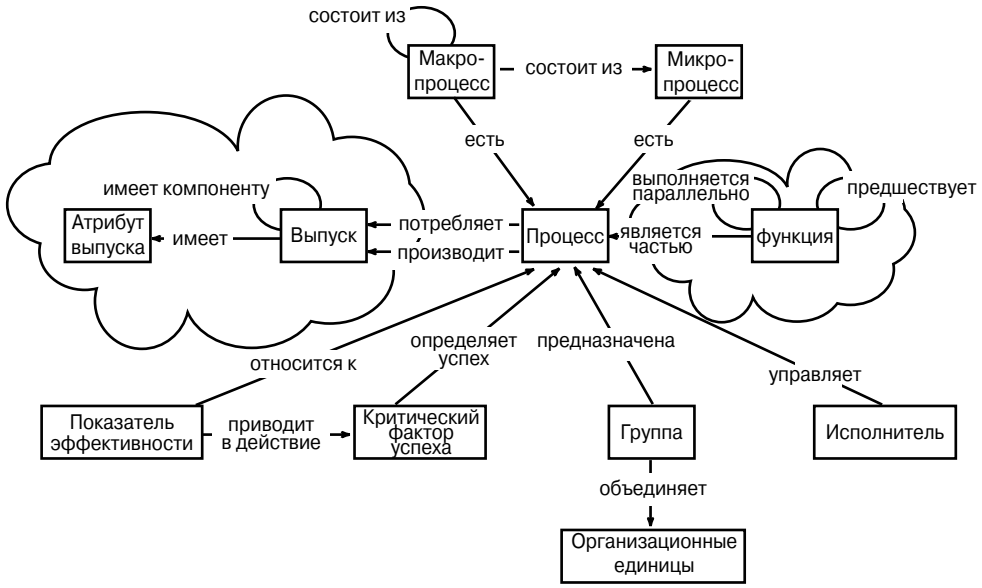


Рис. 14. Метамодель PROMET

Преимущество PROMET состоит в строгом структурировании объектов и последовательном использовании стратегических понятий (концепции Портера). Используя преимущества компьютерных технологий, PROMET позволяет документировать моделирование бизнес-процессов. Очевидно, для усовершенствования процессов необходимо увязать PROMET и критические факторы успеха (цели).

A.I.3 Другие методы стратегического моделирования бизнес-процессов

Существует ряд других методов оценки, приоритизации и стратегического инжиниринга бизнес-процессов. Многие из этих методов используют графические иллюстрации или таблицы. Вот несколько самых популярных методов:

- портфельный анализ бизнес-процессов *(см. Mc Farlan/Mc Kenney/Pyburn, Information Archipelago 1983)*;
- информационная интенсивность выходного потока, согласно Портеру *(см. Porter/Millar, How Information Gives You Competitive Advantage 1985)*;
- портфельные иллюстрации Boston Consulting Group.

A.II

Моделирование отдельных аспектов ARIS

A.II.1 Моделирование функционального аспекта

На рис. 15 показано, какое место в “домике ARIS” занимает блок под названием “Функции”. Функции часто описывают относительно других компонент. Они тесно связаны с данными, так как управленческие функции направлены на процесс преобразования входных данных в выходные данные. Функции также часто описывают в контексте с организационными объектами, особенно при составлении штатного расписания.

В концепции ARIS функции рассматриваются как отдельный аспект описания бизнес-процессов.

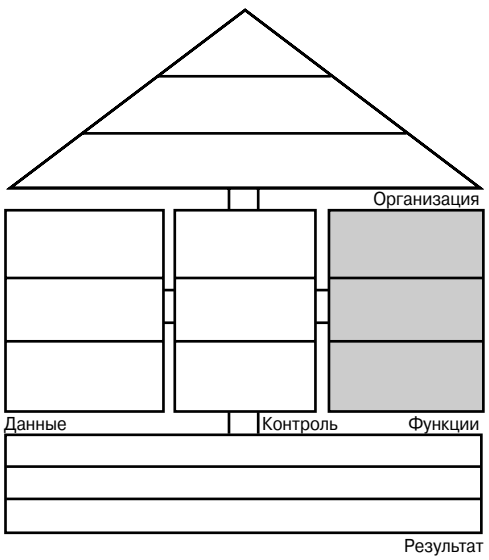


Рис. 15. Функциональный аспект концепции ARIS

A.II.1.1 Описание требований функционального аспекта

Стратегия бизнес-процесса предполагает функции, которые предприятие должно выполнять эффективно *(см. Mertens, Wirtschaftsinformatik 1995, p. 40)*. Понятие функции мы рассматриваем наряду с такими понятиями, как процесс, деятельность или задача. Наименование комплексной функции, такой как обработка заказов, используется для обозначения бизнес-процесса. Описание поведения функций, т.е. динамический контроль функций от начала до конца, составляет часть бизнес-процесса. Однако в простых описаниях функций главное внимание уделяется иллюстрированию статической структуры функций.

В иллюстрациях используются различные символы для обозначения функций. На рис. 16 приведены некоторые из этих символов. В данной книге мы используем прямоугольник с закругленными углами.

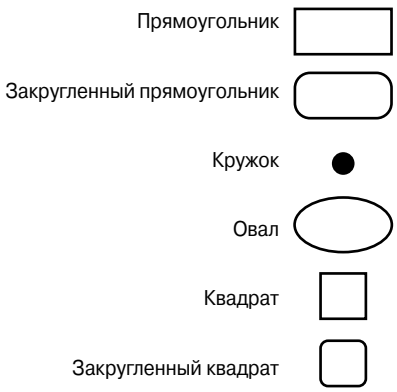


Рис. 16. Символы для обозначения функций в иллюстрациях
(Olle et al., Information Systems Methodologies 1991, p. 274)

При проектировании бизнес-процессов основу для моделирования функций составляет информационно-ориентированная стратегическая концепция, согласно которой цели должны поддерживаться функциями. Согласно Рокарту, цели соответствуют критическим факторам успеха *(см. Rockart, Critical Success Factors 1982)*.

Функции определяются как операции, применяемые к объектам для достижения одной или нескольких целей. Цели могут быть взаимосвязаны; различают доминирующие и подчиненные цели (рис. 17). Таким образом, структура взаимосвязанных целей в сети приводит к ассоциации *:* в классе ЦЕЛЬ (рис. 18). Чтобы различать две линии между классами ЦЕЛЬ и СТРУКТУРА ЦЕЛИ, мы употребили ролевые имена. Поскольку нет ни одной цели, превышающей самые высокие доминирующие цели, в то время как многие доминирующие цели превышают подчиненные цели, линия “доминирует” сопровождается обозначением (0..*) в нотации (min, max). Линия “подчиняется” сопровождается таким же обозначением, так как нет ни одной цели, которая бы подчинялась целям самого нижнего уровня.

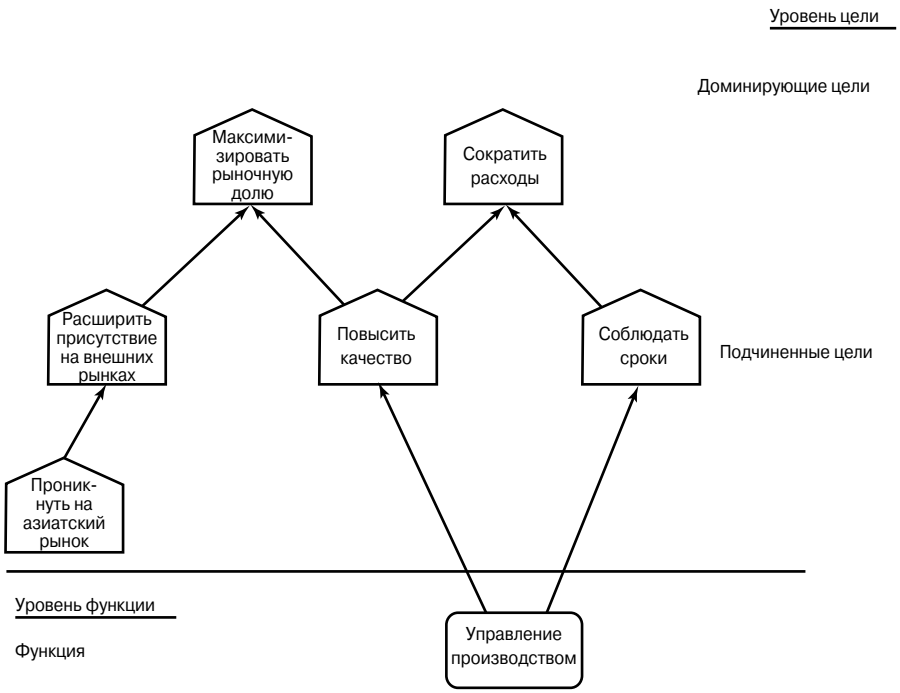


Рис. 17. Взаимосвязи целей и функций

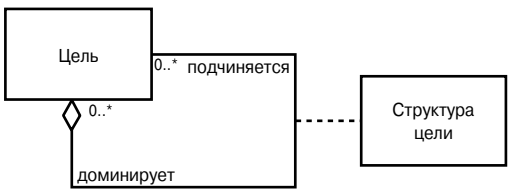


Рис. 18. Диаграмма классов для моделирования структуры целей

Функции могут поддерживать множество целей. Ассоциация между функциями и целями передается на верхние уровни иерархии, т.е. если некоторая функция поддерживает цель нижнего уровня, она поддерживает и вышестоящую цель в рамках данной связи. В примере на рис. 17 функция “Управление производством” поддерживает также доминирующую цель “Сократить расходы”.

A.II.1.1.1 Структура функций

Функции могут быть описаны с различной степенью детализации. Самый высокий уровень обобщения и, следовательно, стартовая позиция в нашем обсуждении, — это сложные функции, включающие много видов деятельности (complex function bundle — “клубок” функций). Чтобы упростить процесс, сложные функции структурируются, т.е. разделяются на подфункции, как показано на иерархических диаграммах. На рис. 19, а приведен пример иерархи-

ческой диаграммы для (преимущественно) функций преобразования информации. Рис. 19, б иллюстрирует (преимущественно) функции промышленного производства.

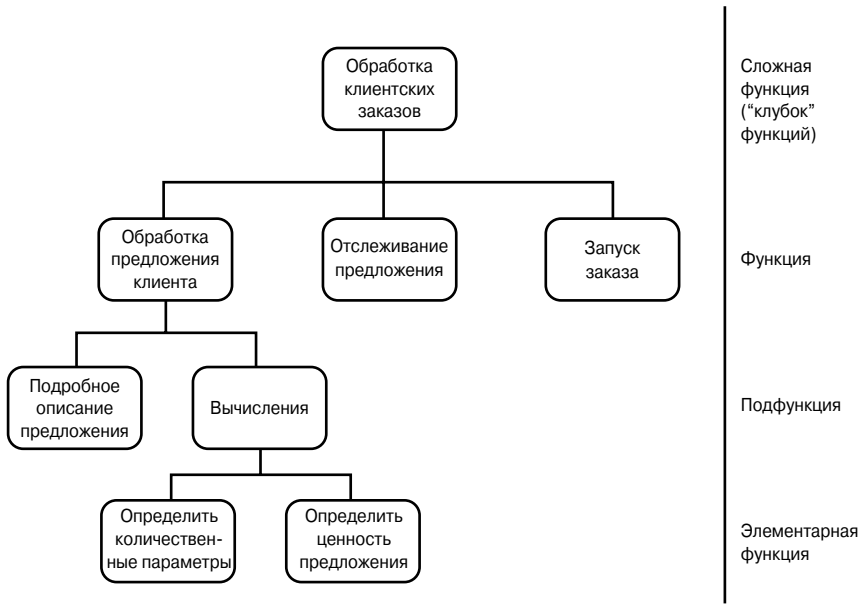


Рис. 19, а. Пример иерархической диаграммы функций преобразования информации

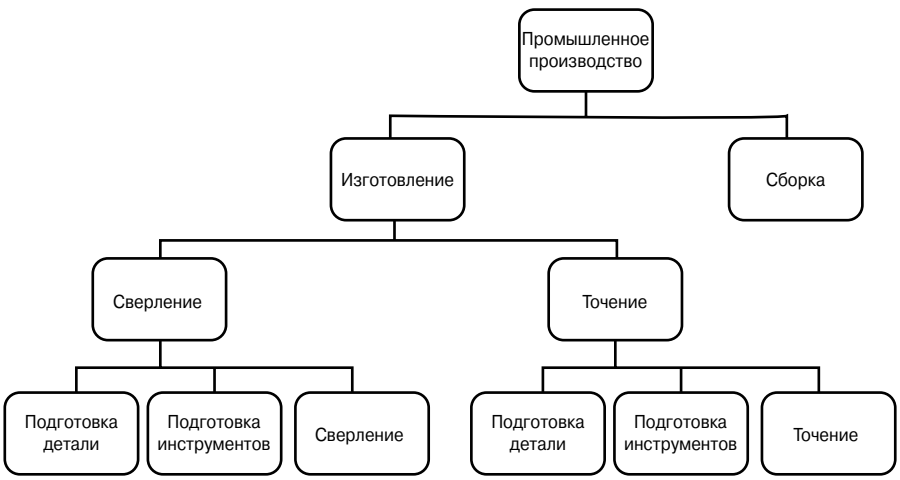


Рис. 19, б. Пример иерархической диаграммы функций промышленного производства

На рис. 19, а и 19, б показаны два класса ключевых функций, различающихся объектами, на которые направлены функции. Мы сосредоточимся на функциях обработки информации в сфере управления, хотя те же рассуждения можно отнести к промышленному производству.

В общем случае термин “функция” можно использовать на каждом уровне иерархии, хотя часто это понятие разбивают на подчиненные понятия. Пример детализации приведен на рис. 19, а.

- **Сложная функция** — комплексная функция, состоящая из многих действий.
- **Функция** — сложная деятельность, которую можно разбить на подфункции. Вместе с другими функциями составляет сложную функцию (“клубок” функций).
- **Подфункция** — деятельность, которую можно разбить на элементарные функции. Вместе с другими подфункциями составляет вышестоящую функцию.
- **Элементарная функция** — деятельность, которую нельзя разбить на составляющие. Например, деятельность, выполняемая на конкретном рабочем месте или внутренняя процедура, не имеющая альтернатив.

Зачастую эти понятия можно отделить друг от друга только условно, поэтому в данной книге мы преимущественно будем обходиться общим понятием “функция”.

Хотя общепринято разбивать функции посредством иерархических диаграмм, этот подход “сверху вниз” имеет ряд недостатков. Например, зачастую не хватает строгих правил классификации, что затрудняет контроль содержательности функции на определенном уровне. С другой стороны, противоположный подход, состоящий в объединении элементарных функций в более крупные функциональные единицы, может быть выполнен более систематично. Вот почему на практике следует использовать *оба* подхода. Сначала функции разбиваются на составляющие в рамках подхода “сверху вниз”, чтобы получить представление об элементарных функциях. Затем элементарные функции перегруппируются снизу вверх. Мартин и Олле с соавторами предлагают некоторые интересные примеры иерархии функций *(см. Martin, Information Engineering II 1990, pp. 45; Olle et al., Information Systems Life Cycle 1988, p. 57)*.

Иерархические структуры функций можно построить в соответствии с таким принципом: “идентичные процедуры, идентичные информационные объекты и идентичное распределение должны применяться к идентичным бизнес-процессам” *(см. Nüttgens, Koordiniert-dezentrales Informationsmanagement 1995, p. 97)*. Учитывая параметры их “бизнес-процесса”, только статические функции сгруппированы, как противоположные динамическому описанию бизнес-процессов. На рис. 20 представлены несколько примеров классификационных параметров. Подгруппы созданы с помощью дальнейшей дифференциации параметров.

Выбор параметров классификации зависит от модели. При восстановлении структуры бизнес-процесса имеет смысл классифицировать функции в соответствии с этими параметрами. Если система должна быть разработана позже, функции следует классифицировать по видам деятельности, что-

бы получить некоторые возможности для повторного использования функциональных модулей. Это позволяет одновременно осуществлять несколько видов классификации.

Параметры классификации	**Характеристики**	**Примеры**
Вид деятельности	Групповые функции с идентичными (подобными) правилами преобразования	Вести учет накладных Вести учет клиентов Вести учет зарплаты
Объект	Групповые функции, направленные на один объект	Запустить заказ Отозвать заказ Выполнить заказ
Бизнес-процесс	Групповые функции, объединенные в процесс	Выбрать поставщиков Навести справки Оформить заказ на поставку

Рис. 20. Параметры классификации функций

При построении метамодели на рис. 21 этот факт был принят во внимание. Здесь одновременно осуществлена классификация по поддерживающим операциям и процессам. Чтобы каждая функция была представлена один раз, создан класс ОБЩАЯ ФУНКЦИЯ, включающий функции независимо от параметров классификации. Таким образом, приемка заказов и проверка наличия определяются один раз, как элементы класса ОБЩАЯ ФУНКЦИЯ.

Каждая характеристика функции, не вовлеченная во внутренние взаимосвязи процесса, описывается как атрибут этого класса. Согласно предположению, сделанному Олле с соавторами, мы можем различать имена новых элементов и сами эти элементы, т.е. наименование общей функции и саму эту функцию *(см. Olle et al., Information Systems Life Cycle 1988)*. Это облегчает обращение с синонимами и омонимами в мультилингвальных конструкциях. С целью упрощения мы опускаем этот частный аспект в данной книге.

Чтобы различать функции, которые преимущественно преобразуют информацию, и функции сферы материального производства, создадим подклассы.

Классификация по видам деятельности соответствует классу ассоциаций ОПЕРАЦИОННАЯ СТРУКТУРА ОБЩЕЙ ФУНКЦИИ. Кардинальности *:* предполагают сетевую структуру, при которой определенная функция может входить в состав многих доминирующих функций. Если бы функции были самостоятельными, это привело бы к бесконечной последовательности.

Бизнес-процессы поддерживают доминирующие корпоративные цели. Эта взаимосвязь устанавливается посредством связи ПОДДЕРЖКА КОРПОРАТИВНОЙ ЦЕЛИ между классами БИЗНЕС-ПРОЦЕСС и КОРПОРАТИВНАЯ ЦЕЛЬ. Минимальная кардинальность равна 1. Бизнес-процесс, не поддерживающий корпоративной цели, не имеет смысла, так же как и корпоративная цель, которую не поддерживает ни один бизнес-процесс. Бизнес-процесс

можно разделить на подпроцессы, что показывает класс ассоциаций СТРУКТУРА БИЗНЕС-ПРОЦЕССА. В этом смысле подпроцессы могут входить в состав многих доминирующих (ключевых) процессов.

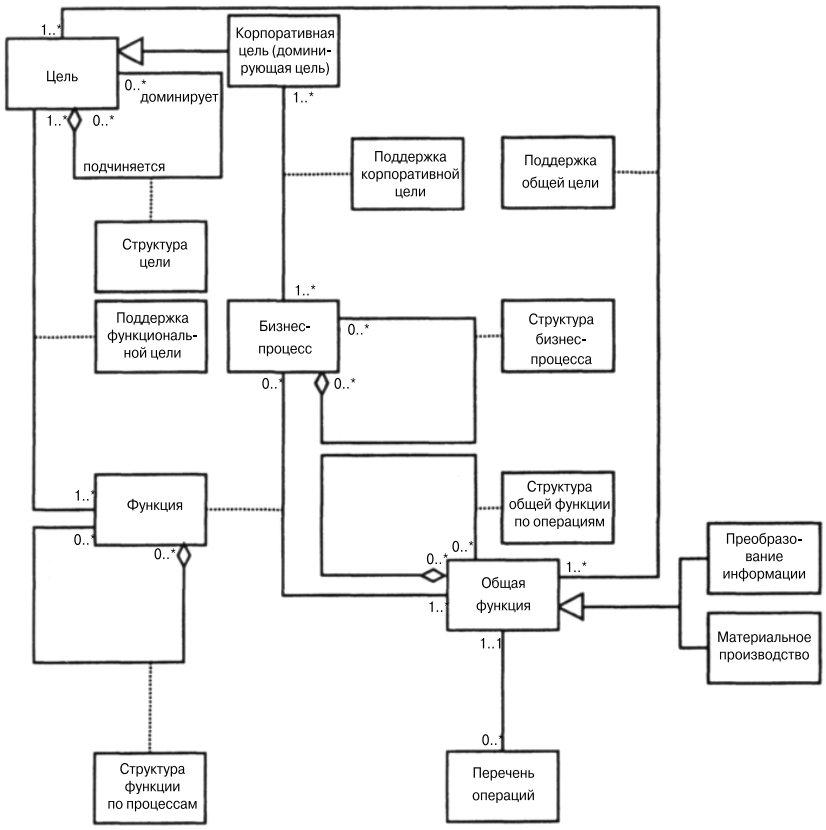


Рис. 21. Метамодель структуры функций и структуры целей

Общие функции связаны с соответствующими бизнес-процессами с помощью класса ассоциаций ФУНКЦИЯ. Согласно этому подходу, ориентированному на бизнес-процессы, наименование функции не определено, пока она не взаимосвязана с бизнес-процессом. Класс ассоциаций ФУНКЦИЯ может иметь атрибуты общей функции. С другой стороны, классу ассоциаций можно присвоить дополнительные атрибуты в контексте бизнес-процессов. Функции разбиваются на подчиненные функции в соответствии с бизнес-процессами. Поскольку каждая фукция может существовать многократно в пределах одного бизнес-процесса, также используется сетевая структура.

Функции на высшем уровне иерархии принадлежат непосредственно ключевым бизнес-процессам и не подчиняются доминирующей функции. Аналогично, элементарные функции на низшем уровне иерархии не имеют подчи-

ненной функции. Вот почему их минимальная кардинальность равна 0. Цели также относятся к конкретным функциям.

На первый взгляд, статистическая структура бизнес-процесса и структура функций в разрезе бизнес-процессов очень похожи. На рис. 22, а изображена структура бизнес-процесса “Планирование производства и контроль”, включающего несколько функций.

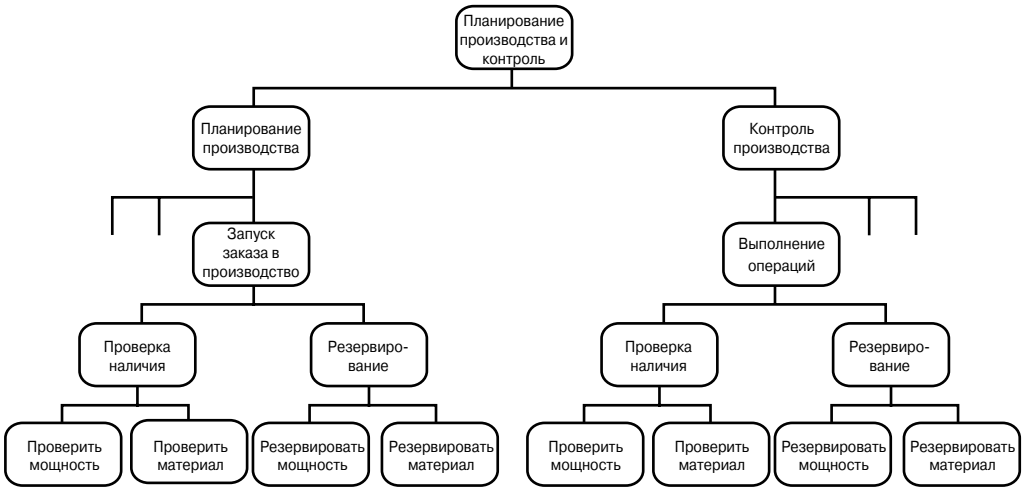


Рис. 22, а. Структура бизнес-процесса “Планирование производства и контроль”

На рис. 22, б представлена сетевая структура, позволяющая избежать многократного изображения одних и тех же функций посредством кардинальностей (0..*):(..*).

Если функции, принадлежащие многим подпроцессам, должны быть обозначены самостоятельно, они могут получить ролевые имена (рис. 23). В таком случае функции определяются своими ролевыми именами, бизнес-процессами и своим базовым значением. Ролевые имена можно присвоить, если остальные элементы бизнес-процесса еще не определены. Например, на рис. 22 процесс планирования и контроля распределен по нескольким организационным единицам, тогда ролевое имя может представлять собой наименование организационной единицы. Так, проверка наличия среднесрочных чеков может производиться на уровне операционного подразделения, а проверка краткосрочных чеков — на уровне департамента.

Предложенное решение позволяет каждой общей функции в контексте процессов получить специальные характеристики. С другой стороны, если функции определены настолько четко, что их можно использовать в любом приложении, и состоят из тех же подфункций, то эти функции можно определить как исходные строительные блоки — функциональные объекты. На рис. 22, б проверка наличия и резервирование могут быть такими функциональными объектами. Обращаясь к диаграмме классов на рис. 21, мы видим, что класс ассоциаций СТРУКТУРА ОБЩЕЙ ФУНКЦИИ отражает контекст строительных

блоков, и кардинальность между классами ОБЩАЯ ФУНКЦИЯ и БИЗНЕС-ПРОЦЕСС имеет характер (1..*):(0..*). Теперь только доминирующие строительные блоки (функциональные объекты) могут относиться к процессу, и не каждая общая функция будет поддерживать процесс.

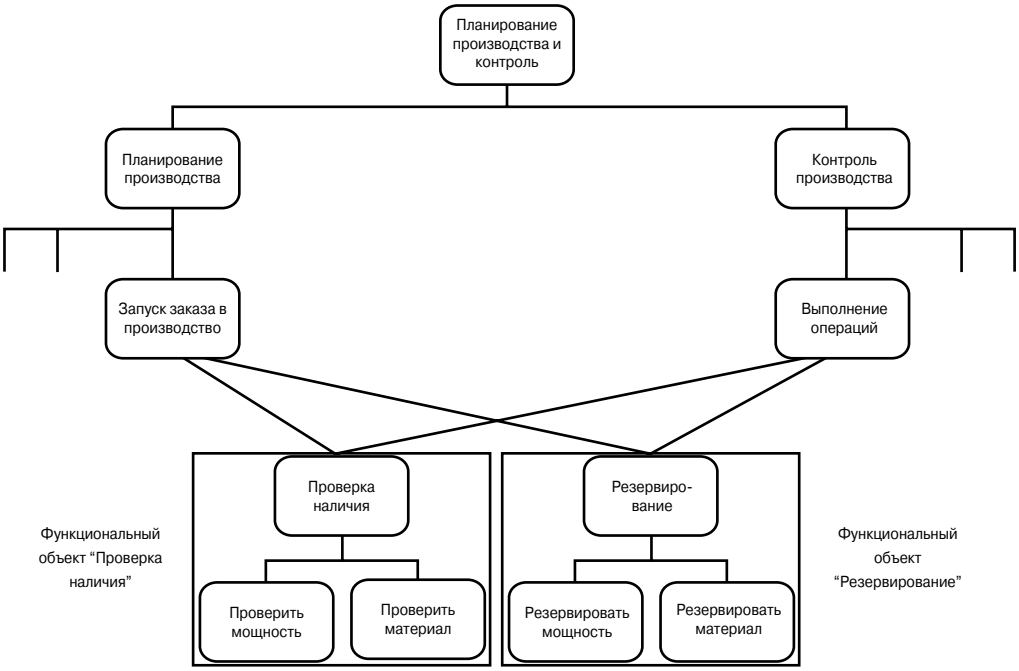


Рис. 22, б. Сетевая структура бизнес-процесса "Планирование производства и контроль"

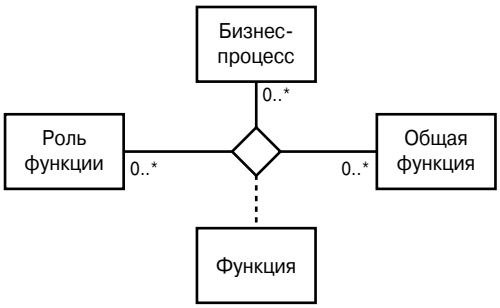


Рис. 23. Присвоение ролевых имен

В метаструктурах обозначение ФУНКЦИОНАЛЬНЫЙ ОБЪЕКТ изображается как специфическая версия класса ОБЩАЯ ФУНКЦИЯ (рис. 24). Структуры сложных функций могут состоять из функциональных объектов. В дальнейшем мы усилим этот вывод, объединяя функциональный аспект ARIS с остальными аспектами при формировании бизнес-объектов.

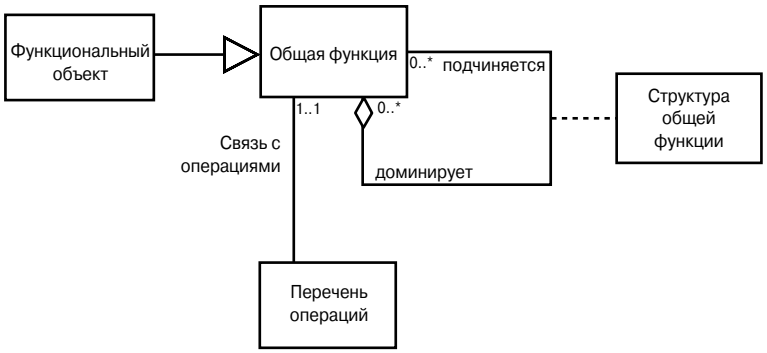


Рис. 24. Метамодель функционального объекта

Помимо классификации по видам деятельности, класс ассоциаций СТРУКТУРА ОБЩЕЙ ФУНКЦИИ может моделировать также классификацию по процессам или объектам.

В рамках макроподхода характеристики функции детализируют задачу, выполняемую с помощью функции (отвечают на вопрос “Что делается?”). Микроподход отвечает на вопрос “Из каких операций состоит эта функция?”. Например, функцию “Деловая поездка” можно разбить на такие операции: обоснование командировки; проверка соответствия поездки с корпоративными правилами; проверка проездных документов; оценка расходов; разрешение.

Эти шаги выполняются в контексте задачи. Они не классифицируют функцию, они скорее есть часть функции. Соответствующая документация предназначена для проверки выполнения функции. Эта документация может иметь вид текста, структограмм или таблиц решений *(см. Nüttgens, Koordiniert-dezentrales Informationsmanagement 1995, p. 95)*.

В метамоделях ПЕРЕЧЕНЬ ОПЕРАЦИЙ изображается как отдельный класс, связанный с классом ОБЩАЯ ФУНКЦИЯ.

Если внутренние процессы не документируются, как это обычно бывает в случае креативных функций типа принятия решений или сбора идей, это приводит к внутренне не структурированным функциям, которые могут поддерживаться только групповыми приложениями, а не операционными информационными прикладными системами.

A.II.1.1.2 Последовательность функций в процедурах

Помимо изучения структуры функций, на этапе определения требований анализируется последовательность функций в процедурах, что прямо ведет к описанию процессов. В отличие от описаний управленческих процессов, рассматриваемых позже, определяются логические последовательности функций, а не триггеров. Это уместно, когда инициирующие события или сообщения не предоставляют никакой дополнительной информации в ходе определения требований или когда триггеры добавляются позже, на этапе спецификации дизайна.

Методы описания процедур исходят из методик построения сетевых диаграмм, описывают различные связи подчинения и доминирования, измеряют расстояние между событиями, фиксируют любые наложения событий или их густоту. Более того, логически увязываются входные и выходные элементы соответствующих взаимосвязей.

На рис. 25 представлена в виде процедуры часть иерархической диаграммы, изображенной на рис. 19, а. Рис. 25 подтверждает, что контекст процедуры нельзя вывести из древовидного индекса. После вычислений с использованием приближенных значений (“ballpark figures”) и количественной структуры заказа, имеет место узел решений, из которого выходят три ветви: составить новое техническое предложение, если вычислена нереальная цена; приостановить процесс, если есть основания полагать, что предложение будет отвергнуто; обслужить заказ, если клиент принял предложение. Вероятности различных альтернатив указаны рядом с соответствующими стрелками. Так как эти альтернативы взаимоисключающие, сумма вероятностей равна 1.

Данная концепция взята из методики графического анализа GERT *(см. Elmaghraby, Activity Networks 1977; Scheer, Projektsteuerung 1978)*.

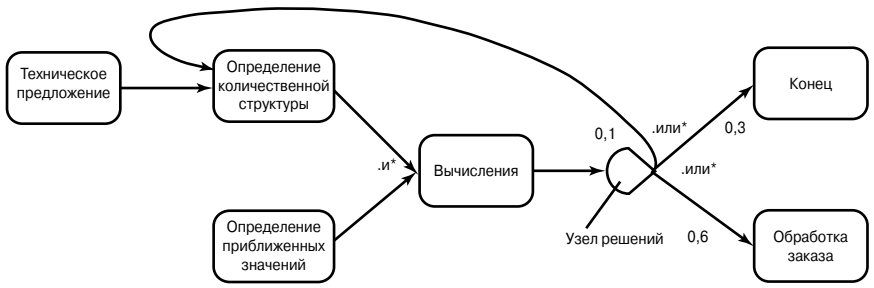


Рис. 25. Последовательность функций в процедуре

Узел решений, представленный здесь, можно рассматривать как отдельный элемент, но его также можно интерпретировать как нормальное событие нулевой продолжительности или как часть предыдущего события.

Упорядочивание взаимосвязей формирует новый класс ассоциаций ПОЗИЦИОНИРОВАНИЕ внутри класса ассоциаций ФУНКЦИЯ. Каждая позиция определяется предшествующей и последующей функцией. Добавляя класс ассоциаций ПОЗИЦИОНИРОВАНИЕ на рис. 26, получаем возможность определить расстояние между событиями, наложение или отсрочку событий.

Логические зависимости между предельными значениями размещены как атрибуты связей позиционирования.

Для функций сферы материального производства последовательность функций в процедурах описывается в рабочем задании. Понятие “рабочее задание” уже подразумевает описание процесса, потому что элементы других аспектов ARIS (организационные единицы, ресурсы, материальный продукт) содержатся в рабочем задании. Однако управление событиями не присутствует явно в рабочем задании, следовательно, динамика процессов не представлена в явном виде.

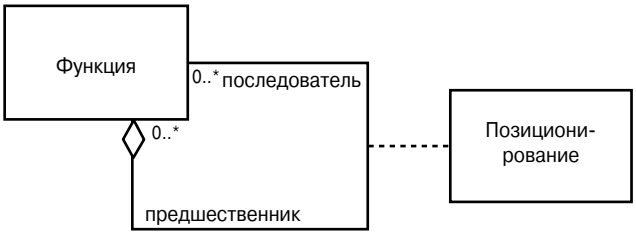


Рис. 26. Позиционирование

Рабочие задания касаются производства деталей различных уровней сложности. Рабочие задания для подклассов создаются на уровне типа, как показано на упрощенном примере на рис. 27, а. Каждая операция описывает функцию, а рабочее задание описывает, как из операций составляется процесс.

РАБОЧЕЕ ЗАДАНИЕ НА ОБРАБОТКУ ЛИСТОВОГО МЕТАЛЛА

№ операции	Наименование операции	Средняя продолжительность	Операционное подразделение
1	Сверление	5	Группа 1
2	Фрезеровка	7	Группа 5
3	Снятие заусениц	4	Группа 4
4	Мытье	2	Группа 7

Рис. 27, а. Рабочее задание на уровне типа детали

Операции соответствуют техническим процедурам. Технические процедуры описываются независимо от контекста рабочего задания, а затем детализируются в контексте рабочего задания или процесса. Диаграммы классов соответствуют рис. 27, б. Технически это совпадает с метамоделью структуры функций на рис. 21 (БИЗНЕС-ПРОЦЕСС, ОБЩАЯ ФУНКЦИЯ, ФУНКЦИЯ). Таким образом, последовательность технических функций можно трактовать так же, как и административных функций.

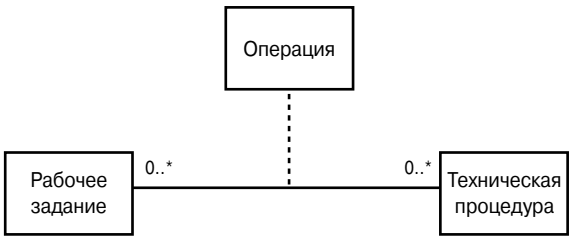


Рис. 27, б. Диаграмма классов для рабочих заданий

Документирование рабочих заданий — одна из классических задач при моделировании процессов планирования производства и контроля в информационных системах. Рабочие задания анализируются на уровне деталей и классов деталей; создаются большие базы данных.

Диаграмма классов для управления рабочими планами на уровне деталей приведена на рис. 27, б *(см. Scheer, Business Process Engineering 1994, p. 216)*. Контекст процесса выясняется с помощью определения взаимосвязей деталей.

Не принято использовать функции на уровне деталей в моделировании бизнес-процессов. Это делается только для особо важных конечных продуктов.

Рабочие задания на уровне типа и экземпляра, которые мы обсуждали до этого момента, похожи на описание обработки основных данных, так как они не зависят от контекста заказов (хотя экземпляры, управляемые системами документооборота, зависят от заказов). В данном случае рабочие задания на уровне типа и экземпляра похожи на шаблоны. В системах планирования производства и контроля параллельно разрабатываются рабочие задания, зависящие от основных данных и заказов.

A.II.1.1.3 Типы обработки

Чтобы отличать функции, поддерживаемые средствами информационной системы, от функций, выполняемых вручную, класс ФУНКЦИЯ разбивается на классы СИСТЕМНАЯ ФУНКЦИЯ и РУЧНАЯ ФУНКЦИЯ" (рис. 28).

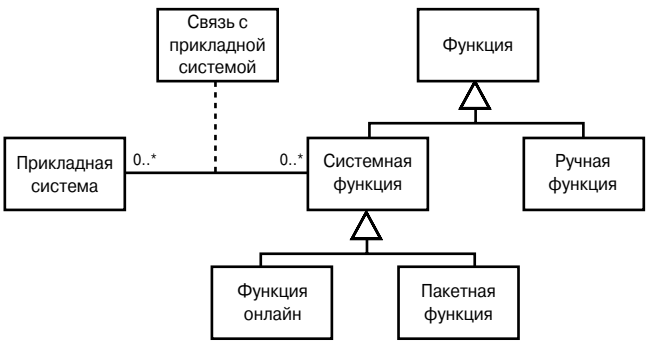


Рис. 28. Детализация понятия функции

Системные функции предназначены для обслуживания клиентских заказов, сопровождения баз данных, создания статистики с помощью информационных систем. Если прикладная система известна, соответствующая информация также характеризует системную функцию. Информация о прикладной системе имеет общий характер (предполагается, что описание прикладной системы не входит в спецификацию дизайна).

Определение требований также включает предварительную оценку типа обработки для системных функций. Один из ключевых параметров, характеризующих тип обработки, — это индикатор, сигнализирующий о том, может ли пользователь управлять процессом и корректировать его (обработка в режиме онлайн) или функция выполняется без вмешательства пользователя (пакетная обработка). Как показано на рис. 28, чтобы оценить, подходит ли системная функция для обработки в режиме онлайн, используется много параметров (рис. 29). Вот почему класс СИСТЕМНАЯ ФУНКЦИЯ разбит на подклассы ФУНКЦИЯ ОНЛАЙН и ПАКЕТНАЯ ФУНКЦИЯ.

Свойства / Цели	Управляемость событиями	Достоверность	Возможность интерактивных решений	Исключение пиков	Улучшение управления	Повышение качества
Экономия времени	●	●	●	●	●	
Экономия труда		●		●	●	
Сбор информации	●		●			●
Улучшение условий труда	●	●		●	●	●
Рационализация организационного процесса		●	●		●	

Рис. 29. Параметры и цели обработки в режиме онлайн
(Scheer, Principles of Efficient Information Management 1991, p. 81)

A.II.1.1.4 Модели принятия решений

Кроме административных целей, информационные системы также используются для поддержки принятия решений, например с целью оптимизировать планирование производства.

Модель линейного программирования (ЛП) — типичный пример модели принятия решений. В моделях ЛП переменные определены таким образом, чтобы максимизировать целевую функцию (рис. 30). Модели ЛП не привязаны к какому-либо конкретному приложению и иллюстрируют модели принятия решений на метауровне. На рис. 31 изображена модель ЛП II-го уровня абстракции для планирования производства (об уровнях абстракции в моделировании *см. Scheer, ARIS — Business Process Frameworks 1998, pp. 120–125*).

Целевая функция	$\sum_j c_j x_j \rightarrow \max$
Secondary conditions	$\sum_j a_{ij} x_i \leq A_i$ для всех i $x_j \geq 0$ для всех j
Переменные	x_j
Коэффициенты	a_{ij}, c_j

Рис. 30. Структура модели ЛП

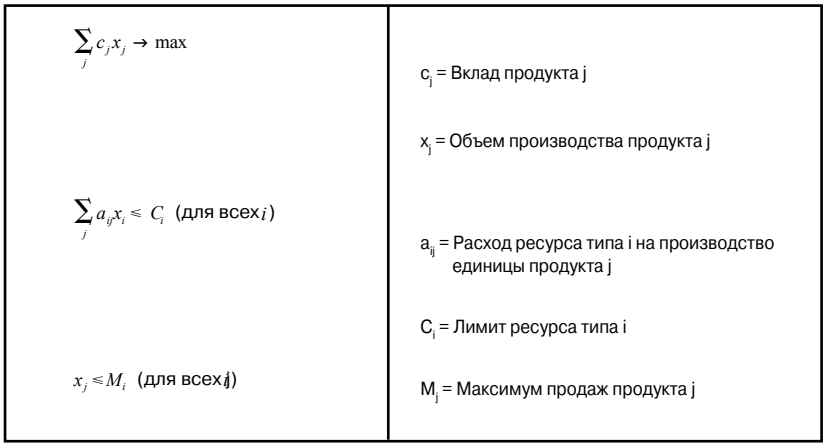


Рис. 31. Модель ЛП для планирования производства

Представление модели ЛП в виде диаграммы классов основано на объектах метамодели ЛП, представленных на рис. 30. Модели ЛП включают элементы ПЕРЕМЕННАЯ, УРАВНЕНИЕ (как целевая функция или как ограничение) и КОЭФФИЦИЕНТ.

Класс МОДЕЛЬ ПРИНЯТИЯ РЕШЕНИЙ создан для конкретных моделей принятия решений (рис. 32). Многие модели принятия решений могут использоваться для одной функции (например, планирование производства). С другой стороны, одна модель принятия решений может использоваться для многих функций. Вот почему используются обозначения кардинальности (0,..*).

Одна модель принятия решений включает много уравнений (например, одно и то же ограничение на величину материальных затрат входит в модель краткосрочного планирования производства и в модель планирования инвестиций). Такие переменные, как объем производства, объем продаж и инвестиции, используются во многих моделях принятия решений.

Взаимосвязь между определенными параметрами (колонки матрицы ЛП) и уравнениями (строки матрицы ЛП) устанавливается с помощью коэффициентов. Для каждой переменной (колонки) с помощью коэффициентов строится много уравнений. И наоборот, каждая строка (уравнение) включает многие переменные.

Матричные генераторы, переменные, уравнения и коэффициенты модели увязываются с базой данных путем определения всех допустимых комбинаций индексов, исходя из логического контекста *(см. Scheer, Business Process Engineering 1994, pp. 525)*. Формат системы математического программирования предполагает стандартную форму описания модели.

На рис. 32 представлена логическая структура моделей принятия решений — структура репозитория прикладных моделей *(см. Scheer, Principles of Efficient Information Management 1991, p. 157)*.

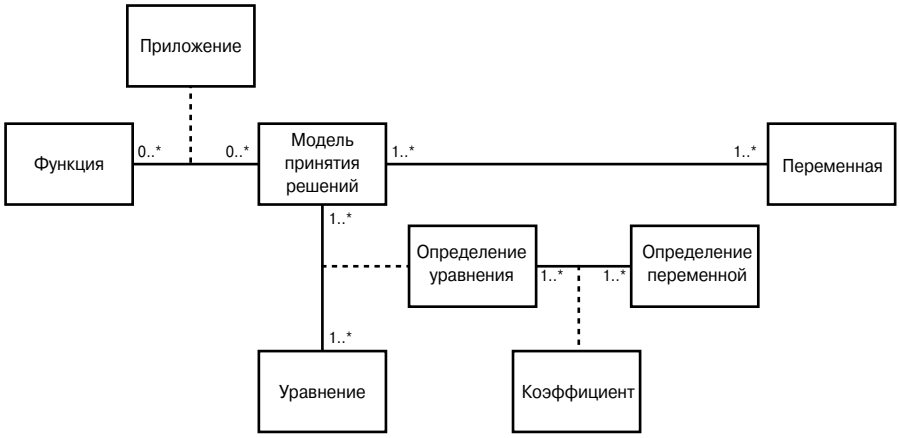


Рис. 32. Логическая структура моделей принятия решений

A.II.1.1.5 Сводное описание требований функционального аспекта

Метамодели для описания требований функционального аспекта сведены на рис. 33.

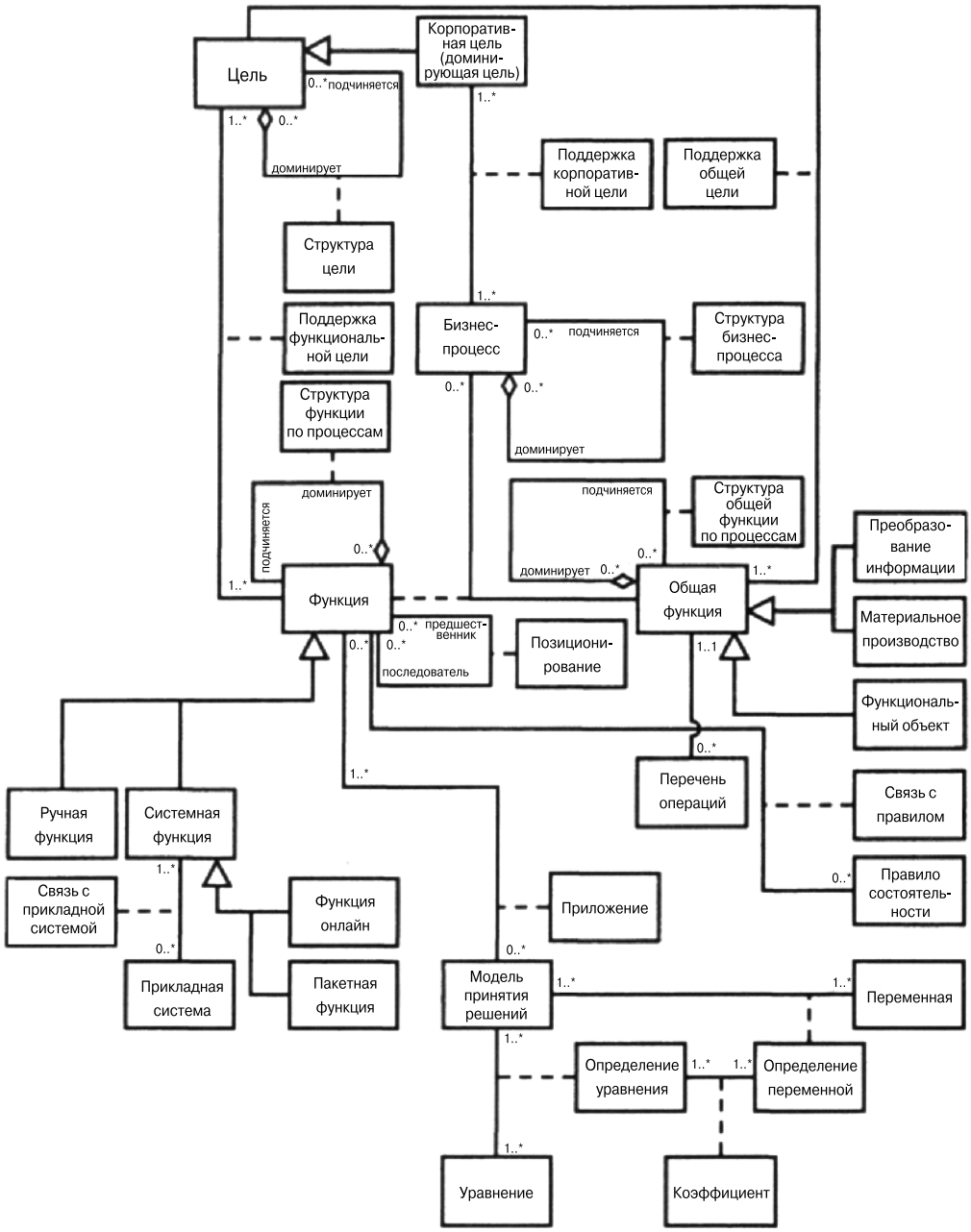


Рис. 33. Метамодель определения требований функционального аспекта

A.II.1.2 Конфигурирование функций

Интерфейсы систем управления процессами, систем технологического контроля и прикладных систем моделируются с соблюдением ряда требований.

Прежде всего класс прикладной системы (система управления; текстовый редактор; специализированный пакет) увязывается с функциями. Характеристики прикладной системы определяются как можно детальнее. Кроме того, следует указать, предполагают ли прикладные системы обмен данными (рис. 34).

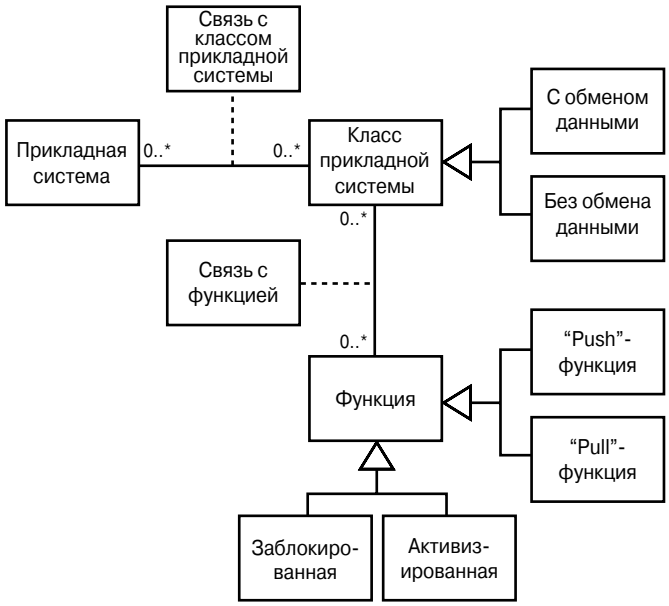


Рис. 34. Прикладные системы

Модели функций и перечни операций обеспечивают фундамент для функционального анализа, дифференцированного учета затрат (расчета себестоимости по видам деятельности) и управления бизнес-процессами. Показатели, необходимые для дифференцированного учета затрат, увязываются с функциями.

При планировании ресурсов содержание соответствующих функций конфигурируется согласно моделям функций. Это позволяет сфокусировать системы планирования ресурсов на конкретном приложении в рамках производственного или организационного проекта.

Метамодели функций, которые мы обсудим, подходят для моделирования технологических приложений *(см. Galler, Vom Geschäftsprozeßmodell zum Workflow-Modell 1997, p. 62)*. Определяя атрибуты функций, рекомендуется включать конкретные временны*е* показатели, например, среднее время выполнения функции.

По отношению к функциям различают принципы “pull” (тянуть) и “push” (толкать). По принципу “pull” сотрудники *извлекают* соответствующую функцию из почтового ящика вместе с перечнем работ. По принципу “push” функция события *посылается* сотруднику для обработки.

При моделировании конкретных процессов управления потоком должны быть удовлетворены дополнительные требования. Эти требования копируются из определения требований на уровне функций, но могут изменяться при необходимости.

Когда приложения увязаны с функциями, особую важность приобретает возможность активизировать или заблокировать определенную функцию в существующей первоначальной модели. Путем активизации или блокирования функций создаются конкретные модели. Состоятельность модели оценивается с учетом логических взаимосвязей между функциями.

Чтобы обеспечить состоятельность модели в целом, держите под контролем активизацию или блокирование функций *(см. Scheer, ARIS — Business Process Frameworks 1998, Fig 50)*. Фрагмент SAP R/3 модели, изображенный на рис. 35, показывает пример блокирования функций в среде ARIS Toolset. Структура конкретного проекта для SAP R/3 (нижнее окно) следует непосредственно из функциональной модели (верхнее окно).

В процессе детального рассмотрения функции устанавливаются конкретные параметры в соответствии с определением требований. Программа SAP R/3 Business Engineer, имеющая прямой доступ к системе IMG, обеспечивает интерфейс “вопрос–ответ” с этой целью. На рис. 36 изображен фрагмент прототипа SAP AG Business Engineer. Модель проекта включает дерево функций приложения. Функции (например, “Обслуживание кредита” — “Credit Processing”) ссылаются на необходимые подфункции (окно слева внизу), которые связаны с параметрами (окно справа вверху). Нижнее правое окно показывает, как функция включается в контекст процесса.

A.II.1.3 Спецификация дизайна функционального аспекта

Спецификация дизайна для определения требований функционального аспекта включает несколько вертикальных уровней. Это так называемый **программный дизайн** — функции впоследствии реализуются в виде подпрограмм. Широко известен термин “крупномасштабное программирование”, а последующая реализация с помощью конкретного языка программирования называется “мелкомасштабным программированием” *(см. Balzert, Lehrbuch der Software-Technik 1996, p. 632, 927)*. Основные этапы спецификации дизайна: разработка структуры модулей; детализация содержания модулей (“мини-спецификация”); создание входных и выходных экранов.

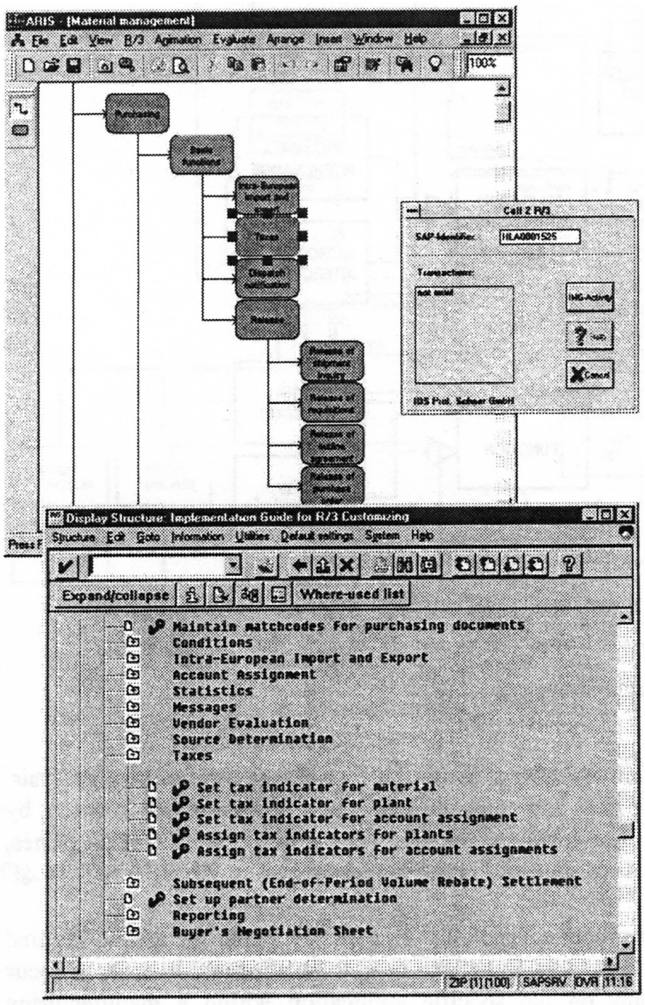


Рис. 35. Пример блокирования функций в среде ARIS Toolset (IDS Prof. Scheer GmbH)

Если функции превращают входные данные в выходные данные, они особенно тесно взаимосвязаны с аспектом данных. Любые ограничения существующих информационных систем также принимаются во внимание при определении требований функционального аспекта. Принцип абстракции позволяет создать функциональную структуру без знания модели данных или конкретной архитектуры аппаратных средств. Это означает, что рассматриваются только общие свойства моделей архитектуры, а не их индивидуальные особенности. Это достигается за счет абстрактных данных, когда определяется только тип данных, а не их физическое воплощение. Например, при описании интерфейса функции элемент данных “номер клиента” рассматривается как целое число без указания таких характеристик этого элемента данных, как тип сущности, имя

связи, тип записи или даже адрес записи и имя поля. То же самое относится к компонентам оборудования, когда при описании функций экрана определяется виртуальный терминал с базовыми свойствами и не указывается конкретная модель терминала.

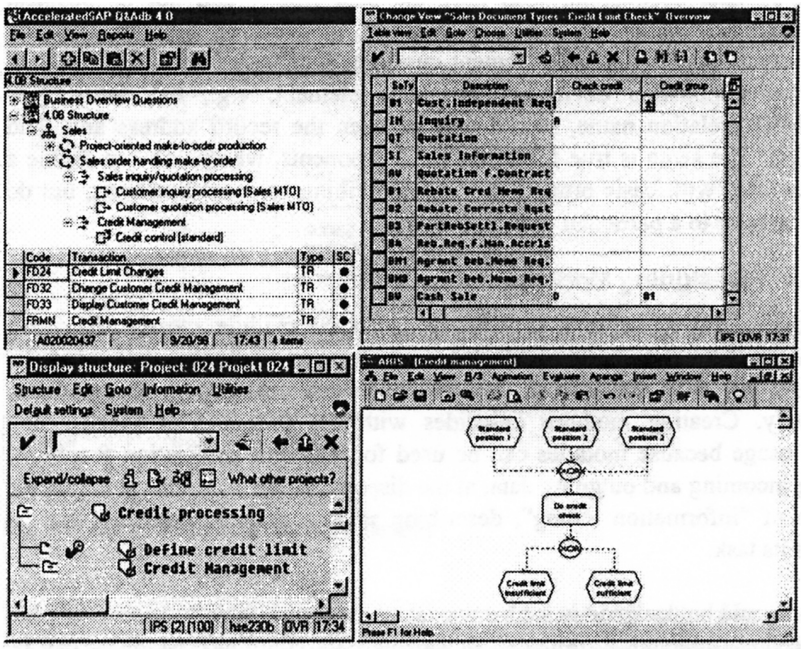


Рис. 36. Функция, подфункции, параметры в SAP R/3

A.II.1.3.1 Разработка модулей

Модуль — один из центральных элементов программного дизайна. Модуль четко описывает автономную часть функции, ее входные и выходные данные. Модуль включает описание данных, управляющую логику и инструкции. Создание модулей согласовывается с принципом локальности и многократного использования, поскольку модули можно использовать для различных приложений. Определение входных и выходных данных с участием пользователя также соответствует принципу “сокрытия информации”, когда описывается, **что** делает модуль, а не **как** он выполняет свою задачу.

Модули разрабатываются так, чтобы максимизировать их “внутреннюю деятельность” и минимизировать их взаимодействие (принцип слабого сцепления модулей). Модули разрабатываются “сверху вниз” или “снизу вверх”. Разработка “сверху вниз” начинается на самом высоком уровне, становится более детальной по мере движения вниз и заканчивается на уровне базовых модулей, которые реализуются с помощью существующих элементов базового программного обеспечения.

При разработке “снизу вверх” первоначально создаются модули самого нижнего уровня, которые затем объединяются в модуль более высокого уровня. Метод “снизу вверх” особенно подходит для работы с библиотекой модулей, которые разработаны раньше. Базовые модули извлекаются из архивов и объединяются в более крупные модули *(см. Balazere, Lehrbuch der Software-Technik 1996, pp. 853)*.

Модули называют процедурами; модули верхнего уровня называют также программами. Модули обладают многими характеристиками. На рис. 37 приведен пример очень детальной иерархии характеристик модуля.

• Класс прикладной системы	например, текстовый процессор
• Тип прикладной системы	например, MS-Word для Windows 6.0
• Прикладная система	например, MS-Word для Windows 6.0 на PC 3417a
• Класс модуля	например, орфографическая программа
• Тип модуля	например, орфографическая программа для MS-Word для Windows 6.0
• Модуль	например, орфографическая программа для MS-Word для Windows 6.0 на PC 3417a

Рис. 37. Иерархия характеристик модуля

Чтобы определить направление процесса разработки модулей (“снизу вверх” или “сверху вниз”), используется уровень определения требований. На рис. 38 иллюстрируется связь между модулями и функциями. ОБЩАЯ ФУНКЦИЯ — это описание функции без фокусирования на содержании конкретного бизнес-процесса. Это подчеркивает, что модуль соответствует принципу многократного использования.

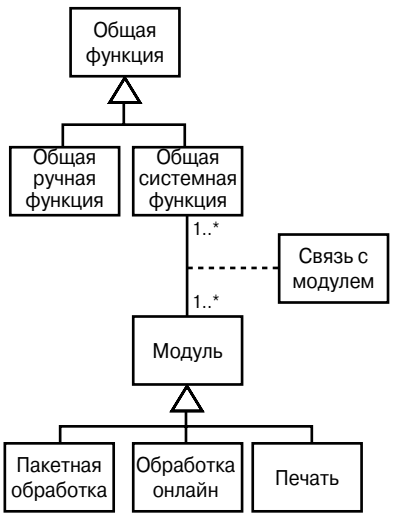
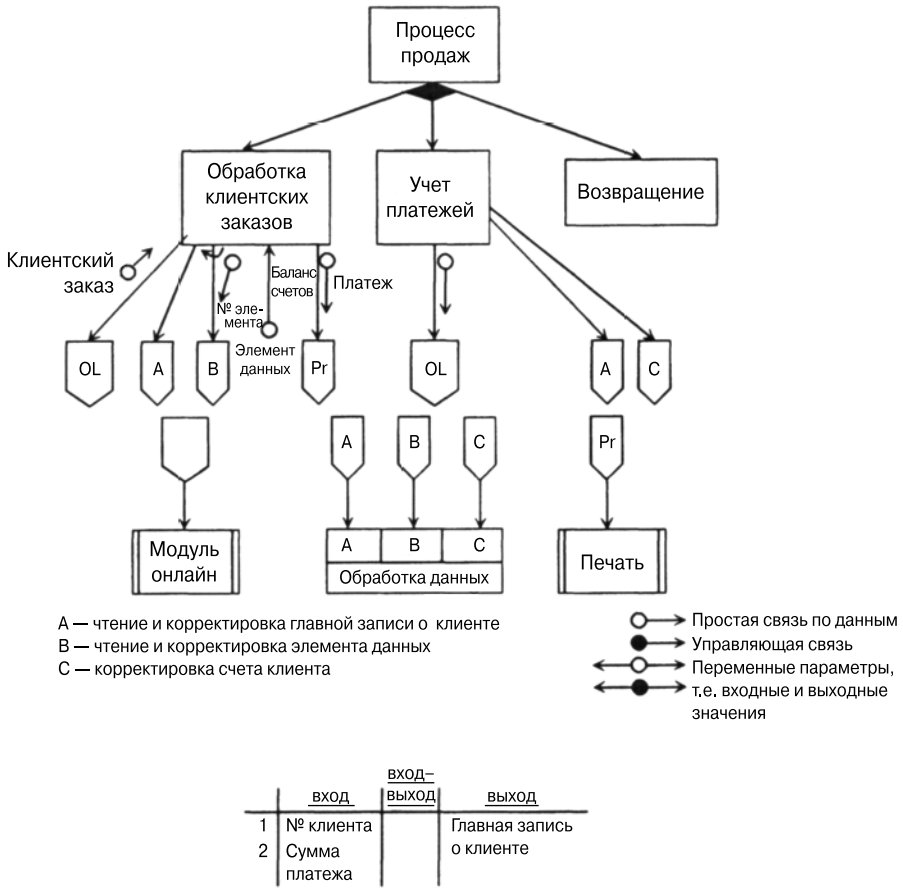


Рис. 38. Связь между модулями и функциями

Так как модули создаются только для функций, которые поддерживаются средствами информационной системы, это означает взаимосвязь с классом СИСТЕМНАЯ ФУНКЦИЯ. Ассоциация *:* с минимальной кардинальностью 1 означает, что модуль используется в системных функциях, а системная функция поддерживается различными модулями. Ассоциация *:* между системными функциями и модулями показывает также, что элементы бизнеса и элементы информационных систем до некоторой степени взаимно независимы.

Модули разделяются на подтипы и связываются в сети. На рис. 39 представлена структурная диаграмма, где модули изображены в виде прямоугольников. Существующие модули, к которым есть доступ, отмечены двойными боковыми линиями.



	вход	вход–выход	выход
1	№ клиента		Главная запись о клиенте
2	Сумма платежа		

Рис. 39. Структурная диаграмма
(Balzert, Entwicklung von Software-Systemen 1982, pp. 356)

Иллюстрации, использующие структурные диаграммы, стали популярными после опубликования таких работ: *Constantine/Yourdon, Structured Design 1979; Page–Jones, Practical Guide to Structured System Design 1980; Balzert, Lehrbuch der*

Software-Technik 1996, pp. 801–862; Sommerville, Software Engineering 1987, pp. 75–103. Операторы (в данном случае для обработки в режиме онлайн) упрощают иллюстрации. Информационные связи между модулями обозначены стрелками; наименования передаваемых данных указаны рядом со стрелками. Существуют также управляющие связи и переменные параметры (входные или выходные). Если информационные взаимосвязи имеют сложный вид, данные нумеруются, а их наименования размещаются в таблице.

Операторы A, B и C связаны с операциями доступа и обработки данных. Ромб в верхней части обозначает управляющие связи (выбор по условию).

Иерархия модулей обусловлена направлением включения. Модули высших уровней включают модули нижележащих уровней. Стрелка между модулями показывает направление включения.

Иерархия модулей создается в соответствии с параметрами “включается” или “включает компонент с именем”.

Модули в иерархической структуре выполняют некоторые свои задачи, используя собственный программный код; остальные задачи выполняют вызываемые модули *(см. Lockemann/Dittrich, Architektur von Datenbanksystemen 1987, p. 102).*

Иерархия модулей недостаточно освещается в документации по программным системам. Различные этапы разделения модулей описаны в работе *Lockemann/Dittrich, Architektur von Datenbanksystemen 1987, p. 103.*

На рис. 40 приведена классификация модулей с помощью ассоциации 1:* между классами МОДУЛЬ и ТИП МОДУЛЯ (управление данными; пакетная обработка; обработка в режиме онлайн).

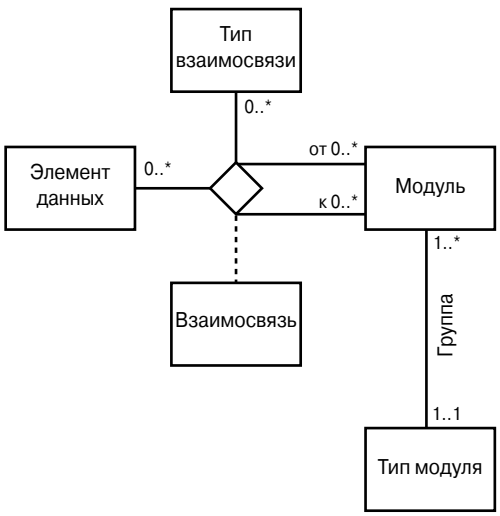


Рис. 40. Классификация модулей

Взаимосвязи модулей характеризуются классом ассоциаций ВЗАИМОСВЯЗЬ. Класс ТИП ВЗАИМОСВЯЗИ описывает тип связи (простая связь по данным или управляющая связь). Данные, которыми обмениваются модули, имеют на-

именования. Класс ТИП ВЗАИМОСВЯЗИ связан с классом ЭЛЕМЕНТ ДАННЫХ. Элементам данных присваиваются номера.

Хотя структурные диаграммы — только один из нескольких методов проектирования систем, структуры классов, разработанные в рамках этого подхода, имеют такой же характер для других методов моделирования с той же логикой *(см. Sommerville, Software Engineering 1987, p.77, p. 106)*.

Наряду с термином "модуль" используется также термин "программа". В общем случае программа представляет собой последовательность команд (инструкций), выполнение которых обеспечивает решение задачи *(см. Stetter, Softwaretechnologie 1987, pp. 12–16)*. Программы, состоящие из взаимодействующих подпрограмм, составляют классы программ или прикладных систем. Подпрограммы, отвечающие требованиям, предъявляемым к модулям, называются модульными.

Степень детализации при проектировании модулей зависит от типа обработки. Последовательные операции проектируются для обработки в режиме онлайн. В зависимости от степени детализации бизнес-модели, операции могут отражать самый нижний уровень иерархии функций бизнеса (элементарная функция).

Объем классификаций при разработке дизайна также зависит от характеристик конкретной информационной системы. Некоторые информационные системы подходят для обработки большого количества мелких операций, а другие — для обслуживания немногих крупных операций *(см. Olle et al., Information Systems Methodologies 1991, p. 256)*. Однако при разработке спецификации дизайна влияние информационной системы следует принимать во внимание в ограниченной степени.

A.II.1.3.2 Мини-спецификация

При проектировании информационных систем содержание модульных подпрограмм описывается с помощью мини-спецификаций. Некоторые подходы основаны на использовании псевдокода и структограмм для описания управляющих структур (контролирующих алгоритм) и исполняемых команд. Управляющие структуры представляют собой последовательное выполнение, выбор по условию и циклы. Простая диаграмма на рис. 41 иллюстрирует использование структограмм и псевдокода.

Команды представляют собой вызов процедур или модульных подпрограмм, а также алгебраические операции, выполняемые над элементами данных. В зависимости от сложности модулей управляющие структуры могут быть вложены друг в друга. Листья дерева модулей обеспечивают самое детальное описание, тогда как на старших уровнях содержание модульных подпрограмм обычно состоит из управляющих структур и команд вызова.

На рис. 42 иллюстрируются управляющие структуры, определенные сочетанием классов ТИП УПРАВЛЯЮЩЕЙ СТРУКТУРЫ, УРОВЕНЬ ИЕРАРХИИ и МОДУЛЬ. Управляющая структура обрамляет блок команд в целом. Модули включают несколько управляющих структур, соответствующих различным уровням иерархии. Команды, из которых состоит управляющая структура, представлены с помощью ассоциации 1:*.

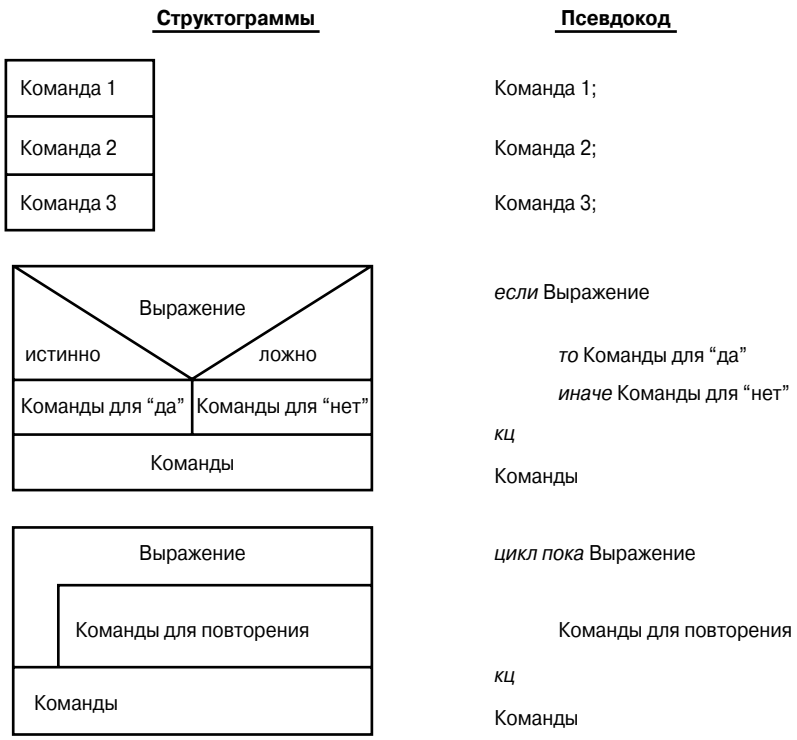


Рис. 41. Управляющие структуры

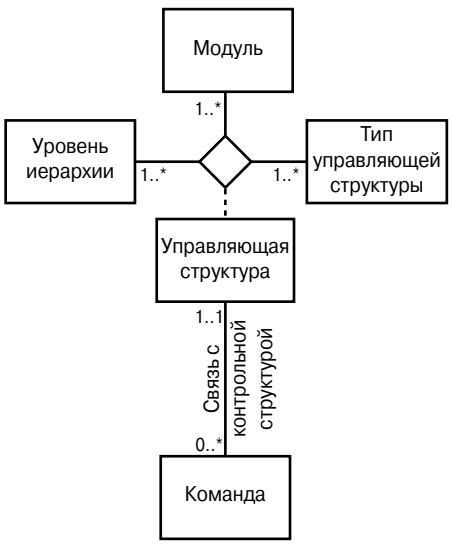


Рис. 42. Метамодель управляющей структуры

A.II.2 Моделирование организационного аспекта

На рис. 47 иллюстрируется место организационного аспекта в концепции ARIS. Функциональный аспект, обсужденный выше, представлен в виде затененного прямоугольника. В спецификации дизайна топология сети исходит из описания иерархической структуры организации. В описании внедрения определены конкретные протоколы коммуникации.

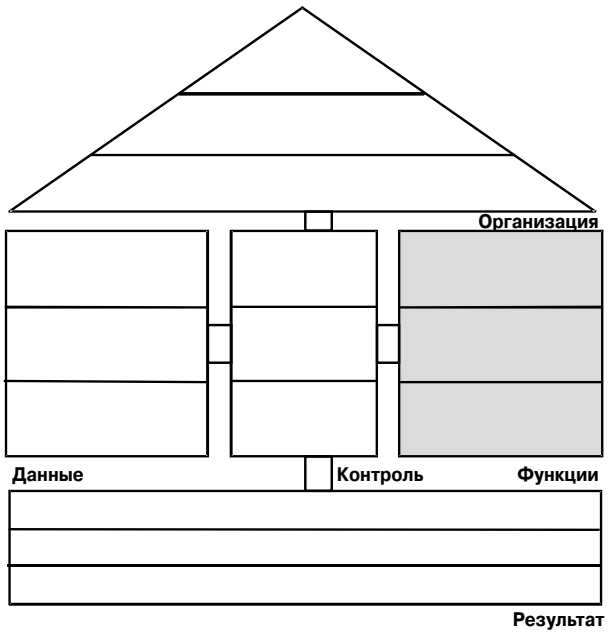


Рис. 47. Организационный аспект в концепции ARIS

A.II.2.1 Описание требований организационного аспекта

Организационная модель предприятия, ориентированного на бизнес-процессы, описывает иерархическую структуру организации, т.е. организационные единицы и взаимосвязи между ними. Ролевая концепция определяет профайл требований организационной единицы, что особенно важно для систем управления потоком.

В большинстве бизнес-приложений иерархическая структура организации не так четко определена как данные, функции или процессы. Однако описание структуры организации необходимо для внедрения прикладного пакета на практике.

A.II.2.1.1 Иерархическая структура организации

Цель построения иерархической структуры организации — объединить группы сотрудников, выполняющих подобные задания, в организационные единицы. На рис. 48, *а*–48, *в* изображены три примера органиграмм. Рис. 48, *а* и 48, *б* — это примеры на уровне типа. Рис. 48, а включает обобщенные понятия, а рис. 48, *б* описывает наименования типов организационных единиц. Рис. 48, *в* касается конкретных организационных единиц.

Рис. 48, а. Органиграмма: уровень обобщенного типа

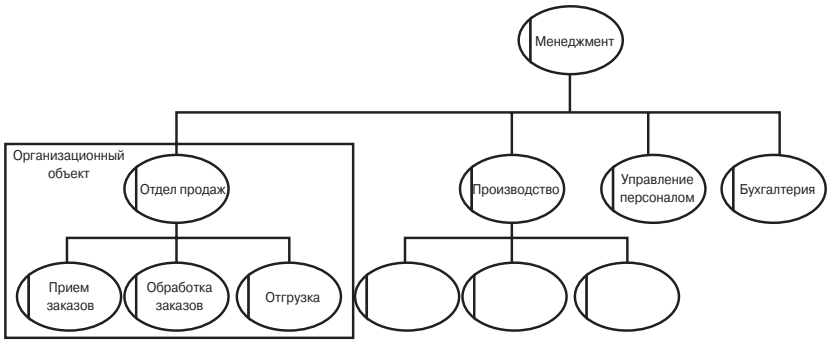


Рис. 48, б. Органиграмма: уровень типа

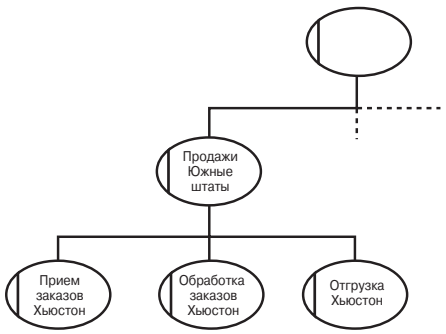


Рис. 48, в. Органиграмма: уровень конкретной организационной единицы

Обычно в моделях бизнес-процессов используются наименования на уровне типа организационной единицы (например, отдел продаж, группа обработки заказов). Однако если бизнес-процесс фокусируется на конкретных организаци-

онных единицах, разрабатываются процедуры для этих организационных единиц (например, для конкретного завода в определенном городе).

Организационные единицы, как и функции, соответствуют видам деятельности (операциям), параметрам объекта или процесса. Рис 48, б иллюстрирует иерархическую структуру, основанную на операционном принципе.

На рис. 49 приведен пример организационной структуры предприятия, ориентированного на бизнес-процессы. Организационные единицы увязаны с бизнес-процессами в рамках логистической цепочки от клиента до процесса производства. Содержательная взаимосвязь организационных единиц характеризуется отношениями “предшественник–последователь” между бизнес-процессами.

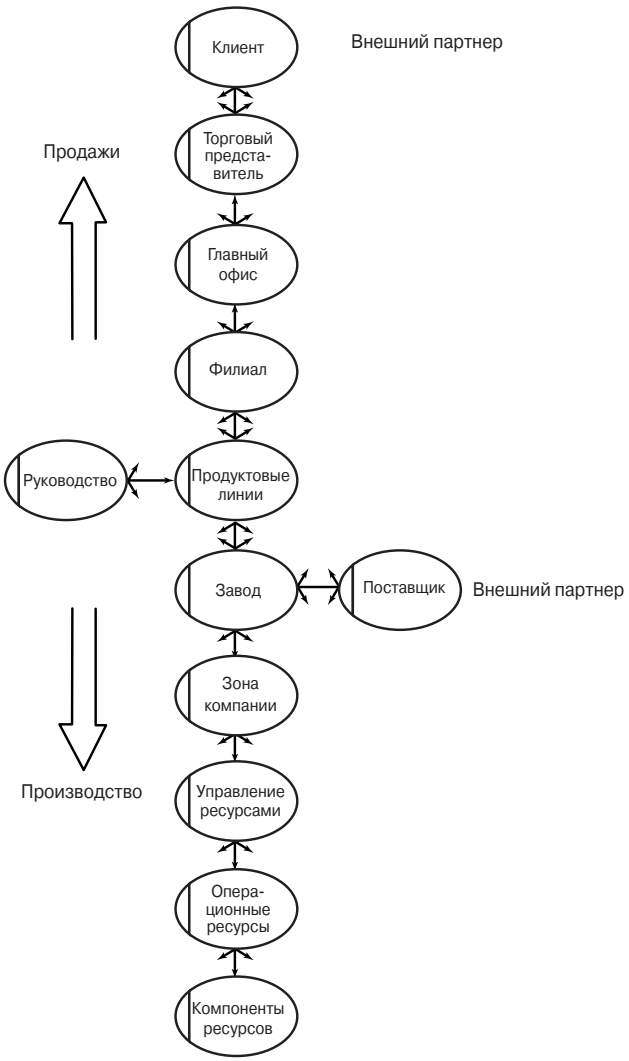


Рис. 49. Организационная структура предприятия, ориентированного на бизнес-процессы

Обычно органиграммы имеют древовидную структуру. Мы используем сетевые взаимосвязи, потому что одна организационная единица (например, отдел продаж) отвечает за несколько продуктовых линеек, а конкретной продуктовой линейкой занимаются многие подразделения. Больше информации по этому поводу содержится в *Bullinger, Prozeβorientire Structuren 1994*.

Наименования организационных единиц (на уровне типа или конкретного представителя) образуют класс “Организационная единица” на метауровне (рис. 50).

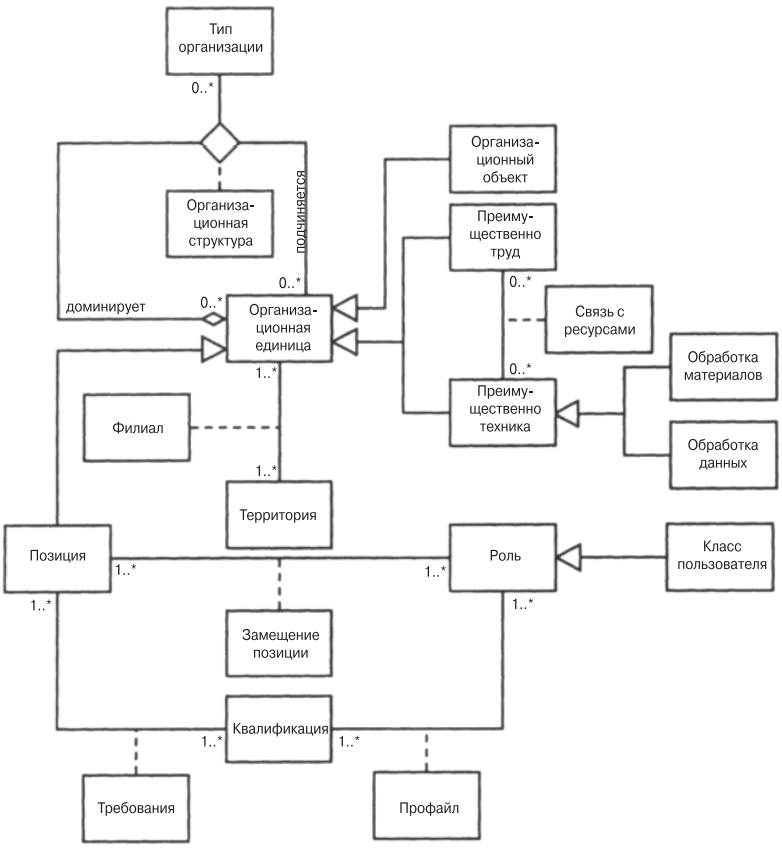


Рис. 50. Метамодель иерархической организации

Машинно-ориентированные процессы структурируются подобно ручным операциям, что приводит к созданию таких организационных единиц, как обрабатывающие центры, машинные отделы, складские системы для производственных предприятий или информационные отделы, рабочие станции и сети персональных компьютеров для организаций, ориентированных на обработку данных. Иерархическая структура организации имеет одинаковый вид для организаций, использующих преимущественно труд или преимущественно материальные ресурсы. В метамоделях класс “Организационная единица” разбивается

на подклассы “Преимущественно труд” и “Преимущественно техника”; последний подкласс разделяется на подклассы “Обработка материалов” и “Обработка данных”. Учитывая, что организационные единицы создаются для производства или продуктов труда, или продуктов машинного производства, рассматривается класс ассоциаций “Связь с ресурсами”. Внешние бизнес-партнеры (клиенты, поставщики или государственные организации) рассматриваются как элементы класса “Организационная единица”.

Географическое распределение организационных единиц обозначается отношением “Филиал” между классами “Территория” и “Организационная единица”.

Самая мелкая единица организационной структуры — это “позиция”, которая определяется частью функции, выполняемой одним сотрудником. Элементы класса “Позиция” — это наименования конкретных должностей.

Позиции входят в более крупные единицы в соответствии с типом организации. Поэтому введен класс “Тип организации”. Иерархические взаимоотношения организационных единиц обозначает класс “Организационная структура”.

A.II.2.1.2 Ролевая концепция

Наряду с организационными единицами, при моделировании цепочки бизнес-процессов описываются также типы должностей, такие как специалист по продажам, бухгалтер, оператор ПК или агент по снабжению. Однако конкретные сотрудники редко закрепляются за конкретными функциями, иначе концепцию бизнеса пришлось бы корректировать при каждом изменении персонального состава организации. Понятие “роль” описывает определенный тип сотрудника определенного уровня образования и квалификации (*см. Galler, Vom Geschäftsprozeßmodell zum Workflow-Modell 1997, pp. 52–58; Rupietta, Organisationsmodellierung 1992; Esswein, Rollenmodell der Organisation 1992*).

Наименования квалификаций составляют класс “Квалификация” и связываются с классом “Роль” с помощью класса ассоциаций “Профайл” (см. рис. 50). Например, роль инженера по продажам состоит из таких квалификаций: высшее техническое образование и практический опыт продаж. Квалификационные требования могут быть усилены для конкретной позиции. Конкретные роли могут связываться с определенными должностями.

При определении ролей различаются классы пользователей для разработки или применения информационных систем, что позволяет предоставить избранным пользователям привилегированный доступ к данным и функциям. В соответствии с компетентностью и частотой использования информационных систем различаются такие классы пользователей (*см. Martin, Application Development 1982, p. 102–106; Davis/Olson, Management Information Systems 1984, p. 503–533*):

- случайные (разовые, эпизодические) пользователи;
- пользователи-профессионалы;
- эксперты.

Чтобы характеризовать классы пользователей, вводим специальное отношение “Класс пользователя”.

На уровне приложения организационные метамодели напоминают структуру данных систем управления персоналом (*см. Scheer, Business Process Engineering 1994, p. 491*). Однако подчеркнем, что системы управления персоналом рассматривают конкретных сотрудников, тогда как ролевые концепции описывают только типы должностей. В целом, системы управления персоналом включают намного более детальные описания объектов.

A.II.2.2 Конфигурирование организаций

Организационные модели обеспечивают определение центров затрат в рамках затратного метода вычисления себестоимости, что позволяет конфигурировать управление бизнес-процессами. Интерпретируя организационные единицы как отделы, рабочие группы и т.п., исходные системы управления ресурсами (такие как MS Project) адаптируются к конкретному приложению.

Метаструктуры, которые уже обсуждались, подходят для моделирования систем управления потоком, ориентированных на бизнес-процессы. Определение ролей имеет ключевое значение, так как для каждой выполнимой функции необходимо знать ее роль, что, в свою очередь, позволяет оценить вклад конкретного исполнителя. Более того, важно также определить замещения. Иногда имеет смысл оперировать детальными характеристиками, например, указывать конкретных сотрудников, а не роли.

В бизнес-приложениях организационный аспект описывается детально, с учетом особенностей приложения и возможностей программного обеспечения.

Организационные модели в бизнес-приложениях определяют такие важные параметры, как коды компаний, их подразделений и клиентов. Это создает основу для последующей привязки функций и данных к организационным единицам, что повышает степень распространения прикладной системы.

В системах управления персоналом организационные модели определяют основу для всех функций учета и контроля. В бухгалтерском учете они обеспечивают коды компании и структуру центров затрат.

A.II.2.3 Спецификация дизайна организационного аспекта

На этапе определения требований для организационного аспекта указываются организационные единицы предприятия и их взаимосвязи. Спецификация дизайна воплощает организационную модель бизнеса в топологию информационной и коммуникационной системы. В частности, определяются топология сети, потребность в ресурсах, типы пользовательского доступа к конкретным узлам и доступные типы компонентов.

A.II.2.3.1 Топология сети

На рис. 51 представлена типичная конфигурация сети промышленного предприятия, объединяющая различные топологии сети, такие как “звезда”, “кольцо” и “шина” (рис. 52).

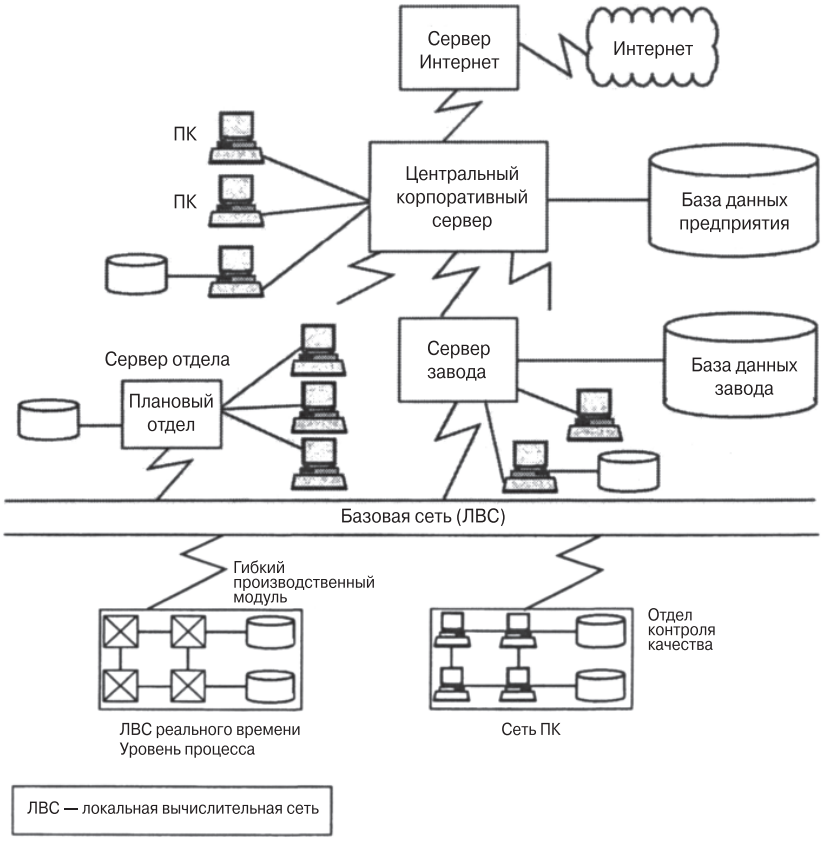


Рис. 51. Пример конфигурации сети

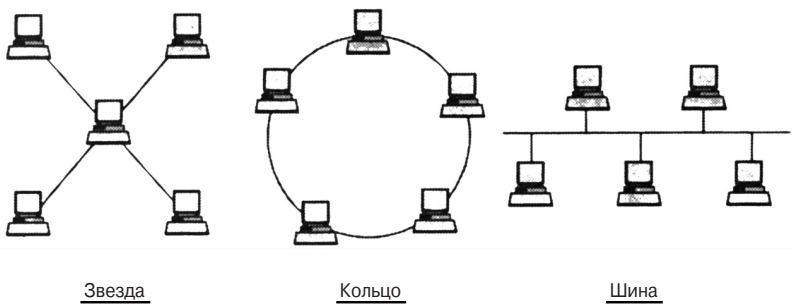


Рис. 52. Топологии сети

Сетевая топология обладает характеристиками надежности, доступа к данным и скорости операций (*см. Hutchinson/Mariani, Local Area Networks 1985; Sikora/Steinparz, Computer & Kommunikation 1988; Sloman/Kramer, Verteilte Systeme und Rechnernetze 1988; Tannenbaum, Computer Networks 1988; Kauffels, Lokale Netze 1997; Taylor, Network Architecture Design Handbook 1997*). Кроме того, различают широкомасштабные вычислительные сети (Wide Area Network — WAN) и ло-

кальные вычислительные сети (Local Area Network — LAN). Широкомасштабные вычислительные сети позволяют передавать информацию на значительные расстояния с использованием коммутируемых и выделенных линий или специальных каналов связи. Локальная вычислительная сеть — это группа компьютеров и периферийное оборудование, объединенные одним или несколькими автономными высокоскоростными каналами передачи цифровых данных в пределах одного или нескольких близлежащих зданий.

В зависимости от типа устройств, включенных в сеть, различают сети терминалов, связывающие терминалы ("тощие клиенты") с серверами, и сети, объединяющие интеллектуальные рабочие станции ("жирные клиенты").

Остов базовой сети — это комплекс сетей с различными характеристиками. Одно из ключевых свойств сети — это производительность в режиме реального времени, характеризуемая скоростью приема–передачи и контролем прерываний.

Еще одна важная характеристика сети — это протокол, принятый в конкретной сети. Существует несколько ориентированных на приложения протоколов Интернет, таких как SMTP, FTP и HTTP, а также протоколы ISO/OSI, такие как X 400. Для доступа к данным подходят, например, протоколы, в основе которых лежит принцип CSMA/CD.

Свойства архитектуры сети определяются независимо от конкретного оборудования или используемого программного обеспечения.

На рис. 53 представлена метамодель конфигурации сети. Класс "Тип сети" объединяет различные типы сетей, характеризующиеся различными топологиями и протоколами. Конкретные сети составляют класс "Сеть".

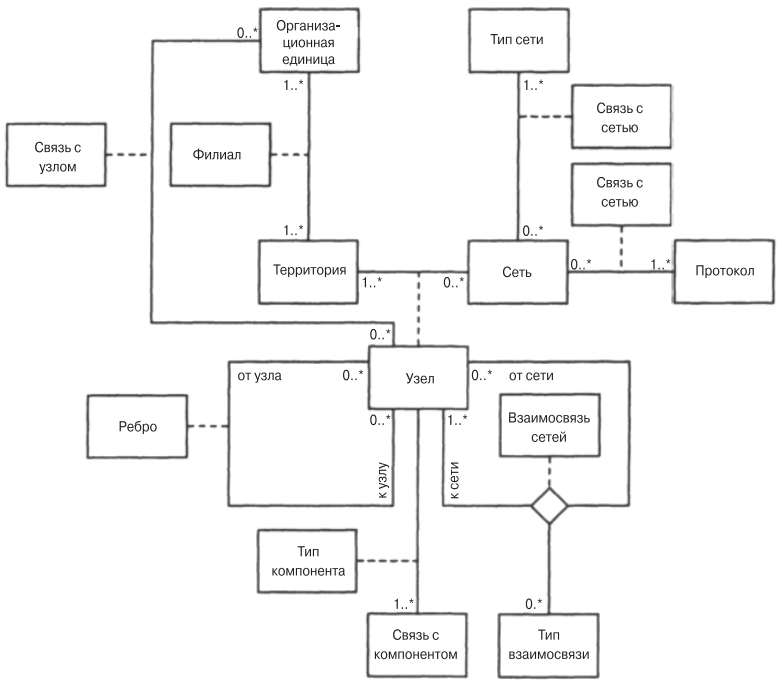


Рис. 53. Метамодель конфигурации сети

При спецификации дизайна обсуждается только логическая структура сети и не рассматриваются физические характеристики, такие как оптоволоконный кабель, коаксиальный кабель или технология передачи данных. Эти детали освещаются в описании применения. Сеть может относиться к нескольким типам сети в зависимости от параметров классификации. Сети определяются своими узлами и ребрами. Местонахождение узла сети характеризуется термином "Территория". Класс "Узел" обозначает связь между классами "Территория" и "Сеть". Сеть включает от 1 до n узлов, хотя некоторые территории могут содержать 0 или n различных узлов логической сети.

Топологии сети выражаются через позиционные взаимосвязи между узлами, поэтому рассматривается связь "Ребро" внутри класса "Узел". В конкретном узле начинается от 0 до n ребер, точно так же в конкретном узле может заканчивать от 0 до n ребер (здесь 0 обозначает первый или последний узел в шинной архитектуре).

Чтобы можно было различать направление ребра, используются ролевые имена "от узла" и "к узлу".

Основные группы передаваемых данных (например, заказы на изготовление) рассматриваются как атрибуты.

Различные сети в организации не изолированы, а взаимосвязаны друг с другом. Связи между узлами сети иллюстрирует класс ассоциаций "Взаимосвязь сетей". На рис. 51 показано, что между сетями существуют различные способы передачи данных, в зависимости от уровня сетевого протокола. Так называемые шлюзы преобразовывают данные из одного сетевого формата в другой на каждом уровне сети. С другой стороны, если некоторые уровни (как правило, верхние) согласованы (т.е. только протоколы нижних уровней нуждаются в преобразовании), используются маршрутизаторы и мосты. Типы взаимосвязи сетей (шлюзы, маршрутизаторы, мосты и т.п.) объединяются на рис. 53 в класс "Тип взаимосвязи".

Описание требований организационного аспекта и сетевая топология увязаны с помощью классов "Территория" и "Организационная единица". Если многие организационные единицы совместно используют центральный узел или конкретный узел доступен только для некоторых организационных единиц данной территории, то рассматривается класс ассоциаций "Связь с узлом" между классами "Узел" и "Организационная единица" (см. рис. 53).

A.II.2.3.2 Типы компонентов

До сих пор узлы определялись только их местоположением и принадлежностью к сети; характеристики используемой аппаратуры не были определены.

Для каждого узла может использоваться сложная компьютерная система, станция ввода–вывода или децентрализованная рабочая станция, имеющая доступ к основным системам. Класс "Тип компонента" объединяет наименования типов используемых физических устройств. Как правило, в одном узле используются устройства разных типов.

Класс ассоциаций "Связь с компонентом" можно интерпретировать несколькими способами. Если узел определен так, что каждому узлу соответствует отдельное устройство, то с точки зрения узла кардинальность будет иметь вид (1..*). Это означает, что один узел соответствует одному типу компонента.

Один тип компонента используется во многих узлах. Если же узел рассматривается просто как сетевой порт, используемый во многих устройствах типа персональных компьютеров или станций вывода, тогда один узел связывается с компонентами многих типов. В этом случае атрибутом связи выступает количество компонентов определенного типа.

A.II.2.4 Внедрение организационного аспекта

Как показано в верхней части диаграммы классов на рис. 54, внедрение начинается с сетевой топологии спецификации дизайна. Сети и узлы, определенные в спецификации дизайна, могут иметь различные физические характеристики на практике. Поэтому понятия, относящиеся к спецификации дизайна, имеют префикс “Лог.”, а понятия физического уровня — префикс “Физ.”. На рис. 55 представлен пример неоднородной сетевой архитектуры.

Предположим, что некоторый сетевой протокол (такой как TCP/IP) моделируется физически на этапе внедрения. Физическая сеть характеризуется конкретной средой передачи данных, будь то коаксиальный кабель, оптоволоконный кабель или витая пара. Между классами “Лог. сеть” и “Физ. сеть” существует ассоциация *:*. Одна физическая сеть соответствует многим логическим сетям, а одна логическая сеть воплощается в различные физические сети.

Согласно диаграмме классов спецификации дизайна, узел физической сети представляет собой связь между физической сетью (класс “Физ. сеть”) и местонахождением (класс “Территория”). Существует взаимосвязь *:* между физическим узлом и логическим узлом, так как один логический узел может соответствовать многим физическим узлам.

В то же время, возможны физические узлы без прямой связи с логическими узлами спецификации дизайна — например, в случае, когда физический узел представляет собой просто технический усилитель и не обладает функциями устройства или приложения. Физические сети определяются физическими узлами и физическими ребрами. Между классами “Лог. ребро” и “Физ. ребро” также существует взаимосвязь *:*.

Однако подчеркнем, что эти взаимосвязи не имеют критического значения для построения информационной модели. Скорее всего достаточно отслеживать связи между логической сетью и физической сетью или между логическими узлами и физическими узлами. Определение физических ребер необходимо только для целей внедрения, связи с логическими ребрами показывать не обязательно.

Используемые устройства, т.е. компьютерные компоненты, характеризуются термином “Физ. тип компонента”, который связан с уровнем спецификации дизайна с помощью класса “Лог. тип компонента”. Например, конкретное наименование устройства с указанием модели и бренда соответствует запоминающей системе на логическом уровне.

Иерархическая связь между доминирующей системой и подчиненными системами иллюстрируется с помощью класса “Иерархия систем”, что позволяет представить развернутую структуру сложных компьютерных или запоминающих систем.

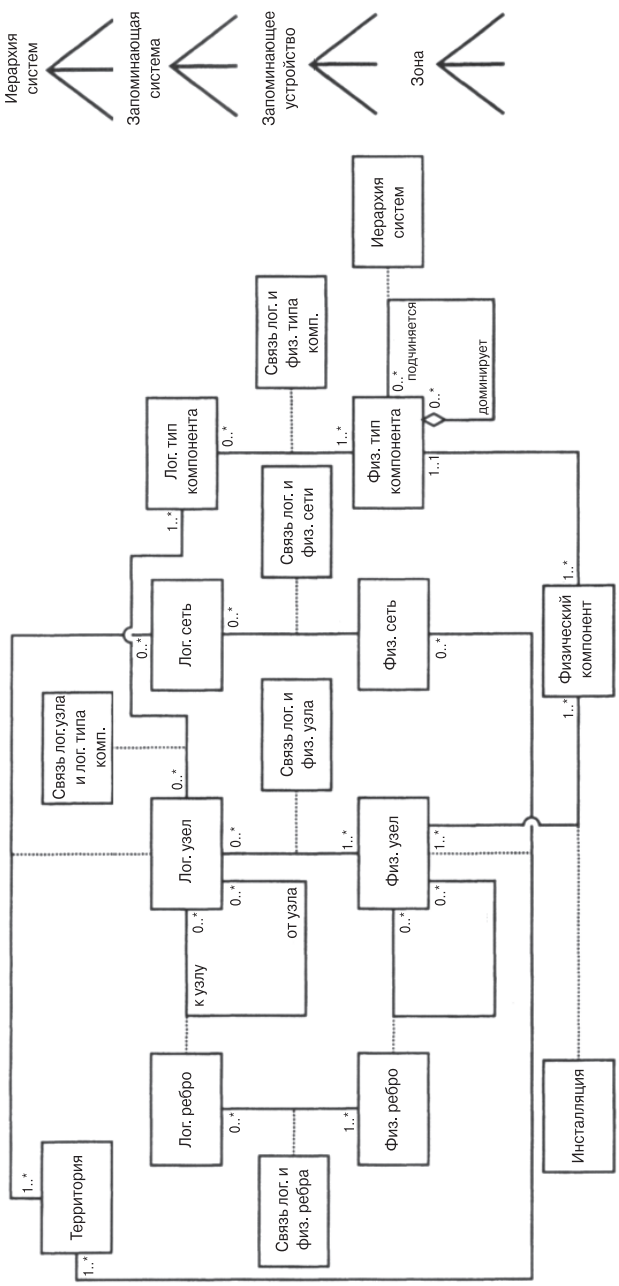


Рис. 54. Моделирование сетей на логическом и физическом уровнях

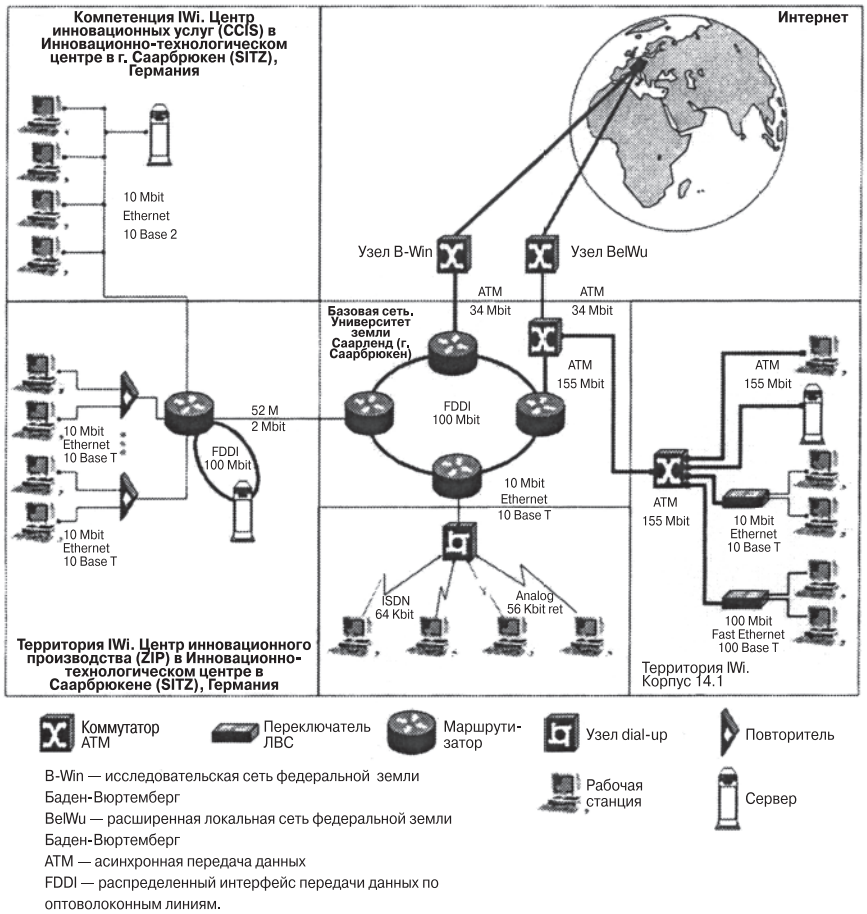


Рис. 55. Пример неоднородной сетевой архитектуры

Различные физические компоненты каждого типа объединены в класс “Физические компоненты”, который не только группирует физические компоненты по типам, но и связывает их с физическими узлами сети. Физические узлы связываются со многими физическими компонентами компьютерных систем. С другой стороны, физические компоненты связываются со многими физическими узлами.

Компоненты программного обеспечения рассматриваются аналогично физическим компонентам компьютерных систем. Для управления сетями используются специальные программы административного контроля.

Один тип компонента используется во многих узлах. Если же узел рассматривается просто как сетевой порт, используемый во многих устройствах типа персональных компьютеров или станций вывода, тогда один узел связывается с компонентами многих типов. В этом случае атрибутом связи выступает количество компонентов определенного типа.

A.II.2.4 Внедрение организационного аспекта

Как показано в верхней части диаграммы классов на рис. 54, внедрение начинается с сетевой топологии спецификации дизайна. Сети и узлы, определенные в спецификации дизайна, могут иметь различные физические характеристики на практике. Поэтому понятия, относящиеся к спецификации дизайна, имеют префикс “Лог.”, а понятия физического уровня — префикс “Физ.”. На рис. 55 представлен пример неоднородной сетевой архитектуры.

Предположим, что некоторый сетевой протокол (такой как TCP/IP) моделируется физически на этапе внедрения. Физическая сеть характеризуется конкретной средой передачи данных, будь то коаксиальный кабель, оптоволоконный кабель или витая пара. Между классами “Лог. сеть” и “Физ. сеть” существует ассоциация *:*. Одна физическая сеть соответствует многим логическим сетям, а одна логическая сеть воплощается в различные физические сети.

Согласно диаграмме классов спецификации дизайна, узел физической сети представляет собой связь между физической сетью (класс “Физ. сеть”) и местонахождением (класс “Территория”). Существует взаимосвязь *:* между физическим узлом и логическим узлом, так как один логический узел может соответствовать многим физическим узлам.

В то же время, возможны физические узлы без прямой связи с логическими узлами спецификации дизайна — например, в случае, когда физический узел представляет собой просто технический усилитель и не обладает функциями устройства или приложения. Физические сети определяются физическими узлами и физическими ребрами. Между классами “Лог. ребро” и “Физ. ребро” также существует взаимосвязь *:*.

Однако подчеркнем, что эти взаимосвязи не имеют критического значения для построения информационной модели. Скорее всего достаточно отслеживать связи между логической сетью и физической сетью или между логическими узлами и физическими узлами. Определение физических ребер необходимо только для целей внедрения, связи с логическими ребрами показывать не обязательно.

Используемые устройства, т.е. компьютерные компоненты, характеризуются термином “Физ. тип компонента”, который связан с уровнем спецификации дизайна с помощью класса “Лог. тип компонента”. Например, конкретное наименование устройства с указанием модели и бренда соответствует запоминающей системе на логическом уровне.

Иерархическая связь между доминирующей системой и подчиненными системами иллюстрируется с помощью класса “Иерархия систем”, что позволяет представить развернутую структуру сложных компьютерных или запоминающих систем.

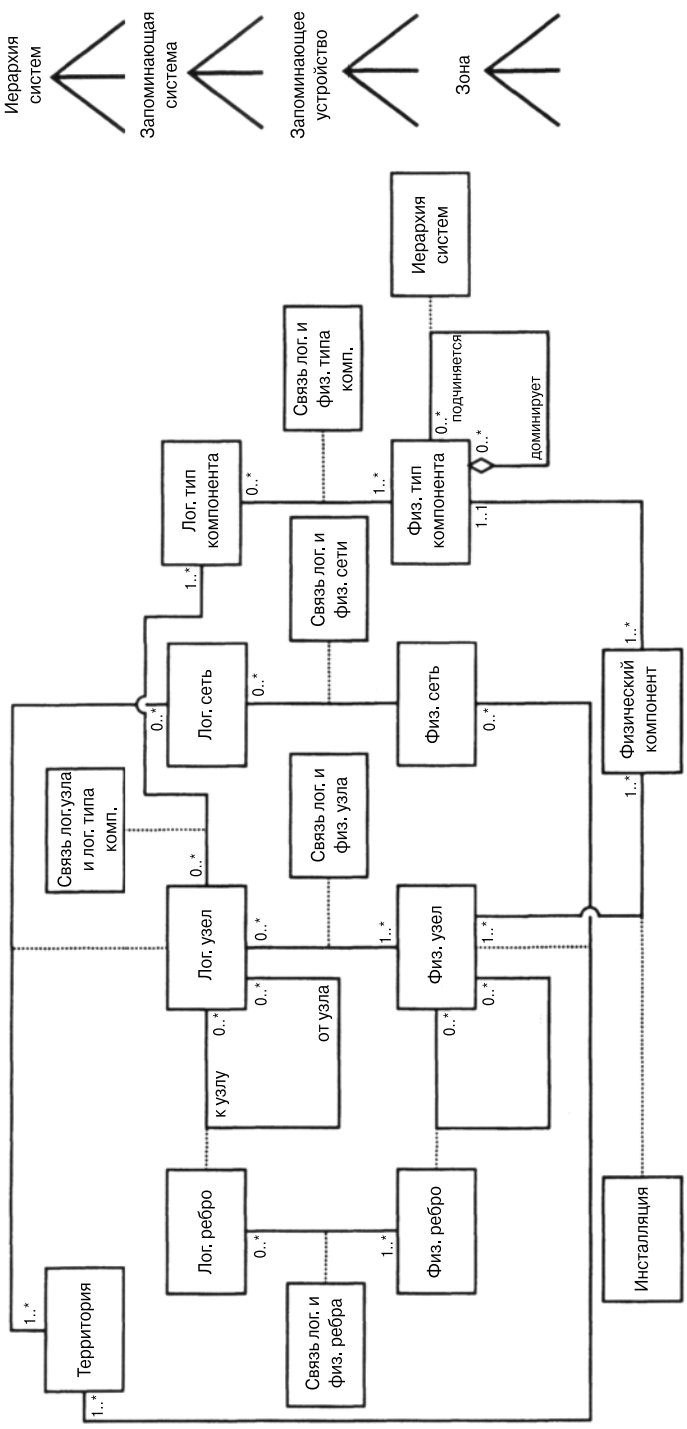


Рис. 54. Моделирование сетей на логическом и физическом уровнях

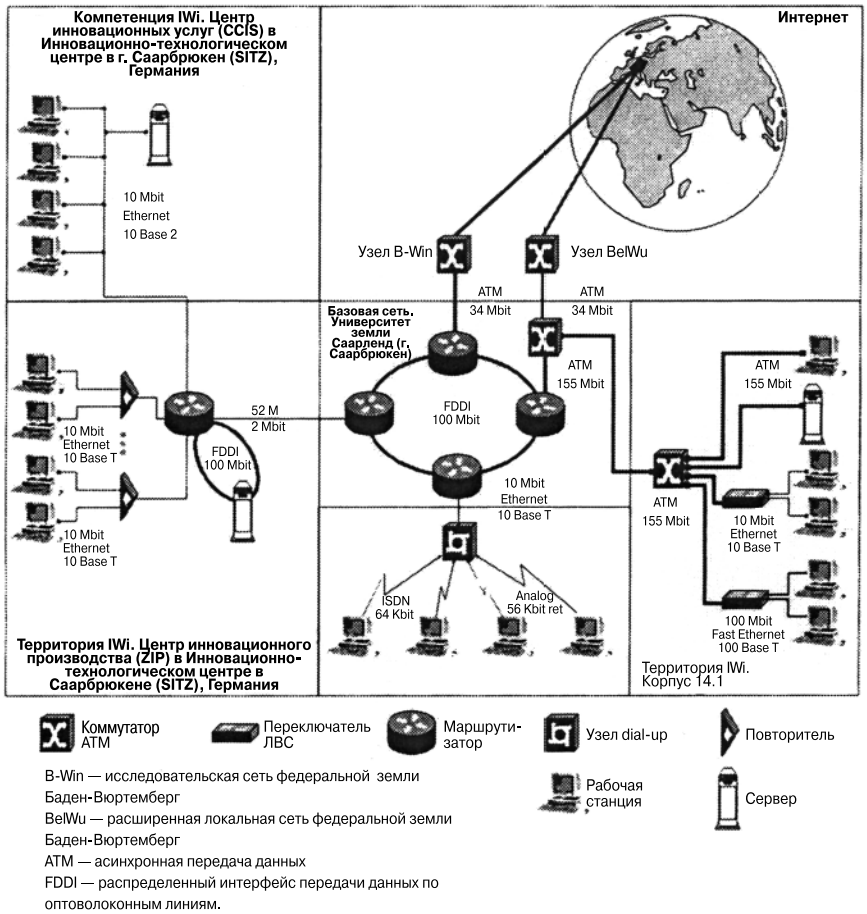


Рис. 55. Пример неоднородной сетевой архитектуры

Различные физические компоненты каждого типа объединены в класс “Физические компоненты”, который не только группирует физические компоненты по типам, но и связывает их с физическими узлами сети. Физические узлы связываются со многими физическими компонентами компьютерных систем. С другой стороны, физические компоненты связываются со многими физическими узлами.

Компоненты программного обеспечения рассматриваются аналогично физическим компонентам компьютерных систем. Для управления сетями используются специальные программы административного контроля.

A.II.3 Моделирование данных

Моделирование данных включает описание объектов данных, необходимых для выполнения функций. Объекты данных, передаваемые другим организационным единицам, служат также для описания выходного результата. Из объектов данных, спроектированных на этапе определения требований, формируется класс объектно-ориентированного метода моделирования.

На рис. 56 иллюстрируется место данных в концепции ARIS. Функциональный и организационный аспекты, обсужденные выше, представлены в виде затененных фигур.

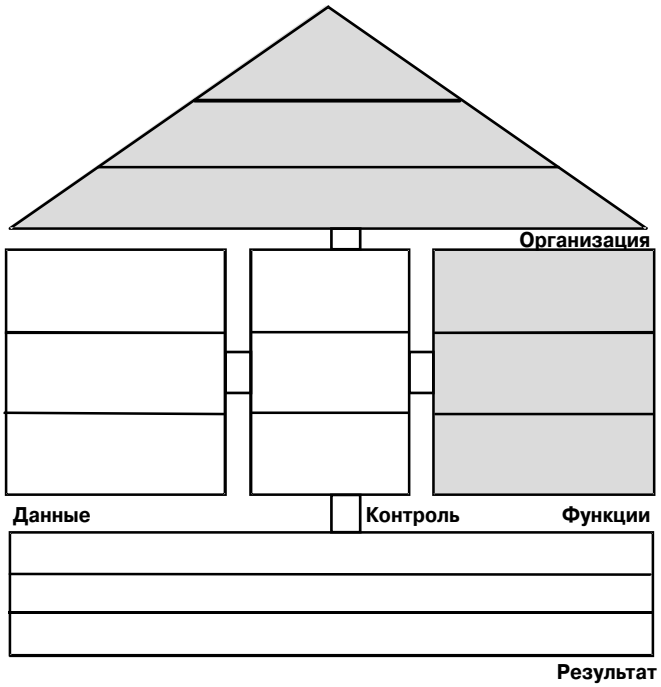


Рис. 56. Данные в концепции ARIS

A.II.3.1 Описание требований аспекта данных

Используемые в моделях объекты данных характеризуются различной степенью детализации. На рис. 57 иллюстрируются примеры объектов данных, включая конкретные наименования некоторых объектов. Объекты данных разделяются на макрообъекты (например, голосовое сообщение или почтовое отправление) и микрообъекты (такие как тип сущности, атрибут и тип связи).

Термин “объект данных” имеет несколько значений. С одной стороны, он обозначает широкий диапазон типов документов, как показано на рис 57. Кроме того, объекты могут взаимодействовать с объектно-ориентированными базами данных. Чтобы избежать неоднозначного толкования, понятие объекта данных обычно дополняется необходимыми деталями.

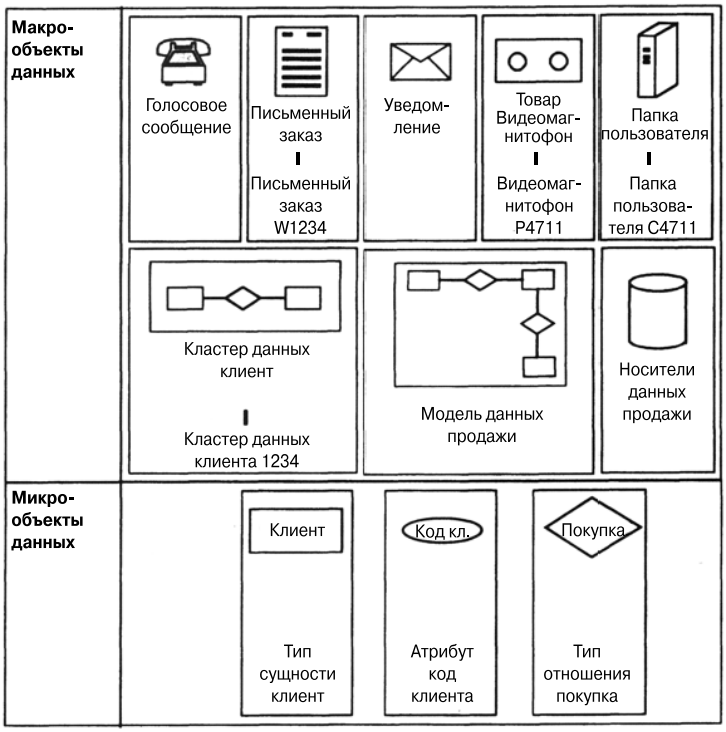


Рис. 57. Объекты данных

Объекты данных выполняют различные роли (рис. 58). Они описывают события или сообщения, относящиеся к управлению бизнес-процессом. Кроме того, объекты данных описывают состояние среды. Результат работы функций, преобразующих входные данные в выходные, относится одновременно и к аспекту выходного результата, и к аспекту данных.

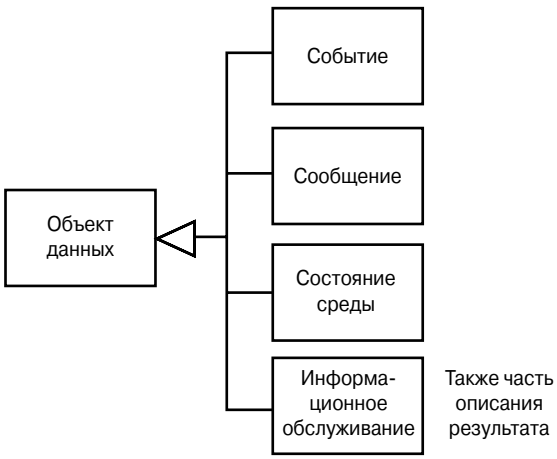


Рис. 58. Роли объектов данных

A.II.3.1.1 Макрообъекты данных

Данные, которые разбиваются на более мелкие составляющие, относятся к макроданным. Для того, чтобы проиллюстрировать данные, необходимые для бизнес-процесса, обычно не требуется детальной разбивки макроданных. Такая детальная разбивка используется только на этапе углубленного анализа бизнес-процесса.

На рис. 59 иллюстрируется метамодель макрообъектов данных. Здесь термин МАКРООБЪЕКТ ДАННЫХ используется как общее наименование набора данных.

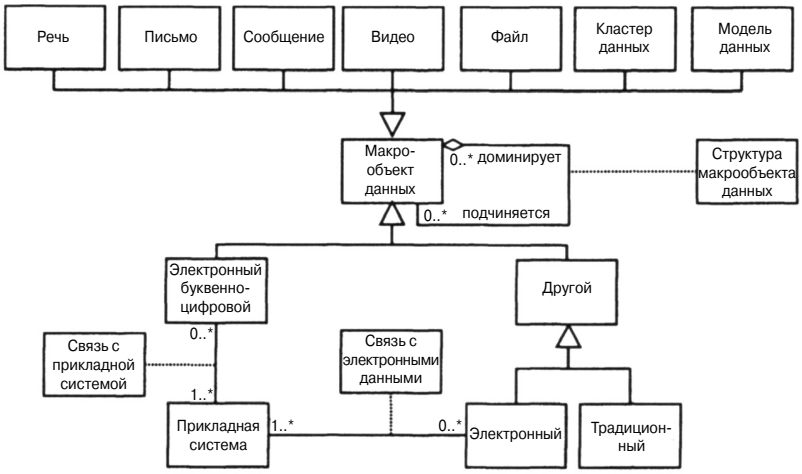


Рис. 59. Метамодель макрообъектов данных

Макрообъекты данных взаимосвязаны; например, пользовательский файл может содержать данные нескольких типов (заказы, уведомления, деловые письма и т.п.).

Модель данных предприятия содержит ряд зональных моделей, включающих многие кластеры данных. Поскольку кластеры данных принадлежат многим зональным моделям, внутри класса МАКРООБЪЕКТ ДАННЫХ существует ассоциация *:* (рис. 60).

Объекты данных имеют буквенно-цифровой формат или состоят из звуков, графических объектов или традиционных бумажных документов. Следовательно, класс МАКРООБЪЕКТ ДАННЫХ разбивается на подклассы ЭЛЕКТРОННЫЙ БУКВЕННО-ЦИФРОВОЙ и ДРУГОЙ; последний в свою очередь состоит из подклассов ЭЛЕКТРОННЫЙ и ТРАДИЦИОННЫЙ.

Как правило, прикладные системы ориентированы на электронные объекты данных типа электронных почтовых отправлений. Если проектируется ситуация “как есть”, то спецификация дизайна включает описание фактически используемых прикладных систем. Целевое проектирование предполагает стратегические решения относительно использования тех или иных прикладных систем.

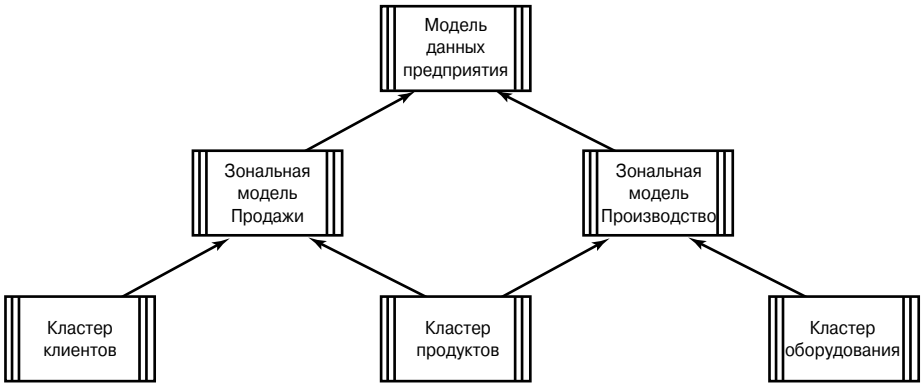


*Рис. 60. Ассоциация *:* между макрообъектами данных*

A.II.3.1.2 Микрообъекты данных

Детальная структура бизнес-приложения моделируется с помощью объектно-ориентированных диаграмм классов или методов ERM (модель “сущность–связь”). В последующих главах обсуждается структура репозитория этого метода, так как метод ERM стандартно используется для моделирования бизнес-процессов на практике. Диаграммы классов рассматриваются более детально в разделе A.III.2.1.1.1 в контексте объектно-ориентированных моделей, связывающих данные и функции.

Элементы ERM иллюстрируются как диаграммы классов в нотации UML. Рассмотрим сначала простую модель ERM, а затем несколько усложним ее.

Простые модели “сущность-связь” для структурирования данных бизнес-приложения состоят из типов сущности и связи, соединенных с помощью ребер. Расширенные модели ERM усиливаются за счет спецификации кардинальностей, добавления операторов специализации и обобщения, реинтерпретации типов связи в типы сущности. Кардинальности в рамках модели “сущность–связь” и диаграмм классов в нотации UML иллюстрируются на рис. 3.

Простая модель “сущность–связь”

На рис. 61 изображена структура данных о продажах в рамках ERM.

КЛИЕНТ, ТОВАР и ВРЕМЯ — это типы сущности, связанные с помощью типов связи ПОКУПКА и ЗАКАЗ. Заказы рассматриваются как транзакционные объекты данных, связанные с типом сущности ВРЕМЯ; остальные элементы представляют основные данные.

Элементы характеризуются ключевыми и описательными атрибутами. Количество экземпляров типа связи, допустимых с точки зрения типа сущности, указано в кардинальностях.

На рис. 62 иллюстрируются классы ТИП СУЩНОСТИ и ТИП СВЯЗИ, представляющие типы сущности и типы связи на уровне бизнеса. КЛИЕНТ, ТОВАР, ВРЕМЯ, ПОКУПКА и ЗАКАЗ — экземпляры классов ТИП СУЩНОСТИ и ТИП СВЯЗИ.

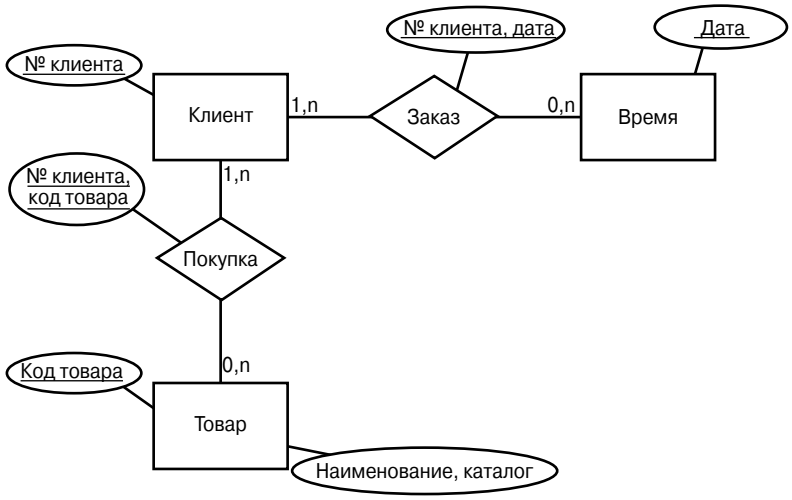


Рис. 61. Структура данных о продажах в рамках ERM

На уровне бизнеса типы сущности и типы связи соединяются с помощью ребра; поэтому вводится класс РЕБРО. Количество допустимых вариантов направления связи указано в атрибутах класса РЕБРО. Поскольку от типа сущности к типу связи могут вести много ребер, кардинальность имеет вид (0..*). Если как минимум две сущности участвуют в связи, кардинальность составляет (2..*).

Между типом сущности и типом связи может существовать много ребер — например, ребра, которые характеризуют связи доминирования и подчинения в структуре спецификации материалов. В подобных случаях класс РОЛЬ РЕБРА позволяет однозначно определить ребро.

Каждый тип сущности имеет определяющий ключевой атрибут. Все ключевые атрибуты формируют класс КЛЮЧЕВОЙ АТРИБУТ. Между классами ТИП СУЩНОСТИ и КЛЮЧЕВОЙ АТРИБУТ существует ассоциация 1:1; поэтому как минимальная, так и максимальная кардинальность равна 1. Например, КОД КЛИЕНТА относится к типу сущности КЛИЕНТ как уникальный определяющий ключевой атрибут; аналогично КОД ТОВАРА относится к типу сущности ТОВАР.

Тип связи определяется ключевыми атрибутами типов сущности; поэтому нет необходимости явно указывать связь между классами ТИП СВЯЗИ и КЛЮЧЕВОЙ АТРИБУТ. Ключевые атрибуты типа связи неявно следуют из связи между классами КЛЮЧЕВОЙ АТРИБУТ и ТИП СУЩНОСТИ. Однако для большей наглядности на рис. 62 показана избыточная связь между классами КЛЮЧЕВОЙ АТРИБУТ и ТИП СВЯЗИ.

Классы ТИП СУЩНОСТИ и ТИП СВЯЗИ характеризуются атрибутами и формируют класс ИНФОРМАЦИОННЫЙ ОБЪЕКТ. Этот класс связывается с классом МАКРООБЪЕКТ ДАННЫХ. Все типы сущностей и типы связи, включая ребра, характеризуют макрообъект данных (например, модель данных департамента маркетинга). Элементы описания характеризуют документы или видеоданные.

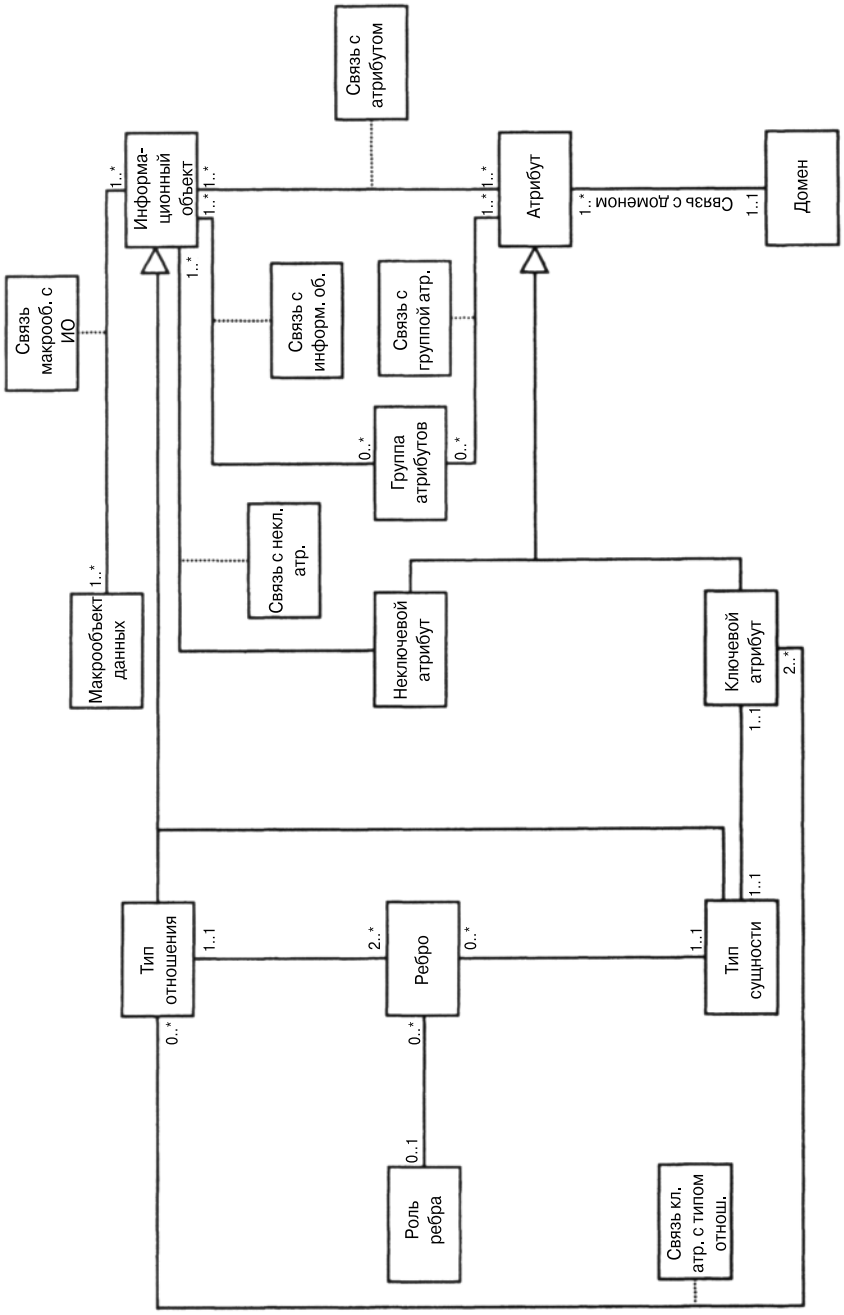


Рис. 62. Простая метамодель ERM

Следующий шаг после проектирования типов сущности и типов связи — определение неключевых атрибутов и их связей. Классы КЛЮЧЕВОЙ АТРИБУТ и НЕКЛЮЧЕВОЙ АТРИБУТ вместе формируют класс АТРИБУТ.

Связь классов НЕКЛЮЧЕВОЙ АТРИБУТ и ИНФОРМАЦИОННЫЙ ОБЪЕКТ имеет вид (1..*):(0..*); это означает, что один информационный объект может иметь много неключевых атрибутов, а один атрибут может характеризовать много информационных объектов. Например, атрибут “Наименование” может относиться как к информационному объекту “Потребитель”, так и к информационному объекту “Поставщик”. Класс ассоциаций СВЯЗЬ С АТРИБУТОМ включает связи с ключевыми и неключевыми атрибутами.

Атрибуты, имеющие общее содержание, объединяются в группы. Например, группа атрибутов “Адрес” включает атрибуты “Почтовый индекс”, “Город”, “Улица”, “Номер дома” и т.п. Группы атрибутов могут пересекаться, поэтому связь между классами АТРИБУТ и ГРУППА АТРИБУТОВ имеет вид (1..*):(0..*). Группа атрибутов включает по меньшей мере один атрибут, при этом не каждый атрибут принадлежит к группе атрибутов. Информационные объекты могут напрямую связываться с группами атрибутов.

Множество значений атрибута характеризует класс ДОМЕН. Каждый атрибут относится только к одному домену.

Связь (1..*):(0..*) между классами АТРИБУТ и ИНФОРМАЦИОННЫЙ ОБЪЕКТ обеспечивает управление атрибутами и доменами.

A.II.3.1.2.2 Расширенная модель “сущность–связь”

Для расширеной модели ERM характерны такие дополнительные свойства:

- реинтерпретация типов связи в типы сущности;
- специализация и обобщение сущностей;
- создание комплексных объектов из типов сущности и типов связи.

Более детальная спецификация кардинальностей касается атрибутов класса ассоциаций РЕБРО и не представляет собой усиления информационной модели.

В метамодели, показанной на рис. 64, реинтерпретация типов вязи в типы сущности требует введения не только общего класса ТИП СУЩНОСТИ, но и специализированных классов как для исходных, так и реинтерпретированных типов сущности. Таким образом, реинтерпретированные типы отслеживаются дважды. С одной стороны, они представляют собой элементы специализации общего класса ТИП СУЩНОСТИ, а с другой — элементы класса ТИП СВЯЗИ.

Добавление операторов специализации и обобщения ведет к созданию класса ОБОБЩЕНИЕ (СПЕЦИАЛИЗАЦИЯ). На рис. 63 иллюстрируется пример расширенной модели ERM. Тип сущности КЛИЕНТ разбивается на типы ЗАРУБЕЖНЫЙ КЛИЕНТ и ВНУТРЕННИЙ КЛИЕНТ. Один элемент специализации соответствует нескольким типам сущности (ЗАРУБЕЖНЫЙ КЛИЕНТ и ВНУТРЕННИЙ КЛИЕНТ). В то же время один тип сущности соответствует только одному элементу специализации.

Поскольку в специализации ключевые атрибуты доминирующего типа сущности относятся к подчиненному типу, один ключевой атрибут характеризует несколько типов сущности (например, номер клиента относится как к общему типу сущности КЛИЕНТ, так и к специализациям ЗАРУБЕЖНЫЙ КЛИЕНТ

и ВНУТРЕННИЙ КЛИЕНТ). Следовательно, кардинальность между классами КЛЮЧЕВОЙ АТРИБУТ и ИСХОДНЫЙ ТИП СУЩНОСТИ имеет вид (1..*):(1..1).

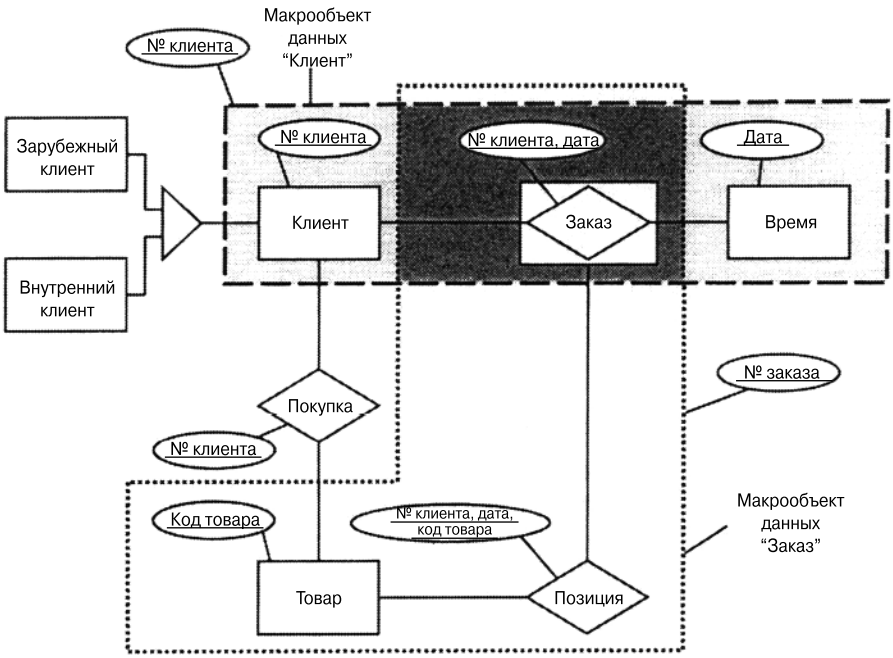


Рис. 63. Пример расширенной модели ERM

Ключевые атрибуты типов связи формируются из ключевых атрибутов соответствующих типов сущности. В целях иллюстрации на рис. 64 изображен отдельный класс ассоциаций ТИП СВЯЗИ–КЛ. АТР.

В моделях ERM сложные сценарии разбиваются на более простые структуры, хотя связь с общей структурой не всегда наглядна. Поэтому вводится класс ОБЪЕКТ, объединяющий многие типы сущности и связи, относящиеся к объекту (*см. Dittrich, Nachrelationale Datenbanktechnologie 1990; Harder, Relationale Datenbanksysteme 1989; Lockemann, Weiterentwicklung relationaler Datenbanken 1991; Kilger, C.: Objektbanksysteme 1996*).

Заказы и данные, относящиеся к определенному клиенту, также представляют собой сложные объекты (см. рис. 63). Таким образом, отдельные объекты на уровне бизнеса (такие как "Чертеж", "Контракт", "Заказ") — это экземпляры класса ОБЪЕКТ на метауровне. Объекты могут пересекаться, поэтому используются кардинальности (0..*) или (1..*).

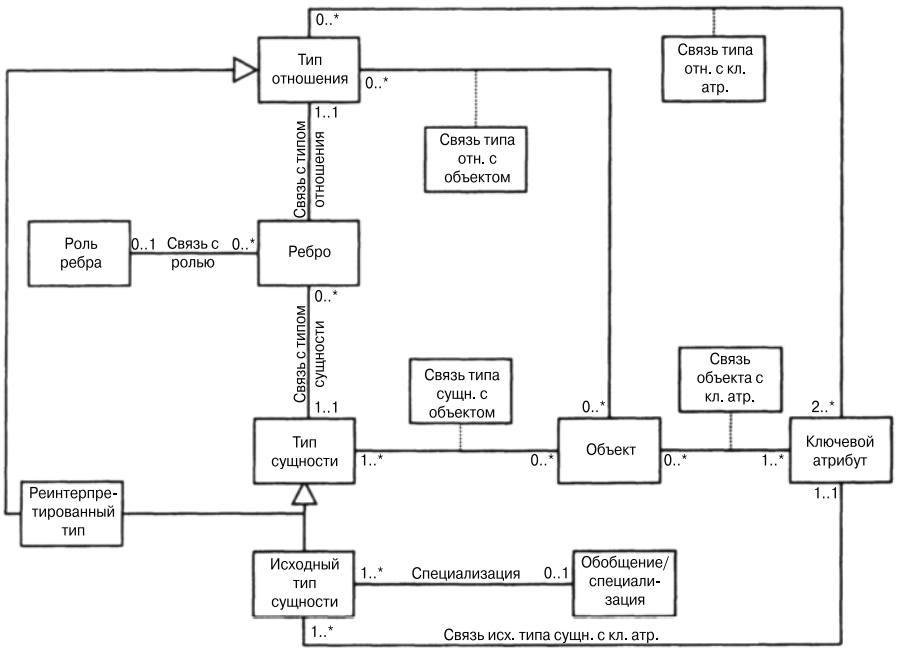


Рис. 64. Расширенная метамодель ERM

A.II.3.2 Конфигурация данных

Модель данных описывает типы издержек и ставки затрат, необходимые для вычисления стоимости процесса с помощью операционно-ориентированного учета затрат.

Модель данных включает объекты данных, необходимые для целевого проектирования и планирования ресурсов.

В управлении потоками данных особую важность имеет контроллинг. Следует различать данные, физически передаваемые в ходе процесса, и данные, хранимые в базах данных, к которым процесс только обращается.

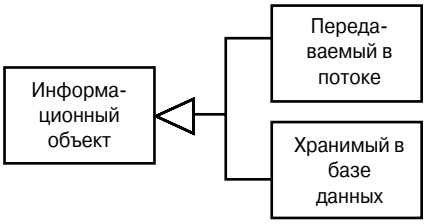


Рис. 65. Группы данных

К первой группе относятся преимущественно нетрадиционные данные, такие как мультимедийные документы, передаваемые в потоке. Вторая группа содержит только ссылки на данные, хранимые в базах данных. Доступ к данным контролируется с соблюдением ограничений и привилегий.

Объекты метамоделей, обсужденные выше, достаточны для описания требований к данным бизнес-процессов (*см. Galler, Vom Geschäftsprozeßmodell zum Workflow-Modell 1997, p. 67*). Иногда имеет смысл моделировать индивидуальные экземпляры данных (конкретные документы), если они используются в каждом бизнес-процессе данного типа. В этом случае модель данных необходимо дополнить средствами администрирования конкретных документов.

С точки зрения бизнеса, структуры данных в бизнес-приложениях все больше документируются с помощью моделей данных. На рис. 66, *а* и 66, *б* иллюстрируются фрагменты моделей данных для приложений SAP R/3 и ARIS. Обе иллюстрации основаны на концепции ERM. Модель данных SAP R/3 сочетает модели ERM согласно Чену (*Chen, Entity–Relationship Model 1976*), структурирует их согласно нотации Синца (*Sinz, Datenmodellierung in SERM 1993*) и Бэчмена (*Bachman, The Programmer as Navigator 1973*).

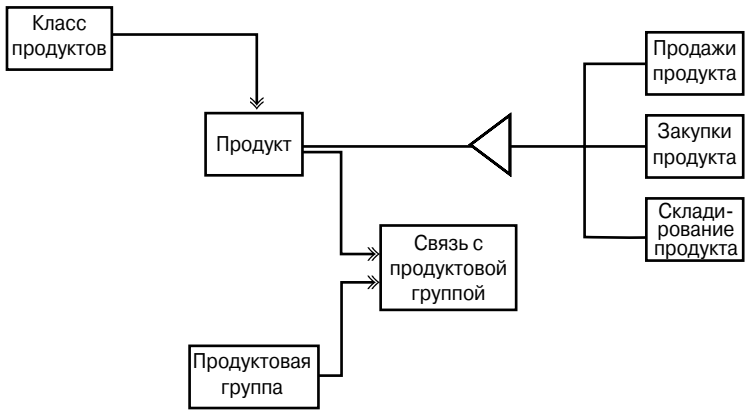


Рис. 66, а. Модель данных SAP R/3

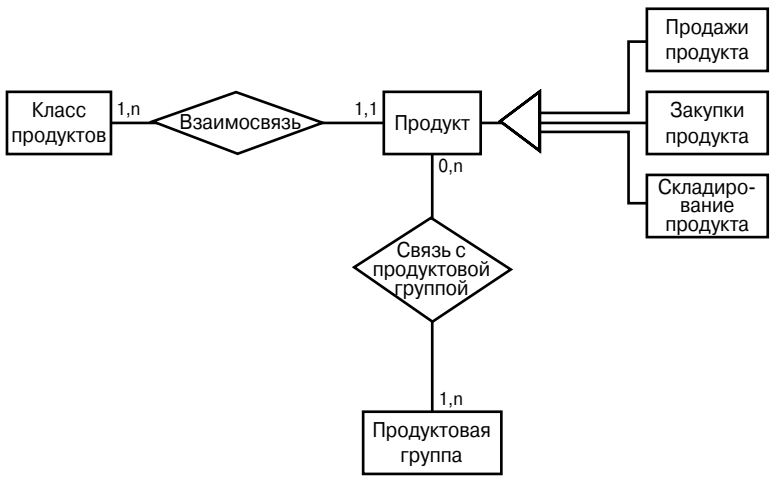


Рис. 66, б. Модель данных приложения ARIS

С моделями данных стандартных приложений обычно проводятся такие действия:

- удаление информационных объектов;
- удаление атрибутов;
- корректировка разрядности атрибутов;
- добавление атрибутов;
- добавление объектов данных.

При выполнении первых трех действий необходимо следить за тем, чтобы приложения, использующие данные, также корректировались. На рис. 67 иллюстрируется, как при изменении разрядности атрибута автоматически корректируется пользовательский интерфейс бизнес-приложения.

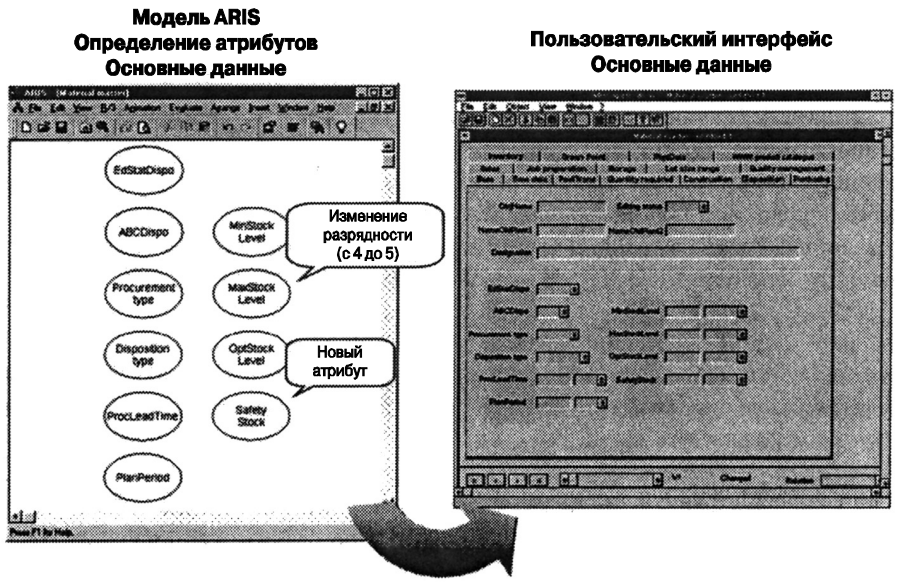


Рис. 67. Конфигурация пользовательского интерфейса

Если новые атрибуты добавляются только с информационной целью и не связаны внешне с прикладной системой, то к существующему пользовательскому интерфейсу добавляются только новые поля. Это делается вручную. Добавление новых атрибутов, поддерживаемых извне, приводит к необходимости создания новых пользовательских интерфейсов для администрирования на уровне экземпляра (добавить, изменить, удалить). Новые интерфейсы должны генерироваться автоматически. Приложения, которые будут работать с новыми структурами данных, должны быть соответственно скорректированы.

A.II.3.3 Спецификация дизайна данных

В процессе спецификации дизайна генерируются конкретные языковые интерфейсы из семантической модели данных. Эти интерфейсы согласуются с определенными моделями данных, хотя их следует отличать от семантических моделей данных. Различают иерархические, сетевые, реляционные и объектно-ориентированные модели данных. Поскольку иерархические модели данных представляют только исторический интерес, а важность сетевых моделей уменьшается, мы сфокусируемся на реляционных моделях данных и частично рассмотрим объектно-ориентированные модели данных.

На первом шаге информационные объекты из описания требований трансформируются в отношения с соблюдением определенных правил.

На втором шаге отношения оптимизируются с помощью нормализации; устраняются все аномалии операций “вставить”, “изменить” или “удалить”. При этом детализируются исходные отношения, которые перешли из описания требований. Если чрезмерная детализация препятствует эффективности, возможна также денормализация.

На третьем шаге определяются условия целостности, которые могут или передаваться из описания требований и приводиться в соответствие с ограничениями реляционной модели, или генерироваться на этапе спецификации дизайна. Условия целостности, передаваемые из описания требований, нуждаются в дополнительной обработке по причине языковых ограничений реляционной модели. Условия целостности необходимо формулировать на специальном языке манипулирования данными.

На четвертом шаге реляционная схема трансформируется в язык описания данных соответствующей системы управления базами данных. При этом добавляются логические пути доступа для поддержки обработки записей. За этим шагом следует этап внедрения.

Реляционная схема формулируется средствами языка описания данных (Data Description Language — DDL). Использование языка описания данных максимально приближает этап внедрения, когда остальные аспекты ARIS взаимодействуют с аспектом данных. Приложения должны взаимодействовать со схемой базы данных, выраженной средствами DDL.

A.II.3.3.1 Создание отношений

Отношения (R_i) определяются списком наименований атрибутов A_{ij}. Отношения удобно представить в виде таблицы (рис. 68). Математически отношение представляет собой декартово произведение доменов атрибутов.

Придерживаясь относительно простых требований, отношения создаются из описания требований к данным в рамках ERM. Каждый тип сущности и каждый тип связи n:n приводит к созданию нового отношения. Тип связи n:n означает, что максимальное значение по меньшей мере двух кардинальностей соседних типов сущностей равно n.

(1) $R_i (A_{i1}, A_{i2}, ..., A_{iz})$ A_{ij} — Атрибут j в отношении i

Деталь (Номер детали, Наименование, Запас)

Деталь	Номер детали	Наименование	Запас
	4717	Гайка	526
	4728	Болт	768

(2) $R_i \subseteq D_{i1} \times D_{i2} \times ... \times D_{iz}$

где D_{ij} — домен A_{ij}

Рис. 68. Представление отношений

С другой стороны, типы связи 1:n не создают нового отношения. Отношения корректируются с помощью добавления ключевого атрибута к типу сущности, из которого исходит кардинальность, максимальное значение которой равно 1 (рис. 69). Добавленный ключевой атрибут называется внешним ключом.

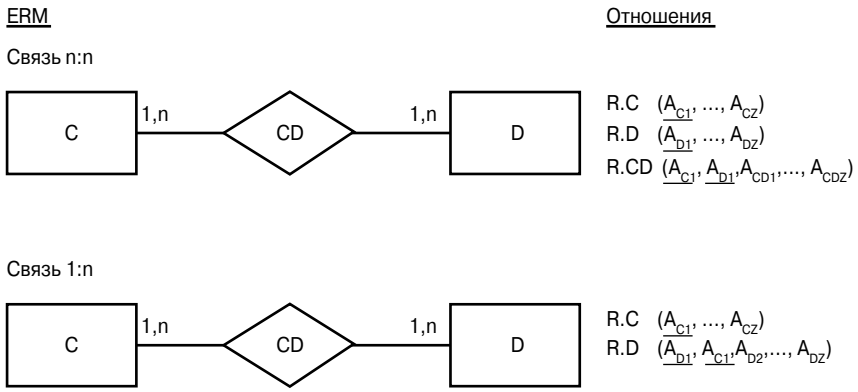


Рис. 69. Создание отношений

В метамодели создания отношений на рис. 70 вводится класс ОТНОШЕНИЕ. Класс ассоциаций СОЗДАНИЕ ОТНОШЕНИЯ обозначает связь с классом ИНФОРМАЦИОННЫЙ ОБЪЕКТ. В соответствии с условиями формирования, информационный объект может иметь 0 или максимум 1 отношение, тогда как одно отношение может характеризовать один или несколько информационных объектов. Атрибут класса ОТНОШЕНИЕ — это собственное имя отношения, которое может совпадать с именем исходного информационного объекта ERM.

Имена атрибутов, определяющих отношение, можно взять из описания требований, хотя их можно изменить. Если изменений не происходит, атрибуты создаются с помощью связи СОЗДАНИЕ ОТНОШЕНИЯ между классами ОТНОШЕНИЕ и ИНФОРМАЦИОННЫЙ ОБЪЕКТ. Однако чтобы подчеркнуть самостоятельную роль спецификации дизайна в создании атрибутов,

атрибуты отношения связаны с общими атрибутами из описания требований с помощью класса ассоциаций СВЯЗЬ ОТНОШЕНИЯ С АТРИБУТАМИ.

Если имена не изменяются при переходе от описания требований, отношения создаются в соответствии с требованиями. Многие прикладные системы поддерживают такую автоматическую миграцию из ERM.

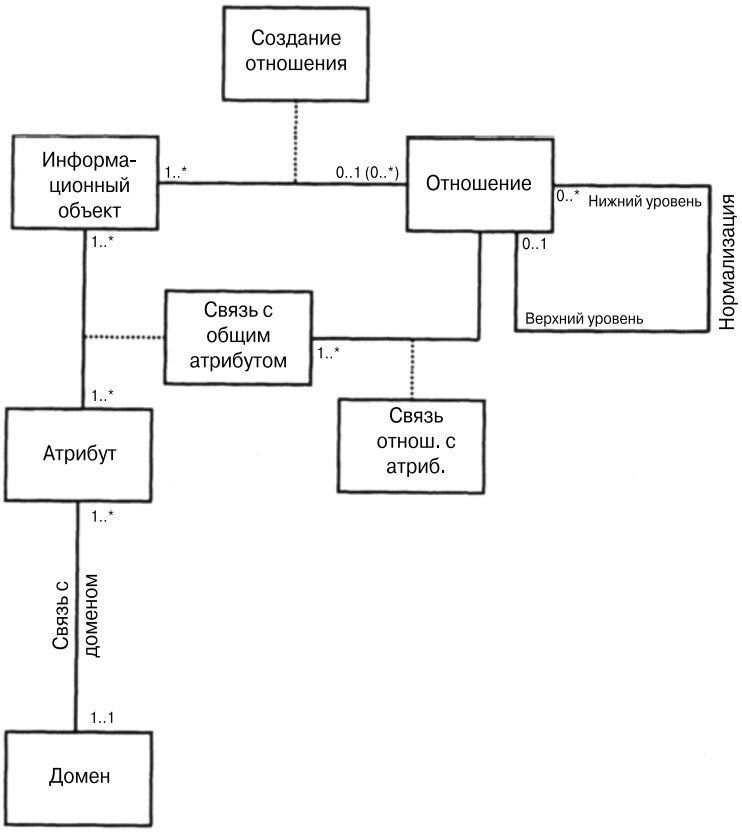


Рис. 70. Метамодель создания отношений

Домены имеют доступ к существующим определениям доменов из описания требований с помощью назначения атрибутов (класс ДОМЕН будет обсуждаться при рассмотрении реляционной модели и условий целостности, относящихся к доменам).

Преобразование типов сущностей и типов связей в отношения реляционной модели не вызывает затруднений. Преобразование сложных объектов связано с необходимостью импортирования процедур или неструктурированных групп данных в среду реляционной модели или даже преобразования модели данных в объектно-ориентированную модель данных.

A.II.3.3.2 Нормализация и денормализация

Трансформация объектов описания требований в исходные отношения реляционной модели данных иногда вызывают нежелательные побочные эффекты — избыточные функциональные зависимости между атрибутами отношений. Эти аномалии устраняются с помощью так называемой нормализации.

Нормализация — процесс преобразования реляционных отношений к виду, отвечающему так называемым нормальным формам. Хотя процесс нормализации разработан именно для реляционных баз данных, его рассматривают как общую процедуру улучшения структур данных и применяют также для других моделей данных. В этой книге каждый отдельный шаг нормализации освещается только на уровне определения. Более детально эта тема рассматривается в других публикациях (*см. Schlageter/Stucky, Datenbanksysteme 1983, p. 183; Wedekind, Datenbanksysteme I 1991, p. 200; Vossen, Datenbank-Management-Systeme 1995, pp. 249–270*). Более того, мы рассмотрим только нормальные формы от первой по третью (так называемые нормальные формы Бойса-Кодда), так как нормальные формы от четвертой и выше используются редко.

Начнем со следующего примера (*см. Schlageter/Stucky, Datenbanksysteme 1983, p. 162*), который относится к проектной организации.

ПЕРВАЯ НОРМАЛЬНАЯ ФОРМА (1 NF)
(1,1) R_EMPLOYEE (**EMP_NO**, NAME, ADDRESS, PROFESSION, **DEPT_NO**)
(1,2) R_PROJECT (**P_NO**, P_NAME, P_DESCR, P_MGR)
(1,3) R_EMP_PROJ (P_NO, EMP_NO, PH_NO, PERCENT_WORK_HOURS)
(1,4) R_DEPT (DEPT_NO, DEPT_MGR, BUILD_NO, JANITOR)
ВТОРАЯ НОРМАЛЬНАЯ ФОРМА (2 NF)
(2,1) R_EMPLOYEE* (EMP_NO, NAME, PH_NO, ADDRESS, PROFESSION, DEPT_NO)
(2,3) R_EMP_PROJ* (P_NO, EMP_NO, PERCENT_WORK_HOURS)
ТРЕТЬЯ НОРМАЛЬНАЯ ФОРМА (3 NF)
(3,4) R_DEPT* (DEPT_NO, DEPT_MGR, BUILD_NO)
(3,5) R_BUILD (BUILD_NO, JANITOR)

Определения

- Отношение R находится в первой нормальной форме (1 NF), если каждый атрибут может содержать только одно значение.
- Отношение находится во второй нормальной форме (2 NF), если оно соответствует первой нормальной форме и каждый неключевой атрибут функционально зависит от ключевого атрибута.
- Отношение находится в третьей нормальной форме (3 NF), если оно соответствует второй нормальной форме и ни один из неключевых атрибутов не зависит транзитивно от ключевого атрибута.

Рассмотрим процесс нормализации на предложенном выше примере.

Отношения, находящиеся в первой нормальной форме, содержат аномалии, которые устраняются при переходе ко второй нормальной форме.

- Имеет место **аномалия вставки**. Например, когда в базу добавляются данные о новом сотруднике, который еще не занят ни в одном проекте. В этом случае номер телефона (PH_NO) не может относиться к новому сотруднику, так как PH_NO находится только среди атрибутов отношения "сотрудник–проект" (R_EMP_PROJ).
- Когда проект закончен и отношение (1,3) следует удалить, появляется **аномалия удаления**. Будет удален и номер телефона сотрудника, хотя номер телефона остается адекватным.
- Имеет место также **аномалия изменения.** Например, если номер телефона сотрудника, занятого в нескольких проектах, изменяется, то многие кортежи отношения R_EMP_PROJ нуждаются в изменении. Происходит массовая корректировка записей базы данных, хотя по сути изменяется *единичный* факт.

Эти аномалии устраняются, когда отношения (1,1) и (1,3) преобразуются во вторую нормальную форму. Учитывая, что номер телефона PH_NO зависит только от ключевого атрибута отношения (1,1) EMP_NO, он добавляется к атрибутам этого отношения. Одновременно номер телефона исключается из отношения (1,3). Отношения (1,2) и (1,4) уже находятся во второй нормальной форме.

Когда нанимается новый охранник (janitor), все кортежи отношения (1,4), относящиеся к департаментам здания, в котором работает охранник, подлежат корректировке. Атрибут JANITOR не зависит непосредственно от DEPT_NO (код департамента), а зависит от BUILD_NO (код здания). Поэтому в третьей нормальной форме вместо отношения (1,4) присутствуют два отношения: отношение (3,4) получено из отношения (1,4) с помощью исключения атрибута JANITOR, а отношение (3,5) добавлено. Отношения (1,2), (2,1) и (2,3) уже находятся в третьей нормальной форме.

Технически процесс нормализации заключается в том, что исходные отношения дробятся. Степень измельчения отношений зависит от их исходных качеств. Если исходное состояние — это так называемое универсальное отношение, где объекты из описания требований не отсортированы, процесс нормализации означает полное изменение структуры. С другой стороны, если описание требований выполнено с использованием, например, ERM, информационные объекты находятся на достаточно высоком уровне нормализации.

Имеет смысл проверить систему с помощью нормализации, даже если информационные объекты спроектированы тщательно, ключевые атрибуты определены и неключевые атрибуты будут добавлены позже.

При дроблении класса ОТНОШЕНИЕ в процессе нормализации исходные отношения превращаются в новые отношения. Учитывая, что один информационный объект может вести ко многим отношениям, кардинальность для класса ОТНОШЕНИЕ превращается в (0..*), как показано в скобках на рис. 70.

Связь между отношением реляционной модели и отношением нижнего или верхнего уровня нормализации отображает класс НОРМАЛИЗАЦИЯ.

Описание требований и исходные отношения корректируются при создании новых отношений в процессе нормализации. Однако это ведет не к расширению диаграммы классов, изображенной на рис. 70, а только к созданию новых экземпляров класса СВЯЗЬ ОТНОШЕНИЯ С АТРИБУТАМИ.

A.II.3.3.3 Условия целостности

Выполнение условий целостности данных необходимо для того, чтобы база данных корректно моделировала объективную реальность (*см. Blaser/Jarke/Lehmann, Datenbanksprachen und Datenbankbenutzung 1987, p. 586*).

Поскольку таблицы реляционных моделей не слишком хорошо отображают семантические факты, определяются условия целостности с помощью языка манипулирования данными. Условия целостности могут также определяться внутри прикладной программы. Учитывая принцип локальности и преимущества централизованного контроля целостности данных, имеет смысл включить условия целостности в модель данных. Современные базы данных позволяют хранить много функций (*см. Dittrich/Gatziu, Aktive Datenbanksysteme 1996*).

Условия целостности ссылаются на семантическое содержание и внедрение модели данных. Условия целостности тесно связаны с базой данных и не касаются вопросов пользовательского интерфейса. Вот почему в данной работе мы фокусируемся на условиях семантической целостности.

Условия состоятельности (условия ссылочной целостности) касаются атрибутов, экземпляров кортежей и отношений, производных от типов связей (*см. Blaser/Jarke/Lehmann, Datenbanksprachen und Datenbankbenutzung 1987, p. 588*).

Язык структурированных запросов SQL (Structured Query Language) — универсальный компьютерный язык, применяемый для создания, модификации и управления данными в реляционных базах данных. В SQL условия целостности определяются с помощью команд ASSERT и проверяются при выполнении соответствующих действий.

Например, чтобы при удалении кода сотрудника (EMP_NO) в отношении R_EMPLOYEE связь с навыками сотрудника (в отношении OWNS, которое ссылается на отношение SKILLS) также удалялась, следует использовать следующее определение триггера (*см. Blaser/Jarke/Lehmann, Datenbanksprachen und Datenbankbenutzung 1987, p. 592*).

```
DDEFINE TRIGGER T1
ON DELETE OF EMPLOYEE (EMP_NO):
DELETE OWNS
WHERE OWNS.EMP_NO = EMPLOYEE.EMP_NO.
```

На рис. 71 иллюстрируется несколько примеров определения условий целостности с помощью команд ASSERT языка SQL.

Объяснение	Команды SQL
1) Условие касается атрибута. Значение EMP_NO должно иметь 4 знака.	ASSERT IB1 ON EMPLOYEE: EMP_NO BETWEEN 0001 AND 9999
2) Условие касается многих атрибутов записи. SALARY_SUM (зарплата департамента) должна быть меньше ANNUAL_BUDGET.	ASSERT IB2 ON DEPARTMENT: SALARY_SUM < ANNUAL_BUDGET
3) Условие касается многих экземпляров одного типа записи (отношения). Зарплата одного сотрудника не может превышать больше чем на 20% среднюю зарплату по департаменту.	ASSERT IB3 ON EMPLOYEE X: SALARY * 1,2 ≤ (SELECT AVG(SALARY) from EMPLOYEE WHERE DEPART = X.DEPART)
4) Условие касается многих экземпляров во многих отношениях. Значение SALARY_SUM для департамента должно быть равно сумме значений SALARY всех сотрудников департамента.	ASSERT IB4 ON DEPARTMENT X: SALARY_SUM = (SELECT SUM(SALARY) from EMPLOYEE WHERE DEPART = X.DEPARTNO)

Рис. 71. Условия целостности
(Reuter, Sicherheits- und Integritatsbedingungen 1987, pp. 381, 385)

На рис. 72 иллюстрируются ключевые взаимосвязи, порождаемые условиями целостности. Левый фланг рис. 72, содержащий классы ОТНОШЕНИЕ, АТРИБУТ и ДОМЕН, представляет собой отправную позицию для обсуждения вопросов целостности. Класс ТИП ЦЕЛОСТНОСТИ описывает различные типы условий целостности (*см Reuter, Sicherheits- und Integritatsbedingungen 1987, p. 380*). Условия целостности различают по их рангу, т.е. по типу и количеству объектов, которых касается условие целостности (например, см. рис. 71). Кроме того, условия целостности различают по времени проверки (или условие проверяется всегда, или после выполнения определенного количества операций), по способу проверки (условия состояния или условия перехода) или по вызываемым действиям.

На рис. 72 каждое конкретное условие целостности относится к одному типу целостности. Условие целостности может касаться одного или многих отношений и ассоциаций атрибутов одного или многих отношений. Пределы значений атрибутов проверяются с помощью доменов.

A.II.3.3.4 Логические пути доступа

Выполнение вложенных запросов SQL может привести к потере эффективности базы данных. Чтобы избежать потери эффективности, следует разработать утилиты (обслуживающие программы) для обеспечения доступа к отдельным записям или группам записей базы данных. В частности, следует избегать необходимости последовательного просмотра таблиц.

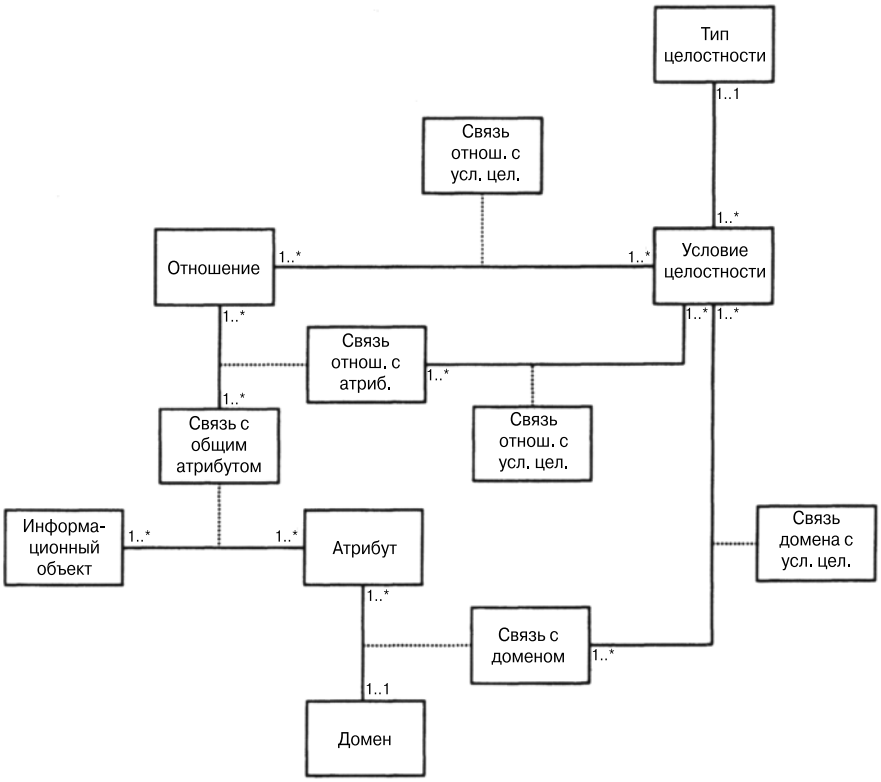


Рис. 72. Метамодель условий целостности

Как правило, утилиты обеспечивают доступ к записям по их ключам и доступ к группам записей в определенной последовательности (сортировка). Логические пути доступа для первичных ключей классифицируются на основании последовательных организационных форм, древовидных индексов или распределенных организационных форм. Пути доступа для вторичных ключей создаются с помощью инвертированных списков (таблиц индексов).

В спецификации дизайна обуславливается, какой тип поддержки требуется для определенных атрибутов. Класс ТИП ЛОГИЧЕСКОГО ПУТИ ДОСТУПА характеризует различные типы поддержки. Связь между атрибутом отношения и типом пути доступа характеризует логический путь доступа. Атрибут определенного отношения участвует в определении многих путей доступа.

Пути доступа в спецификации дизайна певоначально определяются в соответствии с общими предположениями относительно количества кортежей в таблице и ожидаемого количества обращений к таблице. Когда база данных создана и показатели эффективности известны, утилиты доступа могут детализироваться дальше.

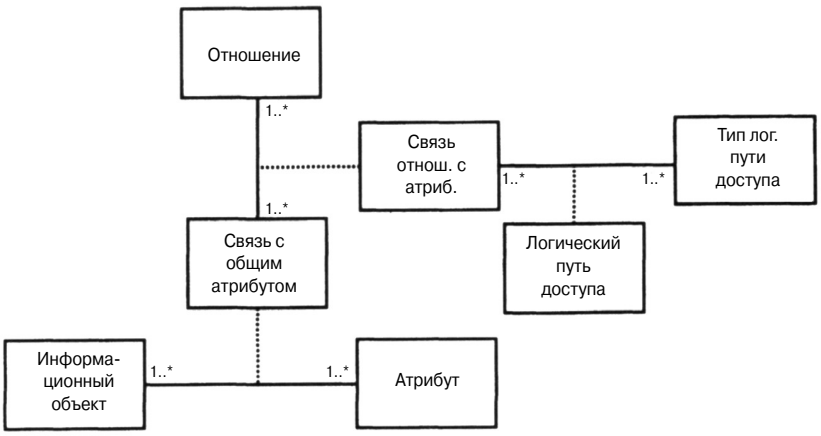


Рис. 73. Логические пути доступа

A.II.3.3.5 Схема базы данных

На заключительном этапе спецификации дизайна структуры данных переводятся на язык определения данных конкретной системы баз данных, например, ORACLE, INFORMIX, CA-INGRES. Учитывая, что реляционные модели определены математически, а условия целостности на языке SQL стандартизованы, перевод схемы реляционной базы данных на язык конкретной системы баз данных не вызывает затруднений.

Если предприятие использует несколько различных систем управления базами данных (СУБД), нейтральная схема реляционной базы данных (рис. 74) может быть преобразована в несколько конкретных схем. Поставщики программного обеспечения, предлагающие продукты для различных баз данных, часто встречаются с такой ситуацией. Конкретные схемы баз данных включают конкретные определения логических путей доступа и условия целостности.

A.II.3.4 Внедрение модели данных

Начнем со спецификации дизайна модели данных, т.е. традиционной схемы базы данных с ее отношениями, атрибутами и условиями целостности. В спецификации дизайна определяются также логические пути доступа к определенным ассоциациям атрибутов с учетом частоты обращений и запросов.

В процессе внедрения концептуальные схемы моделируются во внутренние схемы, которые описывают те же фрагменты реальности, что и концептуальная схема. Семантический аспект не рассматривается на этом этапе. Моделирование внутренней схемы из концептуальной схемы возможно без знания семантического окружения.

Задача администраторов базы данных — структурировать внутреннюю схему, чтобы создать эффективные структуры базы данных с помощью ин-

формационных технологий. Администраторы должны знать параметры использования и отслеживать время отклика на запросы к базе данных от различных приложений. При этом администраторы могут ничего не знать о содержании приложений.

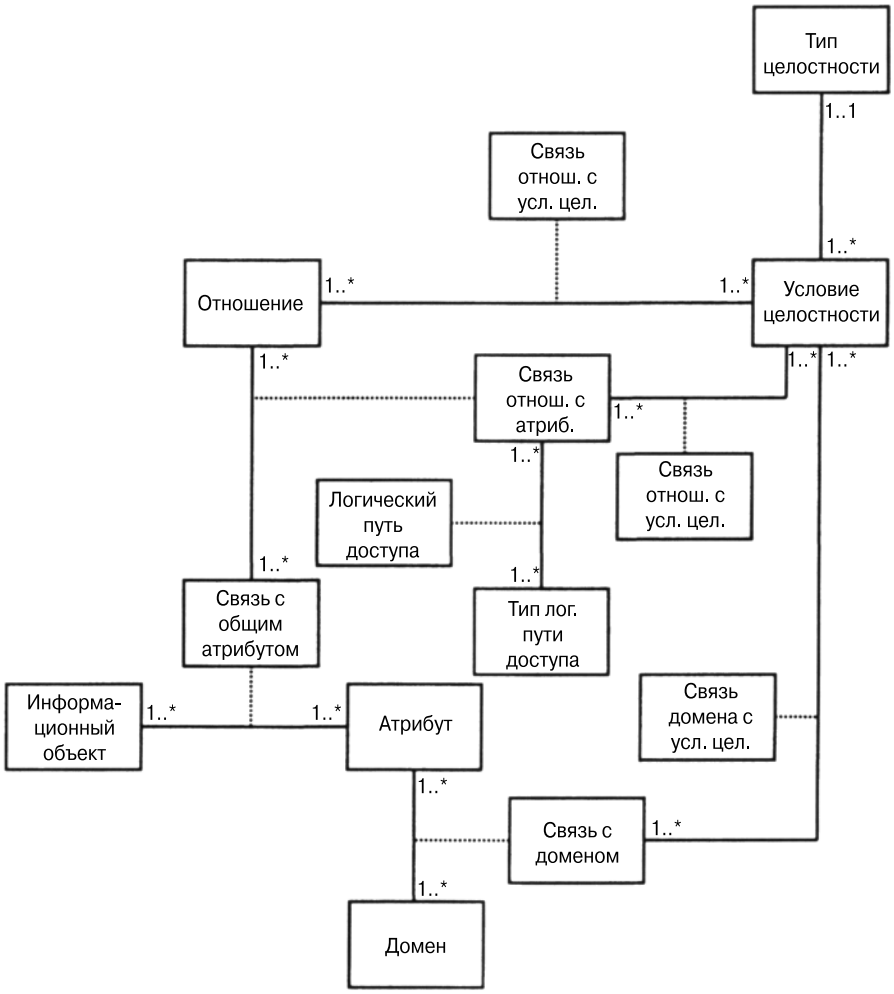


Рис. 74. Нейтральная схема реляционной базы данных

Автономия этапа внедрения поддерживается использованием специфической корпоративной терминологии, которая моделируется на логическом уровне спецификации дизайна. Например, понятия ОТНОШЕНИЕ и АТРИБУТ на этапе внедрения (рис. 75) подчиняются терминам ТИП ЗАПИСИ и ПОЛЕ. Термин ТИП ЗАПИСИ обозначает тип записи, который характеризуется определенной комбинацией атрибутов. Класс СТРАНИЦА включает различные типы записей. Задача администратора состоит в том, чтобы оптимизировать базу данных, размещая часто используемые типы записей физически близко друг к другу.

Внедрение отличается от спецификации дизайна тем, что на этапе внедрения последовательность полей можно изменить, поля переименовать, данные сжать или использовать специфические форматы полей. Более того, создаются виртуальные поля, т.е. задаются правила преобразования полей, если содержание полей создается из содержания других полей (например, итоговые поля).

Возможно различие между отношениями и записями, если отношения разбиваются на многие записи или многие отношения объединяются в одну физическую запись.

Условия состоятельности и целостности, определенные на уровне спецификации дизайна, на этапе внедрения определяются на физическом уровне в виде процедур. Кроме того, логическим путям доступа ставятся в соответствие конкретные методы физического доступа или определяются дополнительные пути физического доступа.

В добавление к именам ЗАПИСЬ и ПОЛЕ, имеющим аналоги в спецификации дизайна, вводятся классы СТРАНИЦА и ЗОНА (см. рис. 75), т.е. создаются дополнительные организационные элементы для оптимизации структур распределения памяти. Исходные организационные элементы лежат в основании физического размещения данных и доступа к данным, находящимся на внешних устройствах.

Потенциал оптимизации страниц и зон очевиден, если речь идет о достоверности данных. Так, эффективнее иметь доступ ко многим записям, размещенным на одной странице, чем обрабатывать записи, распределенные между несколькими страницами. Аналогично, упорядоченный доступ к страницам эффективнее, чем доступ к страницам в произвольном порядке.

Язык описания хранения данных (Data Storage Description Language — DSDL) связывает внутренние и внешние модели, внедряя структуры распределения памяти. Физические пути доступа моделируются с помощью конкретных индексных таблиц, цепочек хэш-функций.

Физические пути доступа определяются на уровне связи записи и поля. Они или отображают логические пути доступа из спецификации дизайна, или создаются на этапе внедрения, когда появляются детальные сведения о параметрах эффективности.

Определения на уровне физической структуры данных подчиняются цели независимости данных. Изменения физических устройств или программного обеспечения должны влиять только на внедрение, но не на концептуальную схему базы данных. Одна концептуальная схема базы данных может включать несколько внутренних схем баз данных. С другой стороны, возможна корректировка концептуальной схемы базы данных без изменения ее физической схемы.

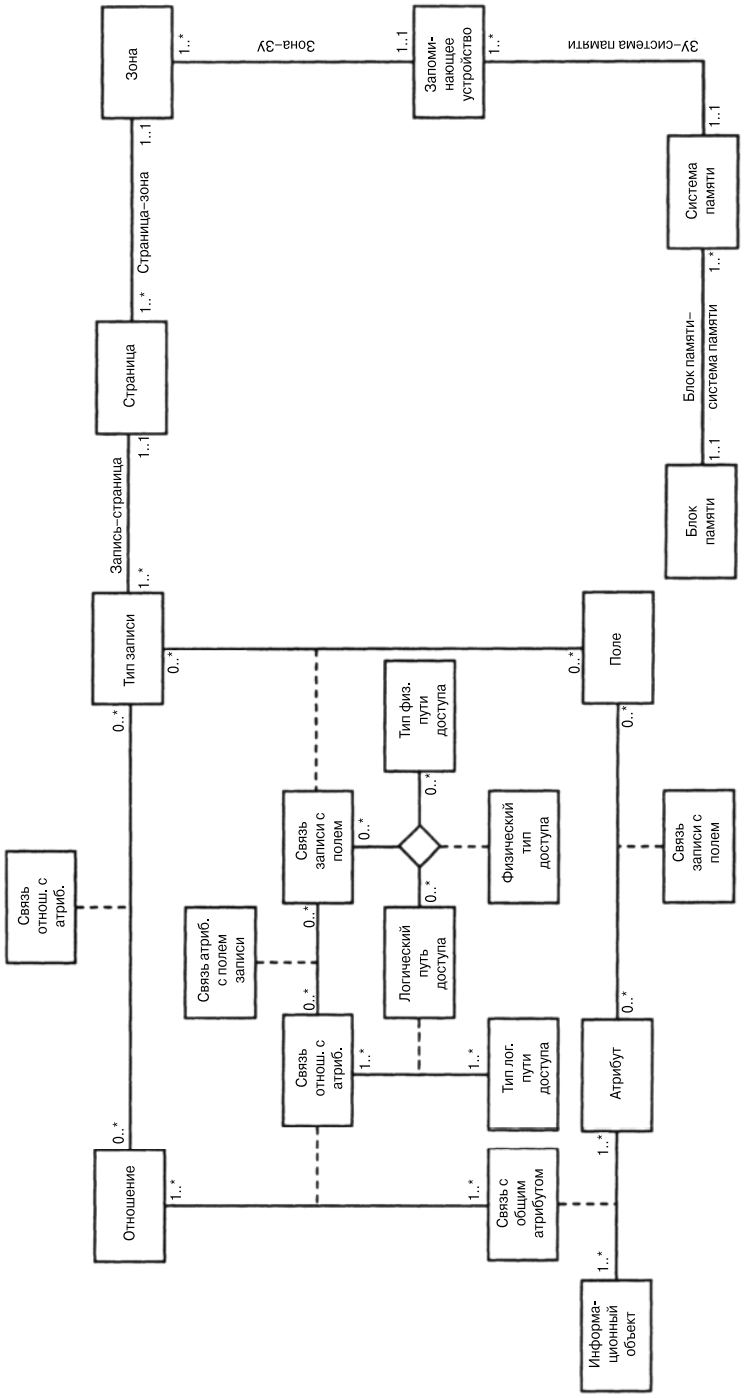


Рис. 75. Внедрение реляционной базы данных

A.II.4 Моделирование результата

На рис. 76 иллюстрируется место выходного результата в концепции ARIS. Аспекты, обсужденные выше, представлены в виде затененных фигур.

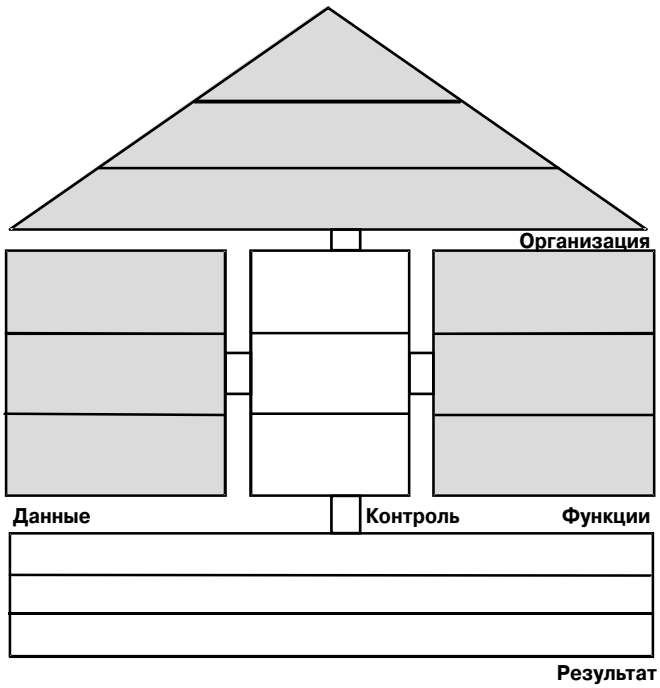


Рис. 76. Результат в концепции ARIS

Выходной результат представляет собой итог выполнения процессов, использующих входные данные. Степень детализации описания результата соответствует степени детализации процессов.

Описание результата — один из ключевых аспектов моделирования бизнес-процессов. Это относится как к стратегическому планированию, так и к моделированию операционных бизнес-процессов.

Термин “Результат” включает различные типы результата (материальный продукт и услуги) и рассматривается с различной степенью детализации. Мы будем использовать этот термин в значении “Продукт”, хотя по отношению к услугам это звучит несколько непривычно.

Спецификация дизайна и описание внедрения не рассматриваются по отношению к результату, потому что результат не внедряется самостоятельно. Все большее количество изделий (станки, автомобили и даже бытовые стиральные машины) оснащено собственными компьютерными системами, которые согласуются с концепцией ARIS.

Информационные услуги моделируются как объекты данных. Необходимо точно указать статус объекта данных, если нужно определить статус результата.

Спецификация дизайна и описание внедрения объектов данных содержатся в разделе “Моделирование данных”.

A.II.4.1 Описание требований результата

Бизнес-моделирование материального продукта широко представлено в научных и практических разработках. STEP Model — эталонная модель для описания геометрических, физических, химических, функциональных и административных характеристик материальных продуктов (*см. Grabowski et al., Integriertes Produktmodell 1993*). Моделирование данных — метод выбора и описания продуктовых моделей как моделей данных.

В отличие от описания материальных изделий, моделирование услуг находится на начальной стадии развития.

Даже некоммерческие организации, специализирующиеся на предоставлении услуг, только начинают описывать свои продукты.

Приведем примеры определения продуктов.

- Продукт — это товар (или группа товаров), предназначенный для использования вне операционного подразделения, где он создан (внутри предприятия или за его пределами; *см. KGSt, Das Neue Steuerungsmodell 1994, p. 11*).
- Продукт — это то, что операционные подразделения предоставляют бизнес-единицам вне своей организации, учитывая запросы других предприятий, соблюдая требования законодательства и независимо от того, должны эти предприятия платить за продукт или нет (*см. KGSt, Wege zum Dienstleistungsunternehmen 1992*).

Документирование продукта требует расходов. Можно ли определенный результат процесса назвать продуктом, зависит от поставленной цели.

В материальном производстве наименование продукта присваивается не после каждой операции, а только после достижения определенного статуса (*см. Scheer, Business Process Engineering 1994, p. 154*). Промежуточные шаги характеризуются информацией относительно операции в целом.

При моделировании процессов, если существует взаимосвязь “поставщик–клиент” со следующей операцией или организационной единицей, каждое изменение статуса после выполнения операции приводит к новому виду продукта.

На рис. 78 иллюстрируется, что выходной результат (входной поток) и продукт разделяются на материальный продукт и услуги, а услуги, в свою очередь, — на информационные услуги и другие услуги. Важнейшая современная тенденция — формирование информационного общества, где сферы материального производства и услуг тесно взаимосвязаны.

Моделирование выходного результата заключается в моделировании иерархических продуктовых структур (продуктовых сетей).

В отличие от расширенных продуктовых моделей данных, включающих намного более сложные и детальные факты, продуктовые структуры характеризуют только ограниченный семантический диапазон. Различные уровни моделирования и типичные примеры объектов моделирования иллюстрируются на рис. 79.

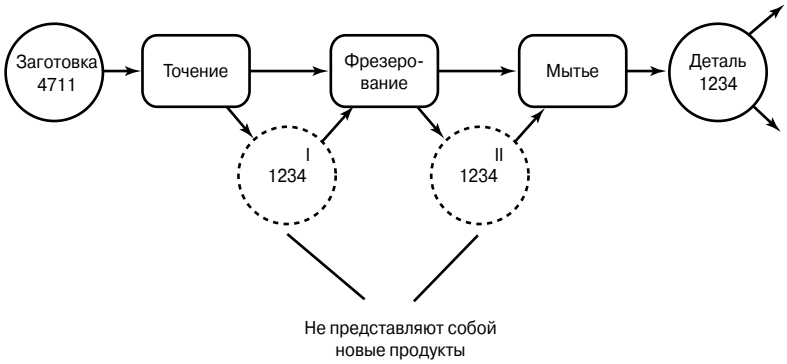


Рис. 77. Создание материального продукта

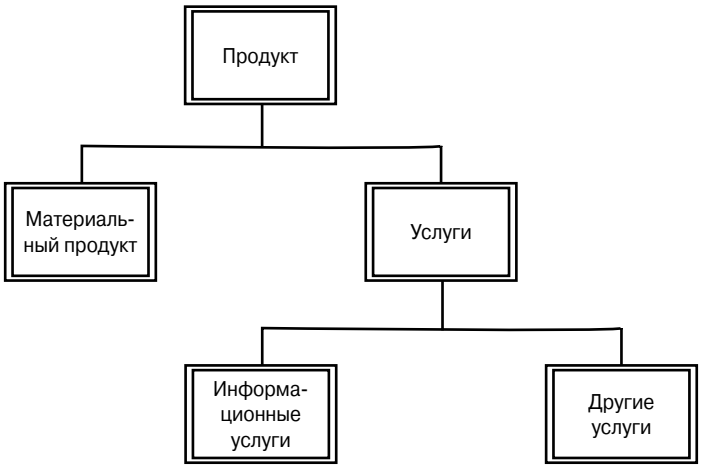


Рис. 78. Типы продукта

Примеры объектов моделирования включают различные подклассы на втором уровне моделирования, иерархические продуктовые структуры и связь между материальным продуктом и услугами. Связи “часть и целое” на втором уровне обозначаются пунктирными линиями.

На рис. 80 рассматривается общий класс ПРОДУКТ наряду с подклассами МАТЕРИАЛЬНЫЙ ПРОДУКТ и УСЛУГИ. Объекты моделирования второго уровня — элементы названных классов. Структура продукта характеризуется классом связей СТРУКТУРА ПРОДУКТА; элементы последнего — это связи “часть и целое” на втором уровне. На рис. 81, *а* и 81, *б* иллюстрируются некоторые примеры продуктовых моделей на втором уровне моделирования.

Если для других аспектов ARIS моделирование на уровне отдельных экземпляров рассматривается как исключение, то при моделировании результата это не так. Промышленные предприятия используют исчерпывающую классификацию продуктов материального производства, поэтому все образцы законченных, промежуточных и исходных продуктов детально описываются в соответствующих файлах.

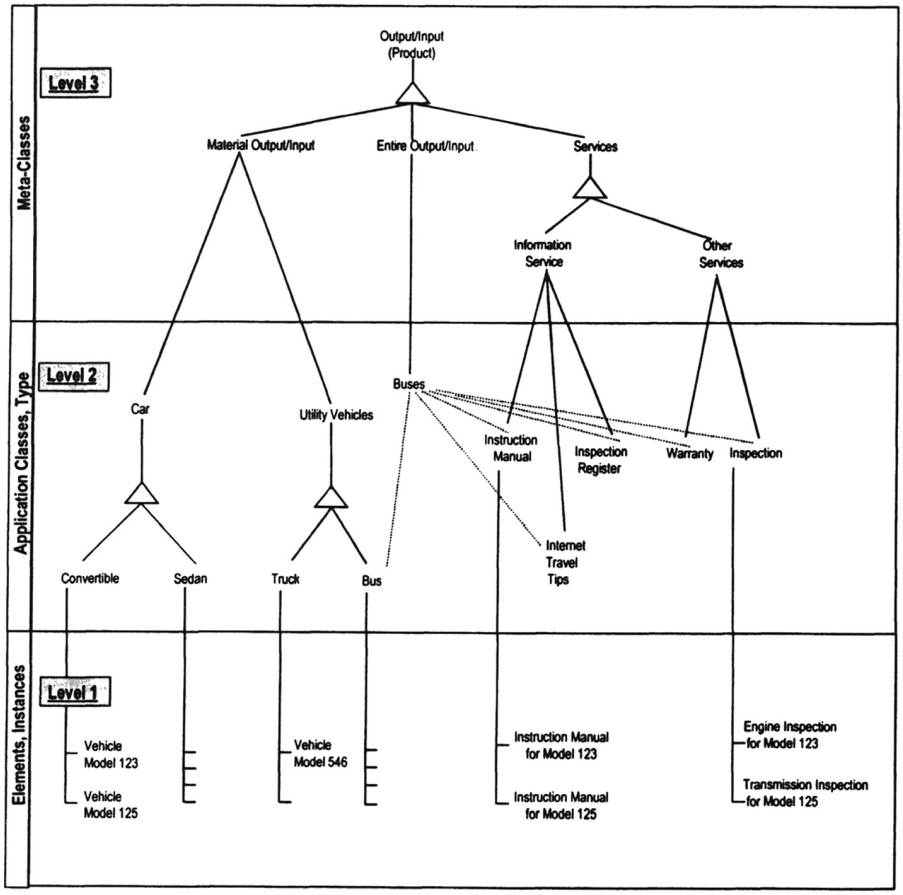


Рис. 79. Объекты моделирования продукта

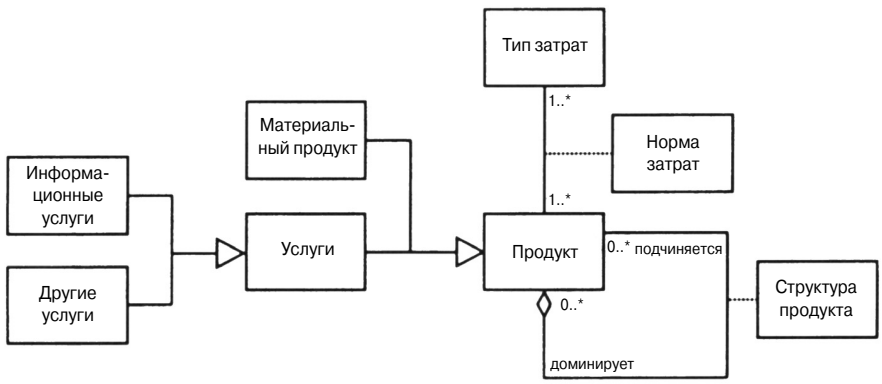


Рис. 80. Метамодель продукта

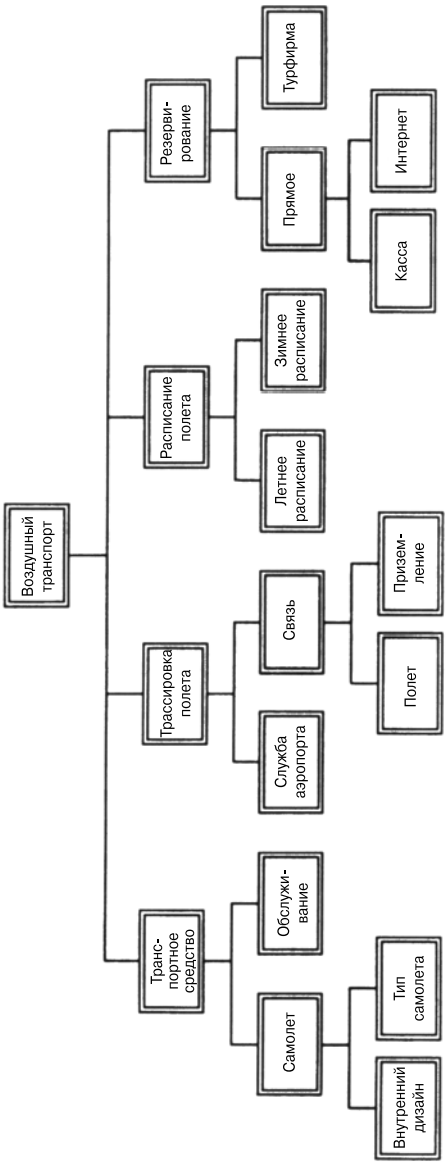


Рис. 81, а. Пример модели услуг

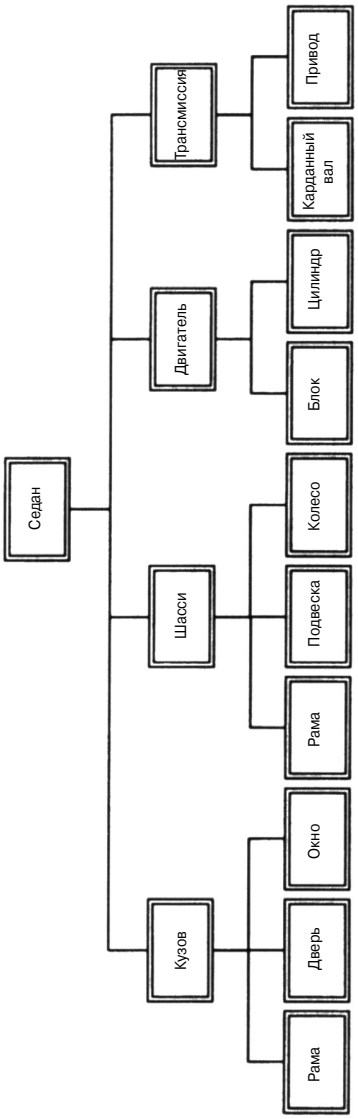


Рис. 81, б. Пример модели материального продукта

На рис. 82 иллюстрируется модель седана на уровне экземпляра.

В сфере услуг сегодня также существует тенденция описывать на втором уровне продуктовых моделей не только классы, но и конкретные продукты. Например, в Берлине сектор административных услуг насчитывает свыше десяти тысяч наименований услуг.

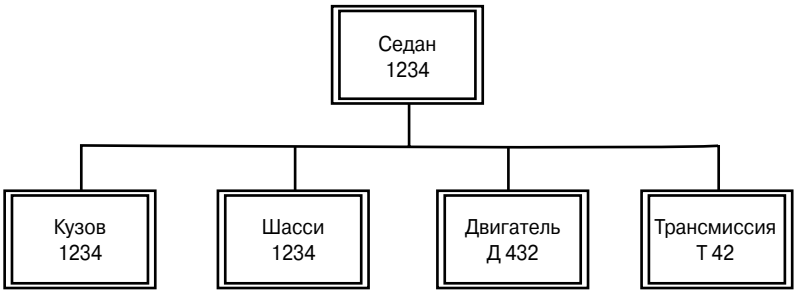


Рис. 82. Модель материального продукта на уровне экземпляра

Описание продукта управляется с помощью систем планирования производства и контроля. Оцениваются затраты на создание продукта. В метамодели на рис. 80 класс ТИП ЗАТРАТ описывает категории издержек (материалы, персонал, амортизационные отчисления), относящиеся к продукту. Связь НОРМА ЗАТРАТ содержит конкретные средние значения затрат определенного типа на единицу выпуска.

Информационные продукты, такие как сертификаты, руководства пользователя и т.п., представляются как объекты данных в концепции ARIS. В аспекте результата они рассматриваются как входные или выходные данные функций. В аспекте данных определения объектов данных должны соответствовать требованиям информационных продуктов.

Как и в случае других аспектов, существует некоторое пересечение аспектов результата и данных в концепции ARIS. Материальный продукт описывается в базах данных с помощью продуктовых структур и классификаторов операций обработки и контроля. С другой стороны, прикладные системы не содержат столь детальной информации об услугах.

В отличие от систем планирования производства и контроля (Production Planning and Control — PPC), только информация, необходимая для организационной оптимизации процессов, требуется для описания результата. В системах планирования производства и контроля используется значительно больше детальных атрибутов и типов данных.

В то время как модель результата представляет продукт логически, модель данных ведет к физическому воплощению. Однако следует заметить, что существует тесная связь между аспектом результата в концепции ARIS и продуктовыми описаниями материального выпуска в системах PPC и CAD. Например, в неизбыточных организациях экземпляры продукта системы PPC можно использовать для моделирования экземпляра ARIS. С другой стороны, программную систему ARIS Toolset можно использовать как удобный инструмент создания интерфейса для ввода данных и вывода презентаций.

A.II.4.2 Конфигурация результата

Системы функционально-стоимостного анализа используют результат процесса, чтобы сосредоточить внимание на определенных факторах затрат.

Операционный контроль требует создания информационных объектов. Информация на уровне экземпляра создается с помощью копирования описаний классов второго уровня.

На промышленных предприятиях описание материального продукта находится в спецификации материалов. Учитывая пересечение аспектов данных и результата, в общем случае достаточно ссылаться на экземпляры системы PPC при описании экземпляров продукта операционных моделей. Если обработка заказа требует дополнительных условий, для описания экземпляров продукта операционных моделей используются дополнительные атрибуты.

Модели результата используются при конфигурировании бизнес-приложений, сфокусированных на соответствующих процессах. Необходимы различные процессы, в зависимости от того, предоставляет ли предприятие услуги, занимается розничной торговлей или производит материальные изделия. Программа SAP R/3 Business Engineer использует информацию о типах продукта на первых двух уровнях конфигурации системы — для выбора бизнес-сценариев и определения вариантов процесса (рис. 83 и 84).

Бизнес-приложения, которые не управляются процессом, не описывают в явном виде модель результата. Информационные сервисные объекты включаются в модель данных как объекты данных, но не определяются явно как продукты функции или процесса.

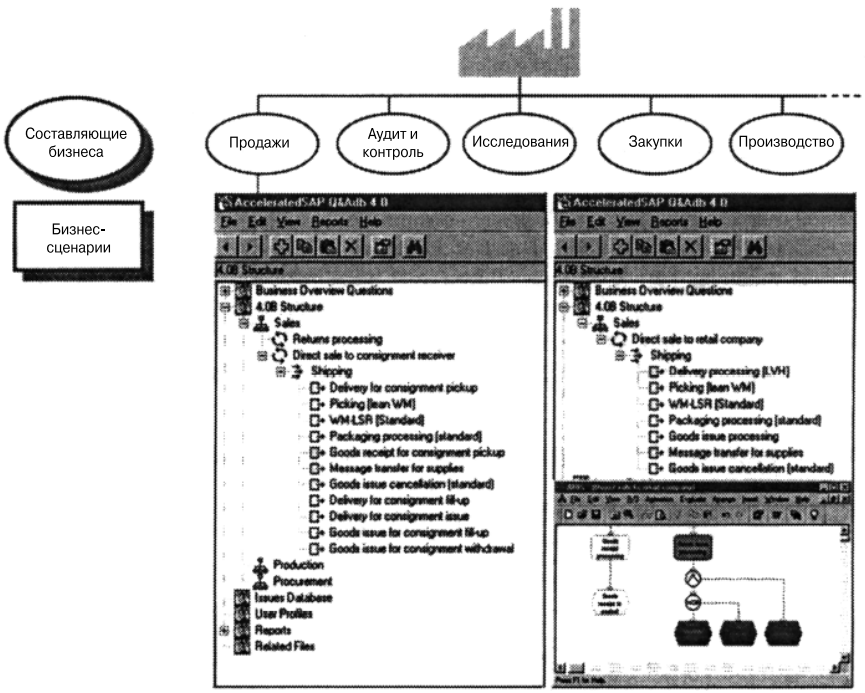


Рис. 83. Выбор бизнес-сценариев
(Schröder, Business Engineer 1997)

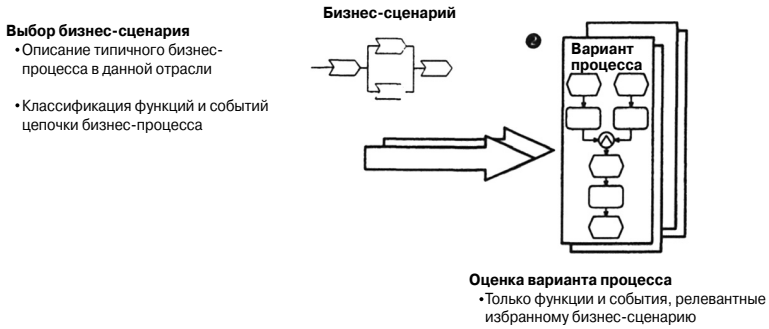


Рис. 84. Определение вариантов процесса
(Schröder, Business Engineer 1997)

A.III Моделирование взаимосвязей между аспектами (управление)

Задача моделирования управления — анализ взаимосвязей аспектов ARIS (функционального, организационного, данных и результата), рассмотренных выше отдельно. Такой анализ необходим для описания модификаций статуса и моделирования динамического поведения системы.

Сначала обсуждаются методы описания взаимосвязей между парами аспектов, а затем анализируются взаимосвязи всех аспектов. При моделировании управления используется знакомая модель жизненного цикла, включающая описание требований, конфигурацию, спецификацию дизайна и описание внедрения.

A.III.1 Взаимосвязи функций и структуры организации

На рис. 85 иллюстрируется место взаимосвязей функционального и структурного аспектов в концепции ARIS. Функциональный и структурный аспекты представлены в виде затененных фигур.

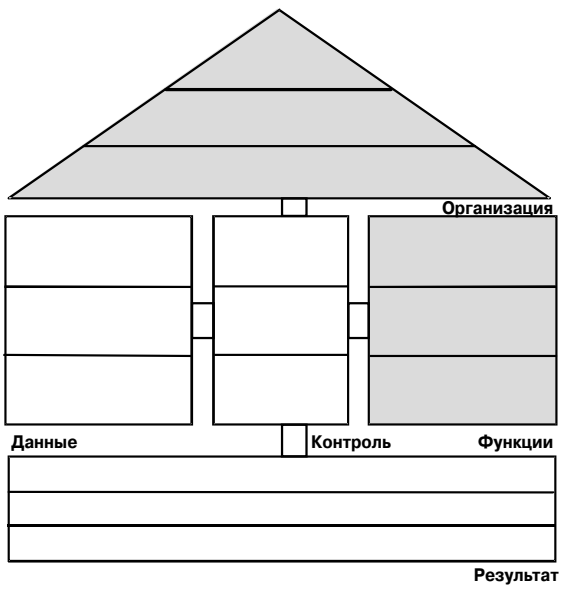


Рис. 85. Взаимосвязи функционального и структурного аспектов в концепции ARIS

На рис. 86 представлена базовая взаимосвязь между бизнес-функцией и организационной единицей. На практике эта взаимосвязь может иметь различные семантические ответвления. Возможны также различные исходные модели и ограничения на распределение функций.

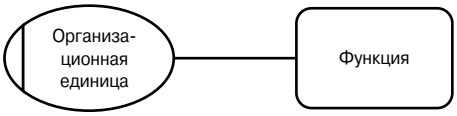


Рис. 86. Базовая взаимосвязь организационной единицы и функции

A.III.1.1 Описание требований

A.III.1.1.1 Диаграммы соответствия функций и организационной структуры

Взаимосвязи функций и структуры организации можно представить в виде диаграмм с различной степенью детализации.

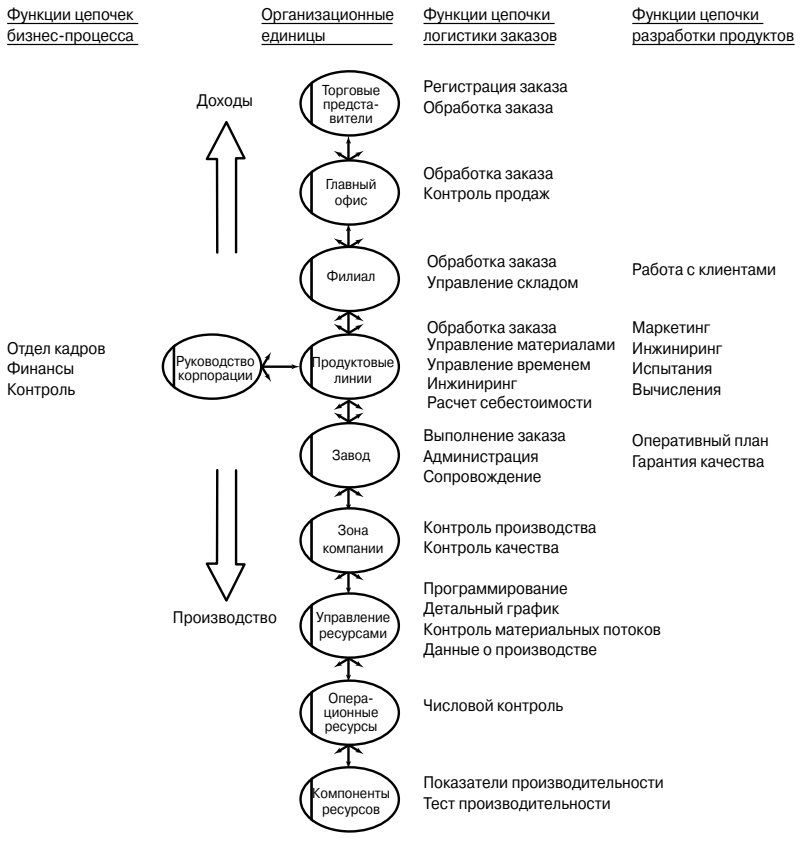


Рис. 87. Модель организационного распределения функций

В моделях распределения функций ключевые функции цепочек процесса поставлены в соответствие определенным организационным единицам. На рис. 87 иллюстрируется распределение функций планового отдела промышленного предприятия (см. рис. 49) на примере цепочек логистики заказов, разработки продуктов и контроля.

На рис. 88 распределение функций представлено в форме матрицы. Пример отображает реальную ситуацию обработки заказов при производстве капитального оборудования. В этом конкретном случае обработка заказов включает около 150 функций, при этом нет разделения на автоматизированные процессы и ручную обработку.

Функции \ Организационные единицы	Менеджмент	Маркетинг	Менеджмент и организация	Менеджмент филиала	Отдел кадров	Финансы и контроль	Продажи	Продажи и распределение	Исследования и развитие	Производство	Закупки	Склад
Анализ рынка	i	r	a	a			i					
Составление производственной программы	r	i	i	a	a	a	i		a	a	a	
Обработка предложений							r					
Обработка заказов							r					
Разработка продуктов	i			a	a	i	i		r	i	i	
Планирование производства			i		i	a	i		i	r	i	a
Закупка материалов											r	i
Управление складом								a				r
Управление производством и контроль						a	a					
Обеспечение качества				a			a	a	i	r	i	r
Отгрузка				i			i	r				
Финансы и контроль	a		a	i	a	r			a	a	a	
Планирование инвестиций	i		r	a	a	i				a	a	
Развитие персонала	i		i	a	r	a				a	a	
Учет запасов	i		a	i		r				a	a	

r — отвечает i — активно участвует a — привлекается

Рис. 88. Распределение функций, представленное в форме матрицы

Если определенная функция относится к нескольким организационным единицам, то степень участия организационной единицы в выполнении данной функции детализируется с помощью информации о том, отвечает ли данное подразделение за функцию, активно участвует в ее выполнении или привлекается в случае необходимости. Другие примеры распределения функций см. в *Martin, Information Engineering II 1990, p. 58.*

Модель организационного распределения функций и матрица распределения показаны в виде связи РАСПРЕДЕЛЕНИЕ ФУНКЦИЙ в метамодели на рис. 89. Это распределение относится к ключевым функциям.

Каждая организационная единица, вовлеченная в бизнес-процесс, определяется связями функций с соответствующими процессами. С другой стороны, каждая функция и цепочка процесса определяется связями с организационными единицами.

Рис. 89. Диаграмма классов для распределения функций

Что касается кардинальностей, то каждая организационная единица вовлечена в выполнение по крайней мере одной функции, а каждая функция относится по меньшей мере к одной организационной единице. С другой стороны, каждая функция может выполняться с участием многих организационных единиц; при этом функции распределяются между многими уровнями планирования внутри цепочки процесса.

Операции с клиентами соответствуют исполнителям и конкретным клиентам (рис. 90). В конкретной операции с клиентами могут участвовать многие исполнители. Конкретный исполнитель может участвовать во многих операциях с клиентами.

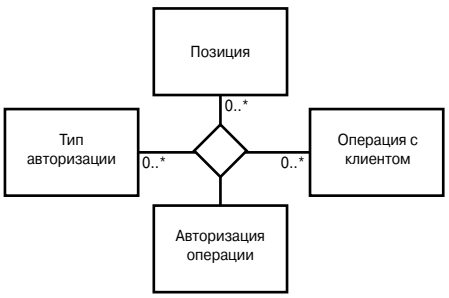


Рис. 90. Авторизация операций с клиентами

A.III.1.1.2 Диаграммы вариантов использования

Диаграммы вариантов использования (use case diagrams) — составная часть языка UML.

Диаграммы вариантов использования описывают, как “актеры” — организационные единицы — связываются с функциями, хотя термин “диаграмма вариантов использования” описывает только часть бизнес-процесса, представляющую собой цельную операцию, т.е. непрерывную во времени и пространстве. Диаграммы вариантов использования помогают одолеть сложные проблемы при моделировании бизнес-процессов. На рис. 91 иллюстрируется пример диаграммы вариантов использования (*см. Oestereich, Objektorientierte Softwareentwicklung 1997, p. 215*).

Диаграммы вариантов использования “обрамляют” определенную ситуацию и ставят ей в соответствие конкретные варианты. Каждая функция, изображенная в виде овала, соответствует определению элементарной функции. Актеры и функции соединяются с помощью линий коммуникаций. Каждый вариант

использования нумеруется. Тесная связь между вариантами, подразумевающая, например, что один вариант противоречит другому или, напротив, фактически включает другой, изображается в виде пунктирной линии (см. на рис. 91 стрелку между размещением заказа и проверкой статуса).

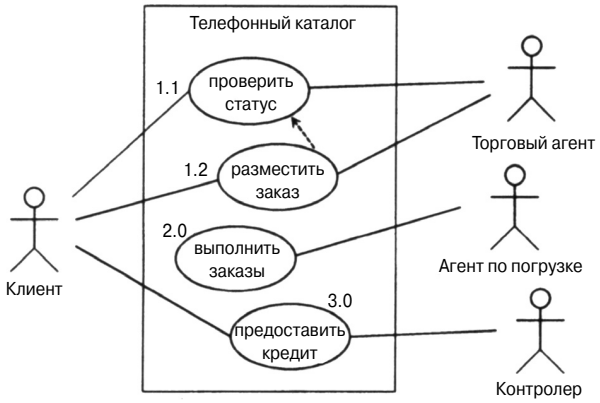


Рис. 91. Диаграмма вариантов использования
(UML Notation Guide 1997, Fig. 33)

Согласно концепции ARIS, диаграммы вариантов использования осуществляют связь между организационной моделью (определяет актеров) и функциональной моделью. На рис. 92 иллюстрируется формирование диаграммы вариантов использования. Диаграммы вариантов использования часто сопровождаются текстами и детальными диаграммами последовательности действий.

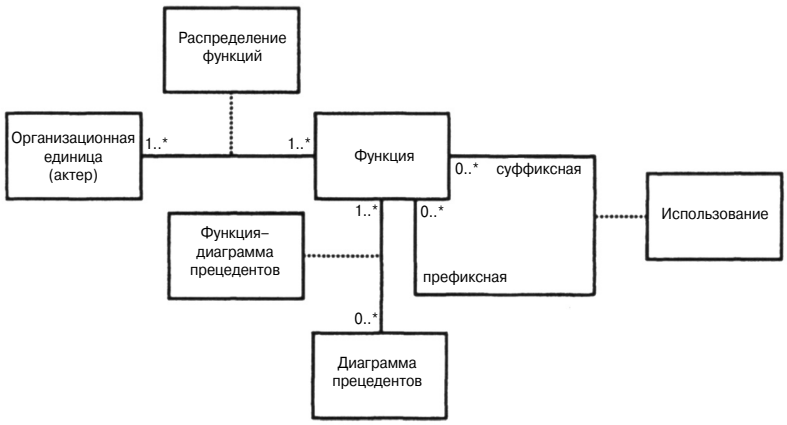


Рис. 92. Метамодель диаграммы вариантов использования

A.III.1.2 Конфигурация

Распределение функций между организационными единицами имеет ключевое значение в конфигурации системы.

При функционально-стоимостном анализе необходимо распределить функции между соответствующими центрами расходов. Это имеет ключевое значение для вычисления норм затрат по функциям.

При планировании ресурсов распределение функций показывает, какая функция в бизнес-процессе использует какой вид ресурсов.

Операционный контроль основан на ассоциациях организационных функций, определяющих, какое место занимает соответствующая функция в операционном процессе.

В конфигурациях макроуровня ассоциации организационных функций определяют степень децентрализации стандартной прикладной системы. Обычно это делается с помощью программ, определяющих, удовлетворяются ли требования системы PPC централизованно на уровне завода или децентрализованно на уровне подразделения. Чтобы обеспечить больше организационной гибкости, ассоциации организационных функций должны иметь свободную конфигурацию.

В конфигурациях микроуровня ассоциации организационных функций контролируют, как пользователи владеют системными функциями, и определяют степень участия каждого рабочего места в процессе. На рис. 93, а–г на примере управления материалами иллюстрируются связи между функциями организационных единиц и цепочками процесса.

В организации, показанной на рис. 93, а и 93, б, получение товаров извне и доступ к складу разделены. Получением товаров извне заняты многие сотрудники отдела получения товаров. На входе в склад скапливается масса товаров, с которыми затем работают сотрудники склада. В таком случае необходимы два различных интерфейса пользователя. Каждый интерфейс обеспечивает выполнение функций сотрудника соответствующего отдела (получения товаров или доступа к складу).

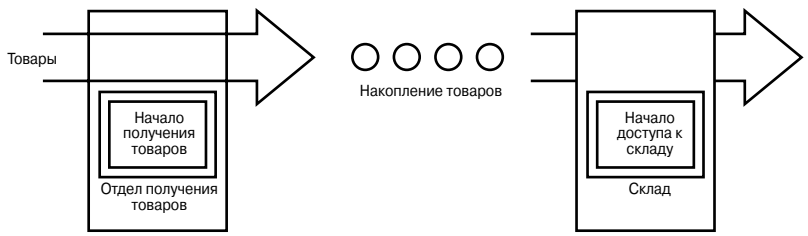


Рис. 93, а. Управление материалами в случае организационного разделения функций

На рис. 93, в и 93, г функции получения товаров и доступа к складу объединены. В этом случае необходим только один интерфейс пользователя. Функциональный процесс в обоих случаях идентичен, за исключением того, что раз-

личное разделение функций в организации ведет к различным процессам, требующим разработки конкретных диалоговых программ.

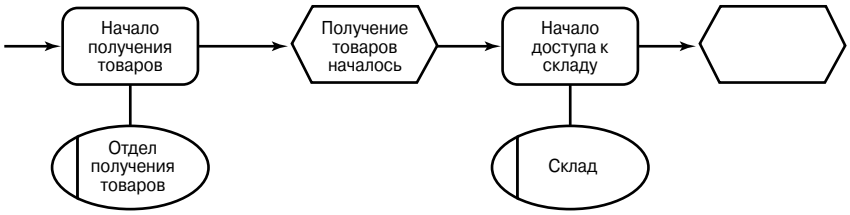


Рис. 93, б. Цепочка процесса, изображенного на рис. 93, а

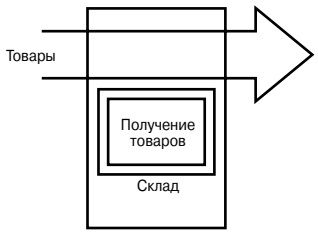


Рис. 93, в. Управление материалами в случае организационного объединения функций

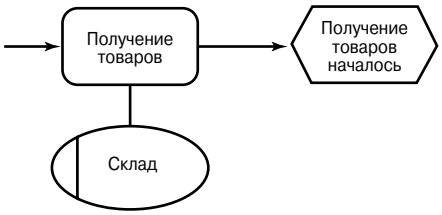


Рис. 93, г. Цепочка процесса, изображенного на рис. 93, в

Конфигурация систем стандартного программного обеспечения включает программирование пользовательских транзакций и экранов в соответствии с распределением функций между подразделениями организации. Это требует большой гибкости стандартных систем, чтобы они могли обеспечить продуктивную работу конкретных пользователей. Поэтому один из ключевых вопросов создания конкретной прикладной системы, ориентированной на пользователя, — это способность адекватно отобразить распределение функций в организации. Такую способность имеет стандартное программное обеспечение, основанное на модели ARIS (*см. IDS, ARIS-Applications 1997*).

Конфигурация прикладных систем включает авторизацию пользовательских транзакций, которая содержится в описании требований. Некоторые типы авторизации разрешают пользователям запускать определенные модули, использовать экраны ввода-вывода или подписываться на рассылки.

На рис. 94 иллюстрируется два пути авторизации. Класс ПРОГРАММНЫЙ ОБЪЕКТ состоит из подклассов МОДУЛЬ, ЭКРАН и ЛИСТИНГ. Если функции авторизации для каждого пользователя содержатся в матрице авторизации, класс ассоциаций АВТОРИЗАЦИЯ связывает классы ТИП АВТОРИЗАЦИИ, ОРГАНИЗАЦИОННАЯ ЕДИНИЦА и ПРОГРАММНЫЙ ОБЪЕКТ. В таком случае пользователи с идентичными параметрами авторизации характеризуются отдельно, что делает администрирование громоздким и избыточным.

Альтернативный путь авторизации позволяет уменьшить сложность администрирования. Определенные параметры авторизации сначала формируют класс ПАСПОРТ с помощью класса ассоциаций АВТОРИЗАЦИЯ–ПАСПОРТ. Класс ПАСПОРТ связывается с группами пользователей или индивидуальными пользователями с помощью класса ассоциаций ПОЛЬЗОВАТЕЛЬ–ПАСПОРТ. Такая непрямая авторизация существенно уменьшает избыточность информации.

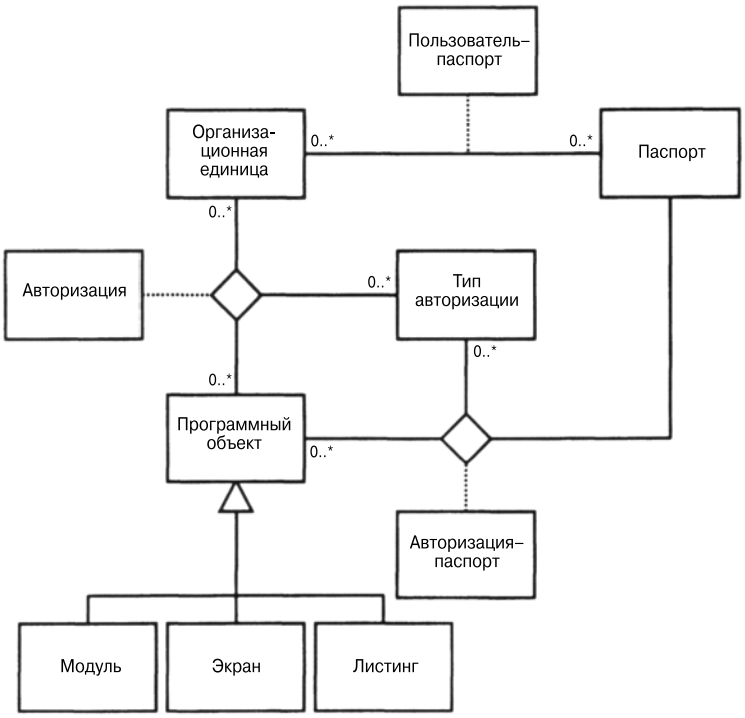


Рис. 94. Два пути авторизации

A.III.1.3 Спецификация дизайна

При спецификации дизайна программные объекты (модули, бизнес-объекты, пользовательские транзакции) ставятся в соответствие конкретным узлам компьютерной сети. Они осуществляют физический доступ к данным с помощью либо стандартных методов (удаленный вызов процедур, CORBA, DCOM и т.п.),

либо собственных методов, таких как RFC (SAP) или ALE (SAP). На рис. 95 класс МЕТОД ДОСТУПА включает некоторые из этих методов.

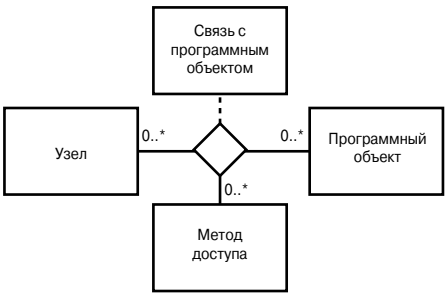


Рис. 95. Методы физического доступа

A.III.2 Взаимосвязи функций и данных

На рис. 96 иллюстрируется место взаимосвязей функций и данных в концепции ARIS. Рассматриваемые аспекты представлены в виде затененных фигур.

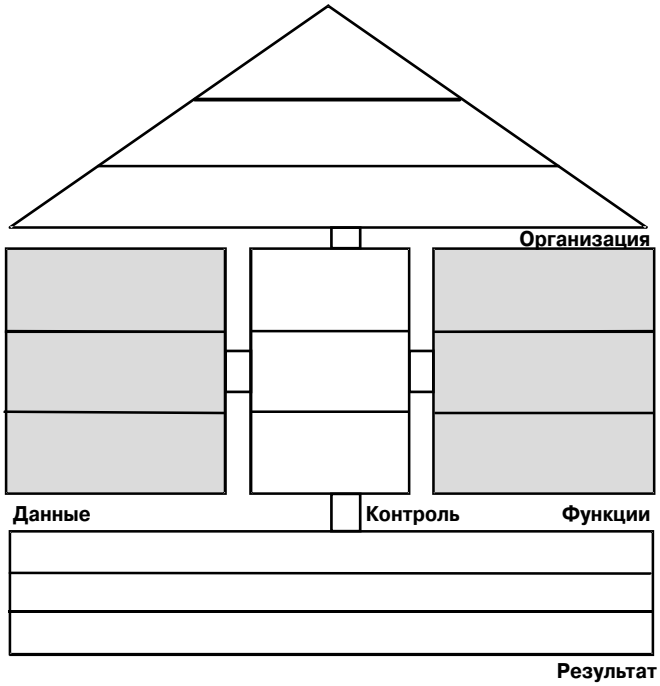


Рис. 96. Взаимосвязи функций и данных в концепции ARIS

В рамках общей модели бизнес-процесса ARIS существует два вида взаимосвязей функций и данных:

- функции преобразуют входные данные в выходные данные;
- функции создают события (модификации статуса данных); сообщения об изменениях статуса приводят в действие следующие функции.

Взаимосвязи первого вида привели к созданию многих методов проектирования, где данные и функции тесно взаимосвязаны. К таким методам относятся, например, структурный анализ и проектирование (Structured Analysis and Design Technique; SADT), диаграммы Де Марко и объектно-ориентированные диаграммы классов.

Событийное управление функциями — ключевой вопрос в описании поведения системы.

A.III.2.1 Моделирование описания требований

A.III.2.1.1 Соответствие функций и данных

A.III.2.1.1.1 Объектно-ориентированные диаграммы классов

Диаграммы классов описывают системную структуру объектно-ориентированной бизнес-модели. Классы характеризуются определением, атрибутами и применяемыми методами. Понятие "метод", используемое в объектно-ориентированном анализе, можно уравнять с понятием "функция". Учитывая, что классы чаще всего представляют собой классы данных, они олицетворяют связь между аспектами данных и функций. Диаграммы классов описывают ассоциации и ограничения между классами.

Поведение системы описывается с помощью динамических моделей, характеризующих любой обмен сообщениями между объектами или взаимодействие между методами одного объекта. Эта тема подробнее анализируется в разделе, посвященном управлению событиями и сообщениями.

Некоторые свойства объектно-ориентированных классов были проанализированы при описании аспекта данных. Остановимся кратко на характеристиках создаваемых объектно-ориентированных классов.

Объектно-ориентированный анализ (ООА) не относится к стандартизованным методам. Несколько авторов разработали родственные подходы (*см. Coad/Yourdon, Object-Oriented Analysis 1991; Booch, Object-Oriented Design 1991; Rumbaugh et al., Object-Oriented Modeling and Design 1991; Jacobsen, Object-Oriented Software Engineering 1996*). Сравнить эти подходы нелегко, поскольку авторы используют различные графические символы.

Язык UML, разработанный Г. Бучем, Д. Рамбо и А. Джекобсоном, позволяет упростить изложение различных подходов ООА, поэтому будем использовать терминологию UML.

Объекты, каждый из которых имеет индивидуальные особенности и идентификационный код, описываются с помощью набора атрибутов (характеристик). Функции (методы), которые применяются к объекту, определяют поведение объекта. Объекты представляют собой экземпляры классов и изображаются в виде прямоугольников (рис. 97, *а*).

Объекты с идентичными атрибутами, функциональностью и семантикой, объединяются в классы объектов или регулярные классы. Например, множество клиентов формирует класс КЛИЕНТ (рис. 97, *б*).

Атрибуты и методы определяют характеристики и поведение экземпляров классов, т.е объектов. Поскольку атрибуты и методы формируют единое целое, классы реализуют принцип инкапсуляции. Помимо атрибутов и методов, определяющих объекты, используются атрибуты и методы, характеризующие классы, а не объекты. Примеры атрибутов класса: "Количество клиентов" и "Добавление нового клиента".

Одно из существенных свойств объектно-ориентированного подхода — это наследование атрибутов. В результате классы имеют доступ к характеристикам

(атрибутам) и поведению (методам) других классов. Наследуемые атрибуты могут быть переопределены принимающим классом.

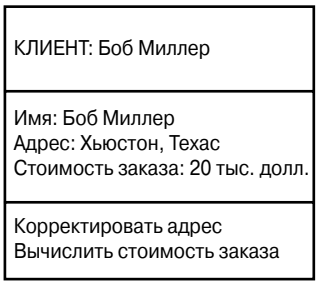


Рис. 97, а. Объект

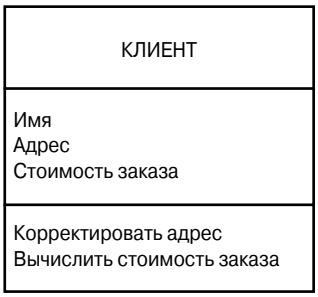


Рис. 97, б. Класс объектов

Наследование атрибутов имеет место в пределах иерархии классов между доминирующими и подчиненными классами. Это демонстрируют операции обобщения и специализации в моделях данных.

Класс может также унаследовать свойства нескольких доминирующих классов (множественное наследование); при этом результирующая диаграмма классов формирует сеть.

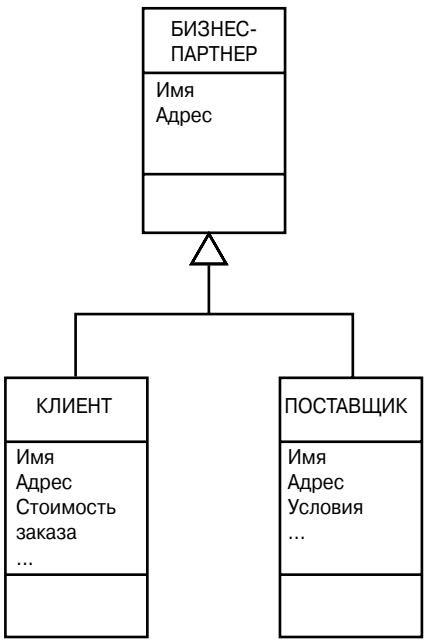


Рис. 98. Наследование

В дополнение к иерархическим связям между классами существуют также связи (ассоциации) между объектами классов одного уровня или между объектами одного и того же класса. Ассоциации эквивалентны связям в модели

“сущность–связь”, хотя здесь они иллюстрируются с помощью только одной линии (рис. 99,а). Иллюстрации следует читать слева направо. Ассоциации характеризуются кардинальностями.

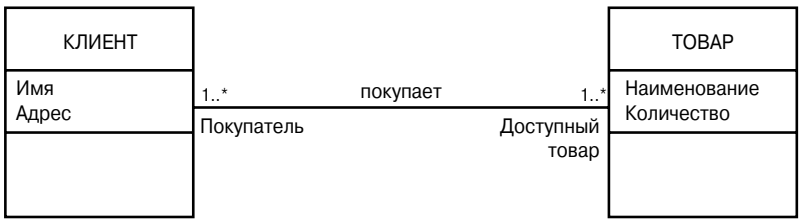


Рис. 99, а. Ассоциация

На каждом конце ассоциации размещается ролевое имя (покупатель, доступный товар и т.п.). Если ассоциации содержат атрибуты, они изображаются как классы (такие как класс ПРОЦЕСС ПОКУПКИ на рис. 99, *б*).

Агрегация — специальный вид ассоциации, описывающий связь “часть и целое” между объектами двух различных классов. На рис. 100 это утверждение иллюстрируется классами ЗАКАЗ и ПУНКТ ЗАКАЗА.

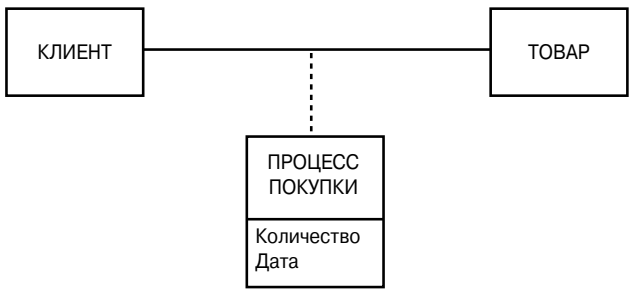


Рис. 99, б. Класс ассоциаций

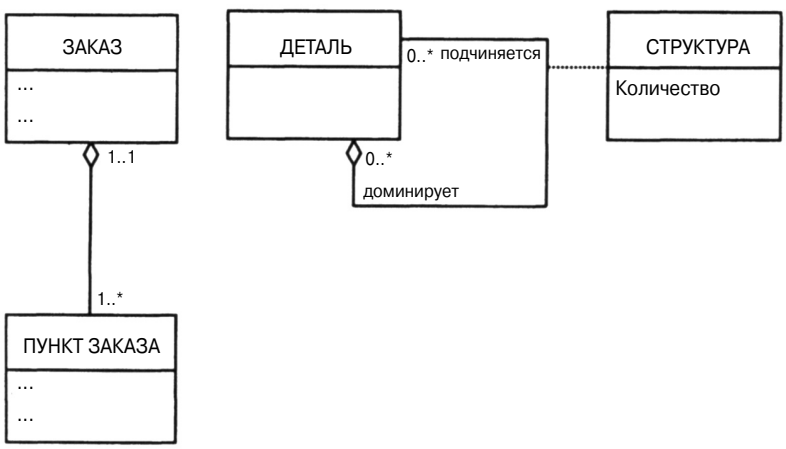


Рис. 100. Агрегация (направленная ассоциация)

Ролевые имена применяются и к агрегациям. Если агрегация имеет атрибуты, это ведет к созданию класса, как показано на рис. 100 на примере класса СТРУКТУРА в случае агрегации данных спецификации материалов.

В отличие от ассоциации, агрегация подчеркивает иерархию между классами, поэтому она также называется направленной ассоциацией.

Понятия класса, наследования, ассоциации и агрегации определяют структурные свойства объектно-ориентированного анализа.

На рис. 101 иллюстрируется первоначальная метамодель структуры объектно-ориентированного анализа. Метамодель фокусируется на понятии КЛАСС. Поскольку классы обычно представляют собой классы данных, между понятиями КЛАСС и ИНФОРМАЦИОННЫЙ ОБЪЕКТ существует ассоциация 1:1. Атрибуты информационных объектов также относятся к классам. Методы, применяемые к классам, описываются в функциональном аспекте.

Размещение атрибутов ссылается на высший класс в иерархии наследования. Отношения наследования иллюстрируются с помощью класса ассоциаций НАСЛЕДОВАНИЕ.

Понятия “независимая ассоциация” и “направленная ассоциация” формируют класс ТИП АССОЦИАЦИИ. Некоторые ассоциации имеют множество значений. Классы ассоциаций регистрируются как специализированные классы.

Предложение UML детально описывает метамодели с помощью средств языка UML. В общем случае это касается диаграмм классов (обычно без методов), ассоциаций и групп объектов модели.

В ключевых концепциях метаописаний UML также содержится информация, которая относится ко второму метауровню в концепции ARIS (*см. Scheer, ARIS — Business Process Frameworks 1998, pp. 120–125*). Метамодель на рис. 102 иллюстрирует пример определения класса в UML.

Метамодели ARIS имеют следующие отличия от метамоделей UML:

1. Метамодели ARIS отлично структурированы согласно концепции ARIS и учитывают фазы жизненного цикла модели.
2. Метамодели ARIS включают усовершенствованные методы и не ограничиваются иллюстрированием объектно-ориентированных методов.
3. Метамодели ARIS включают бизнес-ориентированные приложения (например, стратегическое планирование).

Таким образом, метамодели ARIS представляют собой усовершенствованные метамодели UML, сфокусированные на процессе внедрения.

A.III.2.1.1.2 Диаграмма окружения функции

Классы данных в диаграммах классов явно окружают соответствующие функции. Если функция моделируется независимо от данных, окружение функции формирует ассоциацию *:*, т.е. функция относится ко многим объектам данных. Такой вид моделирования позволяет создавать функциональные объекты, к которым применимы также принципы наследования. Проектирование неизбыточных объектно-ориентированных моделей возможно, если объек-

ты данных *и* функциональные объекты создаются независимо, а соответствие функций и данных устанавливается свободно. Более того, при независимом проектировании объектов данных и функциональных объектов устанавливается связь между аспектно-ориентированным и объектно-ориентированным подходами к моделированию.

На рис. 103 иллюстрируется метамодель диаграммы окружения функции

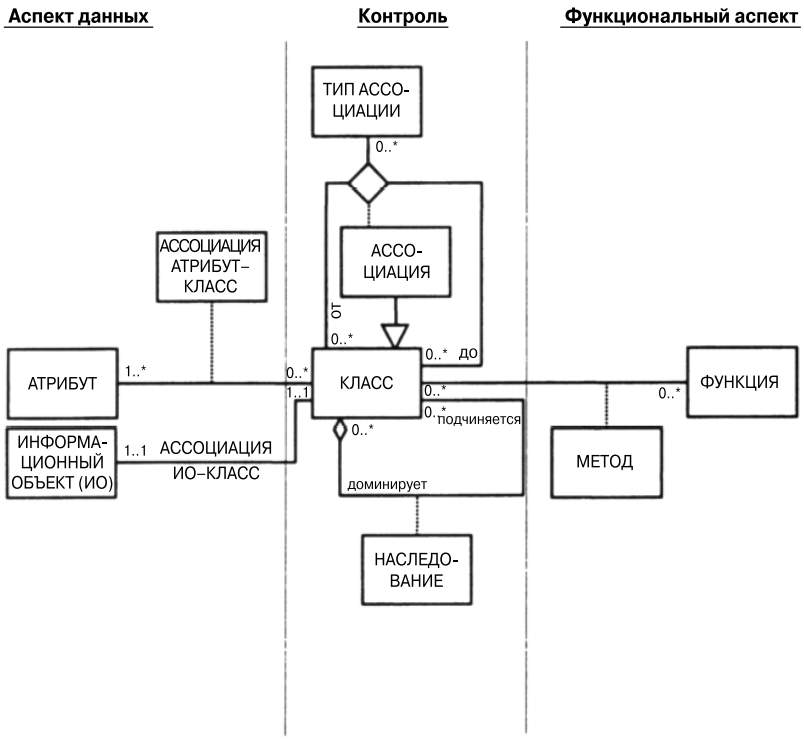


Рис. 101. Метамодель структуры объектно-ориентированного анализа

A.III.2.1.1.3 Поток данных

Объектно-ориентированные диаграммы классов фокусируются на классах. Применяемые методы отнесены к соответствующим классам данных.

В диаграммах потоков данных, напротив, ключевые элементы — это функции, преобразующие входные данные в выходные данные. В традиционном функционально-ориентированном программировании диаграммы потоков данных различной степени детализации играют главную роль при проектировании описания требований.

На рис. 104 обработка клиентского запроса иллюстрируется с помощью диаграммы Де Марко потока данных. Информационные объекты очерчены двумя горизонтальными линиями. Наименования ключевых атрибутов, используемых функциями, сопровождают стрелки. Этот подход позволяет включать системы реального времени (*см. Ward/Mellor, Real-Time Systems 1985*).

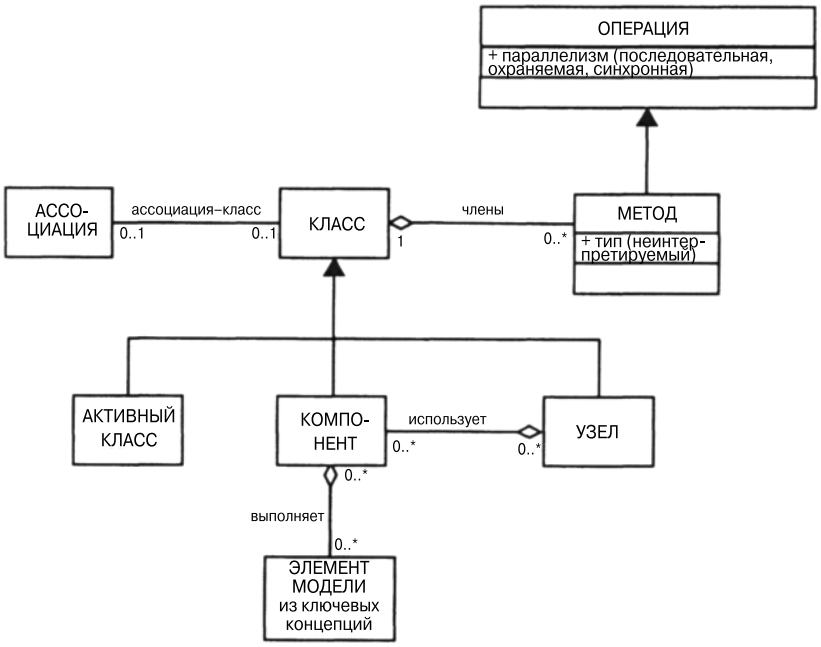


Рис. 102. Метамодель определения класса в UML
(UML Semantics 1997, pp. 35, 44)

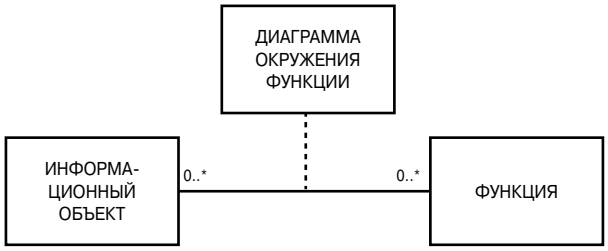


Рис. 103. Метамодель диаграммы окружения функции

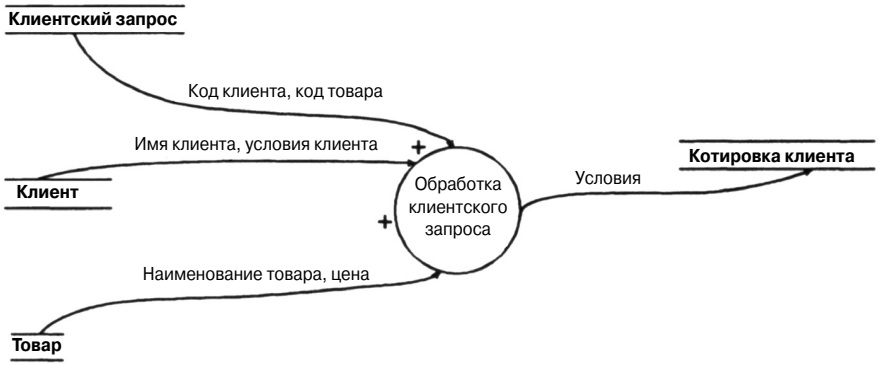


Рис. 104. Диаграмма Де Марко потока данных для обработки клиентского запроса

На рис. 105 иллюстрируется моделирование потока данных с помощью класса ассоциаций ОПЕРАЦИЯ. Класс ОПЕРАЦИЯ позволяет детальнее моделировать операции, которые бизнес-функция выполняет с атрибутами. Эти операции включают:

- создание элементов данных;
- удаление элементов данных;
- корректировку элементов данных;
- только чтение элементов данных.

Следует выбрать наивысший возможный уровень операции для каждого элемента данных. Такое распределение часто демонстрируется также в виде таблицы (*см. Martin, Information Engineering II 1990, p. 272*).

Класс ТИП ОПЕРАЦИИ включает каждый отдельный тип операции, акласс ассоциаций ОПЕРАЦИЯ связывает классы ТИП ОПЕРАЦИИ, ФУНКЦИЯ и АССОЦИАЦИЯ ОБЩИХ АТРИБУТОВ.

Диаграмма Де Марко показывает, что можно установить логическую взаимосвязь между операциями, которые выполняются функцией. Эта возможность иллюстрируется с помощью класса ассоциаций СВЯЗЬ между операциями. ТИП СВЯЗИ подразумевает операции алгебры Буля. Однако следует заметить, что невозможно установить любую возможную логическую взаимосвязь между входными и выходными элементами данных.

Некоторые методы моделирования потоков данных включают ориентацию на данные наряду с функциональной ориентацией. Например, в методе SADT оба подхода обозначены терминами “функциональный блок” (activity box) и “блок данных” (data box). Функциональный блок устанавливает взаимосвязи с входными, управляющими и выходными данными, а процессоры описывают правила преобразования, т.е. операции. Блок данных рассматривается как информационный объект; с его помощью определяют, какие функции создают ИО и какие функции его используют.

A.III.2.1.1.4 Входные и выходные экраны

Функции используют входные и выходные экраны. Например, бизнес-функция “Создание клиентского заказа” использует много экранов, предназначенных для просмотра или ввода данных. С другой стороны, определенные экраны используются многими функциями. Поэтому класс ЭКРАН связан с классом ФУНКЦИЯ ассоциацией *:* (см. рис. 105).

Экраны предназначены для ввода, корректировки или удаления свойств объекта данных. Экраны приложения или функции представляют входные данные или выходные отчеты.

С помощью моделей экранов функции конфигурируются с объектами данных, представленными на экранах. Например, когда функции ввода данных связываются с типом сущности КЛИЕНТ, они предназначены для ввода данных о клиенте. А когда функции ввода данных связаны с типом сущности ПАЦИЕНТ, они подходят для ввода данных о пациенте. Редактирование функций

можно усовершенствовать путем изменения содержания экрана (добавления или удаления атрибутов).

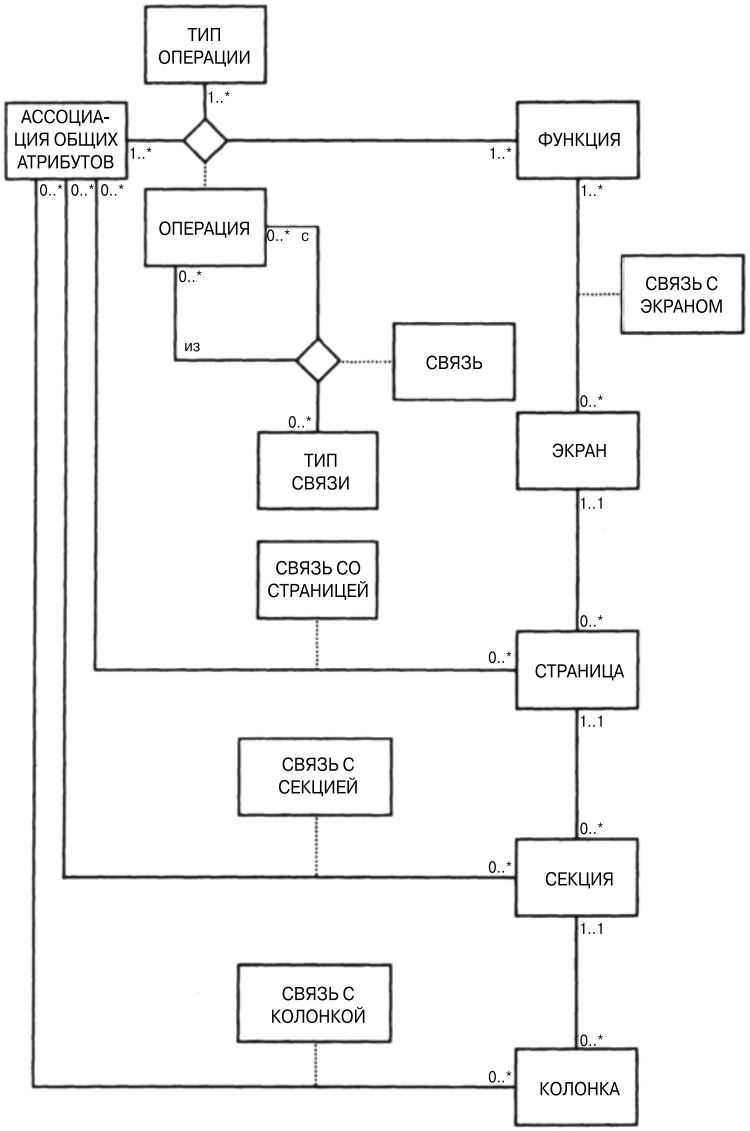


Рис. 105. Моделирование потока данных с помощью класса ассоциаций ОПЕРАЦИЯ

На рис. 106, *а* иллюстрируется модель экрана, включающая атрибуты классов КЛИЕНТ и АДРЕС. Взаимосвязи этих классов показаны на примере обработки учетной записи клиента (рис. 106, *в*). В данном примере учетная запись клиента представляет собой очень сложный объект данных. Правая сторона

рис. 106, а изображает происхождение данных из модели данных. Метамодель на рис. 105 дополняется этими взаимосвязями.

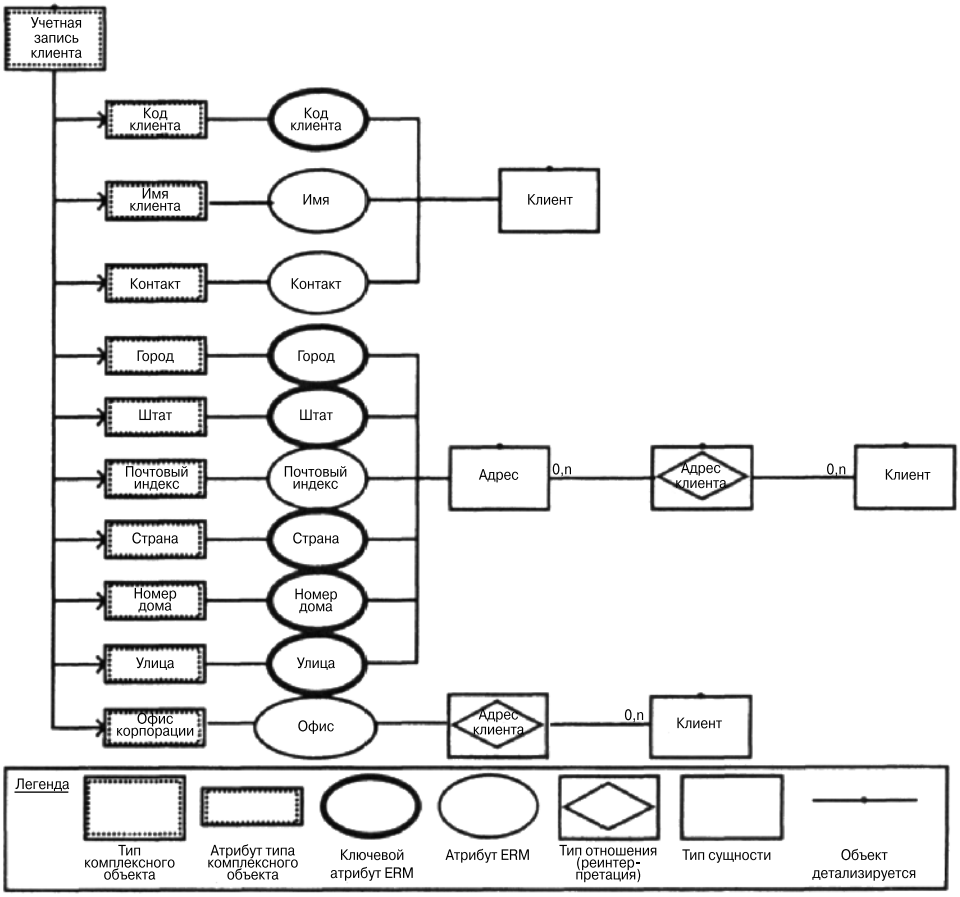


Рис. 106, а. Структура и атрибуты сложного объекта данных "Учетная запись клиента"

Экраны спроектированы с соблюдением иерархии и состоят из страниц, секций и колонок (рис. 106, б).

Подробная модель экрана позволяет автоматически создать экран, показанный на рис. 106, г (*см. IDS, ARIS-Framework 1997*).

Более того, на рис. 106, б модель рис. 106, а дополнена контактом. Повторяющаяся группа значений адреса представлена в виде таблицы внутри экрана.

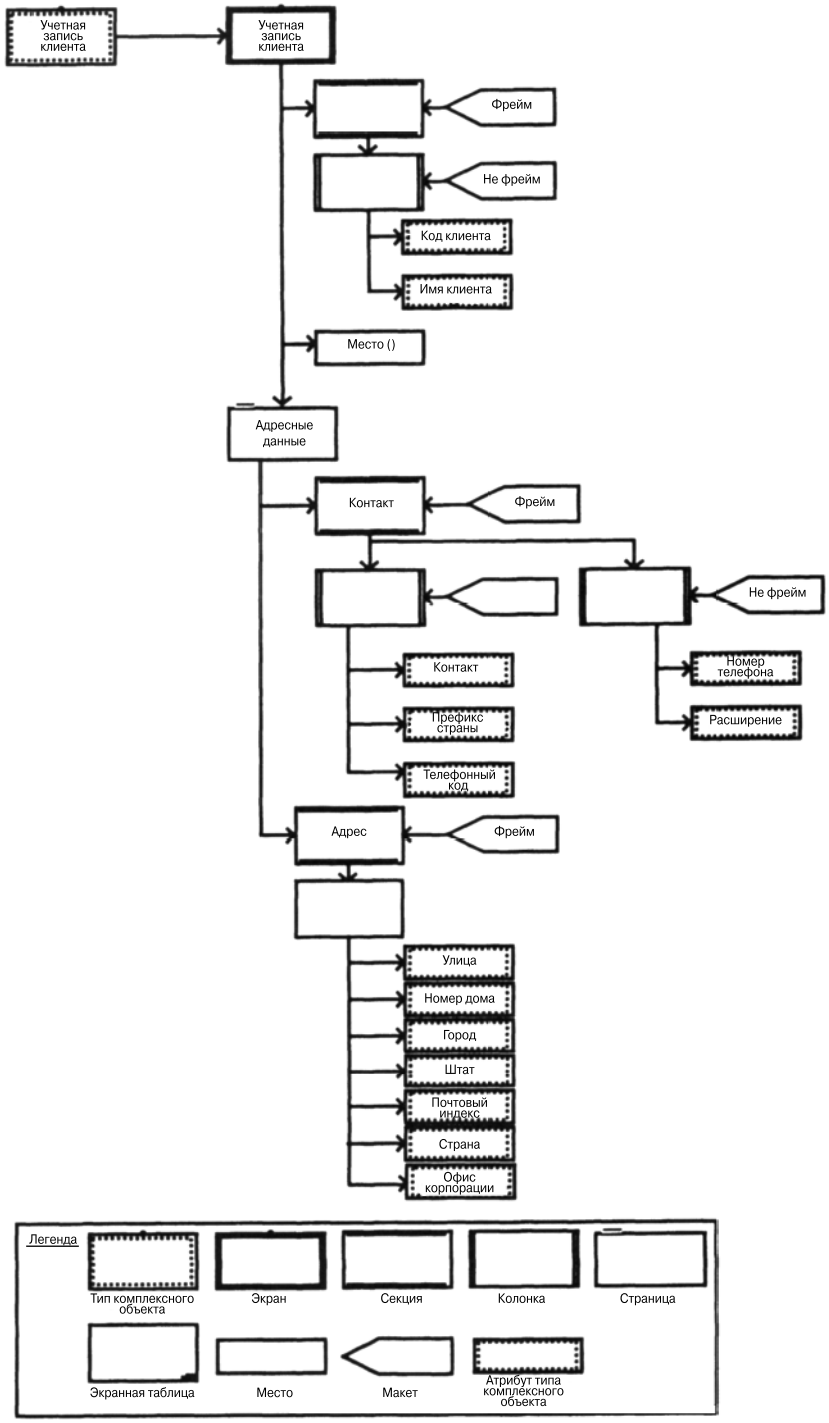


Рис. 106, б. Пример модели экрана

Рис. 106, в. Модель данных для учетной записи клиента

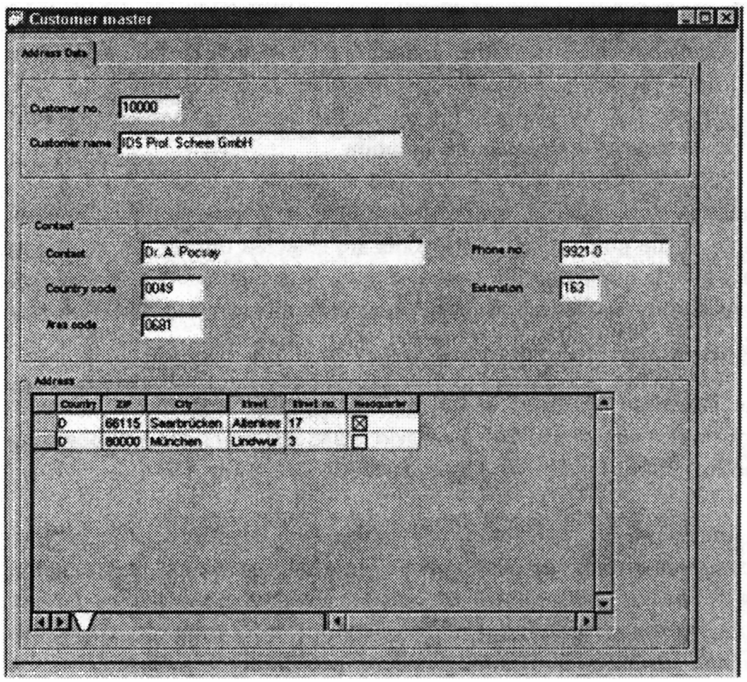


Рис. 106, г. Представление данных на экране

A.III.2.1.2 Управление событиями и сообщениями

Как главный элемент событийно-управляемой цепочки процесса, поток управления событиями описывает динамику бизнес-процесса. В объектно-ориентированных концепциях внутреннее поведение объекта описывается диаграммами состояния, что также включает управление событиями.

События инициируют сообщения, передавая многим адресатам информацию о том, что событие произошло. Эти сообщения дополняются другой информацией и, в свою очередь, инициируют новые функции.

Обмен сообщениями описывает также динамическое поведение объектов в рамках объектно-ориентированного подхода.

Учитывая успехи в моделировании бизес-процессов и важность объектно-ориентированного подхода, введем понятия, которые помогут нам увязать процессно-ориентированный и объектно-ориентированный подходы к моделированию.

A.III.2.1.2.1 Правило “событие-условие-действие”

В компьютерной науке правило “событие–условие–действие” (Event Condition Action — ECA) используется для регулирования потоков управления (*см. Dittrich/Gatziu, Aktive Datenbanksysteme 1996, p. 10*), хотя это правило характеризует также бизнес-процессы (*см. Herbst/Knolmayer, Geschäftsregeln 1995*).

События включают факты (*что*), которые случаются в определенный момент времени (*когда*). *Условия* определяют обстоятельства соответствующего события. *Действия* описывают реакцию на происходящее событие.

В ARIS-моделях бизнес-процессов события создаются с помощью обрабатывающих функций или актеров вне модели. В процессе моделирования отбираются релевантные события, поэтому в модель включаются только события, оказывающие влияние на бизнес-процесс. Следовательно, условия включаются в определение события, и правило ECA становится правилом EA (“событие-действие”). Вместо события “Стоимость заказа известна” и проверки условия “Стоимость заказа больше 5 тыс. долл.” (сценарий a на рис. 107) рассматриваются два возможных релевантных события (сценарий б на рис. 107).

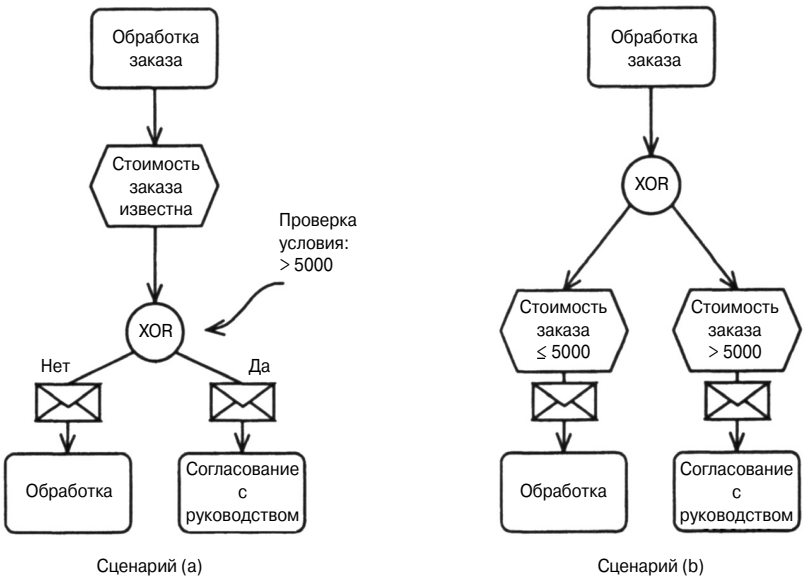


Рис. 107. Различные способы моделирования событий

Действия осуществляются с помощью функций. Стрелки иллюстрируют передачу сообщений о событии (используется символ почтового конверта) и активизацию соответствующих функций. Сообщения помещаются в очередь для обработки. Сообщения могут содержать дополнительные атрибуты для передачи обрабатывающей функции специальной информации.

События могут связываться друг с другом различными сложными способами. Например, перед запуском функции может быть необходимо осуществление нескольких событий; иногда последовательность событий имеет значение. Эти составные события описываются с помощью таких операций алгебры событий, как объединение, пересечение, дополнение и отрицание (*см. Dittrich/Gatziu, Aktive Datenbanksysteme 1996, p. 26*).

A.III.2.1.2.2 Событийно-управляемые процессные цепочки

Метод событийно-управляемых процессных цепочек (Event-Driven Process Chain — EPC) был разработан в Институте информационных систем в г. Саарбрюкен, Германия, совместно с SAP AG (*см. Keller/Nüttgens/Scheer, Semantische Prozeßmodellierung 1992*). Это ключевой компонент концепций бизнес-моделирования SAP R/3.

В основе метода EPC лежат концепции стохастических сетей и сетей Петри. Упрощенные варианты не рассматривают условия и сообщения и включают только модели "событие-действие".

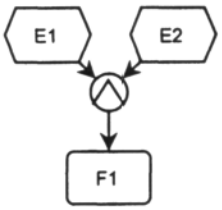
Событие может повлечь за собой выполнение многих функций. С другой стороны, иногда многие функции должны быть выполнены, прежде чем событие состоится. Логические взаимосвязи иллюстрируются с помощью символов "и" ($\wedge$), "включающее или" ($\vee$) и "исключающее или" (XOR). На рис. 108 иллюстрируются некоторые типичные примеры взаимосвязей событий.

В случае более сложных логических взаимосвязей между группами функций (например, завершенными функциями и функциями, выполняемыми в текущий момент) используются таблицы решений для предшествующих и последующих функций.

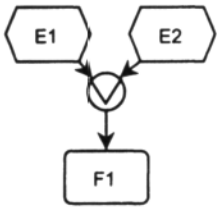
В информационных системах события представлены обновленными данными. Создание клиентского заказа или исполнение заказа на отгрузку — два типичных события, которые обеспечивают модификацию статуса.

Следовательно, на рис. 110 класс СОБЫТИЕ показан как подкласс класса ИНФОРМАЦИОННЫЙ ОБЪЕКТ. Логические взаимосвязи между событиями моделируются с помощью ассоциации между классами ЛОГИЧЕСКИЙ ТИП ВЗАИМОСВЯЗИ и СОБЫТИЕ.

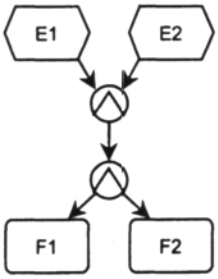
Различные бизнес-приложения обеспечивают возможность конфигурирования конкретных программных решений с помощью моделей бизнес-процессов. Например, на рис. 109 иллюстрируется фрагмент модели бизнес-процесса в методологии BAAN (*см. Baan, Dynamic Enterprise Modeling 1996*). Ниже будет обсуждаться сеть Петри, которая основана на модели, показанной на рис. 110.



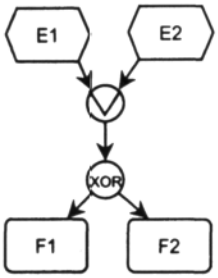
a: Когда происходит событие E1 и событие E2, инициируется функция F1



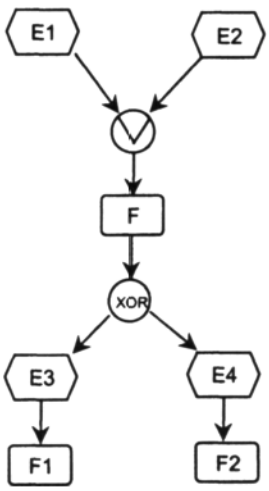
b: Когда происходит событие E1 или событие E2, инициируется функция F1



c: Когда происходит событие E1 и событие E2, инициируются функции F1 и F2



d: Когда происходит событие E1 или событие E2, инициируется функция F1 или функция F2



e: **Когда происходит** событие E1 или событие E2, инициируется функция решений F, чтобы определить, произойдет ли событие E3 или событие E4

Рис. 108. Взаимосвязи событий в событийно-управляемой цепочке (Scheer, Business Process Engineering 1994, pp. 47)

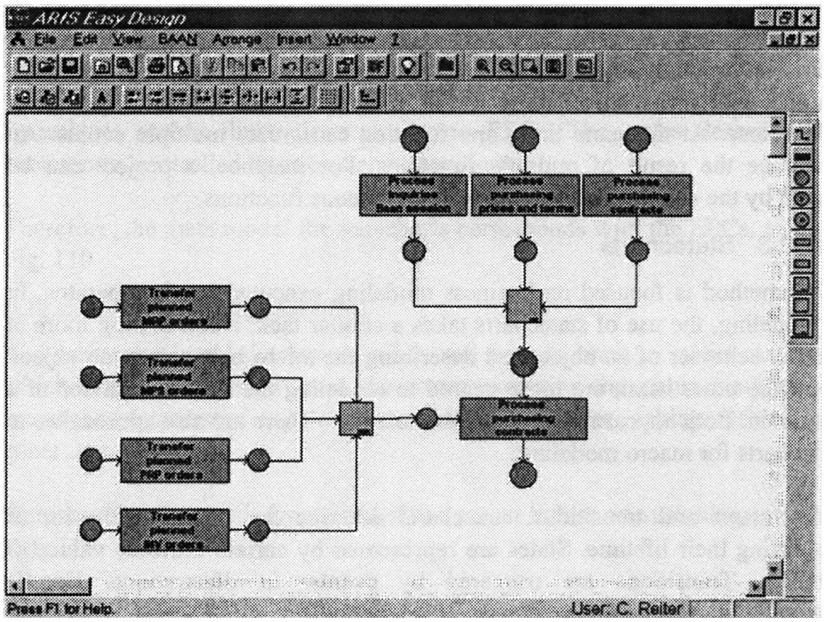


Рис. 109. Модель бизнес-процесса в методологии BAAN
(IDS Prof. Scheer GmbH)

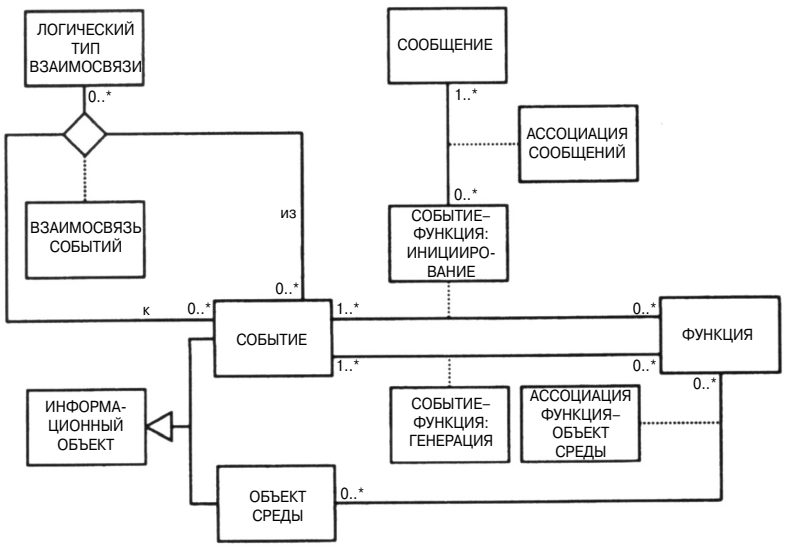


Рис. 110. Метамодель управления событиями и сообщениями в методологии EPC

Прямые взаимосвязи между событием и функцией, т.е. взаимосвязи, не включенные явно в управление сообщениями, представлены с помощью ассоциаций СОБЫТИЕ–ФУНКЦИЯ: ИНИЦИИРОВАНИЕ и СОБЫТИЕ–ФУНКЦИЯ: ПОРОЖДЕНИЕ. Функции могут вызываться одним или несколькими собы-

тиями. В то же время, одна функция может создавать много событий. Событие может быть результатом выполнения многих функций.

A.III.2.1.2.3 Диаграммы состояний

Событийно-управляемые процессные цепочки больше подходят для моделирования бизнес-процесса на макроуровне. В объектном моделировании использование диаграмм состояний позволяет уделить больше внимания внутреннему поведению объекта и описать поведение объекта на микроуровне. Однако два подхода имеют много общего. Диаграммы состояний используются также и при макромоделировании.

Отображая состояния и преобразования, диаграммы состояний описывают внутреннее поведение объектов на протяжении их жизни. Состояния представлены определенными значениями атрибутов объекта. Преобразования вызываются событиями. В иллюстрациях используется номенклатура Д. Харела (*см. Harel, Statecharts 1987, pp. 231–274; Harel, On Visual Formalism 1988, pp. 514–530*), на которую ссылается и Д. Рамбо (*см. Rumbaugh et al., Object-Oriented Modeling and Design 1991*).

На рис. 111 иллюстрируется базовая структура диаграммы состояний, относящейся к объекту.

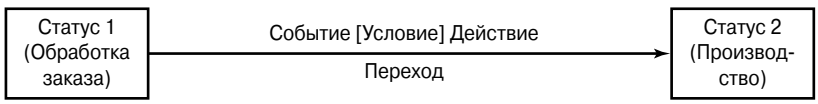


Рис. 111. Диаграмма состояний

На протяжении определенного статуса выполняются действия, такие как “Обработка заказа”. Событие “Заказ обработан” вызывает переход к статусу “Производство”. С событием может быть связано условие, например, “*Успешно* ли обработан заказ?”. Это условие показано в квадратных скобках.

Диаграммы состояний подчиняются правилу ECA. Когда это правило строго применимо, теоретические проверки могут обнаружить любое несоответствие в процессе. Это одно из преимуществ диаграмм состояний. В моделях EPC эвристические методы симуляции помогают достичь подобных результатов.

Диаграммы состояний позволяют создавать модели, подобные EPC. Состояния описывают выполнение функций, управляемых событиями и переходами.

Следовательно, метамодель для диаграмм состояний соответствует метамодели EPC, показанной на рис. 110.

A.III.2.1.2.4 Управление сообщениями

Управление сообщениями предназначено для того, чтобы определить, когда и как реагировать на изменения статуса, сопровождающиеся сообщениями. Сообщение представляет собой обращение отправителя к получателю с требованием генерировать выходной результат.

Диаграммы состояний отображают действия, выполняемые до перехода к новому статусу.

При моделировании простых событийно-управляемых цепочек отдельные сообщения не имеют имен. Они “включаются” в стрелки, соединяющие события и последующие функции.

Если сообщения имеют собственные свойства (такие как атрибуты или инструкции), их имеет смысл моделировать более строго. На рис. 112 представлен пример событийно-управляемой цепочки обработки заказа (*см. Scheer, ARIS-Business Process Frameworks 1998*). Сообщения, изображенные в виде почтовых конвертов, имеют различные свойства. Метамодель на рис. 110 может быть соответственно расширена. Класс СООБЩЕНИЕ связан с классом ассоциаций СОБЫТИЕ–ФУНКЦИЯ: ИНИЦИИРОВАНИЕ. Одно и то же сообщение можно направить различным функциям, но только одно сообщение относится к “инициирующей” стрелке.

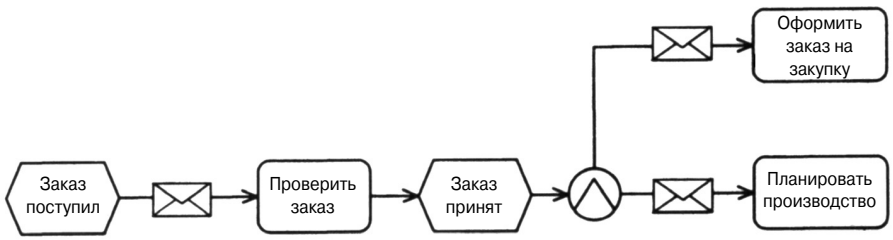


Рис. 112. Пример моделирования сообщений

В объектно-ориентированном моделировании управление сообщениями имеет особую важность, так как поведение системы управляется с помощью потока сообщений между объектами. Потоки сообщений вызывают выполнение задач. Помимо характеристик отправителя и получателя, сообщения содержат наименование функции и необходимые параметры. Отправитель требует от получателя выполнить функцию с заданными входными данными и вернуть выходной результат. Этот процесс иллюстрируется на рис. 113.

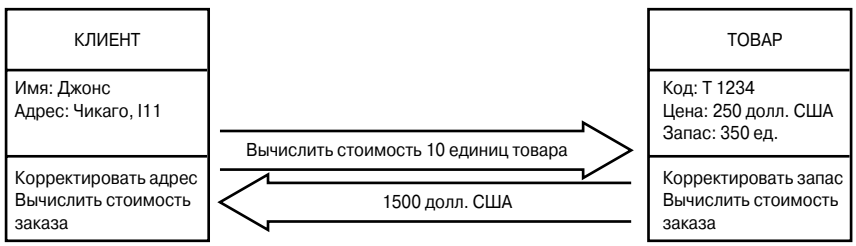


Рис. 113. Обмен сообщениями между объектами

Объект “Клиент Джонс” посылает объекту “Товар 1234” сообщение с требованием вычислить стоимость заказа на 10 единиц указанного товара. Ответ направляется клиенту. Строго говоря, сообщение инициируется функцией “вычислить стоимость заказа”. При этом проверяется, включает ли объект-получатель требуемую функцию. Если не включает, идет поиск по уровням иерархии наследования, пока функция не будет найдена.

Одно из важных свойств объектно-ориентированного подхода — это полиморфизм, когда определенное сообщение посылается объектам различных классов и вызывает различные процессы в зависимости от того, как выполняется запрашиваемая функция.

Пути сообщений иллюстрируются с помощью диаграмм взаимодействия. На рис. 114 продемонстрирован процесс последовательного обмена сообщениями. Каждый конкретный этап процесса представлен в виде задачи, выполняемой двумя объектами.

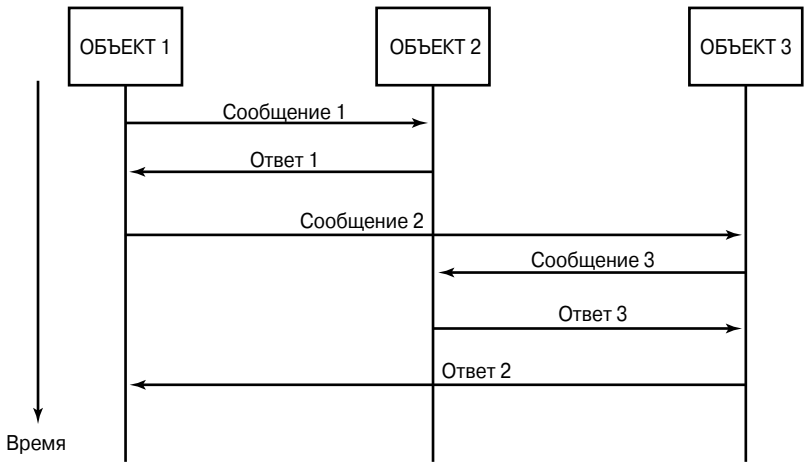


Рис. 114. Диаграмма взаимодействия

В более простом случае диаграммы взаимодействия могут иллюстрировать только базовые сообщения, без детализации последовательности.

Ассоциации уже определяют пути сообщений в объектно-ориентированных диаграммах классов, поэтому моделирование ассоциаций играет такую важную роль в объектно-ориентированном анализе. Помимо путей сообщений, моделируемых ассоциациями, существуют также специальные сообщения, которые пользователи адресуют определенным объектам. Однако пути этих сообщений не включаются в перечень требований, а непосредственно определяются во время выполнения.

Метамодель структуры объектно-ориентированного анализа (см. рис. 101) на рис. 115 включает управление сообщениями.

Сообщения посылаются от функций объектов к функциям других объектов со ссылкой на взаимосвязи, определяемые ассоциациями. Пункты процесса последовательно нумеруются или отмечаются ярлыками, а передаваемые параметры относятся к сообщениям.

Статус может включать выполнение метода, но он может также включать индивидуальные особенности (статус ожидания). В метамоделях это означает кардинальность (0..1) между классами СТАТУС и МЕТОД. Статус вызывается событием, которое, в свою очередь, инициируется статусом.

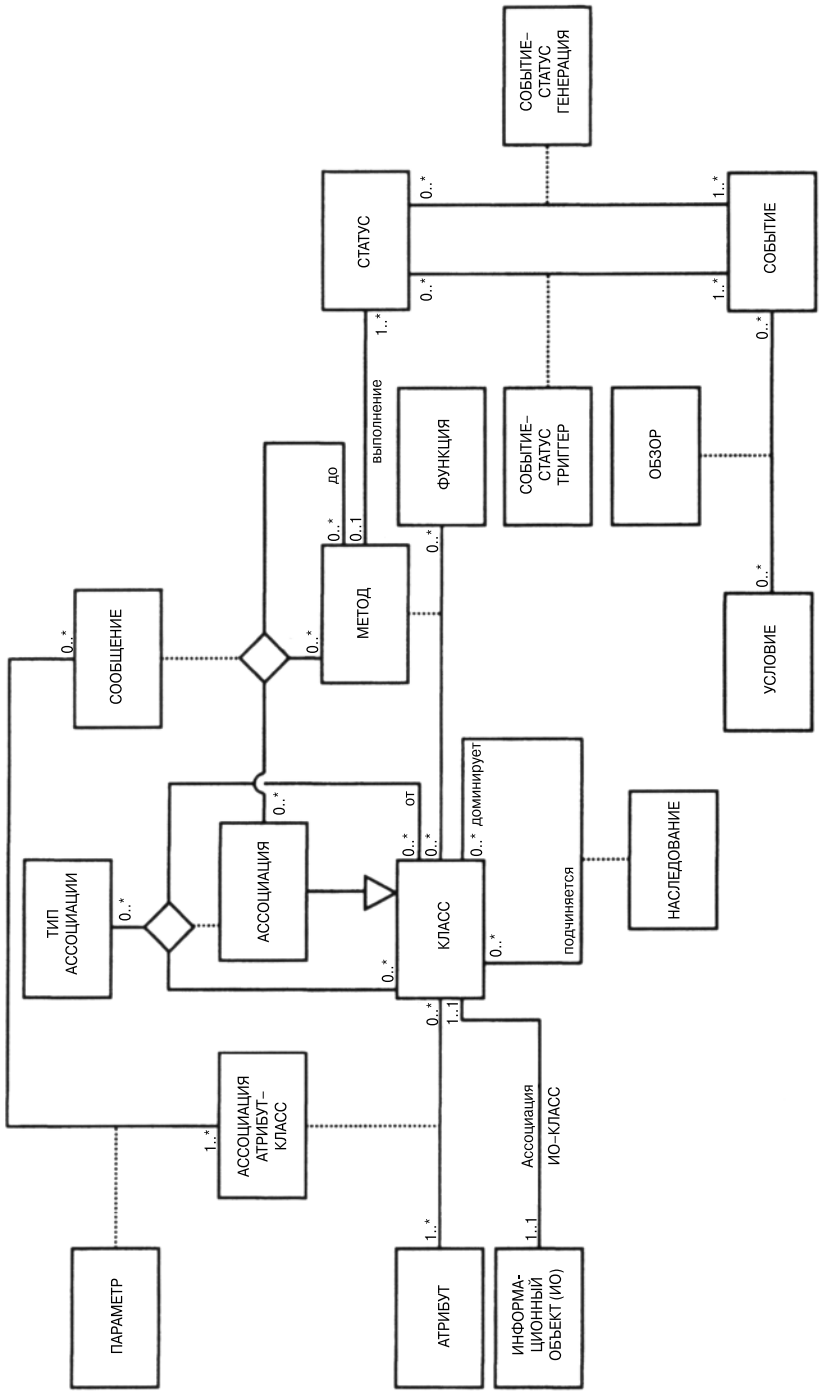


Рис. 115. Метамодель управления сообщениями в объектно-ориентированном анализе

Если понятие статуса приравнивается к понятию функции, метаструктура диаграммы состояний сопоставима с метаструктурой упрощенной событийно-управляемой цепочки, хотя без учета логических взаимосвязей между событиями.

A.III.2.1.2.5 Связь объектно-ориентированного моделирования и метода EPC

Хотя моделирование бизнес-процессов и объектно-ориентированное моделирование соответствуют различным парадигмам, известно немало попыток объединить эти два подхода.

Существует два пути объединения объектно-ориентированной модели и событийно-управляемой цепочки процесса.

Согласно первого подхода, предложенного Бунгертом и Гессом (*см. Bungert/ Heβ, Objektorientierte Geschaftsprozeβmodellierung 1995*), событийно-управляемые цепочки могут быть преобразованы в объектно-ориентированные модели и наоборот. Примеры таких преобразований иллюстрируются на рис. 116, а и 116, б.

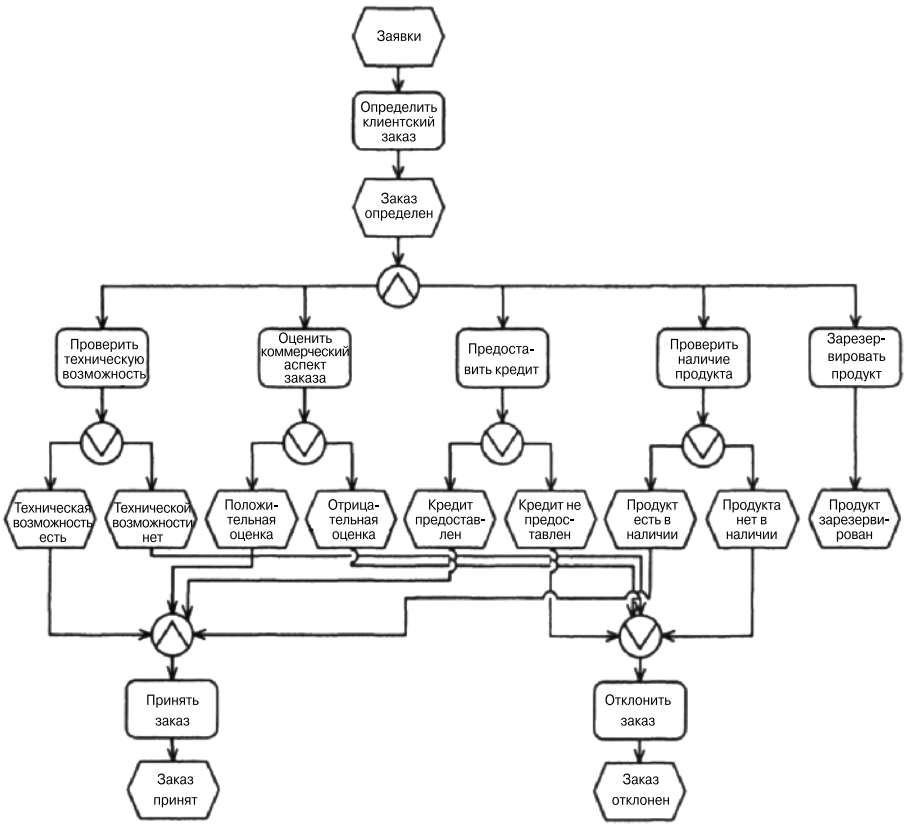


Рис. 116, а. Событийно-управляемая цепочка процесса
(Bungert/Heβ, Objektorientierte Geschaftsprozeβmodellierung 1995, p. 62)

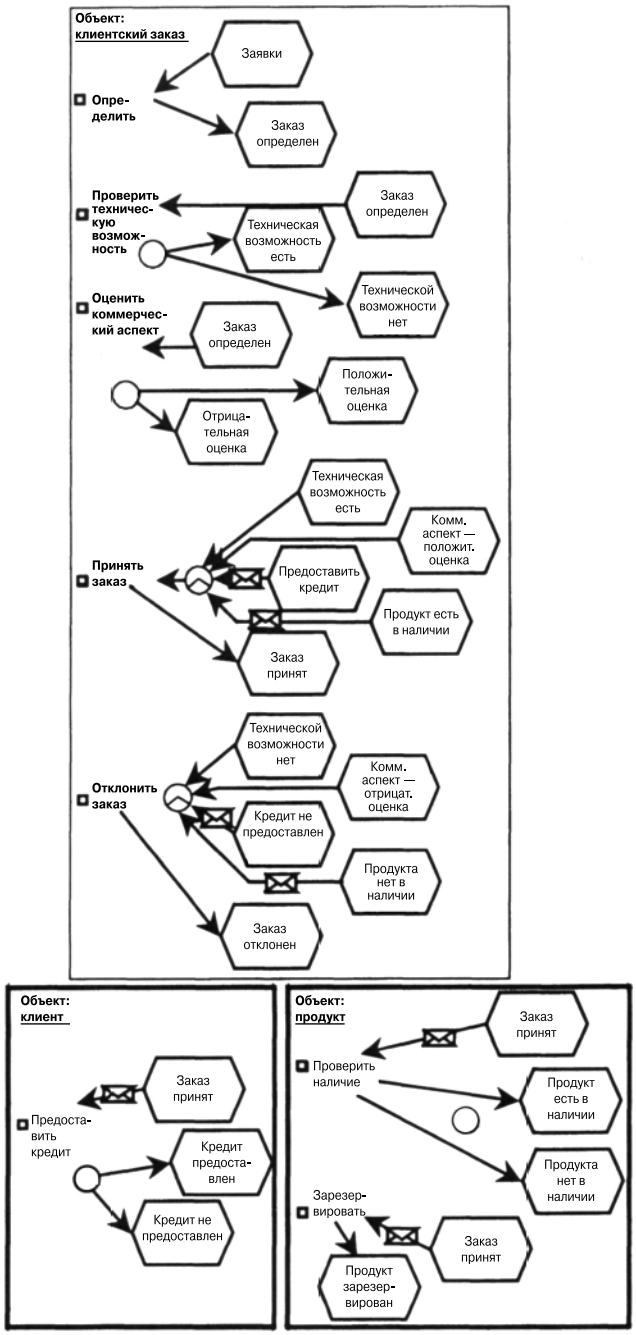


Рис. 116, б. Объектно-ориентированная модель (Bungert/Heβ, Objektorientierte Geschäftsprozeβmodellierung 1995, p. 61)

Здесь предполагается, что информационные объекты, используемые в EPC, могут быть определены как объектно-ориентированные классы. Функции из процессной цепочки относятся к ним, а инициирующие события и события, инициируемые классами, в свою очередь, характеризуют информационные объекты. Инициирующие сообщения могут быть посланы инициируемыми событиями других функций. Внутренние и внешние события характеризуются индивидуально. Метамодель соответствует метамодели EPC (см. рис. 110).

Согласно второй концепции (Натгенса и Циммермана), поток управления событиями EPC преобразуется в поток объектов (*см. Scheer/Nüttgens/Zimmermann, oEPK 1997; Nüttgens/Feld/Zimmermann, Business process Modeling 1998*).

Объектно-ориентированная событийно-управляемая цепочка процесса увязывает управление событиями процессной цепочки с элементами объектно-ориентированного моделирования. Объекты упорядочены согласно потоку управления процесса, а обрабатывающие функции относятся к ним согласно соответствующей процедуре. Это одна из возможных реверсий общей ARIS-модели бизнес-процесса (*см. Scheer, ARIS — Business Process Frameworks 1998, p. 31*).

Моделирование объектно-ориентированной событийно-управляемой цепочки процесса становится критическим в случае, когда обрабатывающая бизнес-функция включает много объектов. В этом случае ключевые объекты должны иметь определения, а другие необходимые объекты только включаются в поток сообщений. Если объект обрабатывается на разных шагах процесса, он перечисляется с различными оперативными функциями. Определения объектов в этой ситуации также вызывают серьезные затруднения.

Этот подход еще раз подчеркивает, что моделирование бизнес-процессов и объектно-ориентированное моделирование следуют различным парадигмам. На рис. 117 иллюстрируется пример объектно-ориентированной событийно-управляемой цепочки процесса, который подчеркивает изложенные выше трудности определения объекта.

A.III.2.2 Конфигурация

Включенные в модель события помогают определить моменты корректировки стоимостных или временны*х* параметров конкретного бизнес-процесса. Когда определенное событие происходит, выполняются соответствующие вычисления, и результаты предоставляются руководству.

Управление процессами в модели документооборота основывается на потоках управления, описанных в модели EPC. Поток управления копируется из соответствующей модели процесса.

Исходя из потока управления, управляющие структуры могут моделироваться все более детально, например, включать задержки перед началом работы, отвергать распределенные задачи или распределять задачи с учетом ограничений (*см. Jablonski, Workflow-Management-Systeme 1995, p. 35*).

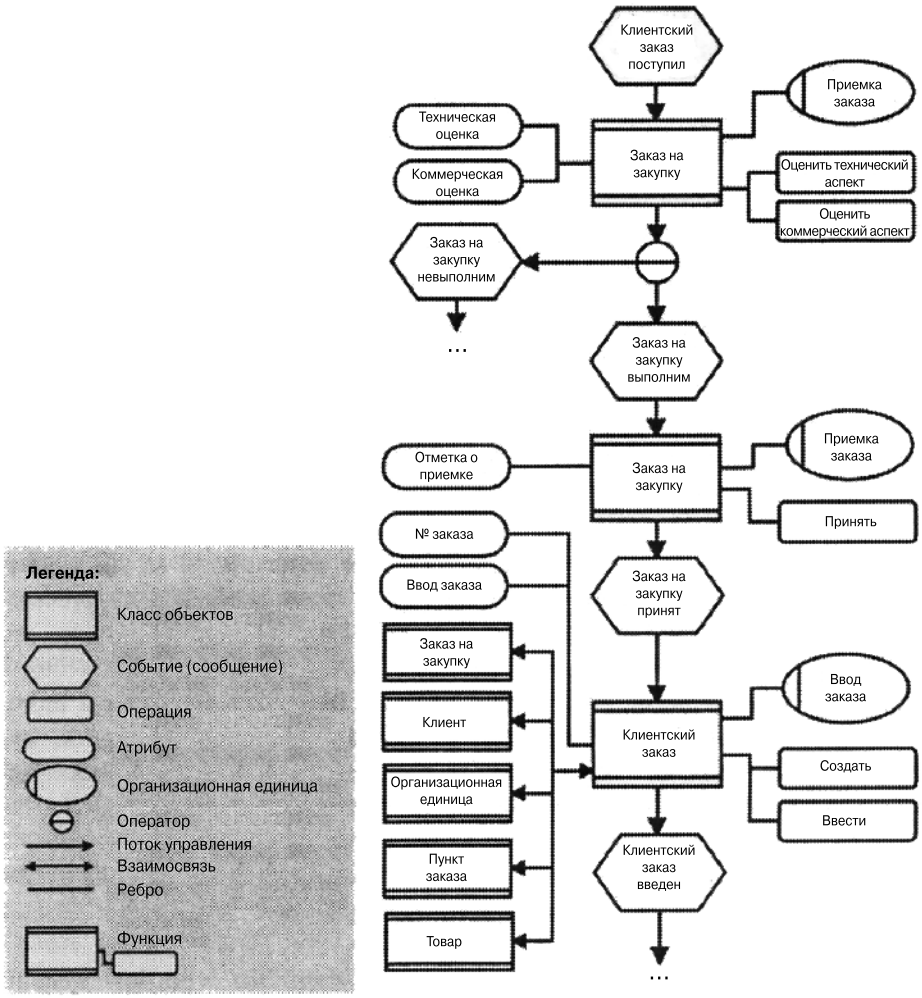


Рис. 117. Объектно-ориентированная событийно-управляемая цепочка процесса

С помощью экранов осуществляется изменение параметров функций.

Бизес-функции определяют, какие прикладные системы и программные модули должна инициировать система управления потоком. На рис. 118 иллюстрируется внедрение модели EPC в модель документооборота.

Подобно потокам управления модели документооборота, потоки управления стандартных программных систем могут быть представлены в методологии EPC. Однако, в отличие от модели управления потоком, эта процедура не свободно конфигурируется, а должна подчиняться параметрам стандартного пакета программ. Это означает, что функции могут быть избраны или определенная последовательность вызовов функций может быть задана.

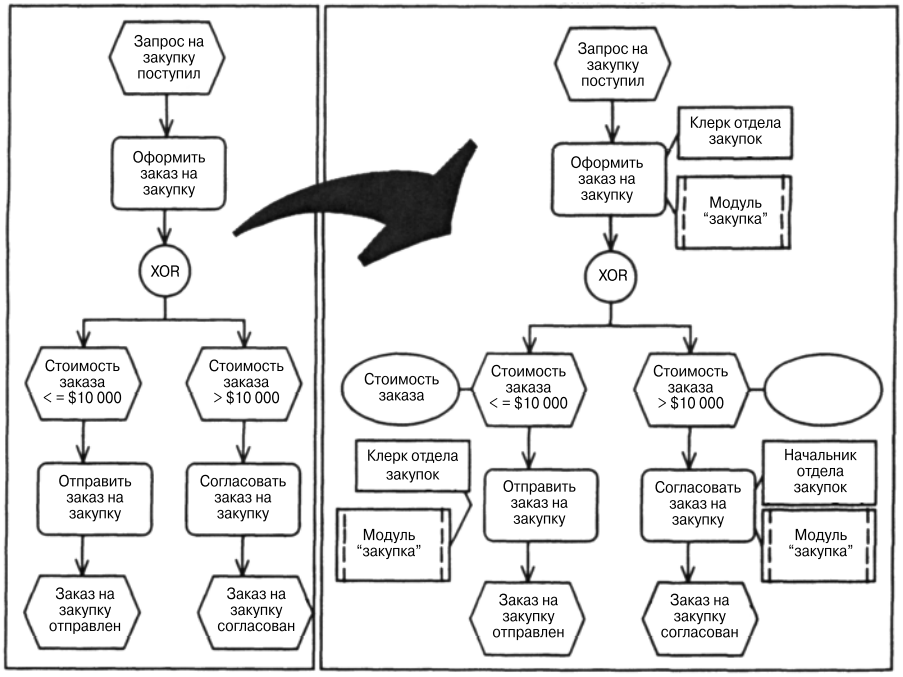


Рис. 118. Внедрение событийно-управляемой цепочки процесса в модель документооборота

Чем больше стандартное программное обеспечение использует новую объектно-ориентированную технологию, тем больше насчитывается вариантов конфигурации. Бизнес-моделирование позволяет проектировать процессы внутри бизнес-объектов и потоки управления между ними.

На рис. 119 иллюстрируется пример конфигурации R/3 с использованием событийно-управляемой цепочки процесса (верхний экран). Закрашенные белым цветом элементы процессной цепочки неактивны. Нижний экран предлагает пользователю прямой доступ к соответствующим операциям, проектной информации и документам.

Функциональные возможности изменяются при изменении моделей экранов. Например, если удалить данные адреса в моделях экранов на рис. 106, *а* и 106, *б*, то пользователь не сможет больше ввести или скорректировать эти данные. Соответствие между объектом данных и функцией также изменится.

A.III.2.3 Спецификация дизайна

A.III.2.3.1 Увязка модулей и баз данных

В спецификации дизайна функционального аспекта модули первоначально создаются без знания схемы конкретной базы данных и опираются только на описание данных. Ниже рассматривается увязка модулей и миниспецификаций с базами данных.

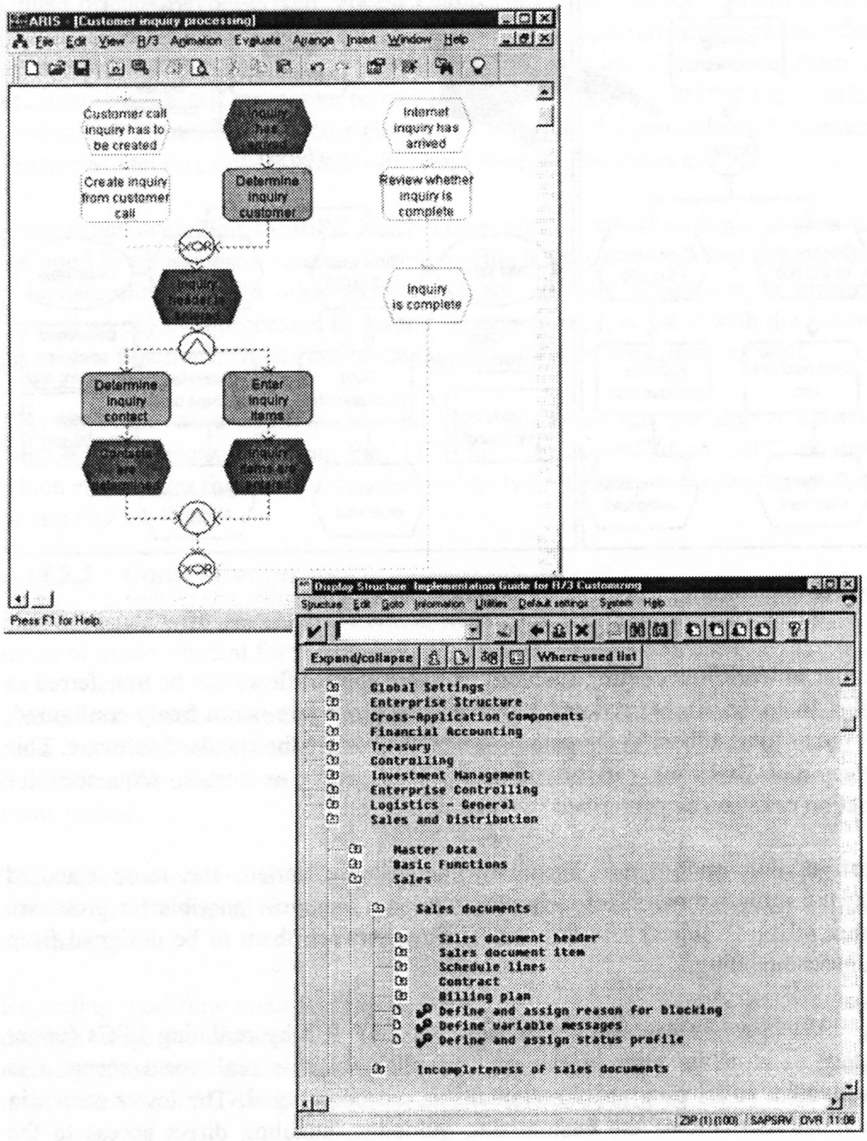


Рис. 119. Конфигурация R/3 с использованием событийно-управляемой цепочки процесса
(SAP AG)

A.III.2.3.1.1 Схема базы данных

Прикладные модули взаимодействуют не со всей концептуальной схемой базы данных, а с ее элементами.

Имеет смысл употреблять индивидуальные имена в концептуальной схеме для конкретных приложений, потому что иногда при автономной разработке

приложения к спецификации дизайна добавляются определения, которые не совпадают с моделью данных, разработанной независимо.

Внешние схемы баз данных, представляющие собой интерфейс между пользователями и концептуальным дизайном, определяют логические версии базы данных для конкретного приложения или конкретного пользователя. Новые отношения создаются из реляционной схемы с помощью удаления атрибутов или определенных кортежей базовых отношений, комбинации базовых отношений согласно определенному критерию или даже разбиения базовых отношений на несколько отношений. Один из важных методов — создание так называемых взглядов (views). Их общая структура имеет такой вид (*см. Mayr/Dittrich/Lockemann, Datenbankentwurf 1987, p. 537*):

DEFINE VIEW [наименование взгляда],

SELECT [выражение].

Метаструктура взглядов иллюстрируется на рис. 120. Концептуальная схема в целом, включающая отношения, атрибуты и условия целостности, представлена комплексным объектом КОНЦЕПТУАЛЬНАЯ СХЕМА. Классы ВНЕШНЯЯ СХЕМА и КОНЦЕПТУАЛЬНАЯ СХЕМА связаны ассоциацией. Много внешних схем может относиться к одному модулю и, наоборот, внешняя схема может соответствовать многим модулям.

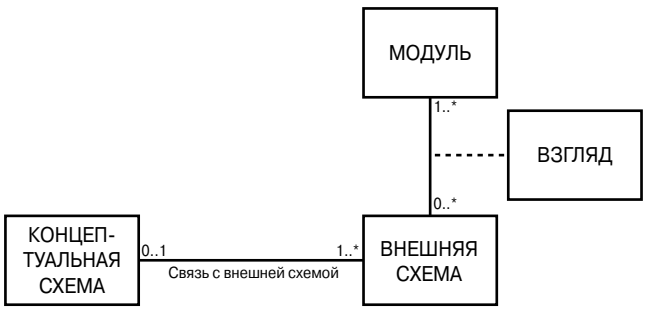


Рис. 120. Связь модулей и схемы базы данных

A.III.2.3.1.2 Управляющие структуры

Как правило, модули состоят из трех частей: объявление данных, управляющая логика и исполняемые команды (инструкции). Управляющие структуры представляют собой последовательное выполнение, выбор по условию и циклы.

Управляющие структуры связаны со структурами данных. Ассоциация 1:1 между классами соответствует последовательному выполнению, ассоциация 1:* соответствует циклу, а операторы специализации (разбивают информационные объекты на подчиненные понятия) соответствуют выбору по условию.

Конкретный счет однозначно соответствует информационному объекту КЛИЕНТ (ассоциация 1:1). Клиент может участвовать во многих бизнес-событиях, хотя одно бизнес-событие соответствует единственному клиенту. БИЗНЕС-СОБЫТИЕ разбивается на ЗАКАЗ и ОТМЕНА.

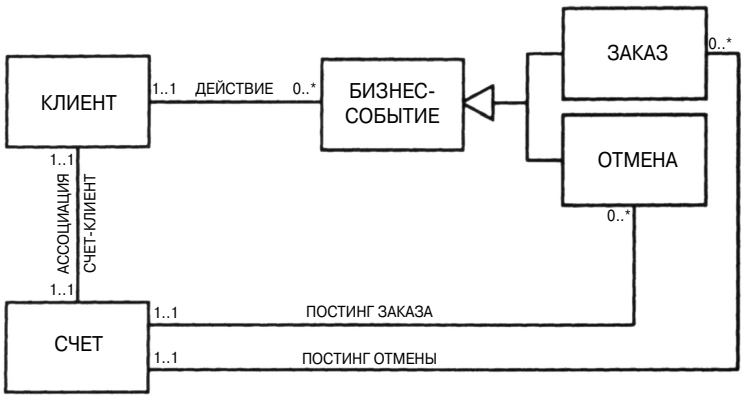


Рис. 121. Взаимосвязь управляющих структур и структур данных

Эти различные бизнес-события инициируют различные постинг-события. Соответствующие управляющие структуры иллюстрируются на структограмме рис. 122. Первоначально читается одна учетная запись клиента. Затем читается соответствующий счет. Оба действия образуют последовательность, так как кардинальность равна 1 (с точки зрения клиента).

Различные бизнес-события с кардинальностью * (с точки зрения клиента) выполняются для клиента, образуя цикл.

В зависимости от типа бизнес-события выполняется различный постинг, согласно их конкретной специализации.

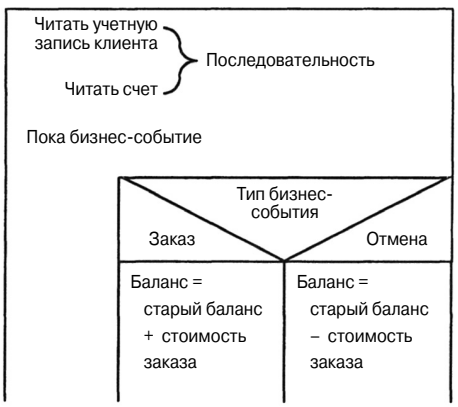


Рис. 122. Пример управляющих структур

Метаиллюстрации опущены. Этот программный дизайн похож на генерацию обмена сообщениями из ассоциаций диаграммы классов в контексте объектно-ориентированного моделирования.

А.III.2.3.1.3 Транзакции в базах данных

Корректировка баз данных основана на понятии транзакции. Свойства транзакции описываются аббревиатурой ACID (Atomarity, Consistency, Isolation, Durability — атомарность, непротиворечивость, изолированность, долговечность). Транзакции включают серии непрерываемых (атомарных) с точки зрения приложения операций с элементами баз данных. Это означает, что, с точки зрения приложения, непротиворечивость базы данных обеспечивается только если транзакция полностью завершена. Если случается ошибка, данные возвращаются в состояние, предшествующее транзакции. Это свойство транзакции называется **атомарностью**. База данных будет скорректирована, только если транзакция успешно завершена.

Помимо атомарности, управление транзакциями должно обеспечить **непротиворечивость**, т.е. транзакции преобразуют базу данных из одного непротиворечивого статуса в другой непротиворечивый статус.

Еще одно свойство — **изолированность**, т.е. промежуточные результаты не могут передаваться другим приложениям в процессе транзакции. **Долговечность** означает, что эффект успешно завершенной транзакции остается неизменным и может быть скорректирован только с помощью новых транзакций.

Транзакции характеризуются шагами “начало транзакции” и “конец транзакции”. Между этими двумя шагами включается произвольное количество команд записи в файл или чтения из файла. Транзакции также используются как единицы измерения шагов обновления данных.

С точки зрения программного дизайна транзакции могут интерпретироваться как модули, поэтому на рис. 123 вводится понятие ТРАНЗАКЦИЯ как конкретизация понятия МОДУЛЬ. Как модули, транзакции могут связываться в сети.

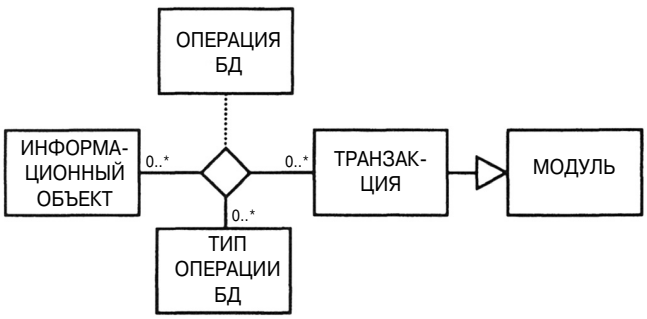


Рис. 123. Концепция транзакции

Множество операций базы данных объединяются в одну транзакцию, образуя ассоциацию ОПЕРАЦИЯ БАЗЫ ДАННЫХ (БД) между классами ТИП ОПЕРАЦИИ БД и упомянутыми классами ТРАНЗАКЦИЯ и ИНФОРМАЦИОННЫЙ ОБЪЕКТ.

A.III.2.3.2 Управление триггерами

Базы данных предназначены не только для хранения данных о предприятии. С ними связаны компоненты, реагирующие на определенные события, связанные с приложением, и инициирующие обновление базы данных (системы активных баз данных). Эти компоненты называются триггерами. Мы упоминали их при рассмотрении условий целостности в спецификации дизайна аспекта данных.

Триггеры инициируют выполнение прикладных функций. Например, в рамках системы планирования складских запасов непрерывно проверяется наличие на складе определенных запасных частей. Событие “объем запаса ниже минимального” инициирует заказ на поставку.

Говоря кратко, триггеры состоят из определения событий, инициирующих их, проверяемых условий и действий, инициируемых при выполнении этих условий. Действия представляют собой операции, т.е. транзакции, направленные на обновление данных. В соответствии с механизмом ETM (Event Trigger Mechanism), триггером называется пара {T = (E,A)}, состоящая из события и действия (*см. Kotz, Triggermechanismus in Datenbanksystemen 1989, p. 54*). Действие A выполняется, как только происходит событие E.

Например, если в процессе разработки продукта некоторая процедура запускается после завершения первой фазы разработки, триггер имеет вид:

EVENT end_of_design_phase_1(design_object: DB_ID);
ACTION verification_procedure_A(verif_obj: DB_ID)
=<проверка verif_obj>;
TRIGGER T1 = ON end_of_design_phase_1
DO verification_procedure_A(design_object);
(*см. Kotz, Triggermechanismus in Datenbanksystemen 1989, p. 64*).

Индивидуальное определение событий, действий и триггеров позволяет различным триггерам использовать одни и те же определения событий и действий. С другой стороны, для определенных событий и действий возможны различные процедуры контроля в зависимости от выбранных параметров.

Используемый здесь синтаксис находится на уровне спецификации дизайна, не зависит от конкретных триггерных систем, конкретных систем баз данных или программной среды.

Механизм “событие–триггер” можно прямо увязать с событийно-управляемой цепочкой процесса, потому что события уже определены, а действия соответствуют функциональным модулям. Триггеры характеризуют контекст между E и A, что соответствует ребрам между событиями и функциями событийно-управляемой цепочки процесса.

Процессы, разработанные в рамках EPC, могут инициироваться триггерами. Модели EPC следует только дополнить символами спецификации дизайна, включая определения событий, триггеров и действий.

На рис. 124 иллюстрируется метамодель концепции триггера, где класс СОБЫТИЕ осуществляет связь с уровнем определения требований.

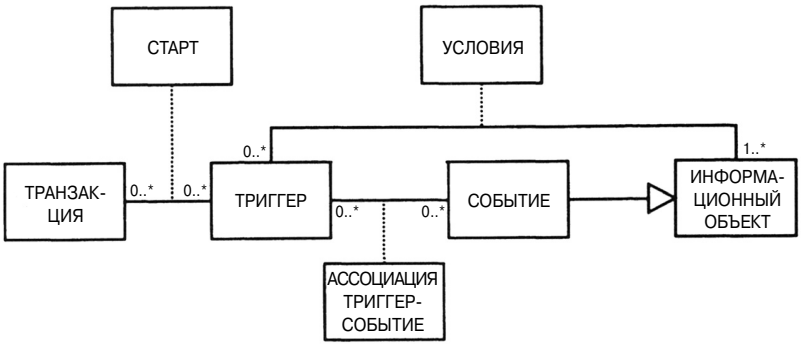


Рис. 124. Структура концепции триггера

Кроме внешних событий, существуют внутренние события, такие как подтверждения заказов. Конкретные моменты времени могут также иллюстрировать события, например, когда определенные события выполняются в конце каждого часа. Поскольку время тоже может быть определено как класс, что делает его информационным объектом, обозначения событий, принятые на этапе определения требований, подходят для триггеров, описанных здесь. Событие может быть инициировано многими триггерами, а конкретный триггер может инициировать много событий.

Когда триггер стартовал, различные соотношения между элементами базы данных проверяются в соответствии с правилами, определенными для триггера. Это отображает класс ассоциаций УСЛОВИЯ, соединяющий классы ТРИГГЕР и ИНФОРМАЦИОННЫЙ ОБЪЕКТ.

Если события выполняются, триггер инициирует одну или несколько транзакций. Конкретная транзакция может быть инициирована различными триггерами. Например, при проверке складских запасов инициирующим событием может быть определенный момент времени (ежечасная проверка наличия необходимого минимума). Другим событием может быть отправка заказа на поставку конкретного элемента запасов.

A.III.2.3.3 Объектно-ориентированная спецификация дизайна

Один из аспектов парадигмы объектно-ориентированного моделирования состоит в существовании тесных взаимосвязей между фазами жизненного цикла. В идеале элементы модели определения требований должны внедряться 1:1; при этом одним из нескольких подходящих методов будет быстрая разработка прототипа (rapid prototyping — быстрое прототипирование). Тем не менее, концепция фаз жизненного цикла типична даже для объектно-ориентированного моделирования. При этом ребра между определением требований, спецификацией дизайна и описанием внедрения не жесткие. В этом свете некоторые авторы (*см. Oestereich, Objektorientierte Softwareentwicklung 1997, p. 66*) относят к уровню определения требований только диаграммы вариантов использования (фаза анализа), а классы, последовательности и графики состояний — к уровню

спецификации дизайна (фаза разработки). Другие авторы применяют одни и те же методы и диаграммы на обоих уровнях, детализируя эти методы на уровне спецификации дизайна.

Мы готовы согласиться с последней теорией. Следовательно, нет необходимости воскрешать в памяти метамодели определения требований. Однако если в ходе преобразований возникает необходимость разделить объекты, возможны взаимосвязи n:n между объектами определения требований и спецификации дизайна. С точки зрения метамодели это будет означать, что дизайн-объект спецификации дизайна связан с каждым дизайн-объектом определения требований с помощью ассоциации n:n.

A.III.2.3.3.1 Общая детализация

Различают внутренние и внешние методы.

Обусловлены такие типы атрибутов классов: гарантии, начальные значения и параметры (см. рис. 125). Гарантии представляют собой условия, которые должен соблюдать объект. Параметры операций сообщают, при каких значениях переменных данные будут преобразованы после запуска операции. На рис. 125 говорится о том, что атрибут “radius” должен быть неотрицательным (гарантия); заданы координаты центра x = 10 и y = 10 (начальные значения); окружностью можно манипулировать, задавая новые координаты центра или новый радиус (параметры).

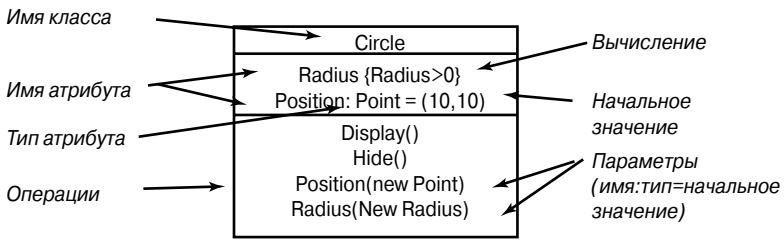


Рис. 125. Технические свойства атрибутов

Динамическое поведение на протяжении жизненного цикла объектов отличается от статуса, последовательности и диаграмм состояния.

Термины “программы” и “модули” также используются в объектно-ориентированном моделировании. Модули представляют собой файлы исходных кодов и объектных кодов.

Модули и их взаимосвязи изображаются на диаграммах компонентов. Метаструктура диаграммы компонентов большей частью соответствует метаструктуре модуля. Это в значительной степени зависит от определения компонентов или модулей.

Понятие “пакет” обозначает группу элементов объектно-ориентированной модели. Иногда понятия “пакет” и “модуль” используются как синонимы.

Как модули, так и пакеты представляют собой компоненты языка UML и иллюстрируются на диаграммах компонентов (рис. 126).

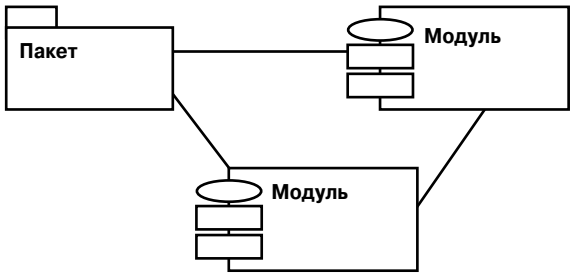


Рис. 126. Диаграмма компонентов в соответствии с UML

A.III.2.3.3.2 Взаимосвязь баз данных

Устойчивые объекты хранятся в объектно-ориентированных базах данных. При этом выполняются принципы наследования, т.е. объектные данные подчиненного класса могут быть устойчиво сформированы из атрибутов доминирующих классов.

С другой стороны, если в объектно-ориентированных моделях используются системы реляционных баз данных, могут возникнуть проблемы, касающиеся взаимосвязи двух парадигм моделирования. Существует два противоположных подхода к моделированию взаимосвязи баз данных (*см. Oestereich, Objektorientierte Softwareentwicklung 1997, p. 136*).

1. Каждый устойчивый объект запоминается в реляционной таблице. Каждый объект каждого класса участвует только в одном отношении, поэтому необходимо корректировать различные виды кортежей в таблице (рис. 127, а).
2. Для каждого класса создается реляционная таблица с ассоциацией 1:1 (рис. 127, б). При этом приходится группировать данные нескольких доминирующих классов для описания объектов подчиненных классов.

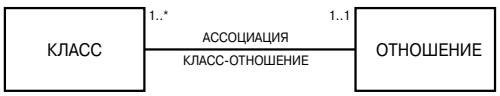


Рис. 127, а. Каждый устойчивый объект запоминается в реляционной таблице

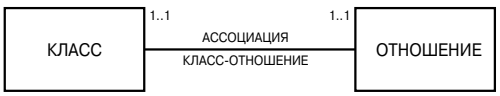


Рис. 127, б. Реляционная таблица создается для каждого класса

В качестве компромисса можно хранить данные на самом нижнем уровне иерархии классов (*см. Oestereich, Objektorientierte Softwareentwicklung 1997, p. 137*). Однако, чтобы получить доступ к вышестоящим классам из нескольких подклассов, сначала необходимо сгруппировать данные из нескольких таблиц.

В общем, моделирование взаимосвязей баз данных вызывает затруднения, так как технология объектно-ориентированных баз данных относительно молода, а интерфейсы между языками объектно-ориентированного моделирования и системами реляционных баз данных несовершенны.

A.III.2.4 Описание внедрения

Взаимосвязи функций и данных внедряются с помощью систем баз данных и языков программирования. Усиливая функциональность соответствующего языка описания данных, системы баз данных позволяют конкретно определить внешние схемы. В частности, системы активных баз данных позволяют внедрить механизмы триггеров.

С помощью объектно-ориентированных языков программирования (C++, Smalltalk, Java и т.п.) методы и диаграммы спецификации дизайна воплощаются в программный код. На рис. 128 иллюстрируется описание класса ОКРУЖНОСТЬ (см. рис. 125) средствами языка C++. Конкретное содержание отдельных операций не рассматривается.

```
class circle
{
                int radius;
                point position;
        public:
                void radius (int newradius);
                void position (point newpoint);
                void display ();
                void hide ();
};
void circle::radius (int newradius)
{
                if (newradius>0)  //вычисления
                {
                ...
                };
};
...
```

Рис. 128. Описание класса объектов из рис. 125
(Oestereich, Objektorientierte Softwareentwicklung 1997, p. 37)

A.III.3 Взаимосвязи функций и результата

Понятие “Результат” включает различные типы результата — материальный продукт и услуги, в том числе и информационные.

Взаимосвязи функций и результата состоят в том, что функции преобразуют входные данные в выходной результат. Процесс, т.е. последовательность функций, соответствует определенному результату.

На рис. 129 иллюстрируется место взаимосвязей функций и результата в концепции ARIS. Рассматриваемые аспекты представлены в виде затененных фигур.

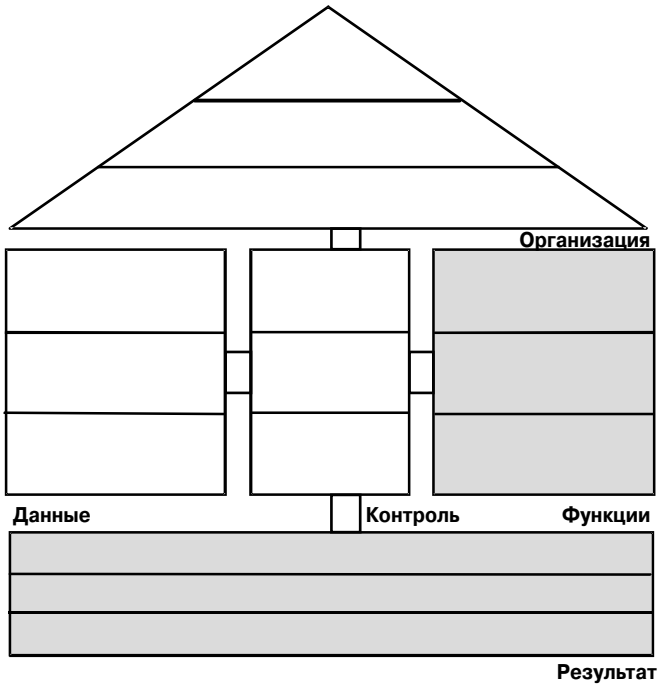


Рис. 129. Взаимосвязи функций и результата в концепции ARIS

A.III.3.1 Моделирование описания требований

На рис. 130 иллюстрируется выходной поток функции, включая количество взаимосвязей функции и результата. На рис. 131 представлена соответствующая метамодель. Здесь предполагается, что выходной результат выполнения каждой функции имеет уникальное наименование.

На рис. 132 и 133 иллюстрируется ситуация, когда после выполнения функции изменяется только статус выходного результата, но не его наименование. Результат выполнения функции определяет исходный статус для выполнения следующей функции. Эта процедура также используется при составлении пооперационного графика в промышленном производстве.

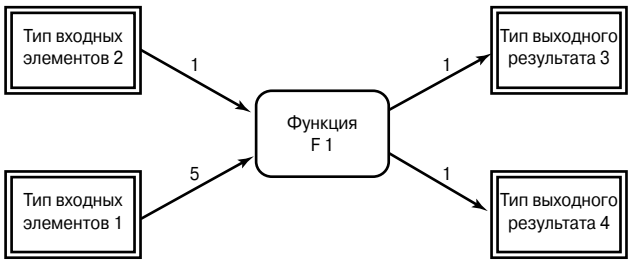


Рис. 130. Выходной поток функции

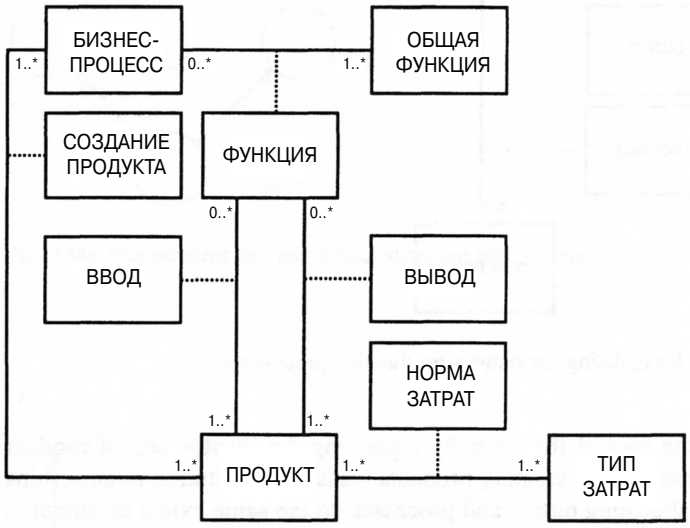


Рис. 131. Метамодель изменения типов выходного результата

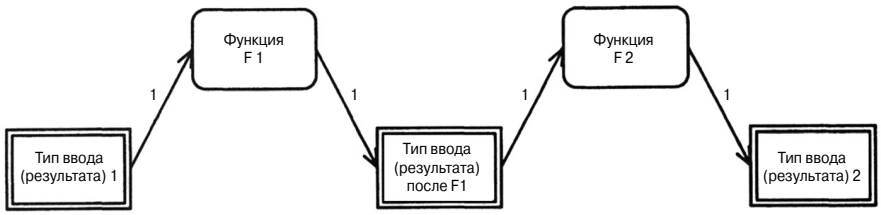


Рис. 132. Поток выходных данных функции без изменения наименования результата

Все, что производится в ходе бизнес-процесса, моделируется с помощью взаимосвязей выходного результата и функции. Выходной поток последней функции бизнес-процесса означает конечный результат всего бизнес-процесса.

Выходному результату соответствуют определенные типы затрат. Внутри процесса рассматривается поток затрат.

Выходной поток функции включает структуру выходного результата, т.е. спецификации материалов.

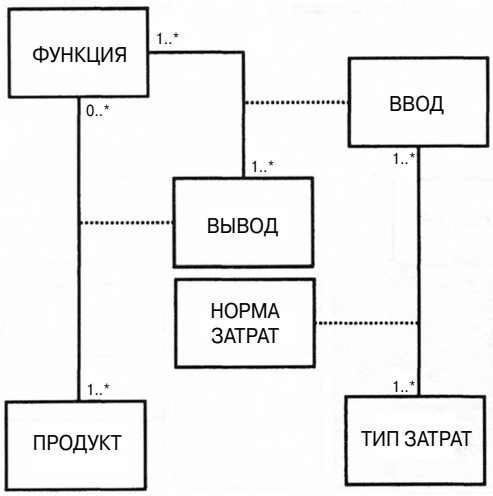


Рис. 133. Метамодель изменения статуса после выполнения функции

При моделировании потока управления строгое разделение структур продуктов и процессов ведет к подстановке взаимосвязей (рис. 134, *а* и 134, *б*). Когда установлено соответствие между продуктами и процессами, цепочки процессов описывают создание промежуточного или конечного продукта из входных составляющих.

На рис. 134, *а* структура выпуска характеризуется двумя промежуточными продуктами. Чтобы создать выходной результат, необходима одна функция на каждый продукт. Это приводит к моделированию четырех событийно-управляемых цепочек процесса. Каждой из этих цепочек соответствует одна функция, описывающая производство одного продукта.

На рис. 134, *б* иллюстрируется структура выпуска, включающего два продукта, без определения промежуточных продуктов. Соответствующий бизнес-процесс имеет более сложную структуру. Параллельное выполнение операций в таком случае отображается не в спецификации материалов, а в описании процесса.

Это объясняет, почему графики операций в промышленном производстве обычно включают лишь несколько (от 7 до 10) операций (функций), а основная часть процесса отображается в спецификации материалов.

При моделировании бизнес-процессов в общем случае речь идет о структуре всего процесса. Эта структура лишь непрямо включает промежуточные продукты. Она также включает разнообразные функции и параллельные структуры. Чем сложнее структура выпуска, тем больше идентичных возможностей существует при раздельном администрировании процессов и продуктов. Это достигается за счет того, что структура выпуска моделирует структуру ключевого процесса, а структуры процессов только моделируют прямые процессы для каждого продукта.

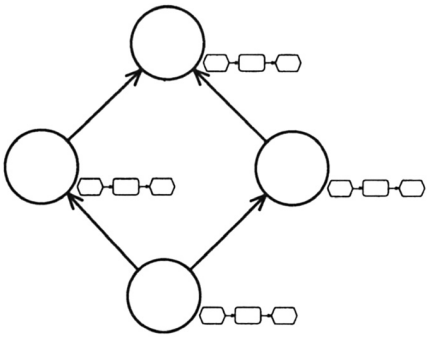


Рис. 134, а. Структура процесса и структура выпуска с промежуточными продуктами

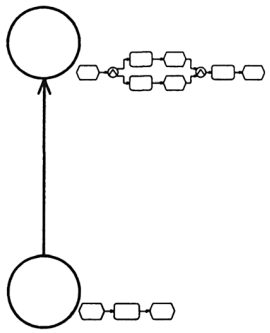


Рис. 134, б. Структура процесса и структура выпуска без промежуточных продуктов

Язык UML позволяет моделировать поток объектов, обрабатываемых функциями, с помощью диаграмм действий (рис. 135). Изменение статуса в результате выполнения функции интерпретируется как выходной результат, т.е. информационный продукт. Метамодель (см. рис. 133) уже отображает этот поток объектов. Кроме того, на диаграммах действий указываются организационные единицы.

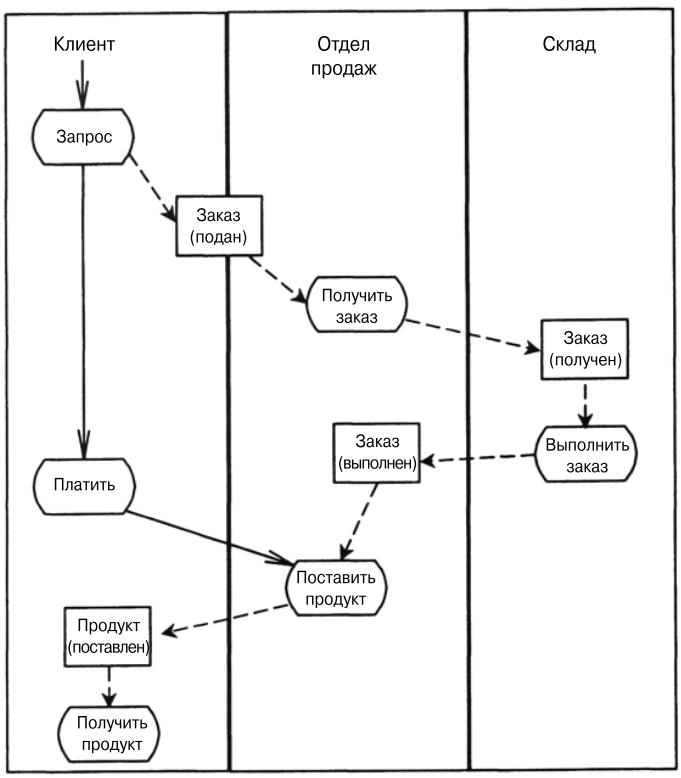


Рис. 135. Действия и поток объектов
(UML Notation Guide 1997, Fig. 56)

A.III.3.2 Конфигурация

Когда выполнение функции завершено, сообщения о выходном результате и соответствующих затратах поступают на уровень управления бизнес-процессом. В зависимости от типа выходного результата (материальный продукт или услуги) эти сообщения распространяются с помощью систем сбора технологических данных в материальном производстве или систем документооборота в информационной сфере.

В ходе конфигурации сведения о типах выходных результатов и типах затрат передаются в систему управления процессом.

Характеристики информационных услуг конфигурируют систему документооборота и служат для определения электронных носителей данных.

A.III.4 Взаимосвязи организации и данных

На рис. 136 иллюстрируется место взаимосвязей организации и данных в концепции ARIS. Рассматриваемые аспекты представлены в виде затененных фигур.

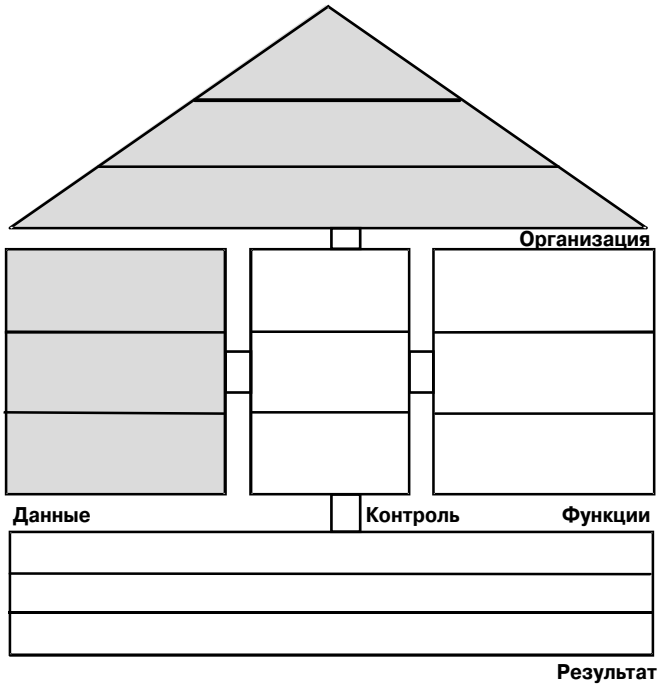


Рис. 136. Взаимосвязи организации и данных в концепции ARIS

A.III.4.1 Моделирование описания требований

Как и в модели взаимосвязей функций и организации, где ключевые функции поставлены в соответствие определенным организационным единицам предприятия (см. рис. 87), в модели взаимосвязей организации и данных (рис. 137) основные объекты данных увязаны с подразделениями предприятия. Взаимосвязи между организационными единицами и данными можно детализировать с помощью таких понятий, как “использование”, “только чтение” или “ответственность за корректировку”. На рис. 138 иллюстрируется соответствующая метамодель.

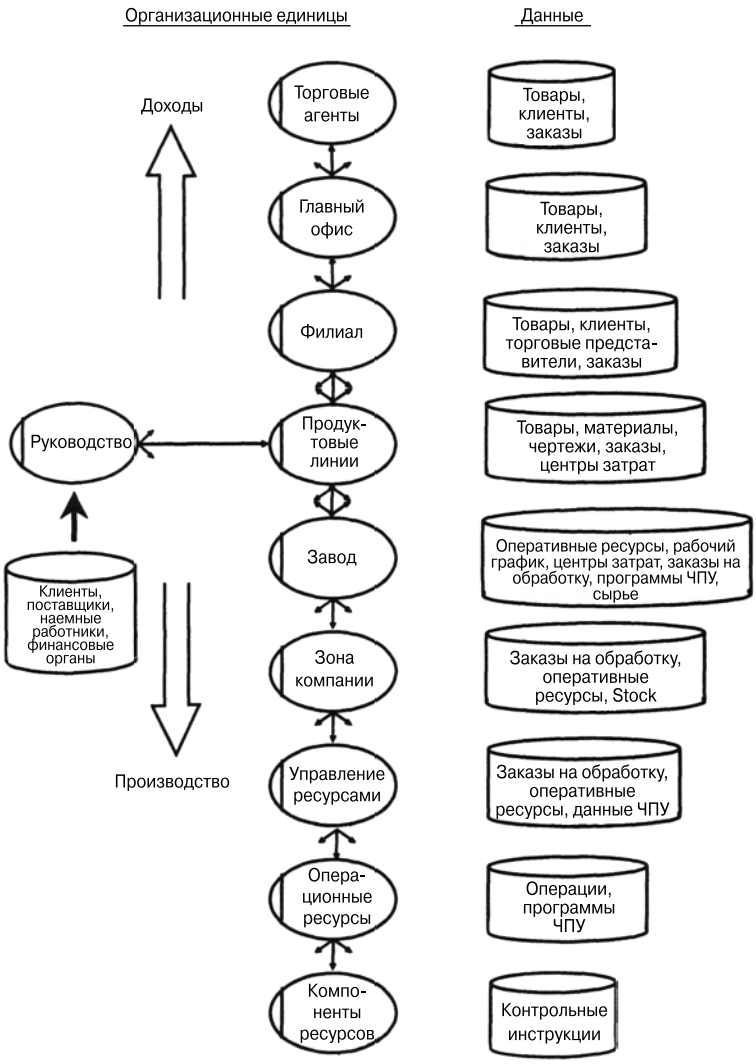


Рис. 137. Модель организационного распределения данных

Рис. 138. Метамодель взаимосвязей организации и данных

При авторизации права доступа пользователей на уровне атрибутов определяются с помощью таких операций, как “создание”, “удаление”, “корректировка” и “чтение”. Авторизацию представляют в виде таблицы. На рис. 139 иллюстрируется метамодель аторизации. Пользователи определяются как отдельные позиции в составе конкретных организационных единиц.

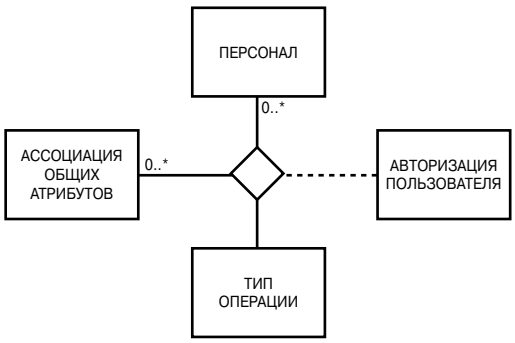


Рис. 139. Метамодель авторизации

Необходимо сформулировать ограничения на типы атрибутов и типы пользователей конкретного процесса. Если возможно, следует определить дополнительные ограничения на числовые значения атрибутов. Например, сотрудник управления по работе с персоналом может читать величину заработной платы, если она не превышает заданного предела. Эти ограничения определяются с помощью значений атрибутов ассоциации АВТОРИЗАЦИЯ ПОЛЬЗОВАТЕЛЯ. Чтобы описать взаимосвязи на уровне позиций, определяют различные типы авторизации для различных типов операций. Например, сотрудник управления по работе с персоналом может корректировать данные до определенного уровня заработной платы, а только читать — до другого заданного уровня.

Действия организационных единиц инициируются определенными событиями. Подобно триггерам функций, действия организационных единиц осуществляются после преобразования соответствующих элементов данных в инициирующие сообщения (рис. 140). Это особенно важно для управления производственным процессом, где распоряжения к действию распространяются как электронные файлы и отображаются в моделях событийно-управляемых процессов согласно распределению функций в организации. Более упрощенно обмен сообщениями между организационными единицами иллюстрируется в виде диаграмм взаимодействия (см. рис. 114).

A.III.4.2 Конфигурация

Установление соответствия между данными и организационными единицами с определенными ограничениями доступа позволяет обеспечить сотрудников, занятых управлением бизнес-процессами, необходимыми инструментами обработки данных.

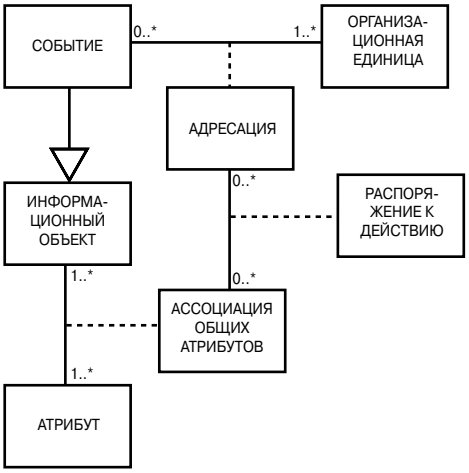


Рис. 140. Событийное управление действиями организационных единиц

Авторизация позволяет в рамках систем документооборота конфигурировать потоки документов, доступных для организационных единиц. С этой целью событийно-управляемые цепочки процессов усилены с помощью организационного распределения функций.

Компоненты бизнес-приложения конфигурируются с помощью авторизации при проектировании экранов и доступа к данным.

A.III.4.3 Спецификация дизайна

A.III.4.3.1 Детализация авторизации

На этапе спецификации дизайна возможности доступа к данным описываются более детально. На рис. 141 иллюстрируются различные возможности определения привилегий доступа. Ключевые привилегии — это чтение, создание, корректировка и удаление данных. Соответствующие объекты классифицируются. Объектом может быть отношение, отдельный атрибут, кортеж или комбинация кортежей. Логические соотношения используются в качестве критерия отбора. Например, руководитель отдела может читать учетные записи только сотрудников своего отдела или сотрудник управления по работе с персоналом может читать величину заработной платы, если она не превышает заданного уровня.

Метамодель авторизации в целом соответствует рис. 139.

В дополнение к прямой взаимосвязи между классами ПЕРСОНАЛ и АССОЦИАЦИЯ ОБЩИХ АТРИБУТОВ (см. рис. 139) устанавливается непрямая связь с помощью назначения привилегий конкретным паролям и конкретным пользователям. Это позволяет устранить избыточность в модели (например, два пользователя с одинаковыми правами доступа получат одинаковые привилегии пароля). Метамодель авторизации иллюстрируется на рис. 142.

Пользователь	Объекты базы данных				
U_1	P_1, P_1		P_3		P_i
U_2		P_1	P_2, P_3		P_1
U_3		P_1, P_2	P_2		
U_m	P_1, P_1	P_i	P_1		P_i, P_k

Обозначения:
U_i — пользователь
O_i — объект базы данных
P_i — привилегии

Рис. 141. Таблица авторизации
(Reuter, Sicherheits-und Integritätsbedingungen 1987, p. 352)

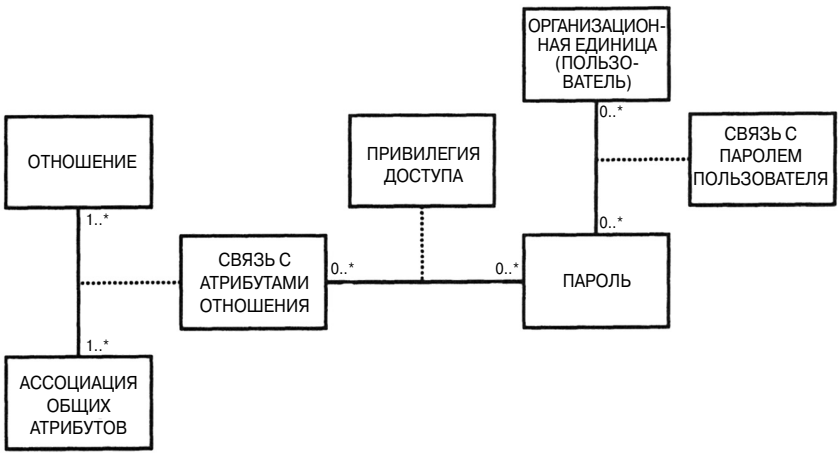


Рис. 142. Метамодель авторизации

Механизм безопасности можно представить в виде таблицы, где характеризуются ключевые запросы к базе данных. Система управления реляционными базами данных CA-Ingres предусматривает специальные возможности, где правила доступа определяются как так называемые ограничения диапазона. Когда запрос выполняется, содержание запроса и ограничения диапазона формируют инструкцию языка DML и обрабатываются в рамках одного синтаксиса. Рис. 143 не требует пояснений.

Словесное содержание	Инструкции CA-Ingres QUEL DML
Ограничения диапазона	
Начальнику отдела Миллеру разрешен доступ ко всем записям отдела D43.	RANGE OF X IS EMPLOYEE RESTRICT ACCESS FOR 'MILLER' TO EMPLOYEE WHERE X.DEPT.MEMB = ''D43'
Содержание запроса	
Составить список всех сотрудников, заработная плата которых превышает 30 тыс. долл. США в год.	RANGE OF X IS EMPLOYEE RETRIEVE INTO LIST (X.NAME) WHERE SALARY > 30000
Запрос с ограничениями	
	RANGE OF X IS EMPLOYEE RETRIEVE INTO LIST (X.NAME) WHERE SALARY > 30000 AND X.DEPT.MEMB = 'D43'

Рис. 143. Авторизация в QUEL
(Reuter, Sicherheits-und Integritatsbedingungen 1987, p. 361)

A.III.4.3.2 Распределенные базы данных

На этапе определения требований данные логически поставлены в соответствие организационным единицам независимо от физических носителей данных. При спецификации дизайна элементы схемы базы данных распределяются между узлами. Этими элементами управляют различные СУБД, взаимодействующие между собой. Следовательно, системы баз данных имеют две задачи: управление локальными данными, размещенными в узле, и связь с данными, размещенными в других узлах.

Распределенные базы данных позволяют облегчить доступ к данным, обеспечить своевременную актуализацию данных, уменьшить затраты и повысить гибкость использования данных (*см. Nerreter, Zur funktionalen Architektur von verteilten Datenbanken 1983, p. 2; Jablonski, Datenverwaltung in verteilten Systemen 1990, p. 5*). Вместе с тем для управления распределенными базами данных требуются значительные усилия.

Крис Дейт сформулировал 12 правил, характеризующих системы распределенных баз данных. Правило 0 представляет собой фундаментальный принцип распределенных баз данных (*см. Date, Distributed Database Systems 1987*).

Правило 0: **Фундаментальный принцип**

Для пользователей система распределенных баз данных должна выступать как централизованная база данных.

Это правило позволяет поддерживать равномерно определенные условия целостности и права доступа, потому что они не зависят от распределения данных.

Правило 1: **Локальная автономия**
Каждый узел должен управлять размещенными в этом узле локальными данными, так что безопасность, целостность и сохранность данных должны контролироваться на месте. Это гарантирует, что выполнение чисто локальной операции в распределенной системе не вызывает никаких затруднений.

Правило 2: **Независимость локальных компонент от центрального компонента**
Это означает, что каждый узел "равноправен". Если бы локальные компоненты подчинялись централизованной системе, это повлияло бы отрицательно на систему в целом в случае аварии.

Правило 3: **Безостановочная работа**
Если отдельный узел отключается, это не должно препятствовать работе системы.

Правило 4: **Локальная прозрачность**
Пользователям нет необходимости знать, где хранятся данные.

Правило 5: **Прозрачность фрагментации**
Чтобы повысить эффективность системы, отношения разбиваются на фрагменты. Пользователи продолжают работать так, как будто наборы данных не разделены.

Правило 6: **Прозрачность копирования**
Копии идентичных данных могут быть размещены в различных узлах. Пользователям не требуется знать о существовании копий данных.

Правило 7: **Распределенное выполнение операций баз данных**
Для распределенного выполнения операций баз данных используются оптимизаторы.

Правило 8: **Распределенное управление транзакцией**
В распределенной системе каждая отдельная транзакция может включать процессы корректировки в других узлах. Если корректировка не завершена, операция отвергается.

Правило 9: **Прозрачность оборудования**
Система управления базами данных должна поддерживать различные системы оборудования.

Правило 10: **Прозрачность операционных систем**
СУБД должна поддерживать различные операционные системы.

Правило 11: **Прозрачность сети**
СУБД должна поддерживать различные сетевые системы.

Правило 12: **Прозрачность баз данных**
Распределенное использование должно быть возможно для различных локальных СУБД, включая использование мостов между различными базами данных.

На рис. 144 иллюстрируется информационная модель, разработанная в соответствии с этими правилами. Используется терминология спецификации дизайна организационного аспекта (сетевая топология) и аспекта данных (реляционная схема); см. *Jablonski, Datenverwaltung in verteilten Systemen 1990, p. 198.*

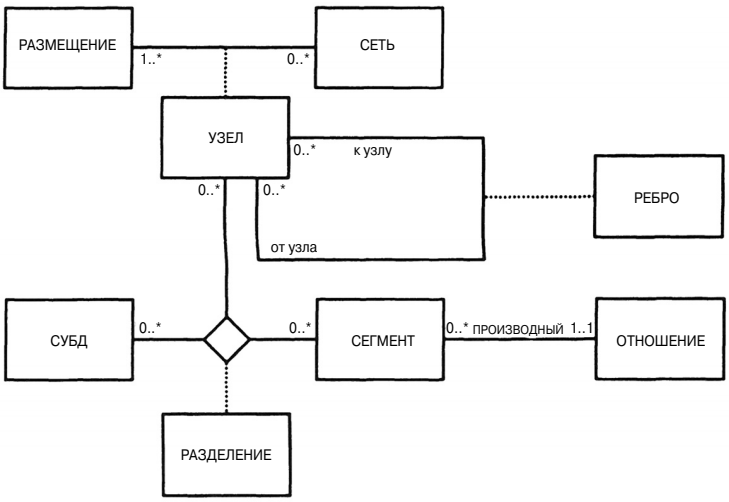


Рис. 144. Распределенные базы данных

Фрагментирование означает, что отношения разделяются горизонтально или вертикально, при этом ключ базового отношения передается каждому фрагменту. Фрагменты отношения могут пересекаться (рис. 145). Фрагменты иллюстрируются на рис. 144 с помощью класса СЕГМЕНТ, где каждый сегмент однозначно соответствует отношению. Соответствие сегмента отдельному узлу или СУБД иллюстрируется с помощью класса РАЗДЕЛЕНИЕ (см. *Jablonski, Datenverwaltung in verteilten Systemen 1990, p. 193*).

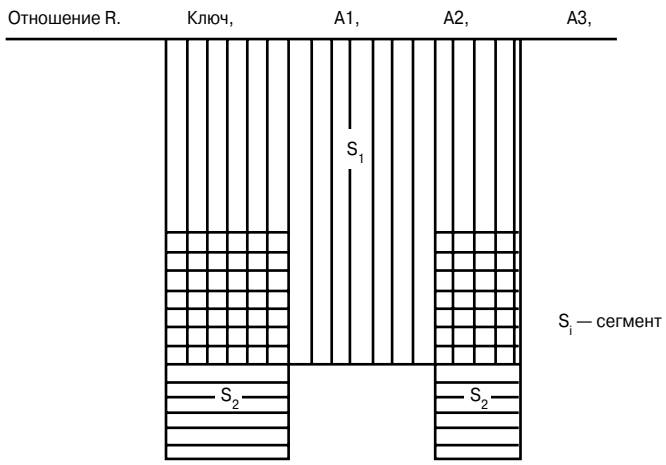


Рис. 145. Сегментация отношения

Система “клиент–сервер” представляет собой упрощенную систему распределенных баз данных без условий прозрачности. Клиенты отправляют свои запросы на сервер базы данных, где запросы удовлетворяются с помощью внутренних операций управления данными. Установление соответствия отношений и сервера базы данных — это составная часть метамодели. На этапе внедрения устанавливается соответствие данных и физических запоминающих устройств. Администрирование прозрачных распределенных систем сталкивается с проблемами координации, и современные СУБД находятся на начальной стадии разрешения этих проблем.

A.III.5 Взаимосвязи организации и результата

На рис. 146 иллюстрируется место взаимосвязей организационного аспекта и результата в концепции ARIS. Рассматриваемые аспекты представлены в виде затененных фигур.

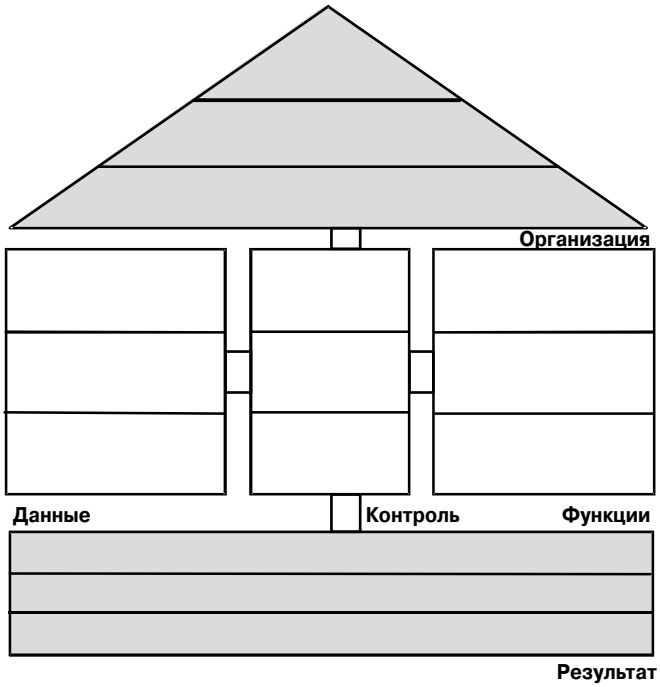


Рис. 136. Взаимосвязи организации и результата в концепции ARIS

A.III.5.1 Моделирование описания требований

В соответствии с распределением функций и данных по организационным единицам предприятия на рис. 147 иллюстрируется взаимосвязь организационной структуры и результата. Например, результат включает базовые стандарты для контроллинга, ориентированного на результат. Некоторые результаты — такие как контакты торговых агентов с целью продажи с учетом принятых предложений — производятся в рамках одной организационной единицы, а некоторые — такие как предложения, переданные в главный офис для дальнейшей обработки, — передаются другим организационным единицам. Однако промежуточные продукты также представляют собой результат конкретной функции организационной единицы, необходимый для выполнения следующей функции той же организационной единицы. Здесь рассматриваются различные уровни организационных единиц (рабочие группы, рабочие места). Количество промежуточных продуктов зависит от глубины детализации организационного аспекта.

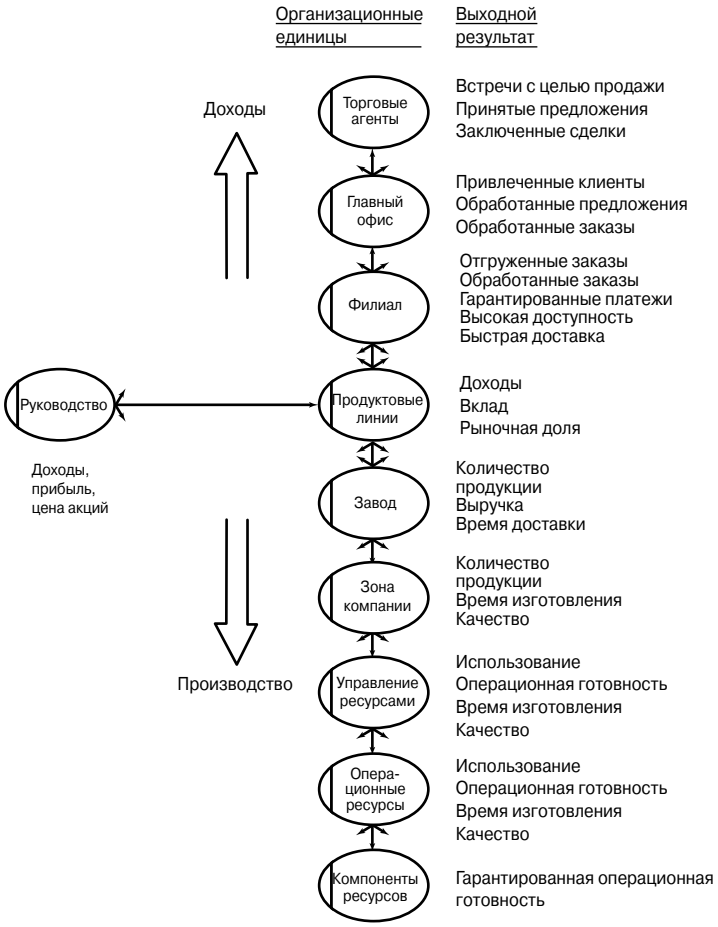


Рис. 147. Модель организационного распределения результата

Часто моделируется только поток результата организационных единиц, полученный от других организационных единиц. На рис. 148 иллюстрируется детализированный выходной поток между организационными единицами “торговый агент” и “главный офис”. Показан также выходной поток для пользователя.

Внешние актеры, такие как потребители, поставщики, правительство и т.п., могут также быть определены как организационные единицы. Диаграммы взаимодействия используются для моделирования предварительных выходных потоков.

Одна из особенностей результата состоит в том, что получатель должен быть готов заплатить за него требуемую цену, т.е. должен знать его стоимость.

В теории бизнеса выходные потоки между организационными единицами имеют важное значение при расчете себестоимости. Показатели выпуска, известные как ссылочные значения, определяются для планирования затрат затратного центра. Простой пример вычисления стоимости выходного результата компании иллюстрируется на рис. 149. Нормы затрат определяются для ссылочных значений.

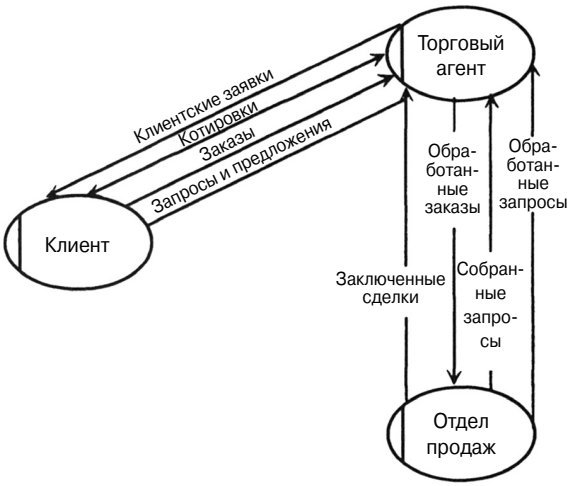


Рис. 148. Выходной поток между организационными единицами

Если затратный центр использует продукцию других затратных центров, то для вычисления себестоимости требуется оценить выходной результат, полученный другими центрами затрат. Это затрудняет определение норм затрат при вычислении стоимости выходного результата компании.

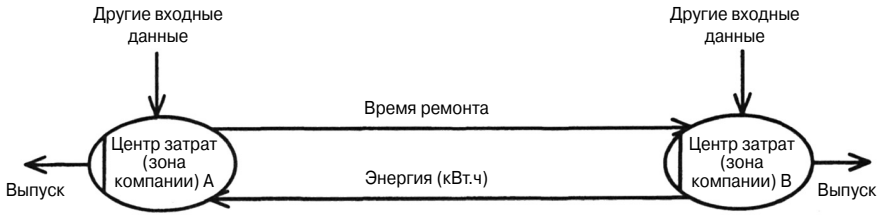


Рис. 149. Вычисление стоимости выходного результата компании

На рис. 150 иллюстрируется метамодель выходного потока между организационными единицами.

Ассоциация ПОСТАВКА иллюстрирует взаимосвязи выходного результата и организационных единиц. Ассоциация ПОСТАВКА–ВРЕМЯ содержит конкретные взаимосвязи во времени с учетом количественных и стоимостных характеристик.

A.III.5.2 Конфигурация

Качественные и количественные характеристики выходного результата определенной организационной единицы имеют ключевое значение для планирования и контроля. Любые отклонения фактических значений от плановых можно использовать для корректировки целевых значений стоимостных пока-

зателей, поэтому выходной поток, соответствующий организационной единице, конфигурирует системы вычисления себестоимости и выпуска.

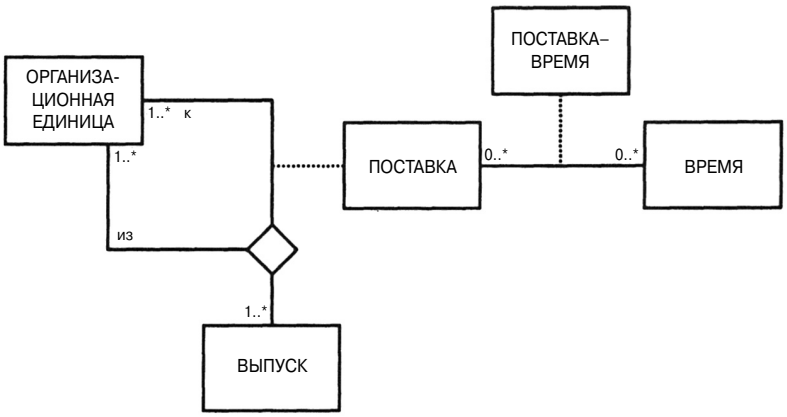


Рис. 150. Метамодель выходного потока

Выходной поток организационной единицы определяется непосредственно из контекста модели, что позволяет детализировать организационные единицы до уровня конкретного рабочего места. В сфере информационных услуг определение выпуска соответствует определению статуса передаваемых массивов данных.

Проще говоря, выходной поток организационной единицы определяет требуемые функции. Например, разные функции необходимы в случае, когда одна организационная единица обрабатывает только простые заказы, а другая — как простые, так и сложные заказы.

Вместо прямого назначения функций организационным единицам можно поставить функции в соответствие выходному результату, если известны взаимосвязи между выпуском и функциями.

A.III.6 Взаимосвязи данных и результата

На рис. 151 иллюстрируется место взаимосвязей данных и результата в концепции ARIS. Рассматриваемые аспекты представлены в виде затененных фигур.

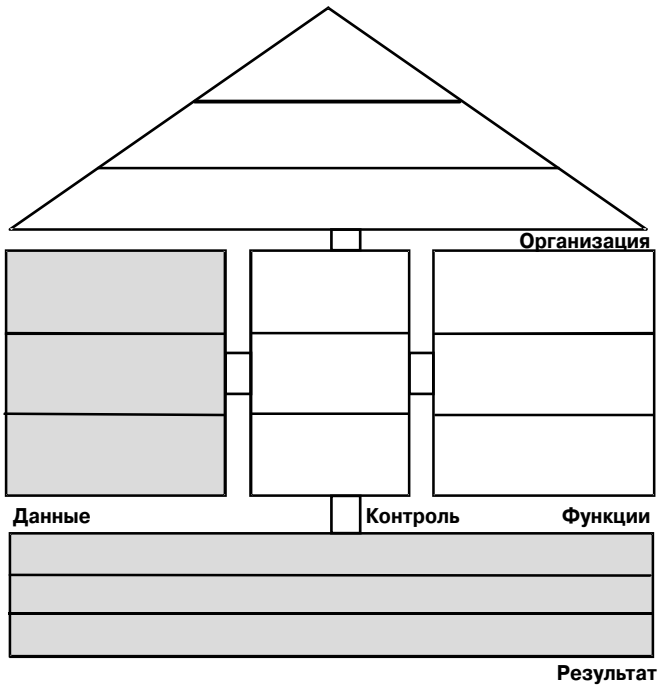


Рис. 151. Взаимосвязи данных и результата в концепции ARIS

A.III.6.1 Моделирование описания требований

Выходной результат и данные связаны уже тем фактом, что информационные услуги представляют собой данные. На рис. 152 информационные услуги (результат выполнения информационных функций) связаны с процессом ввода и рассмотрения данных заказа. Выходной результат включает отметку о статусе данных. При вводе данных заказа их статус ("введены") определяется тем, что соответствующие данные фактически существуют, но отметка об их рассмотрении ("введены и рассмотрены") возможна только в случае, если инспектор проверил данные и отметил их.

Чтобы документировать оба вида информации, в объекте данных "Заказ" создается атрибут "Статус", содержащий отметку о статусе заказа и, следовательно, статусе результата. Материальный продукт и другие услуги представляются собственными материальными или нематериальными моделями, хотя они тоже описываются в информационных системах с помощью объектов данных. Например, материальный продукт документируется с помощью ведомости материалов или чертежей компьютерного дизайна, а услуги отмечаются специ-

фикациями функций или другими документами. Используются также мультимедийные описания продукта, такие как видеофильмы.

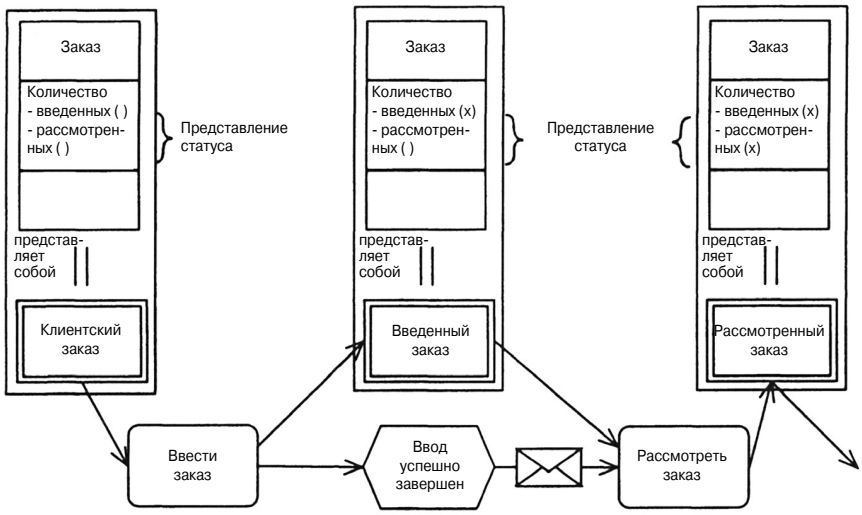


Рис. 152. Взаимосвязь данных и информационных услуг

Сырье и материалы, обрабатываемые в промышленном производстве, существуют как материальный продукт, но им также соответствует информационный объект “Деталь” (рис. 153). Отметка о статусе присутствует в составе этого информационного объекта так же, как и в случае информационных услуг. Единственное отличие заключается в том, что для материального продукта и других услуг объекты данных описывают выходной результат, а для информационных услуг объекты описания и объекты выходного результата могут совпадать.

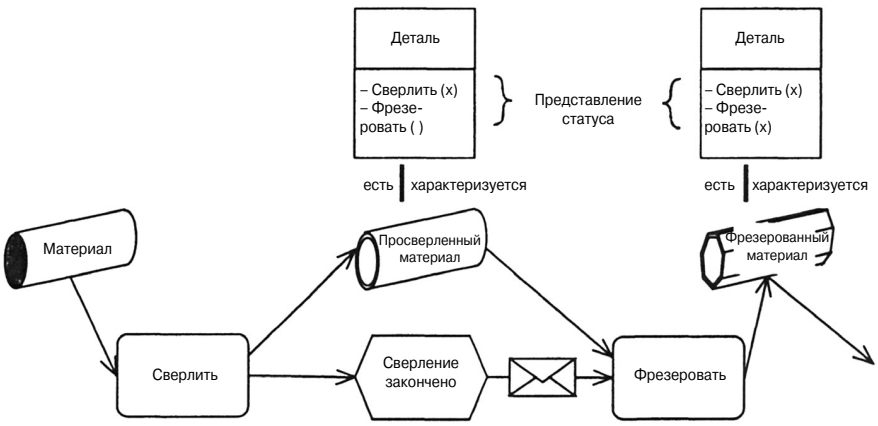


Рис. 153. Взаимосвязь данных и материального продукта

В метамодели отсутствует различие между материальным продуктом и информационными услугами (рис. 154).

Рис. 154. Метамодель взаимосвязи данных и результата

Поскольку выходной результат определен как самостоятельный класс, его статус устанавливается с помощью ассоциации СТАТУС между классами РЕЗУЛЬТАТ и ФУНКЦИЯ. Таким образом, выходной результат может иметь несколько статусов обработки, т.е. не требуется новое определение результата после выполнения каждой функции.

Различные значения атрибутов характеризуют статус обработки, что иллюстрируется классом ассоциаций ПРЕДСТАВЛЕНИЕ СТАТУСА. Представление статуса может иметь вид описания или знака равенства. Результат выражается комбинацией атрибутов многих объектов данных.

На рис. 155 иллюстрируется два вида взаимосвязи данных и результата: СОВПАДЕНИЕ и ХАРАКТЕРИСТИКА. Объекты данных имеют здесь различные роли. Например, проекты компьютерного дизайна в процессе разработки продукта в точности представляют собой результат, т.е. результат и объект данных совпадают. В промышленном производстве чертежи изделия описывают материальный продукт, т.е. характеризуют результат.

A.III.6.2 Конфигурация

Учитывая, что выходной результат и данные иногда пересекаются, различные опции конфигурации рассматриваются только в общих чертах.

Для проверки выходного результата характеристики данных можно прямо поставить в соответствие результату и передать в систему контроля.

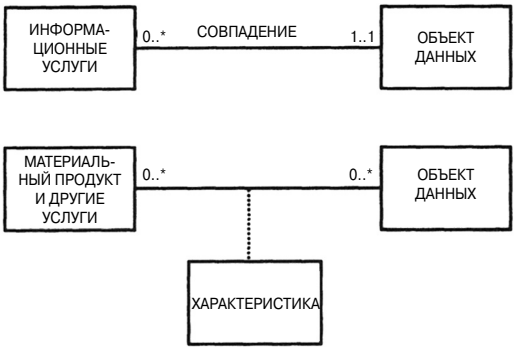


Рис. 155. Общая взаимосвязь данных и результата

Обратная связь от выходного потока конфигурируется с помощью определений данных.

Установление взаимосвязи входных данных и выходного результата позволяет определить необходимые данные (ведомости материалов, результаты контроля качества и т.п.).

A.III.7 Уточненные модели всех аспектов ARIS

Подробно рассматривая все двусторонние взаимосвязи аспектов концепции ARIS, мы обсудили ключевые вопросы моделирования, конфигурации и внедрения бизнес-процессов. В заключение сделаем ударение на усовершенствовании моделей всех аспектов ARIS и не будем детально останавливаться на вопросах внедрения.

A.III.7.1 Моделирование описания требований

По существу, любой аспект ARIS (функции, организация, данные, контроль), вокруг которого монтируются элементы остальных аспектов, может быть центральной точкой обсуждения. Процессы и объекты — вот два базовых аспекта, которые будут находиться в фокусе усовершенствованных моделей.

A.III.7.1.1 Модели процессов

Диаграмма цепочки процесса, приведенная на рис. 12, *а* и 12, *б* для стратегического планирования, иллюстрирует бизнес-процесс в виде таблицы. Каждая линия в диаграмме цепочки процесса показывает, как функция процесса инициируется событием, какие объекты данных обрабатываются с помощью каких атрибутов, какие организационные единицы участвуют в процессе и какой продукт используется или создается, соответственно. Более того, можно установить, какое бизнес-приложение используется для выполнения функций.

На рис. 156 иллюстрируется первичная метамодель диаграммы цепочки процесса, где функция находится в фокусе процедуры.

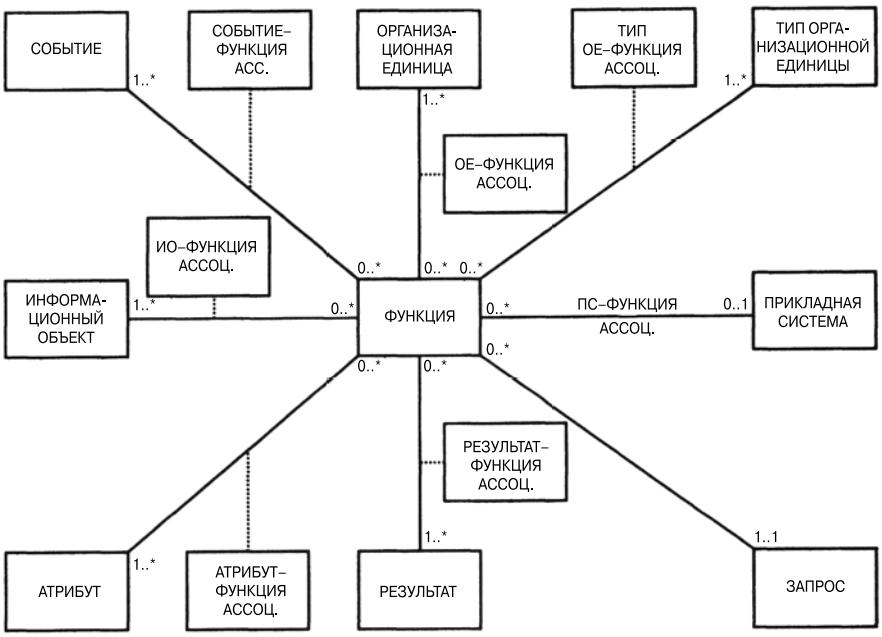


Рис. 156. Метамодель диаграммы цепочки процесса

Вместо таблиц интегрированные бизнес-процессы можно моделировать в свободном стиле, рассматривая событийно-управляемые цепочки процесса в каждом аспекте ARIS. Это соответствует иллюстрации общей модели бизнес-процесса ARIS (см. *Scheer, ARIS — Business Process Frameworks 1998, p. 31*), где метаструктура соответствует диаграмме цепочки процесса (иллюстрируется одна и та же ситуация, хотя представляется по-разному). Обе иллюстрации могут быть преобразованы друг в друга.

A.III.7.1.2 Бизнес-объекты

Модели бизнес-объектов также уточняются. Хотя мы первоначально определили бизнес-объекты как сложные объекты данных или как связи между данными и функциями, бизнес-объекты также включают потоки выходных результатов, организационные потоки и потоки управления.

Следовательно, модель бизнес-объекта включает модели этих элементов (рис. 157). Доступ к методам бизнес-объекта осуществляется с помощью программного интерфейса BAPI (Business Application Programming Interface) в SAP R/3. На рис. 158 иллюстрируется метамодель бизнес-объекта.

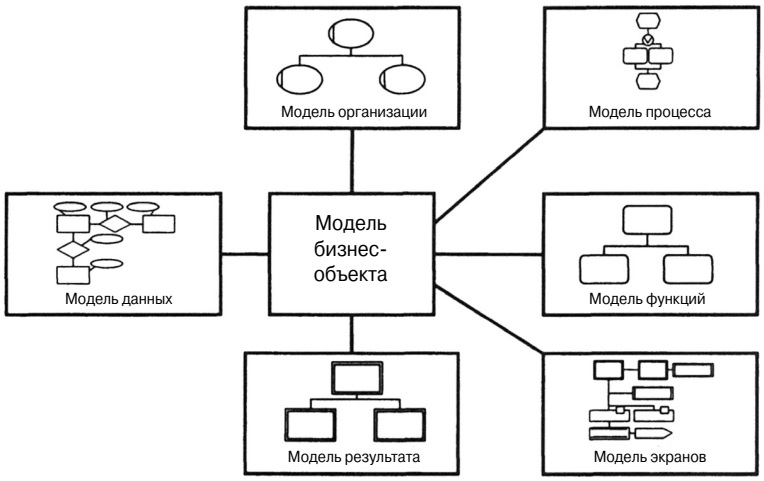


Рис. 157. Модель бизнес-объекта

Чтобы обеспечить внутреннее управление процессом, модели данных, методов и экранов, относящихся к бизнес-объекту, связаны с моделью процесса. Продукт, требуемый или создаваемый бизнес-объектом, иллюстрируется моделью выходного результата.

Приложение состоит из многих бизнес-объектов, управляемых бизнес-процессом, определенным конкретно для этого приложения (рис. 159).

Несколько бизнес-объектов могут быть необходимы для выполнения одной бизнес-функции, или для выполнения нескольких бизнес-функций требуется один бизнес-объект. Бизнес-объекты могут инициировать друг друга без привязки к бизнес-функции; этот поток сообщений обозначен на рис. 159 пунктирной линией.

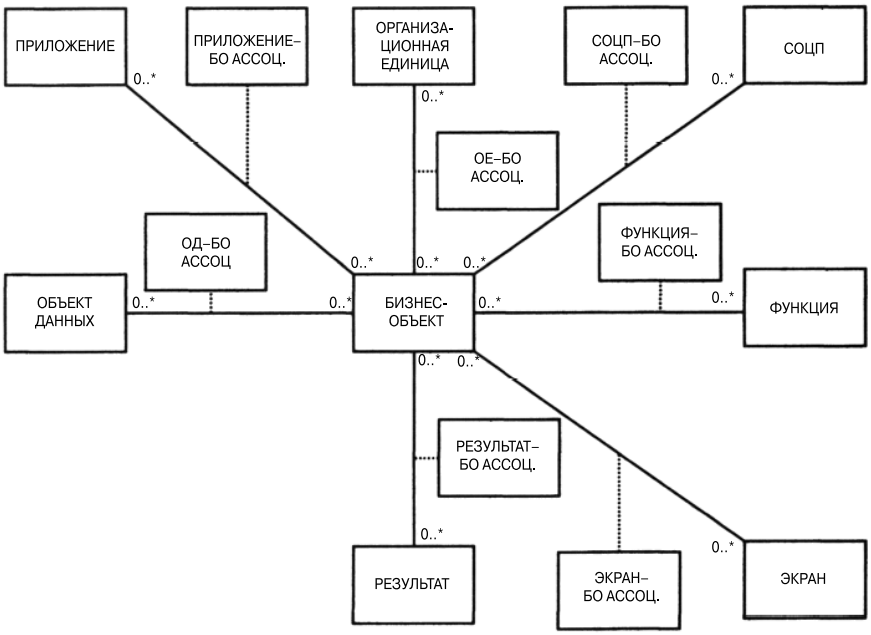


Рис. 158. Метамодель бизнес-объекта

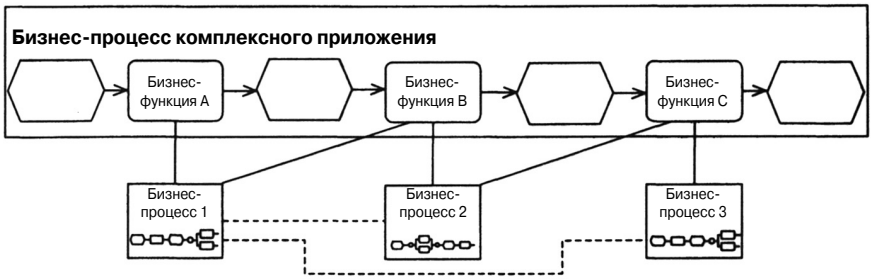


Рис. 159. Бизнес-объекты в составе приложения

A.III.7.2 Конфигурация

При объединении всех аспектов ARIS конфигурированные приложения и модульные приложения интегрируются.

A.III.7.2.1 Использование моделей бизнес-процессов для конфигурации

На рис. 160 иллюстрируются различные варианты конфигурации, возможные в рамках модели бизнес-процесса ARIS (см. *IDS, ARIS-Framework 1997*).

Часть *a* иллюстрирует фрагмент существующей процессной цепочки приложения для процесса закупки или продажи. Будучи инициированы определенным событием, оба метода функции корректировки данных по закупкам или

продажам работают. В прототипе ARIS Framework внедряет эту модель процесса в рабочую систему.

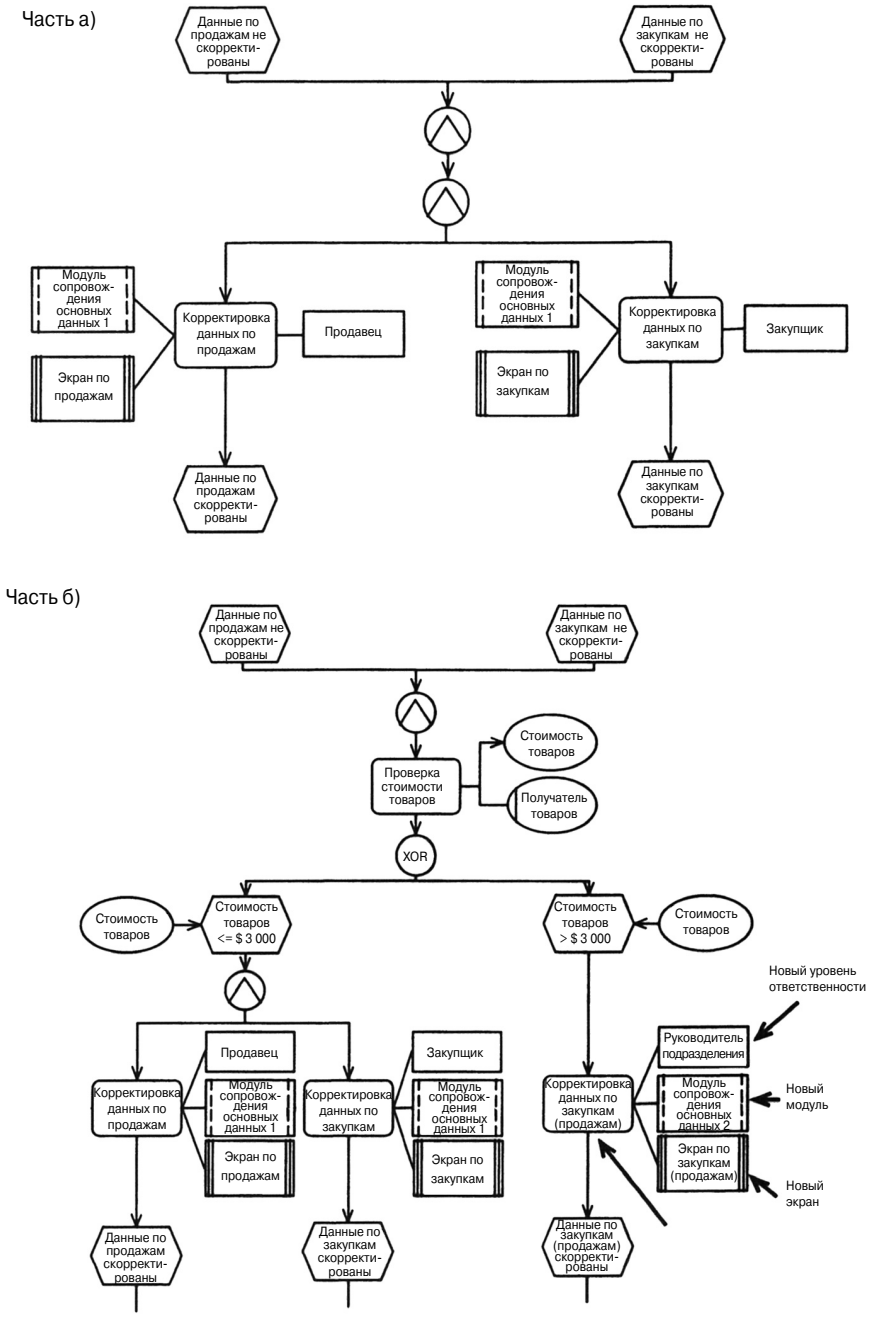


Рис. 160. Варианты конфигурации бизнес-процессов

Часть *б* показывает, как пользователи могут изменять конфигурацию системы с помощью корректировки модели бизнес-процесса. Добавление в модель функции "Проверка стоимости товаров" означает добавление соответствующих функциональных модулей в программную систему.

Используя атрибут "стоимость товаров", организационное подразделение ("получатель товаров") проверяет данные, после чего информация интерпретируется workflow-системой. Если стоимость товаров превышает заданный предел, руководитель подразделения выполняет функцию "Корректировка данных по закупкам (продажам)". Эта функция объединяет две функции: "Корректировка данных по закупкам" и "Корректировка данных по продажам". Новая ветка добавляется в модель бизнес-процесса и интерпретируется с помощью ARIS Framework и системы управления потоком. Новый модуль из дерева модулей выполняет функцию "Корректировка данных по закупкам (продажам)".

Новой функции "Корректировка данных по закупкам (продажам)" ставится в соответствие новый экран. Скорректированная модель бизнес-процесса внедряется в рабочие системы с помощью ARIS Framework и workflow-системы ARIS.

Демонстрируя образец непрерывного усовершенствования процессов при адаптировании систем, этот пример включает такие составляющие конфигурации:

- модификация процессов;
- добавление функций;
- интеграция функций;
- корректировка организационной структуры;
- корректировка экранов.

A.III.7.2.2 Конфигурирование бизнес-объектов

Если бизнес-объекты определены как стартовая точка конфигурации, мы можем сослаться на рис. 157.

На рис. 161 иллюстрируется изменение элементов стандартной модели бизнес-объекта в соответствии с конкретными требованиями клиента.

A.III.7.3 Спецификация дизайна

Если при описании требований модели бизнес-процессов и бизнес-объектов были равноправны (иногда предпочтение даже отдавалось моделям бизнес-процессов), то на этапах конфигурации и спецификации дизайна модели бизнес-объектов будут доминировать. Это способствует достижению таких целей, как возможность многократного использования программ и надежность системы.

Чтобы привести систему в соответствие со спецификацией дизайна, такие бизнес-элементы, как функция, организационная единица, объект данных, выходной результат и т.п., трансформируются в такие элементы, как модуль, узел, отношение и объект результата.

Промежуточные интерфейсы, "посредники" между программами приложения и аппаратными средствами, также относятся к бизнес-объекту.

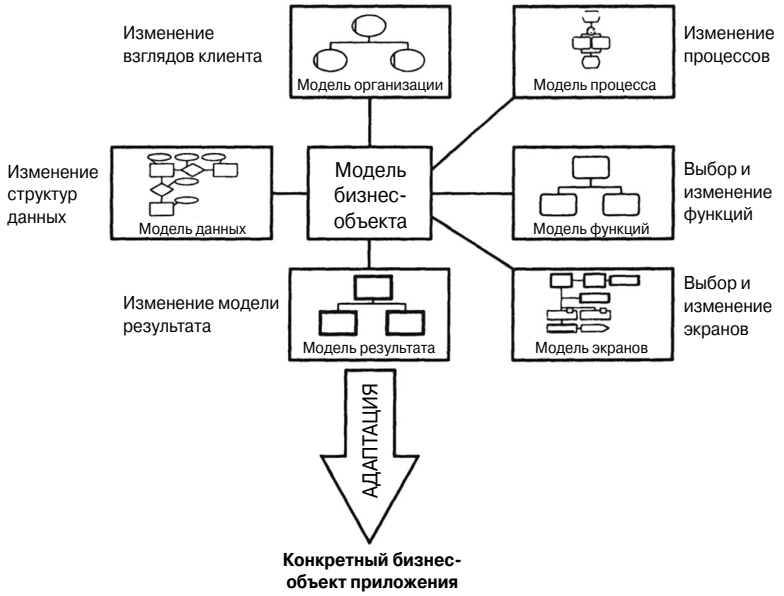


Рис. 161. Модель бизнес-объекта с учетом требований клиента

Особую важность имеют коммуникационные интерфейсы, например, CORBA, COM/DCOM, RMI, предусмотренные отдельными бизнес-объектами для связи с другими бизнес-объектами (рис. 162).

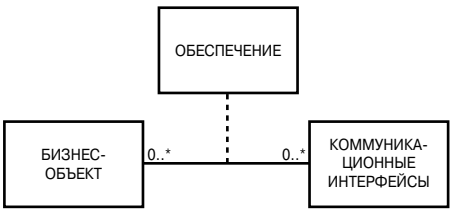


Рис. 162. Коммуникационные интерфейсы бизнес-объектов

Различные элементы объединяются и формируют бизнес-объект согласно спецификации дизайна.

Бизнес-объект теперь содержит все компоненты, необходимые для выполнения отчетов, ведения документов, определения и хранения данных или выполнения в диалоговом режиме таких функций, как калькуляция цен или проверка наличия. Совместно с другими бизнес-объектами он может образовать сложные бизнес-приложения.

B

Процедуральные модели ARIS и избранные приложения

Перейдем к рассмотрению процедур практического использования моделей ARIS для избранных приложений. Остановимся на таких темах:

- внедрение стандартных бизнес-приложений с использованием моделей ARIS;
- внедрение систем управления потоком;
- использование ARIS Framework для разработки приложений;
- моделирование с помощью UML.

Авторы этой главы — признанные эксперты в названных сферах.

Другие примеры использования ARIS в таких сферах, как реинжиниринг бизнес-процессов, сертификация качества в соответствии с ISO 9000, а также управление знаниями, рассмотрены в книге *Scheer, ARIS — Business Process Frameworks 1998*.

B.I

Внедрение стандартных бизнес-приложений с использованием моделей ARIS

Доктор Петер Маттейс (Peter Mattheis), доктор Вольфрам Иост (Wolfram Jost); *IDS Prof. Scheer GmbH*, г. Саарбрюкен, Германия.

B.I.1 Решение критических вопросов, связанных с внедрением стандартного приложения

Даже в настоящее время внедрение стандартных бизнес-приложений на предприятиях связано с рядом затруднений:

- внедрение требует много времени и других затрат;
- сроки и стоимость внедрения нереальны и непредсказуемы;
- передача данных связана с рисками;
- качество процесса внедрения не гарантируется;
- стандарты и контроль неприменимы;
- накопленный опыт не используется.

Для решения перечисленных вопросов требуется структурированный, методический подход. Ключ к успеху лежит в использовании накопленного опыта в разработке соответствующих процедур, моделировании конкретных вертикальных рыночных процессов и оценке системных параметров.

Часто стандартное приложение внедряется как конкретное решение, ориентированное на запросы клиента; при этом уникальность запросов клиента переоценивается. Полученные результаты внедрения, к сожалению, недостаточно прозрачны, поэтому не анализируются общие характеристики различных законченных проектов.

Однако критические вопросы, связанные с внедрениием стандартного приложения, могут быть успешно решены с использованием таких приемов:

- фокусирование на вертикальных рынках, использование опыта внедрения завершенных проектов;
- ориентация на бизнес-процесс, использование знаний о его моделировании в стандартном приложении;
- использование типовых приложений для соответствующего вертикального рынка;
- использование инструментов и накопленного опыта в разработке процедур, моделировании конкретных рыночных процессов и оценке системных параметров.

Рассмотрим примеры внедрения стандартного бизнес-приложения с использованием различных моделей ARIS, иллюстрируя преимущества использования ARIS Toolset в этих проектах.

B.I.2 ARIS Quickstep for R/3

Рассмотрим процедуру внедрения бизнес-приложения, фокусируясь на R/3. Процессы внедрения основываются на модели ARIS Quickstep for R/3 (рис. 163), которая состоит из трех компонент:

- процедуральная модель;
- приложения R/3, типовые для соответствующего вертикального рынка;
- документация типовых приложений R/3 с использованием EPC.

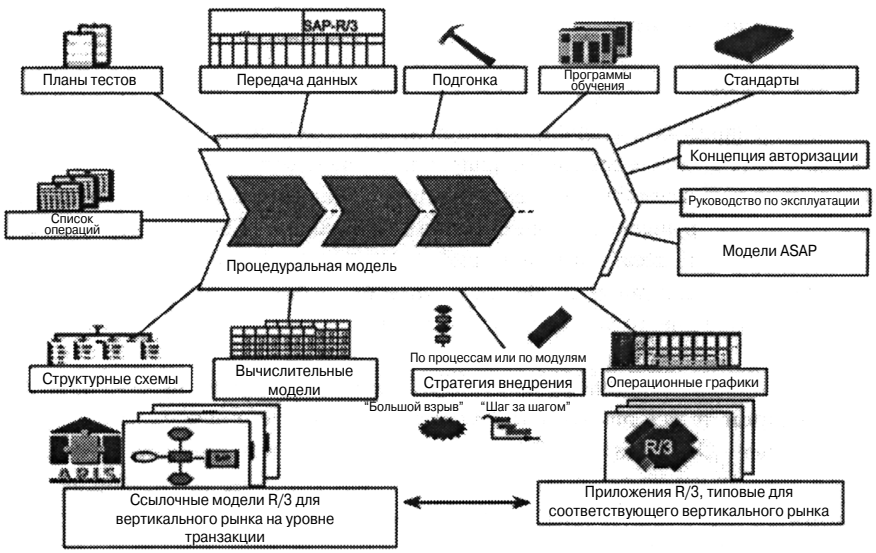


Рис. 163. Компоненты ARIS Quickstep for R/3

Модель ARIS Quickstep for R/3, многократно проверенная в различных проектах внедрения R/3, разработана *IDS Prof. Scheer*.

На рис. 164 иллюстрируются четыре фазы процедуры ARIS Quickstep for R/3. Модель позволяет сократить время внедрения и его стоимость, повысить надежность и качество проекта.

Рис. 164. Процедура ARIS Quickstep for R/3

Описание ARIS Quickstep for R/3 основано на моделях ARIS. Четыре фазы изображены как диаграммы цепочек добавленной стоимости. Каждая фаза детально описана как событийно-управляемая цепочка данных. Диаграммы ввода и вывода описывают входные и выходные данные функций. Этот метод облегчает планирование и понимание проекта, обеспечивает высокую степень прозрачности реализации проекта, позволяет отслеживать эффективность процесса внедрения, оценить влияние любой задержки на срок исполнения и своевременно принять соответствующие меры.

Quickstep for R/3 согласовывается с SAP ASAP (Accelerated SAP) — процедуральной моделью SAP AG, — так как они предназначены для одинаковых целей и предлагают подобные подходы к реализации проекта. Как известно, R/3 уже используется для многих конкретных вертикальных рынков, что служит дополнительным стимулом для использования модели Quickstep for R/3.

B.I.3 ARIS Quickstep for R/3: фазы внедрения SAP

Предложение (заказ)

В общем случае детальный анализ фактической ситуации опускается, хотя существующие бизнес-процессы могут быть проанализированы по желанию клиента. Если клиент заинтересован в получении количественных оценок фактической ситуации (таких как длительность производственного цикла, себестоимость процессов, своевременность выполнения и т.п.), то анализ бизнес-процессов необходим. Используются методы EPC, что обеспечивает четкость

и наглядность даже при моделировании больших сетей. Описание существующей ситуации включает предварительное описание процесса, с одной или двумя степенями детализации.

Клиентские требования относительно внедрения R/3 запрашиваются в ходе совещаний и консультаций и обобщаются. Quickstep for R/3 обеспечивает также процессно-ориентированные вычислительные модели, структурные схемы, списки видов деятельности и операционные графики согласно стратегии внедрения. Это существенно повышает эффективность и качество планирования проекта.

Описания проекта необходимы для того, чтобы оценить затраты на внедрение, учитывая дополнительные расходы, необходимые для нового применения стандартного приложения с учетом требований клиента. Рекомендуется следующая процедура.

Совместно с руководством предприятия разработайте общую концепцию, ориентированную на бизнес-процессы. Например, решите, в какой мере управление цепочкой поставок должно быть внедрено на практике и какие современные прогрессивные технологии (такие как экстрасети с партнерами по бизнесу) будут использованы.

Опираясь на общую концепцию, разработайте продуктовую модель, чтобы определить продукты и услуги предприятия. Эта модель представляет основные характеристики продуктов и услуг, что имеет ключевое значение для принятия покупательских решений.

Модель рынка описывает маркетинговые кампании, каналы продаж и ожидания рынка относительно продуктов и услуг предприятия.

Продуктовая модель и модель рынка должны быть согласованы. Обе модели создаются с помощью ARIS Toolset, что позволяет пользователю отчетливо видеть взаимосвязи между отдельными компонентами, анализировать их и добавлять комментарии.

Цепочки добавленной стоимости, необходимые для разработки и продажи соответствующих продуктов, определяются в соответствии с продуктовой стратегией. Вспомогательные функции, такие как управление персоналом или финансы, рассматриваются отдельно от процессов, которые фактически создают добавленную стоимость. Чтобы обеспечить достаточную наглядность, цепочки добавленной стоимости должны содержать от 30 до 50 элементов, которые представляют стратегическое направление внедрения процессно-ориентированного программного продукта.

Стратегический компонент проекта необязателен. Однако опыт реальных проектов показал, что именно с помощью этого компонента менеджмент может управлять проектом. В значительной степени будущая эффективность проекта зависит от стратегического компонента.

Прототип концептуального дизайна

В начале этой фазы устанавливаются временны*е* рамки и план действий для проекта. Используются операционные графики и списки видов деятельности, входящие в Quickstep for R/3. Сотрудники должны быть проинформированы

о целях проекта и перспективах их достижения, а также о том, какое программное обеспечение необходимо установить и какое оборудование приобрести.

Определение организационной модели R/3 — это еще одна составляющая подготовки к внедрению программного обеспечения. На уровне предприятия необходимо определить перечень корпоративных подразделений, список клиентов, коды компании и коды показателей бухгалтерской отчетности. Следует определить организационные структуры для процессов логистики, продаж и закупок.

Организационная структура запоминается в ARIS Toolset как иерархическая диаграмма, которая сопровождается соответствующими обоснованиями.

Масштаб прототипа зависит от определения соответствующих бизнес-процессов, отражаемых в модулях R/3.

Начиная с цепочки добавленной стоимости, варианты процесса разработаны для таких бизнес-процессов:

- ведение баз данных;
- обработка предложений (заказов);
- закупки;
- производство;
- послепродажное обслуживание.

Например, для обработки заказов существует ряд вариантов, таких как обработка стандартных продуктов; обработка сложных продуктов с использованием продуктовых конфигураторов или обработка сложных продуктов с использованием проектных инструментов. Эти варианты процесса моделируются в ARIS Toolset как процессы. Чтобы уменьшить затраты на разработку и поддержать функциональность R/3, варианты процесса ссылаются на эталонную модель R/3 и модели вертикальных рынков IDS.

Модели вертикальных рынков документируют процессы для приложений R/3, сконфигурированные IDS.

Эти модели должны быть усовершенствованы с учетом потребностей клиентов в соответствующих организационных подразделениях. Для этого требуются дружественные пользователю инструменты, такие как ARIS Easy Design.

Чтобы постепенно обучить команду работе с приложением R/3, коды транзакций хранятся в каждой функции модели процесса. В результате пользователи быстрее осваивают функции и диалоговые экраны, растет их заинтересованность в приобретении нового приложения.

Основываясь на приложениях, сконфигурированных для соответствующего вертикального рынка, прототипы создаются командой сотрудников в тесной кооперации с клиентом.

На рис. 165 иллюстрируется иерархия соответствующих моделей.

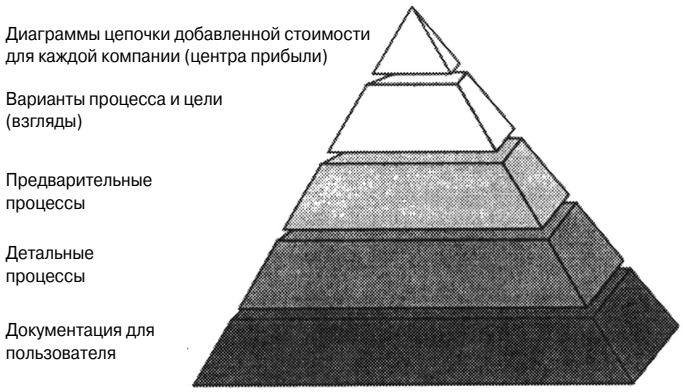


Рис. 165. Иерархия моделей

Внедрение

В процессе внедрения прототипы дорабатываются с использованием данных, определенных на предыдущем этапе. Профайлы авторизации, устанавливающие соответствие между пользователями и функциями, формируются с помощью генератора профайлов.

Первоначально функции приложения тестируются в соответствии с процессами, используя данные, введенные пользователем. После завершения и утверждения отдельных функций выполняются комплексные испытания. Дополнительная проверка выполняется с использованием данных о выпуске продукции, как только эти данные успешно переданы.

Уроки тренинга для пользователей проводятся одновременно с комплексными испытаниями.

Модели процессов детализируются до уровня операции, что создает условия для оформления пользовательской документации. В ходе моделирования линейных процессов конкретные инструкции генерируются для исполнителей, помогая им выполнять их текущие задачи.

Непрерывное усовершенствование

После практического внедрения приложения следует фаза оптимизации, в течение которой приложение и процессы снова проверяются. В фазе оптимизации модели, разработанные на предыдущих этапах, оптимизируются. Если на предыдущих этапах были вычислены показатели стоимости или времени выполнения операций, появляется возможность проанализировать динамику этих показателей и количественно оценить степень усовершенствования процессов.

B.I.4 Выводы

Предложенный метод внедрения стандартного бизнес-приложения с использованием моделей ARIS обладает следующими характеристиками.

1. **Ориентация на стратегию.** Услуги и продукты предприятия описаны в продуктовой модели. Требования рынка отображены в модели рынка. Цепочки добавленной стоимости определяются в соответствии с продуктовой стратегией. Внедрение программного обеспечения опирается на цепочки добавленной стоимости, которые осуществляют связь с корпоративной стратегией.
2. **Эффективность.** Используются модели вертикального рынка, реализованные в приложениях R/3, созданных для конкретной отрасли. Конкретные процессы вертикального рынка документируются в среде ARIS Toolset.
3. **Ориентация на методы и модели.** Используется ARIS Toolset и соответствующие методы и модели.
4. **Стандартность, прозрачность и наглядность.** Процедуральная модель ARIS Quickstep for R/3 позволяет сократить время внедрения и его стоимость, повысить надежность и качество проекта.
5. **Ориентация на процессы.** Организуются процессно-ориентированные тренинг-классы, семинары. Иерахия процессов детализируется от цепочек добавленной стоимости до кодов операций.
6. **Ориентация на вертикальный рынок.** Используется опыт конкретной отрасли.

B.II Внедрение систем управления потоком с использованием моделей ARIS

Доктор Андреас Кронц (Andreas Kronz), Dipl.-Inform; *IDS Prof. Scheer GmbH*, г. Саарбрюкен, Германия.

B.II.1 Факторы успешного внедрения систем управления потоком

В соответствии с концепцией ARIS в *IDS Prof. Scheer GmbH* разработана система управления потоком, которая использует методы моделирования ARIS, подчиняется архитектуре HOBE и интегрируется с ARIS Toolset. Система ARIS Workflow используется преимущественно для создания прототипов, для быстрой разработки выполнимых, процессно-ориентированных приложений на основе моделей бизнес-процессов. ARIS Workflow взаимодействует с другими коммерчески доступными системами управления потоком, такими как WorkFlow (CSE), FlowMark (IBM), WorkParty (SNI), Staffware (Staffware) и Visual WorkFlo (FileNET). В будущем ARIS Workflow будет теснее интегрирован с концепцией ARIS и ARIS Workflow Prototyper.

Структурированные и систематизированные процедуры имеют существенное значение для успешного внедрения систем управления потоком (Workflow Management System — WMS), потому что они уменьшают сложность и устраняют источники ошибок, увеличивая надежность планирования производительности и затрат. Благодаря системному подходу процедуральные модели ARIS для внедрения систем управления потоком обеспечивают эффективное внедрение бизнес-процессов в рамках корпоративных приложений. Как необходимое условие для непрерывного усовершенствования процессов между уровнями I и II, модели ARIS используются для описания вспомогательных процессов и процедуральных моделей — от оптимизации бизнес-процессов до агрегирования текущих данных.

Преимущество WMS состоит в использовании хорошо структурированных и операционно-связанных процессов. К сожалению, внедрение четко структурированных бизнес-процесов в настоящее время еще не соответствует текущей функциональности моделей. Требуются модели, с точностью описывающие

четко структурированные процессы и одновременно поддерживающие концепцию непрерывного усовершенствования.

B.II.2 Процедуральные модели ARIS для внедрения систем управления потоком

Процедуральная модель на рис. 166 делится на восемь фаз и описывает процесс внедрения WMS, от стратегического планирования до производства. Чтобы ускорить процесс внедрения, несколько фаз задачи могут выполняться одновременно. С целью обеспечения качества отдельные фазы могут повторяться.

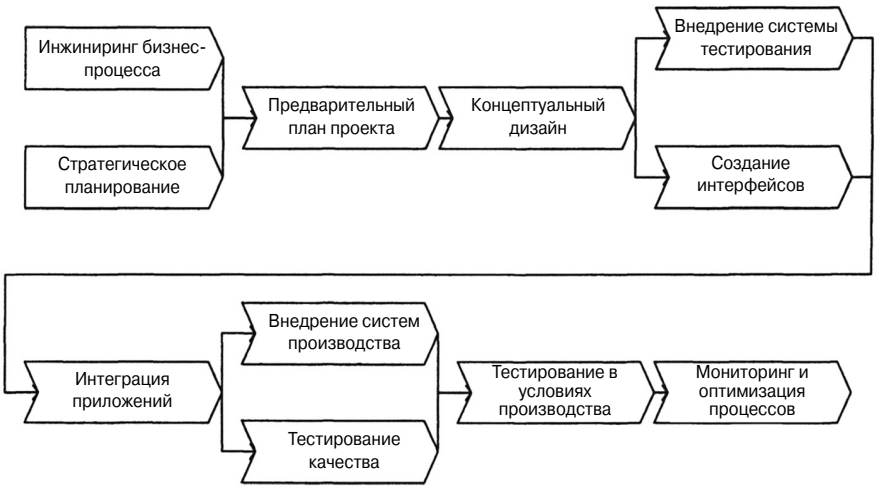


Рис. 166. Процедуральная модель для внедрения систем управления потоком

Процесс внедрения должен сопровождаться качественным обучением персонала. Различные группы пользователей должны изучать разные фазы внедрения. Темы занятий должны охватывать широкий диапазон — от администрирования WMS до внедрения новых бизнес-приложений и оптимизации бизнес-процессов. Чтобы обеспечить успешное внедрение, ответственные подразделения и конкретные сотрудники должны понимать структуру и преимущества бизнес-процессов. Качественная методология и развитый инструментарий играют важную роль в процессе обучения, а также моделирования и мониторинга текущих процессов в WMS. Использование EPC при описании бизнес-процессов обеспечивает прозрачность процессов, особенно когда соглашения о моделировании, включая детализацию связей приложения, отображены в диаграммах распределения функций.

Первая фаза внедрения — это инжиниринг бизнес-процесса как составная часть стратегического планирования предприятия.

Во второй фазе проекта для будущих пользователей проводится презентация системы управления потоком и прототипа бизнес-процесса. Цель презента-

ции — повысить прозрачность системы управления потоком, обосновать выполнимость и определить спецификации функций. Другие цели — сформировать предложение (заказ) для проекта, определить клиентские требования, наметить предварительный план проекта, оценить требования к персоналу, затраты на обучение и общую стоимость проекта. Общие вопросы, такие как численность пользователей, технические требования, необходимое оборудование и т.п., также требуют обсуждения. Один из аспектов анализа выполнимости — прототип для интеграции соответствующего приложения.

Вспомогательные процессы или их предварительная версия должны быть представлены как EPC.

В фазе концептуального дизайна разрабатываются модели выполнимых процессов (рис. 167). Формируются описания организационной структуры (рис. 168), релевантных данных (например, с помощью модели “сущность–связь”), а также процессных цепочек вместе с соответствующими диаграммами распределения функций (рис. 169). Целевые модели бизнес-процесса представляют собой отправную точку этих разработок.

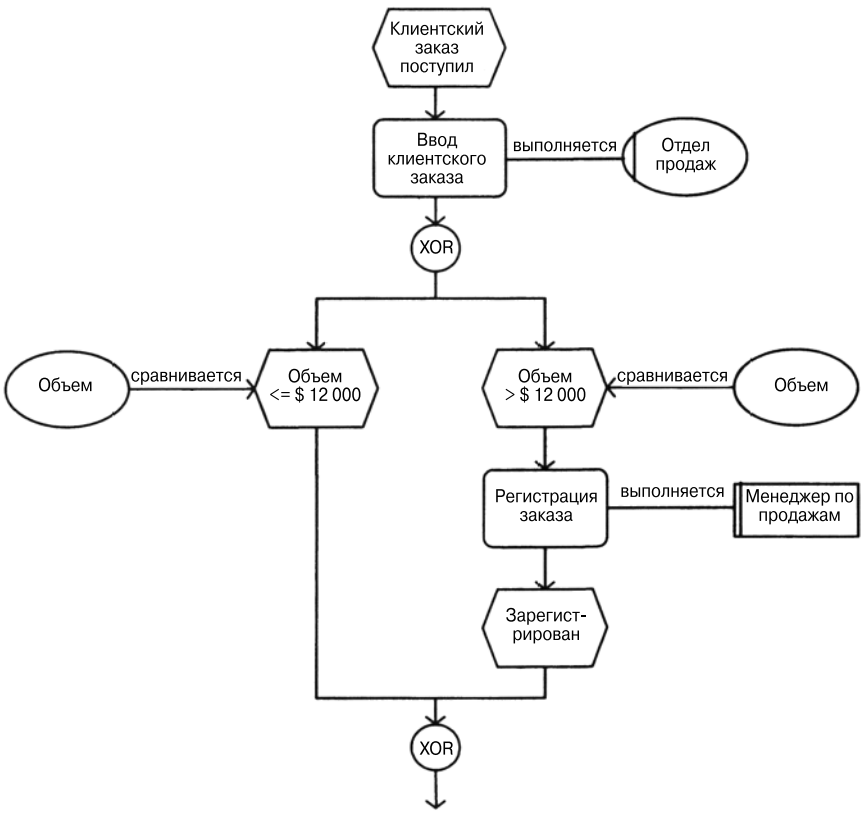


Рис. 167. Фрагмент модели выполнимого процесса

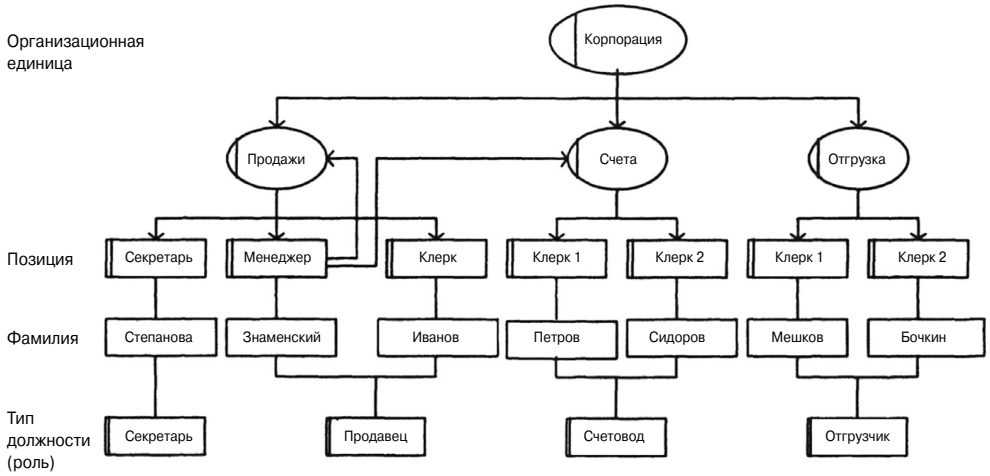


Рис. 168. Фрагмент органиграммы

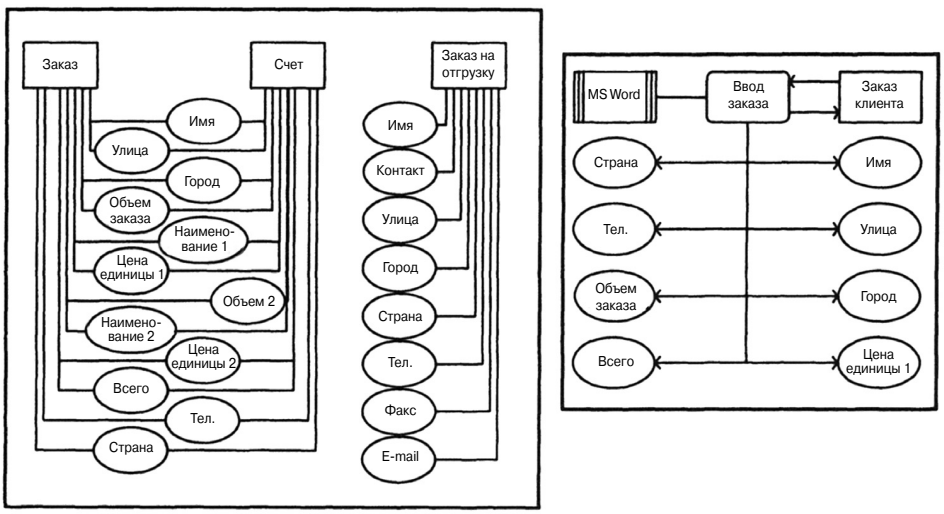


Рис. 169. Фрагмент модели данных и диаграммы распределения функций

Достоверность и состоятельность этих моделей следует проверять с помощью инструментов моделирования.

Дополнительное содержание концептуальной фазы:

- определение ролей и концепции авторизации в той мере, в которой привилегии доступа предусмотрены в системе управления потоком. Привилегии отображаются в органиграммах;
- определение требований к информационной инфраструктуре, начало мероприятий по закупке;
- определение входных (выходных) требований внедряемого приложения;

- определение и тренинг команды для тестирования приложения. Команда состоит из сотрудников и позже будет тестировать систему управления потоком. Следует обучить будущих администраторов системы, чтобы они помогали создать среду для ее тестирования.

Внедрение системы тестирования включает проверку данных, организационных структур и бизнес-процедур соответствующих процессов. Будущие администраторы системы управления потоком должны быть включены в процесс внедрения или даже (под надлежащим руководством) могут сами внедрять систему. Интерфейсы с новыми бизнес-приложениями разрабатываются согласно требованиям и спецификациям.

Приложения интегрируются с соответствующими интерфейсами, после чего модели процессов дополняются конкретными данными и функции проверяются. Затем соответствие WMS проверяется группой тестирования. Любые проблемы на этом этапе могут повлиять на моделирование или интеграцию приложений, что потребует повторения концептуальной фазы.

Внедрение систем производства также включает проектирование концепции эксплуатации, где моделируются авторизация и профайлы пользователей, а обучение пользователей системы управления потоком планируется и выполняется. Заключительный шаг этой фазы — тестирование качества.

В процессе тестирования системы управления потоком реальные сценарии проверяются с использованием достоверных данных. Система управления потоком работает в условиях квази-реалистической загруженности сети и сервера, что дает возможность убедиться в безотказности системы. Надежность концепции безопасности обеспечивается с помощью тестирования функций обратного хода и восстановления. В течение фазы тестирования существующее приложение и новая система управления потоком обычно разрабатываются параллельно, чтобы избежать проблем, которые могут привести к итерациям в процедуральной модели.

Бизнес-процессы контролируются и отслеживаются в ходе практического процесса производства. Наблюдение и моделирование текущих процессов известно как мониторинг (см. *Heß, Monitoring von Geschaftsprozessen 1996*). Это очень важно для последовательного отображения текущих и новых бизнес-процессов с помощью EPC (см. рис. 167). Бизнес-процессы дальше оптимизируются с помощью документирования и вычисления текущих данных, хотя огромное множество входных текущих данных необходимо агрегировать, чтобы получить поддающиеся интерпретации обобщенные данные. По этой причине текущие данные представлены в моделях бизнес-процессов в EPC, где агрегированные данные хранятся как атрибуты (например, среднее время выполнения или частота событий) функций и событий. На втором уровне эти агрегированные модели анализируются, чтобы найти потенциал для их улучшения. Этот процесс в целом известен как цикл непрерывного усовершенствования процессов, в ходе которого процессы моделируются, контролируются, документируются и затем снова моделируются.

Для успешного использования WMS необходимо понимание этого цикла оптимизации сотрудниками. Это обеспечит гибкость WMS, так что бизнес-процессы будут непрерывно адаптироваться и улучшаться.

B.III Использование ARIS Framework для разработки приложений

Саид Имрани (Saeed Emrany), Dipl.-Inform; Ричард Бок (Richard Bock), Dipl.-Inform; *IDS Prof. Scheer GmbH*, г. Саарбрюкен, Германия.

Ранние приложения, ориентированные на бизнес-процессы, было трудно использовать для оптимизации бизнес-процессов. Сегодня программное обеспечение, согласованное с ARIS Framework, предоставляет гораздо больше степеней свободы.

Будучи тесно связанным с ARIS Toolset и ARIS Workflow, ARIS Framework идеально подходит для быстрого создания прототипов бизнес-приложений. Среда ARIS Framework базируется на современной архитектуре клиент–сервер и согласуется с концепцией HOBE. Модели предприятия, структурированные в соответствии с методом ARIS, используются для документации бизнес-процессов и разработки системы управления потоком. Это позволяет развивать и совершенствовать приложения, что, в свою очередь, предоставляет пользователям возможность адаптировать программное обеспечение к изменяющимся условиям и целям.

B.III.1 Общая процедуральная модель

ARIS Framework используется для внедрения и совершенствования приложений, разработанных в соответствии с Framework. Общая процедуральная модель иллюстрируется на рис. 170 в виде событийно-ориентированной процессной цепочки.

Исходные конкретные приложения, ориентированные на пользователя, или создаются с помощью адаптирования и компоновки существующих процессно-ориентированных бизнес-объектов, или разрабатываются с самого начала. Исходные модели, описывающие бизнес-объекты, дорабатываются после выбора объектного метода с учетом конкретных требований пользователей, а затем используются для создания пользовательских приложений с применением ARIS Framework. В процессе доработки бизнес-объекты и соответствующие им процессы модифицируются и затем выполняются с помощью систем управления потоком, интегрированных с ARIS Framework.

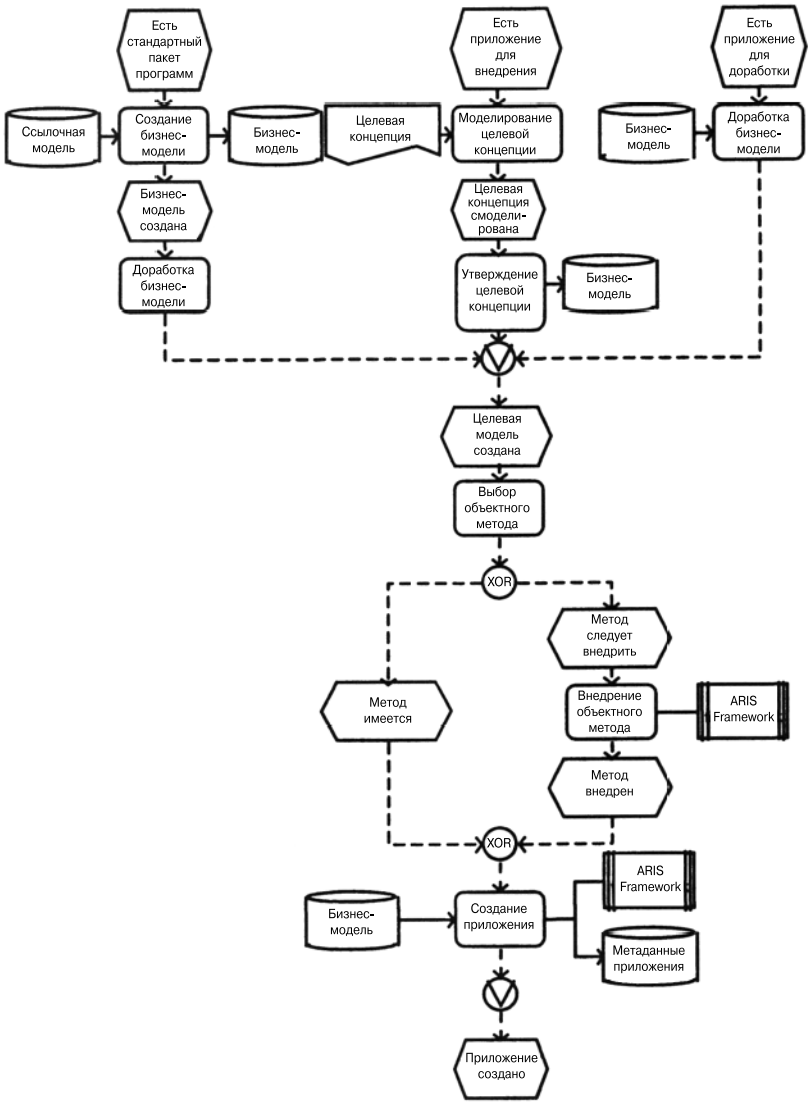


Рис. 170. Процедуральная модель создания приложений, ориентированных на пользователя

Если исходное приложение, ориентированное на пользователя, разрабатывается с нуля, целевая концепция корпорации сначала моделируется с помощью ARIS Toolset. Нет необходимости внедрять существующий объектный метод в среде ARIS Framework раньше создания целевой модели.

Обсудим функцию “моделирование целевой концепции” более детально, показывая ключевые аспекты создания приложения для различных вариантов модели. Описание функции основывается на процедуральной модели.

B.III.2 Процедуральная модель для моделирования целевых концепций

Модели ARIS Framework создаются в соответствии с процедурой, ориентированной либо на процессы, либо на объекты. В первом случае модели основаны на бизнес-процессах, а во втором — на бизнес-объектах. На рис. 171 иллюстрируются необходимые шаги моделирования целевой концепции.

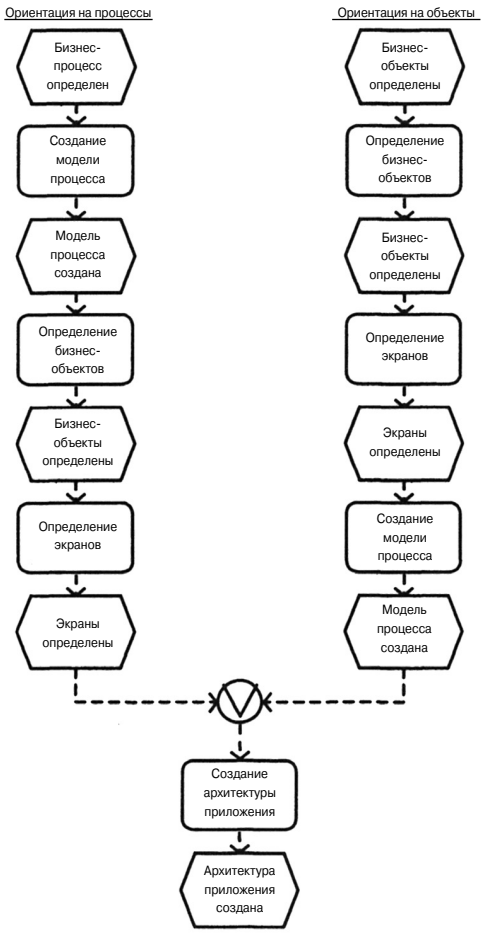


Рис. 171. Процедуральная модель для создания целевых моделей

Создание модели процесса

Помимо таких целей моделирования процессов, как прозрачность и оптимизация, ARIS Framework также определяет структуры приложений в соответствии с процессами. Эта тема будет детальнее рассмотрена ниже. В процедурах,

ориентированных на процессы, модели процессов определяют также требуемые бизнес-объекты. Эти модели процессов создаются с помощью ARIS Toolset как процессные цепочки. Применяются различные степени детализации, чтобы определить бизнес-процессы до уровня выполнимой функции.

Определение бизнес-объектов

Бизнес-объекты представляют элементы данных, необходимые для бизнес-приложения, а также методы, применяемые к элементам данных. Среди элементов данных выделяют объекты данных, их атрибуты, взаимосвязи и методы. Объекты данных и взаимосвязи определяются как модели ERM, а атрибуты определены в ARIS Toolset с помощью диаграмм распределения атрибутов. Чтобы соответствовать требованиям процесса, структуры ERM для конкретного бизнес-объекта корректируются. Соответствие методов и бизнес-объектов иллюстрируется с помощью диаграмм распределения методов. На рис. 172 приведен пример структуры модели бизнес-объекта “заказ”.

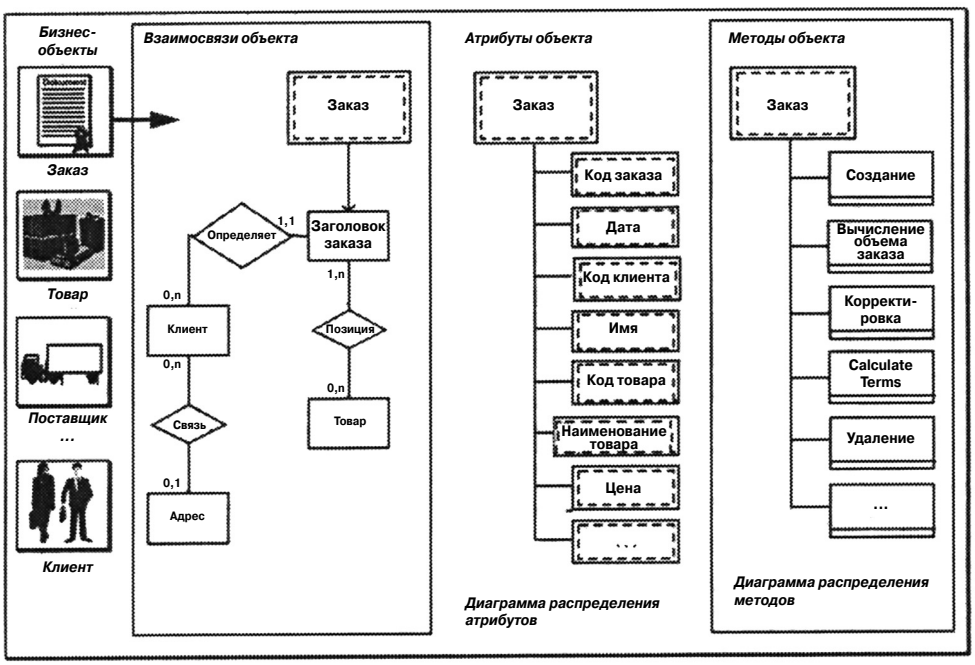


Рис. 172. Структура модели бизнес-объекта

При разработке приложений бизнес-объекты используются для создания схемы физической базы данных в системе реляционных баз данных и для определения взглядов. Создается также мета-информация, что позволяет отобразить конкретные бизнес-объекты в базовых объектах Framework.

Создание бизнес-презентации

Для бизнес-презентации разрабатывается форма презентации бизнес-объекта в рамках приложения.

Бизнес-объекты иллюстрируются разными способами. Конкретные типы иллюстраций описываются с помощью логических схем экранов. Цель моделирования экранов — создание логических схем экранов, соответствующих моделям данных. Логическая структура экрана трансформируется в экран в процессе генерации приложения. На рис. 173 иллюстрируется пример логической схемы и итогового экрана.

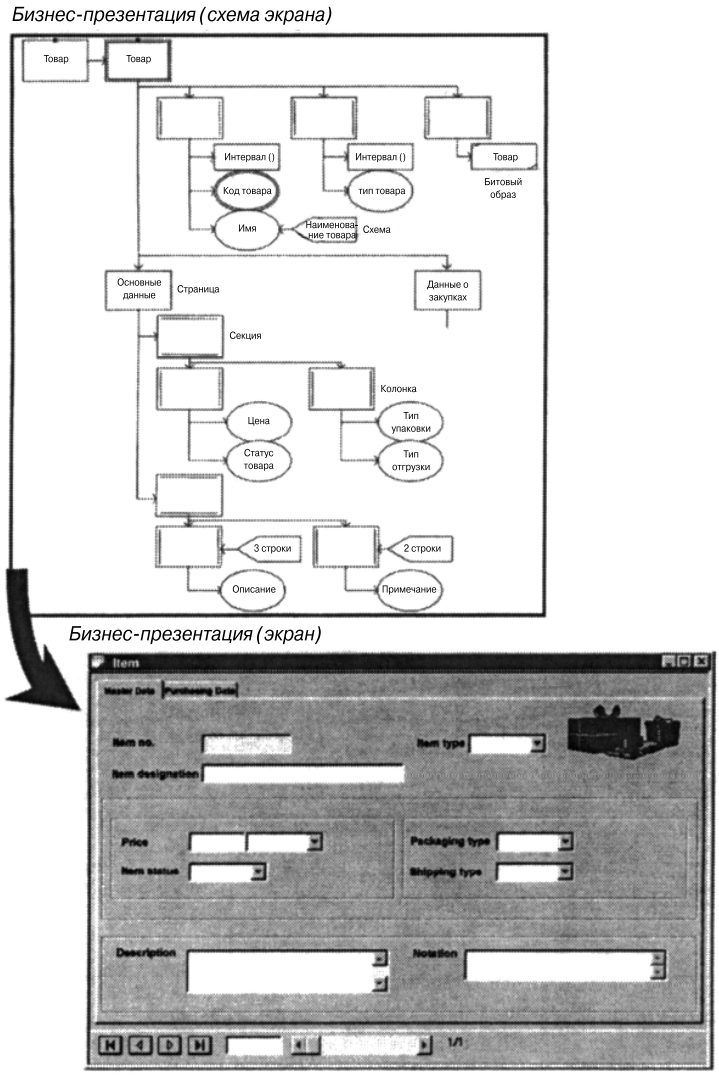


Рис. 173. Создание экрана

Создание архитектуры приложения из EPC

Процессные модели, созданные в ARIS Toolset, обеспечивают фундамент для генерации структур приложения в ARIS Framework. Компоненты приложения, взаимосвязанные с процессными моделями, определяются и моделируются в диаграмме типа приложения, используя соответствующий типам объектов тип приложения, тип модуля и тип информационной функции. На рис. 174 иллюстрируется взаимосвязь между процессной моделью и диаграммой типа приложения.

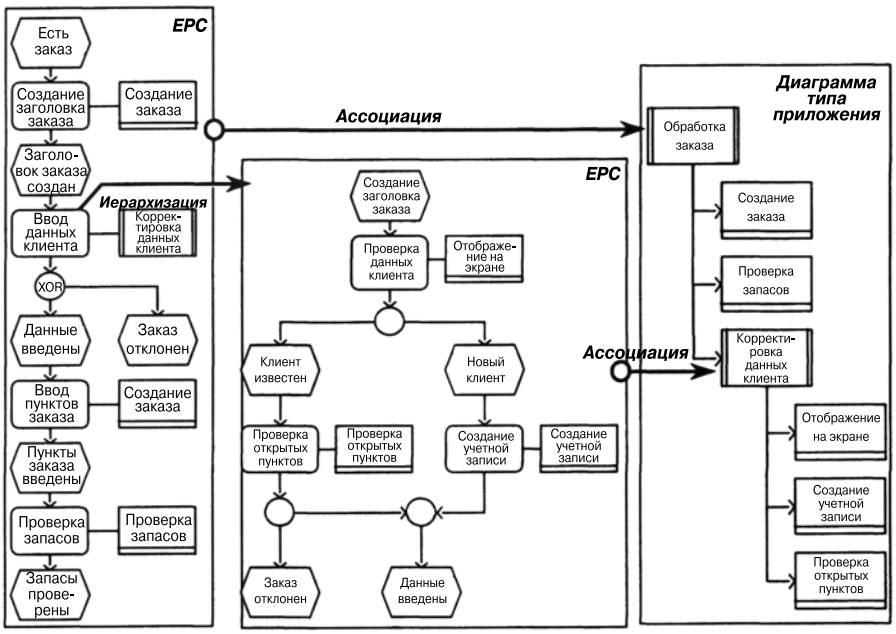


Рис. 174. Взаимосвязь между моделью процесса и диаграммой типа приложения

С использованием диаграмм типа приложения в ARIS Framework создаются так называемые рабочие окружения для приложений, ориентированных на функции, на бизнес-объекты или на бизнес-процессы. На рис. 175 иллюстрируется создание рабочего окружения с использованием диаграммы типа приложения.

Кроме того, при внедрении создается рабочий интерфейс, который зависит от структуры приложения. Таким образом, программное приложение полностью готово.

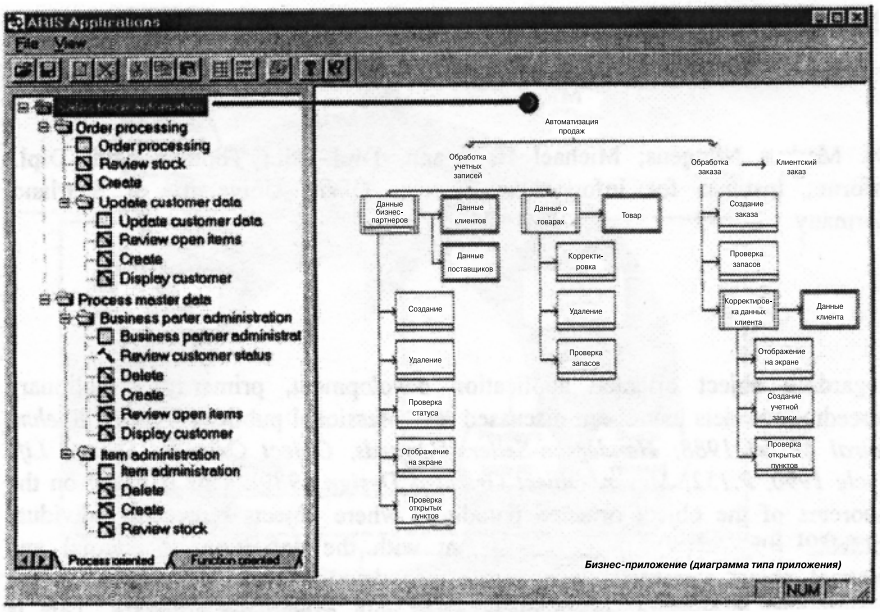


Рис. 175. Создание рабочего окружения с использованием диаграммы типа приложения

B.IV Разработка объектно-ориентированных систем с использованием UML

Маркус Наттгенс (Markus Nüttgens); Майкл Хоффман (Michael Hoffman), Dipl.-Hdl.; Томас Фелд (Thomas Feld), Dipl.-Inform; Институт информационных систем (IWi), Университет земли Саарленд, Германия.

Ряд профессиональных публикаций посвящен разработке объектно-ориентированных систем с использованием преимущественно эволюционных процедуральных моделей *(см. Boehm, Spiral Model 1988; Henderson-Sellers/Edwards, Object Oriented System Life Cycle 1990, p. 152; Meyer, Object Oriented design 1989)*. В основе этого подхода лежат теоремы объектно-ориентированной парадигмы, где объекты представляют отдельные подсистемы "закрытой системы". На основе определений структур внутренних и внешних объектов проектируются расширяемые системы. Результаты проектирования представляют собой реализацию поставленных целей. В эволюционной модели каждый цикл заканчивается выполнимой программой. Каждая из этих программ может тестироваться индивидуально, следовательно, каждая подсистема может тестироваться на ранних стадиях проектирования. Это позволяет избежать тупиковых ситуаций в процессе разработки прикладной системы и своевременно продемонстрировать результаты. Практический опыт внедрения подсистем служит источником непрерывного усовершенствования системы в целом.

B.IV.1 Проектирование и описание процедуральных моделей

Рассмотрим процедуральную модель для разработки объектно-ориентированных приложений с использованием унифицированного языка моделирования UML *(см.UML Notation Guide 1997)*. До настоящего времени ни одна процедуральная модель не была явно определена в среде UML. На рис. 176 иллюстрируется предварительная процедуральная модель, описывающая возможную процедуру разработки объектно-ориентированного приложения.

Разработка объектно-ориентированных приложений обычно основана на модели оптимизации бизнес-процесса *(см. Oestereich, Objektorientierte Softwareentwicklung 1997, p. 85; Yourdon et al., Mainstream Objects 1996, pp. 71)*.

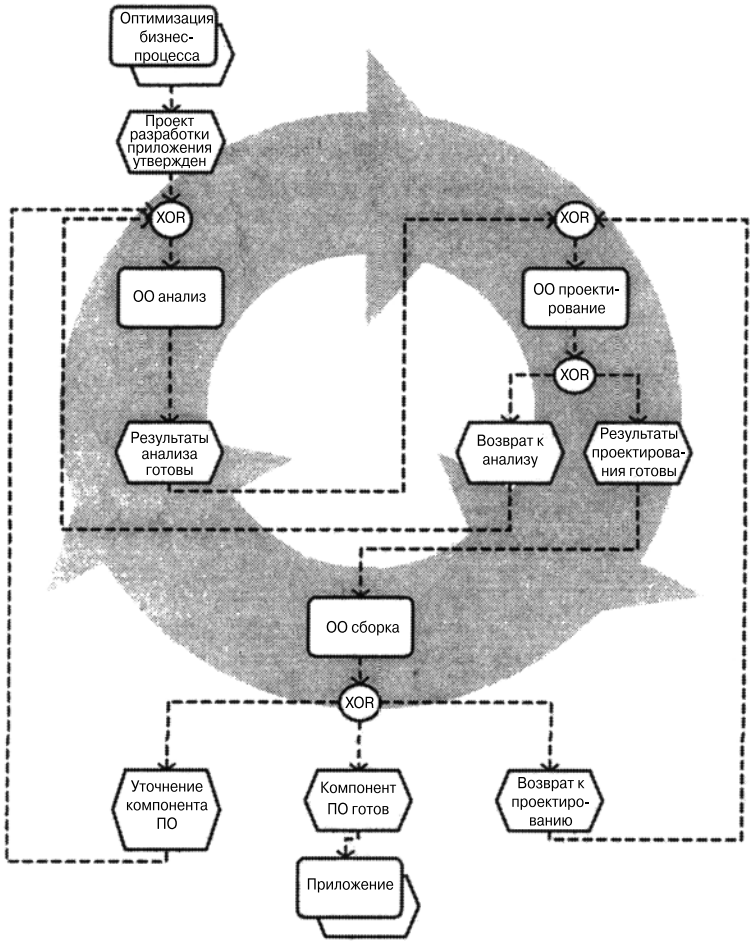


Рис. 176. Предварительная процедуральная модель для разработки объектно-ориентированных приложений

Этапы разработки объектно-ориентированного приложения (анализ, проектирование и сборка) описывают цикл разработки с обратными связями к каждому элементу процедуральной модели. После того как проект разработки объектно-ориентированного приложения утвержден, он анализируется и затем проектируется в рамках объектно-ориентированного подхода. Если при объектно-ориентированном проектировании возникают любые противоречия или проблемы, которые не могут быть решены на этом этапе, повторяется этап объектно-ориентированного анализа.

Результаты объектно-ориентированного проектирования представляют собой входные данные для объектно-ориентированной сборки. Модели проектирования внедряются с помощью генераторов кодов. В случае возникновения каких-либо ошибок в процессе генерации кодов происходит возврат на предыдущий этап. Практическое использование программного обеспечения начина-

ется сразу после готовности соответствующего компонента. Так происходит переход от разработки до внедрения. Если возникает необходимость в изменении компонента программного обеспечения, цикл разработки прикладной системы повторяется, начиная с объектно-ориентированного анализа. Ниже этапы проектирования процедуральной модели рассматриваются более детально.

B.IV.2 Этапы проектирования процедуральной модели

Объектно-ориентированный анализ

На рис. 177 иллюстрируется процесс объектно-ориентированного анализа *(см.Nüttgens, Koordiniert-dezentrales Informationsmanagement 1995, p. 223)*.

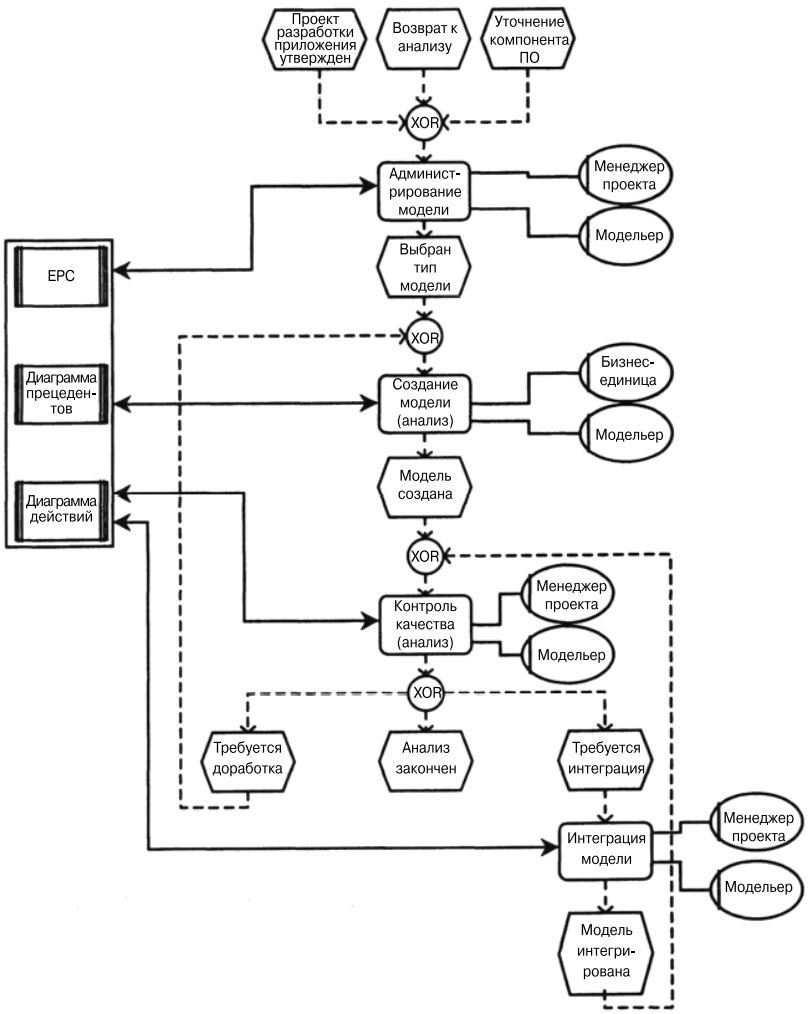


Рис. 177. Процедуральная модель объектно-ориентированного анализа

На первой стадии администрирования модели менеджер проекта и модельеры выбирают типы моделей для системного анализа. Релевантные UML типы моделей: существующие модели бизнес-процессов (такие как EPC); диаграммы вариантов использования; диаграммы действий.

Диаграммы вариантов использования служат в UML преимущественно для первичной оценки организационных сценариев, которые соответствуют отдельным функциональным блокам модели EPC и описываются с использованием нотации UML. Компоненты приложения, поддерживающие отдельные функции, обеспечивают стартовую позицию для структурирования вариантов использования.

Диаграммы действий создаются на основе информации потока управления модели EPC, а затем уточняются при описании конкретных состояний объекта.

Диаграммы действий также обеспечивают фундамент для создания объектно-ориентированных моделей управления потоками.

После выполнения контроля качества компоненты модели объединяются и составляют основу для объектно-ориентированного проектирования.

Объектно-ориентированное проектирование

На рис. 178 иллюстрируется процедуральная модель объектно-ориентированного проектирования, соответствующая процедуральной модели объектно-ориентированного анализа. Диаграммы состояний, диаграммы последовательности действий, диаграммы классов, диаграммы сотрудничества или детальные диаграммы действий представляют собой типы моделей UML. Проектирование диаграмм классов имеет особо важное значение. Отправная точка для проектирования диаграмм классов — это модели бизнес-процессов, соответствующие объектно-ориентированному анализу. Например, функциональные блоки или входные (выходные) данные моделей EPC детализируются в соответствии с объектно-ориентированным подходом и относятся к определенному классу бизнес-объектов.

Если структурная модель данных уже имеется в наличии в виде ERM, ключевые классы и их структурные взаимосвязи проектируются на основе этой модели.

Объектно-ориентированная сборка

На рис. 179 иллюстрируется процедуральная модель объектно-ориентированной сборки. При внедрении моделей объектно-ориентированного проектирования ставится цель достижения наивысшей степени автоматизации с использованием объектно-ориентированных генераторов кодов. В некоторых случаях требуется дополнительное программирование, например, для внедрения методов. Используются любые доступные библиотеки классов. Остальная часть процесса корреспондируется с процедуральной моделью объектно-ориентированного проектирования или анализа. После успешной интеграции компонентов начинается следующий цикл разработки процедуральной модели для остальных прототипов приложения.

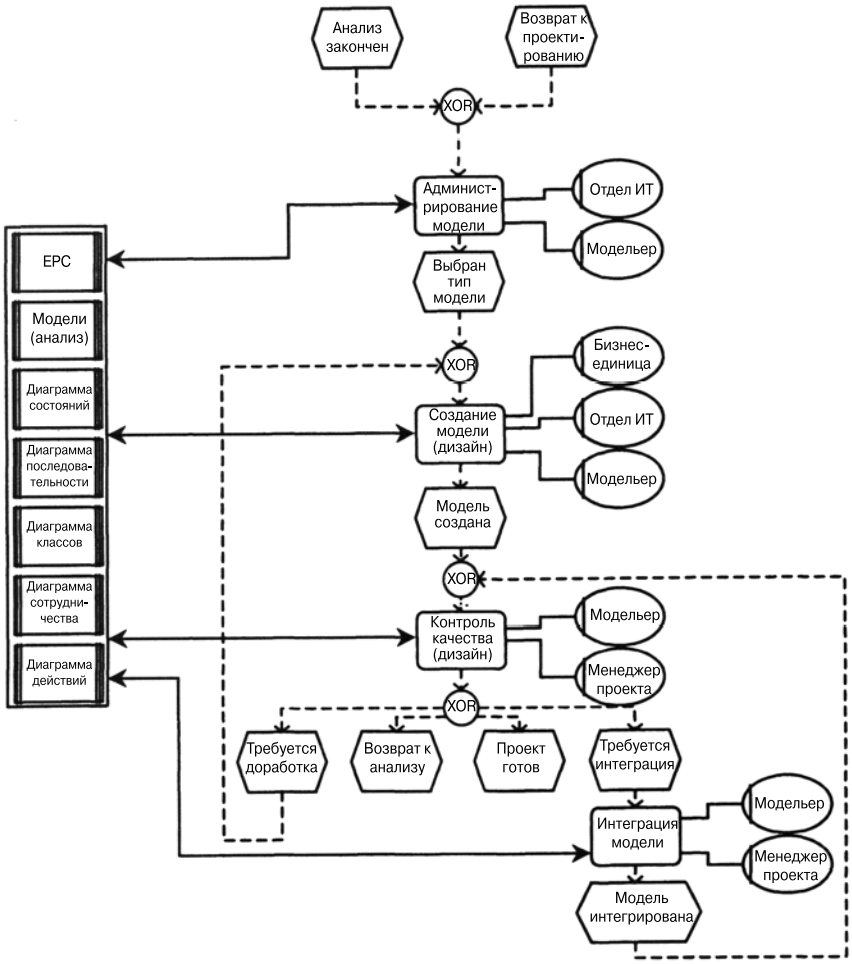


Рис. 178. Процедуральная модель объектно-ориентированного проектирования

B.IV.3 Выводы

Разработка структурных и объектно-ориентированных приложений основывается на концепции оптимизации бизнес-процессов. После оптимизации эти бизнес-процессы обеспечивают основу для разработки приложений как процессно-организационных сценариев. В теории и на практике метод EPC зарекомендовал себя как метод выбора в рамках процессного подхода. Опыт перехода от процессно-организационных и функциональных EPC к моделям объектно-ориентированного анализа и проектирования носит незавершенный характер *(см. Hoffmann/Scheer/Hoffmann, Modellierungsmethoden 1995; Bungert/Heß, Objektorientierte Geschäftsprozeßmodellierung 1995; Scheer/Nüttgens/Zimmermann, oEPK 1997)*.

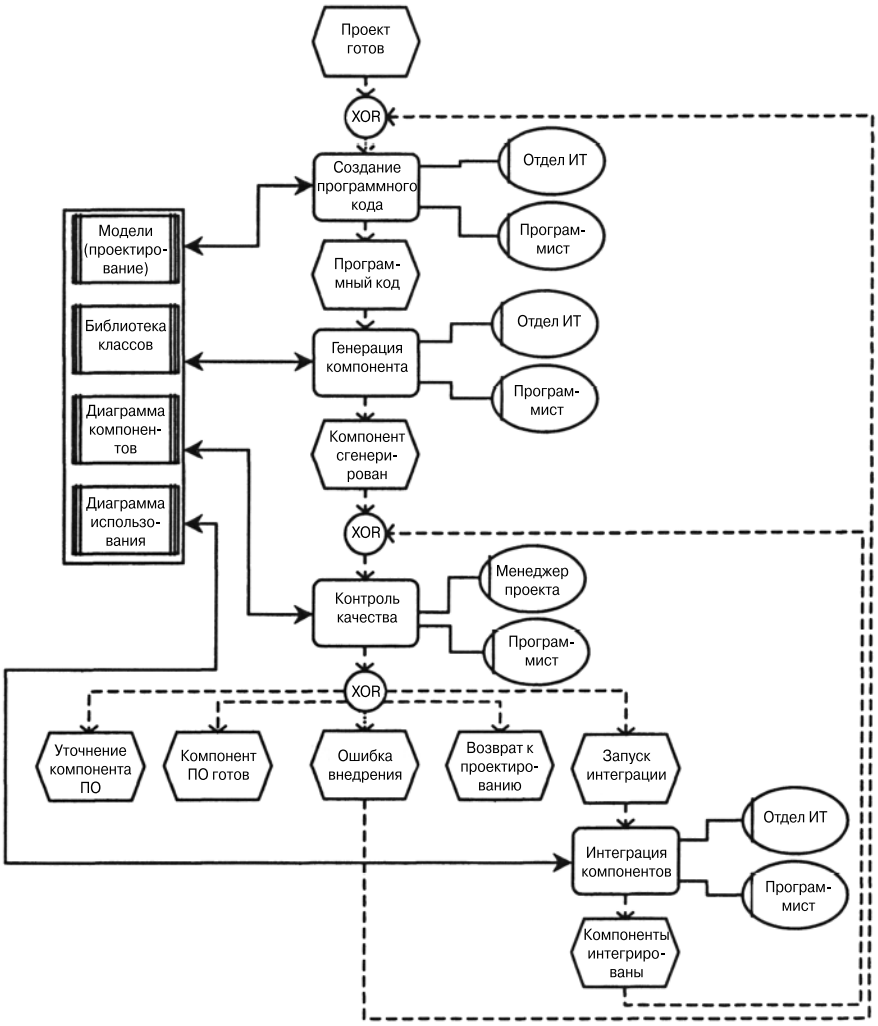


Рис. 179. Процедуральная модель объектно-ориентированной сборки

Процедуральные модели ARIS демонстрируют свои преимущества при разработке прикладных систем и вносят существенный вклад в интеграцию подходов к моделированию.

ARIS

Расшифровка аббревиатур

ACID (Atomarity, Consistency, Isolation, Durability) — требуемые свойства транзакции в СУБД (атомарность, непротиворечивость, изолированность, долговечность).

ARIS (Architecture of Integrated Information Systems) — инструментальное средство моделирования бизнес-процессов.

ATM (Asynchronous Transfer Mode) — асинхронная передача данных.

BAPI (Business Application Programming Interface) — программный интерфейс доступа к методам бизнес-объекта в SAP R/3.

BelWu (Baden-Wurttemberg Extended LAN) — расширенная локальная сеть федеральной земли Баден-Вюртемберг.

B-Win (нем. Broadband-WIssenchafts Netz) — широкополосная исследовательская сеть.

CAD (Computer Aided Design) — компьютерный дизайн.

CSMA/CD (Carrier Sense Multiple Access with Collision Detection) — алгоритм доступа, где все узлы имеют равные возможности доступа к сетевой среде, а при одновременной попытке фиксируется столкновение и сеанс передачи повторяется позднее.

CPI (Continuous Process Improvement) — непрерывное усовершенствование процессов.

DDL (Data Description Language) — язык описания данных.

DML (Data Manipulation Language) — язык управления данными.

DSDL (Data Storage Description Language) — язык описания хранения данных, предназначенный для описания физической структуры базы данных.

ECA (Event Condition Action) — правило “событие–условие–действие”. Правило описывает действие, которое должно быть выполнено при возникновении события, при условии выполнения некоторого требования.

EPC (Event-Driven Process Chain) — событийно-управляемая цепочка процесса.

ERM (Entity Relationship Model) — модель “сущность–связь”.

ETM (Event Trigger Mechanism) — механизм “событие–триггер”.

FDDI (Fiber Distributed Data Interface) — распределенный интерфейс передачи данных по оптоволоконным линиям.

FDL (Flow Defining Language) — специальный язык конфигурации.

GERT (Graphical Evaluation and Review Technique) — методика графического анализа.

HOBE (ARIS House of Business Engineering) — концепция ARIS применительно к бизнес-инжинирингу.

IMG (Implementation Management Guide) — руководство по внедрению.

IWi (Institute for Information Systems) — Институт информационных систем в г. Саарбрюкен, Германия.

LAN (Local Area Network) — локальная вычислительная сеть.

PCD (Process Chain Diagram) — диаграмма цепочки процесса.

PPC (Production Planning and Control) — система планирования производства и контроля.

QA (Quality Assurance) — обеспечение качества.

RMI (Remote Method Invocation) — интерфейс вызова удаленных методов в Java.

SADT (Structured Analysis and Design Technique) — структурный анализ и проектирование.

SAP R/3 — интегрированная автоматизированная система управления предприятием; продукт фирмы SAP (Германия).

SQL (Structured Query Language) — язык структурированных запросов, применяемый для создания, модификации и управления данными в реляционных базах данных.

SWOT (Strengths, Weakness, Opportunities, Threats) — сильные стороны, слабые стороны, возможности, угрозы.

UML (Unified Modeling Language) — унифицированный язык моделирования.

WMS (Workflow Management System) — система управления потоком.

АП — автоматизированное производство.

БД — база данных.

БО — бизнес-объект.

ЗУ — запоминающее устройство.

ИО — информационный объект.

ИСУ — информационная система управления предприятием.

ИТ — информационные технологии.

КД — компьютерный дизайн.

КИ — компьютерный инжиниринг.

ККК — компьютерный контроль качества.

КП — ключевой процесс.

КПВ — вариант ключевого процесса.

КФУ — критический фактор успеха.

ЛВС — локальная вычислительная сеть.

ЛП — линейное программирование.

НИР — научно-исследовательские разработки.

ОД — объект данных.

ОЕ — организационная единица.

ОК — отдел кадров.

ООА — объектно-ориентированный анализ.

ПК — персональный компьютер.
ПО — программное обеспечение.
ПП — планирование с использованием компьютерных программ.
ПС — прикладная система.
СБП — сегмент бизнес-процессов.
СУБД — система управления базами данных.
СЦП — событийно-управляемая цепочка процесса.
ФДС — функция добавленной стоимости.
ФС — функциональный сегмент.
ЧПУ — числовое программное управление.
ЭВ — элемент выпуска.

Ссылки

Baan, Dynamic Enterprise Modeling 1996

Baan Business B.V. (Ed.): Dynamic Enterprise Modeling, A Paradigm Shift in Software Implementation, Deventer 1996.

Bach/Brecht/Hess/Österle, Enabling Systematic Business Change 1996

Bach, V., Brecht, L., Hess, T., Österle, H.: Enabling Systematic Business Change, Wiesbaden 1996.

Balzert, Entwicklung von Software-Systemen 1982

Balzert, H.: Die Entwicklung von Software-Systemen — Prinzipien, Methoden, Sprachen, Werkzeuge, Mannheim et. al. 1982.

Balzert, Lehrbuch der Software-Technik 1996

Balzert, H.: Lehrbuch der Software-Technik, Software-Entwicklung, Heidelberg et. al. 1996.

Barker, Case Method 1990*

Barker, R.: Case* Method, Tasks and Deliverables, Wokingham et. al. 1990.

Blaser/Jarke/Lehmann, Datenbanksprachen und Datenbankbenutzung 1987

Blaser, A., Jarke, M., Lehmann, H.: Datenbanksprachen und Datenbankbenutzung, in: Lockemann, P.C., Schmidt, J.W. (Ed.): Datenbank-Handbuch, Berlin et. al. 1987.

Boehm, Spiral Model 1988

Boehm, B.W.: A Spiral Model of Software Development and Enhancement, in: IEEE Computer 21(1988)5, pp. 14–24.

Booch, Object-oriented Design 1991

Booch, G.: Object-oriented Design with Applications, Redwood City 1991.

Bröhl/Dröschel, V-Modell 1995

Bröhl, A.-P., Dröschel, W. (Ed.): Das V-Modell: Der Standard für die Softwareentwicklung mit Praxisleitfaden, 2nd publ., München-Wien 1995.

Buck-Emden/Galimow, Client-Server-Technologie 1996

Buck-Emden, R., Galimow, J.: Die Client-Server-Technologie des SAP R/3-Systems, 3rd ed., Bonn et. al. 1996.

Bullinger, Prozeßorientierte Strukturen 1994

Bullinger, H.-J.: Prozeßorientierte Strukturen und Workflow-Management bei Dienstleistern, in: Bullinger H.-J. (Ed.): Workflow-Management bei Dienstleistern, Stuttgart 1994, pp. 11–46.

Bungert/Heß, Objektorientierte Gesehäftsprozeßmodellierung 1995
Bungert, W., Heß, H.: Objektorientierte Geschäftsprozeßmodellierung, in: Information Management 10(1995)1, pp. 52–63.

Chen, Entity–Relationship Model 1976
Chen, P.P.: The Entity–Relationship Model: Towards a Unified View of Data, in: ACM (Ed.): Transactions on Database Systems, Nr. 1, 1976, pp. 9–36.

Chen/Doumeingts, GRAI-CIM 1996
Chen, D., Doumeingts, G.: The GRAI-CIM reference model, architecture and methodology, in: Bernus, P., Nemes, L., Williams, T.J. (Ed.): Architectures for Enterprise Integration, London et. al. 1996, pp. 102–126.

Coad/Yourdon, Object-Oriented Analysis 1991
Coad, P., Yourdon, E.: Object-Oriented Analysis, 2nd publ., Englewood Cliffs 1991.

Coad/Yourdon, Objeet-Oriented Design 1991
Coad, P., Yourdon, E.: Object-Oriented Design, Englewood Cliffs 1991.

Codd, OLAP 1994
Codd, E.F.: OLAP — On-Line Analytical Processing mit TM/I, in: M.I.S. GmbH (Ed.), Darmstadt 1994.

Date, Database Systems 1995
Date, C.J.: An Introduction to Database Systems, 6th ed., Reading 1995.

Date, Distributed Databasesystems 1986
Date, C.J.: Twelve Rules for Distributed Databasesystems, in: Computerworld, Juni 1987.

Davis/Olson, Management Information Systems 1984
Davis, G.B., Olson, M.H.: Management Information Systems: Conceptual Foundations, Structure and Development, 2nd publ., New York 1984.

Dittrich, Nachrelationale Datenbanktechnologie 1990
Dittrich , K.R.: Objektorientiert, aktiv, erweiterbar: Stand und Tendenzen der "nachrelationalen" Datenbanktechnologie, in: Informationstechnik: it, Computer, Systeme, Anwendungen, Band 32(1990)5, pp. 343–354.

Dittrich/Gatziu, Aktive Datenbanksysteme 1996
Dittrich, K.R., Gatziu, S.: Aktive Datenbanksysteme: Konzepte und Mechanismen, Bonn et. al. 1996.

Elbling/Kreuzer, Strategische Instrumente 1994
Elbling, O., Kreuzer, C.: Handbuch der strategischen Instrumente — Ubersicht aller Instrumente, Bewertungs- und Einsatzmodelle, Formularanhang für den direkten Praxiseinsatz, Wien 1994.

Elmaghraby, Activity networks 1977
Elmaghraby, S.E.: Activity networks, project planning and control by network models, New York et. al. 1977.

Esswein, Rollenmodell der Organisation 1992
Esswein, W.: Das Rollenmodell der Organisation: Die Berücksichtigung aufbauorganisatorischer Regelungen in Unternehmensmodellen, in: Augsburger, W., Sinz, E.J. (Ed.): Bamberger Beiträge zur Wirtschaftsinformatik, Nr. 18, Bamberg 1992.

Ferstl/Sinz, Wirtschaftsinformatik 1994
Ferstl, O., Sinz, E.J.: Grundlagen der Wirtschaftsinformatik, 2nd publ., Mnchen 1994.

Fingar/Read/Stiekeleather, Next Generation Computing 1996
Fingar, P., Read, D., Stickeleather, J.: Next Generation Computing. Distributed Objects for Business, New York 1996.

Frese, Grundlagen der Organisation 1995
Frese, E.: Grundlagen der Organisation, Konzept – Prinzipien – Strukturen, 6th ed., Wiesbaden 1995.

Galler, Vom Geschäftsprozeßmodell zum Workflow-Modell 1997
Galler, J.: Vom Geschäftsprozeßmodell zum Workflow-Modell, Wiesbaden 1997.

Gutenberg, Die Produktion 1983
Gutenberg, E.: Die Produktion, Grundlagen der Betriebswirtschaftslehre, Bd. I: Die Produktion, 24th ed., Berlin et. aI. 1983.

Gutzwiller/Österle, Referenz-Meta-Modell Analyse 1990
Gutzwiller, T., Österle, H.: CC RIM, Referenz-Meta-Modell Analyse, Forschungsbericht Nr. IM 2000/CCRIM/2, Version 2.0, Institut für Wirtschaftsinformatik, Hochschule St. Gallen, 15.01.1990.

Grabowski et. al., Integriertes Produktmodell 1993
Grabowski, H., Anderl, R., Polly, A., Warnecke, H.-J.: Integriertes Produktmodell, Entwicklungen zur Normung von CIM, Berlin et. al. 1993.

Härder, Relationale Datenbanksysteme 1989
Härder, T.: Grenzen und Erweiterungsmöglichkeiten relationaler Datenbanksysteme für Nicht-Standard-Anwendungen, in: Scheer, A.-W. (Ed.): Praxis relationaler Datenbanken, Proceedings zur Fachtagung, Saarbrücken 1989, pp. 1–25.

Hagemeyer/Rolles/Schmidt/Scheer, Arbeitsverteilungsverfahren 1998
Hagemeyer, J., Rolles, R., Schmidt, Y., Scheer, A.-W.: Arbeitsverteilungs-verfahren in Workflow-Management-Systemen: Anforderungen, Stand und Perspektiven, in: Veröffentlichungen des Instituts für Wirtschaftsinformatik, Saarbrücken 1998, Nr. 144.

Harel, Statecharts 1987
Harel, D.: Statecharts, A Visual Formalism for Complex Systems, in: Science of Computer Programming 8(1987), pp. 231–274.

Harel, On Visual Formalism 1988
Harel, D.: On Visual Formalism, in: CACM, May 1988, pp. 514–530.

Hars, Referenzdatenmodelle 1994
Hars, A.: Referenzdatenmodelle — Grundlagen effizienter Datenmodellierung, Wiesbaden 1994.

Harrington, Business Process Improvement 1991
Harrington, H. J.: Business Process Improvement, New York et. al. 1991.

Henderson-Sellers/Edwards, Object-Oriented System Life Cycle 1990
Henderson-Sellers, B., Edwards, J.M.: The Object-Oriented System Life Cycle, in: Communications of the ACM, 33(1990)9, pp. 142–159.

Herbst/Knolmayer, Geschäftsregeln 1994
Herbst, H., Knolmayer, G.: Ansätze zur Klassifikation von Geschäftsregeln, in: Wirtschaftsinformatik 37(1995)2, pp. 149–159.

Heß, Monitoring von Geschäftsprozessen 1996
Heß, H.: Monitoring von Geschäftsprozessen, in: ZWF Zeitschrift für wirtschaftlichen Fabrikbetrieb, 91 (1996)3, pp. 76–79.

Hildebrand, Software Tools 1990
Hildebrand, K.: Software Tools: Automatisierung im Software-Engineering — Eine umfassende Darstellung der Einsatzmöglichkeiten von Software-Entwicklungswerkzeugen, Berlin et. al. 1990.

Hoffmann/Scheer/Hoffmann, Modellierungsmethoden 1995
Hoffmann, W., Scheer, A.-W., Hoffmann, M.: Überführung strukturierter Modellierungsmethoden in die Object Modeling Technique (OMT), in: Verffentlichungen des Instituts für Wirtschaftsinformatik, Nr. 114. Saarbrücken 1995.

Hollingsworth, Workflow Reference Model 1995
Hollingsworth, D.: The Workflow Reference Model, in: Workflow Management Coalition (Ed.): Document TCOO-1003, Draft 1.1, 1995.

Hutchinson/Mariani, Local Area Networks 1985
Hutchinson, D., Mariani, J.A. (Ed.): Local Area Networks: An Advanced Course, Lecture Notes in Computer Science 184, Berlin et. al. 1985.

IDS, ARIS — Applications 1997
IDS Prof. Scheer GmbH (Ed.): Produktinformation ARlS — Applications, Saarbrücken 1997.

IDS, ARIS — Easy Design 1997
IDS Prof. Scheer GmbH (Ed.): Produktinformation ARIS — Easy Design. Saarbrücken 1997.

IDS, ARlS — Framework 1997
IDS Prof. Scheer GmbH (Ed.): White Paper ARIS — Framework, Saarbrücken 1997.

IDS, ARlS — Toolset 1997
IDS Prof. Scheer GmbH (Ed.): Produktinformation ARIS — Toolset, Saarbrücken 1997.

IDS, Workflow-Modellierung 1997
IDS Prof. Scheer GmbH (Ed.): White Paper Workflow-Modellierung, Vers. 1.1, Saarbrücken 1997.

Information Management Gesellschaft, PROMET 1994
Information Management Gesellschaft (Ed.): PROMET BPR — Methodenhandbuch für den Entwurf von Geschäftsprozessen, Version 1.0, St. Gallen-München 1994.

Jablonski, Datenverwaltung in verteilten Systemen 1990
Jablonski, S.: Datenverwaltung in verteilten Systemen: Grundlagen und Lösungskonzepte, Berlin et. al. 1990.

Jablonski, Workflow-Management-Systeme 1995
Jablonski, S.: Workflow-Management-Systeme, Modellierung und Architektur, Bonn 1995.

Jacobsen, Object-Oriented Software Engineering 1996
Jacobsen, I.: Object-Oriented Software Engineering. A Use Case Driven Approach, Wokingham 1996.

Jost, EDV-gestützte CIM-Rahmenplanung 1993
Jost, W.: EDV-gestützte CIM-Rahmenplanung, Wiesbaden 1993.

Kagermann, Business Framework 1997
Kagermann, H.: Buildung Industry Solutions within the Business Framework, Vortrag Sapphire, Amsterdam Juni 1997,
URL: http://www.sap.com/events/amsterdam/index.htm.

Kauffels, Lokale Netze 1997
Kauffels, F.-J.: Lokale Netze: Grundlagen, Standards, Perspektiven, 9th ed., Bonn et. al. 1997.

Keller/Nüttgens/Scheer, Semantische Prozeßmodellierung 1992
Keller, G., Nüttgens, M., Scheer, A.-W.: Semantische Prozeßmodellierung, in: Veröffentlichungen des Instituts für Wirtschaftsinformatik, Nr. 89, Saarbrücken 1992.

Keller/Schröder, Geschäftsprozeßmodelle 1996
Keller, G., Schröder, G.: Geschäftsprozeßmodelle: Vergangenheit–Gegenwart–Zukunft, in: Management & Computer 4(1996)2, pp. 77–90.

Keller/Teufel, SAP R/3 prozeßorientiert anwenden 1997
Keller, G., Teufel, T.: SAP R/3 prozeßorientiert anwenden: Iteratives Prozeß-Prototyping, Bonn et. al. 1997.

KGSt, Wege zum Dienstleistungsunternehmen 1992
KGSt (Ed.): Wege zum Dienstleistungsunternehmen Kommunalverwaltung, Fallstudie Tilburg, Bericht Nr. 19/1992, Köln 1992.

KGSt, Das Neue Steuerungsmodell 1994
KGSt (Ed.): Das neue Steuerungsmodell: Definition und Beschreibung von Produkten, Bericht Nr. 8/1994, Köln 1994.

Kirchmer, Definition von Geschäftsprozessen 1995
Kirchmer, M.: Markt- und produktgerechte Definition von Geschäftsprozessen, in: Management & Computer 3(1995)4, pp. 267–273.

Kirchmer, Einführung von Standardsoftware 1996
Kirchmer, M.: Geschäftsprozeßorientierte Einführung von Standardsoftware, Wiesbaden 1996.

Kilger, C.: Objektbanksysteme 1996
Kilger, C.: Unterstützung anwendungsspezifischer Zugriffsprofile in Objektbanksystemen, St. Augustin 1996.

Kotz, Triggermechanismen in Datenbanksystemen 1989
Kotz, A.: Triggermechanismen in Datenbanksystemen, Berlin et. al. 1989.

Krcmar, Informationsmanagement 1997
Krcmar, H.: Informationsmanagement, Berlin et. al. 1997.

Kruppke, Software 2000 1997
Kruppke, H.: Software 2000 — ModelIierung und Generierung von individuellen Business Lösungen aus Basiskomponenten, Saarbrücken 1997.

Kueng et. al., Object-Oriented Business Process Model 1996
Kueng, P. et. al.: How to compose an Object-Oriented Business Process Model?, in: Brinkkemper, S. et. al. (Ed.): Method Engineering — Principles of method construction and tool support, Proceedings of the IFIP TCS, WG8.1/WG8.2 Working Conference, London 1996.

Lockemann, Weiterentwicklung relationaler Datenbanken 1991
Lockemann, P.C.: Weiterentwicklung relationaler Datenbanken für Nicht-Standard-Anwendungen, in: Scheer, A.-W. (Ed.): Praxis relationaler Datenbanken, Proceedings zur Fachtagung, Saarbrücken 1989, pp. 2–16.

Lockemann/Dittrich, Architektur von Datenbanksystemen 1987
Lockemann, P.C., Dittrich, K.R.: Architektur von Datenbanksystemen, in: Lockemann, P.C., Schmidt, J.W. (Ed.): Datenbank-Handbuch, Berlin et. al. 1987.

Maier, Qualität von Datenmodellen 1996
Maier, R.: Qualität von Datenmodellen, Wiesbaden 1996.

Mayr/Dittrich/Lockemann, Datenbankentwurf 1987
Mayr, H.C., Dittrich, K.R., Lockemann, P.C.: Datenbankentwurf, in: Lockemann, P.C., Schmidt, J.W. (Ed.): Datenbank-Handbuch, Berlin et. al. 1987.

Martin, Application Development 1982
Martin, J.: Application Development Without Programmers, Englewood Cliffs 1982.

Martin, Information Engineering I 1989
Martin, J.: Information Engineering, Book I: Introduction, Englewood Cliffs 1989.

Martin, Information Engineering II 1990
Martin, J.: Information Engineering, Book II: P1anning and Analysis, Englewood Cliffs 1990.

Mc Farlan/Mc Kenney/Pyburn, Information archipelago 1983
Mc Farlan, F.W., Mc Kenney, J.L., Pyburn, P.: Information archipelago: Plotting a course, in: Harvard Business Review, January-February 1983, pp. 145–155.

Mertens, Wirtschaftsinformatik 1995
Mertens, P.: Wirtschafts informatik — Von den Moden zum Trend, in: König, W. (Ed.): Wirtschaftsinformatik '95 — Wettbewerbsfähigkeit, Innovationen, Wirtschaftlichkeit, Heidelberg 1995, pp. 25–64.

Meyer, Object-oriented Software Construction 1988
Meyer, B.: Object-oriented Software Construction, New York 1988.

Meyer, Object-Oriented Design 1989
Meyer, B.: From Structured Programming to Object oriented Design: Tbe Road to Eiffel, in: Structured Programming, 10(1989)1, pp. 19–39.

Nerreter, Zur funktionalen Architektur von verteilten Datenbanken 1983
Nerreter, U.: Zur funktionalen Architektur von verteilten Datenbanken — Konzepte, Methoden und Beispiele, Dissertation ETH Zürich 1983.

Nüttgens, Koordiniert-dezentrales Informationsmanagement 1995
Nüttgens, M.: Koordiniert-dezentrales Informationsmanagement, Wiesbaden 1995.

Nüttgens/Feld/Zimmermann, Business Process Modeling
Nüttgens, M., Feld, T., Zimmermann, V.: Business Process Modeling with EPC and UML — Transformation or Integration?, in: Schader, M., Korthaus, A. (Ed.): The Unified Modeling Language — Technical Aspects and Applications, Heidelberg 1998, pp. 250–261.

Oberweis, Modellierung von Workflows 1996
Oberweis, A.: Modellierung und Ausführung von Workflows mit Petri-Netzen, Stuttgart et. al. 1996.

Oestereieh, Objektorientierte Softwareentwicklung 1997
Oestereich, B.: Objektorientierte Softwareentwicklung, 3rd ed., München et. al. 1997.

Olle et. al., Information Systems Methodologies 1991
Olle, T.W. et. al.: Information Systems Methodologies: A Framework for Understanding, 2nd publ., Wokingham et. al. 1991.

Olle et. al., Information Systems Life Cycle 1988
Olle, T.W., Verrijn-Stuart, A.A., Bhabuta, L. (Ed.): Computerized Assistance During The Information Systems Life Cycle, Proceedings of the IFIP WG 8.1 Working Conference on Computerized Assistance during the Information Systems Life Cycle, CRIS 88, Amsterdam et. al. 1988.

OMG, Common Business Objcets 1996
Object Management Group (Ed.): Common Business Objects and Business Object Facility, OMG TC Document CF/96-01-04,
URL: http://dataaccess.com/bodtf/DownloadlCFRFP4.doc.

Цsterle, Business Engineering 1 1995
Österle, H.: Business Engineering. Prozeß- und Systementwicklung. Band I: Entwurfstechniken, Berlin et. al. 1995.

Österle/Vogler, Praxis des Workflow-Managements 1996
Österle, H., Vogler, P.: Praxis des Workflow-Managements: Grundlagen Vorgehen, Beispiele, Wiesbaden 1996.

Osterloh/Frost, Prozeßmanagement 1996
Osterloh, M., Frost, 1.: Prozeßmanagement als Kernkompetenz, Wie Sie Business Reengineering strategisch nutzen können, Wiesbaden 1996.

Page, Objektorientierte Software
Page, P.: Objektorientierte Software in der kommerziellen Anwendung, Berlin et. al. 1996.

Page-Jones, Practical Guide to Structured System Design 1980
Page-Jones, M.: The Practical Guide to Structured System Design, New York 1980.

Plattner, Products & Organization 1997
Plattner, H.: Components: Products & Organizations. An Integrated and Innovative Approach to Technology and Organization for Business Solutions for Your Next Millenium, Vortrag Sapphire, Orlando August 1997,
URL: http://www.sap.com/events/amsterdam/index.htm.

Porter, Wettbewerbsvorteile 1992
Porter, M.E.: Wettbewerbsvorteile, 3rd ed., Frankfurt am Main 1992.

Porter/Millar, How Information gives you Competitive Advantage 1985
Porter, M.E., MilIar, V.E.: How Information gives you Competitive Advantage, in: Harvard Business Review, July–August 1985, pp. 149–160.

Pree, Komponentenbasierte Softwareentwicklung 1997
Pree, W.: Komponentenbasierte Softwareentwicklung mit Frameworks, Heidelberg 1997.

Reiter/Wilhelm/Geib, Multiperspektivischen Injormationsmodellierung 1997
Reiter, C., Wilhelm, G., Geib, T.: Toolunterstützung bei der multiperspektivischen Informationsmodellierung, in: Management & Computer 5(1997)1, pp. 5–10.

Reuter, Sicherheits- und Integritätsbedingungen 1987
Reuter, A.: Maßnahmen zur Wahrung von Sicherheits- und Integritätsbedingungen, in: Lockemann, P.C., Schmidt, J.W. (Ed.): Datenbank-Handbuch, Berlin et. al. 1987.

Rockart, Critical Success Factors 1982
Rockart, J.F.: Current Uses of the Critical Success Factors Process, in: Proceedings of the 14th Annual Conference of the Society for Information Management, o. O. 1982.

Rumbaugh et. al., Object-Oriented Modeling and Design 1991
Rumbaugh, J. et. al.: Object-Oriented Modeling and Design, Englewood Cliffs 1991.

Rupietta, Organisationsmodellierung 1992
Rupietta, W.: Organisationsmodellierung zur Unterstützung kooperativer Vorgangsbearbeitung, in: Wirtschaftsinformatik, 34(1992)1, pp. 26–37.

SAP, White Paper Business Framework 1996
SAG AG (Ed.): White Paper Business Framework, Walldorf 1996,
URL: http://www.sap-ag.de/bfw/media/pdf/50016528.pdf.

SAP, White Paper Business Objects 1997
SAP AG (Ed.): White Paper Business Objects, Walldorf 1997,
URL: http://www.sap-ag.de/bfw/media/pdf/50016527.pdf.

Scheer, A.-W.: ARIS — House ofBusiness Engineering 1996
Scheer, A.-W.: ARIS — House of Business Engineering, in: IDS Prof. Scheer GmbH (Ed.): Scheer magazin spezial, Oktober 1996.

Scheer, ARIS — Business Process Frameworks 1998
Scheer, A.-W.: ARIS — Business Process Frameworks, Berlin et. al. 1998.

Scheer, CIM 1994
Scheer, A.-W.: CIM (Computer Integrated Manufacturing) — Towards the Factory of the Future, 3rd ed., Berlin et. al. 1994.

Scheer, Data Warehouse 1996
Scheer, A.-W.: Data Warehouse und Data Mining: Konzepte der Entscheidungs-unterstützung, in: IM Information Management 11(1996)1, pp. 74–75.

Scheer, Principles of Efficient Information Management 1991
Scheer, A.- W.: Principles of Efficient Information Management, 2nd ed., Berlin et. al. 1991.

Scheer, Projektsteuerung 1978
Scheer, A.-W.: Projektsteuerung, Wiesbaden 1978.
Scheer, Business Process Engineering 1994
Scheer, A.-W.: Business Process Engineering — Reference Models for IndustriaI Enterprises, 2nd ed., Berlin et. al. 1994.

Scheer/Nüttgens/Zimmermann, oEPK 1997
Scheer, A.-W., Nüttgens, M., Zimmermann, V.: Objektorientierte Ereignisgesteuerte Prozeßkette (oEPK) — Methode und Anwendung, in.: Veröffentlichungen des Instituts für Wirtschaftsinformatik, Nr. 141, Saarbrücken 1997.

Schlageter/Stucky, Datenbanksysteme 1983
Schlageter, G., Stucky, W.: Datenbanksysteme: Konzepte und Modelle, 2nd ed., Stuttgart 1983.

Schröder, Business Engineer 1997
Schröder, G.: Configure to Order using the R/3 Business Engineer, Vortrag Sapphire, Amsterdam, Juni 1997,
URL: http://www.sap.com/events/amsterdam/sessions/indexl.htm.

Shlaer/Mellor, Object Oriented Systems Analysis 1988
Shlaer, S., Mellor, S.J.: Object Oriented Systems Analysis, Englewood Cliffs 1988.

Siemens Nixdorf, ComUnity 1997
Siemens Nixdorf Informationssysteme AG (Ed.): ComUnity Visual Framework-Overview and Architecture 1997,
URL: http://comunity.sni.de/public/cvf/crf_en/info.htm.

Sikora/Steinparz, Computer & Kommunikation 1988
Sikora, H., Steinparz, F.X.: Computer & Kommunikation, Telekommunikation–Computervernetzung–Kommunikationsarchitektur–PC/Host-Kommunikation. Übersicht, Zusammenhänge und Fallstudien, München 1988.

Sinz, Datenmodellierung im SERM 1993
Sinz, E.J.: Datenmodellierung im Strukturierten Entity-Relationship-Modell (SERM), in: Müller-Ettrich, G. (Ed.): Fachliche Modellierung von Informationssystemen — Methoden, Vorgehen, Werkzeuge, Bonn–Paris 1993, pp. 63–126.

Sinz, Modellierung betrieblicher Informationssysteme 1996
Sinz, E.J.: Ansätze zur fachlichen Modellierung betrieblicher Informationssysteme — Entwicklung, aktueller Stand und Trends, in: Heilmann, H., Heinrich, L.J., Roithmayr, F. (Ed.): Information Engineering, München et. al. 1996, pp. 123–143.

Sloman/Kramer, Verteilte Systeme und Rechnernetze 1988
Sloman, M., Kramer, J.: Verteilte Systeme und Rechnernetze, München-Englewood Cliffs 1988.

Sommerville, Software Engineering 1987
Sommerville, I.: Software Engineering, Bonn et. al. 1987.

Stetter, Softwaretechnologie 1987
Stetter, F.: Softwaretechnologie: Eine Einführung, 4th ed., Mannheim et. al. 1987.

Tannenbaum, Computer Networks 1988
Tannenbaum, A.S.: Computer Networks, Englewood Cliffs 1988.

Taylor, Network architecture design handbook 1997
Taylor, E.D.: Network architecture design handbook: data, voice, multimedia, intranet, and hybrid networks, New York 1997.

UML Notation Guide 1997
Rational Software et. al.: UML Notation Guide, Version 1.1, 01.09.1997,
URL: http://www.rational.com/uml/html/notation.

UML Semantics 1997
Rational Software et. al.: UML Semantics, Version 1.1, 01.09.1997,
URL: http://www.rational.com/uml/html/semantics.

UML Summary 1997
Rational Software et. al.: UML Summary, Version 1.1, 01.09.1997,
URL: http://www.rational.com/uml/html/summary.

Vernadat, Enterprise Modeling and Integration 1996
Vemadat, F.B.: Enterprise Modeling and Integration: Principles and Applications, London 1996.

Vossen, Datenbank-Management-Systeme 1995
Vossen, G.: Datenmodelle, Datenbanksprachen und Datenbank-Management-Systeme, Bonn et. al. 1995.

Ward/Mellor, Real-Time Systems 1985
Ward, P.T., Mellor, S.J.: Structured Development for Real-Time Systems, Vol. 1: Introduction & Tools, Englewood Cliffs et. al. 1985.

Wedekind, Datenbanksysteme I 1991
Wedekind, H.: Datenbanksysteme I: Eine konstruktive Einführung in die Datenverarbeitung in Wirtschaft und Verwaltung, 3rd ed., Mannheim et. al. 1991.

Wirfs-Brock/Wilkerson/Wiener, Objektorientiertes Software-Design 1993
Wirfs-Brock, R., Wilkerson, B., Wiener, L.: Objektorientiertes Software-Design, München 1993.

Yourdon, E. et. al., Mainstream Objects 1996
Yourdon, E. et. al.: Mainstream Objects, München et. al. 1996.

Zachman, Framework for Information Systems Architecture 1987
Zachman, J.A.: A Framework for Information Systems Architecture, in: IBM Systems Journal, 26(1987)3, pp. 276–292.

Zencke, BAPI 1997
Zencke, P: “Unsere Kunden nutzen BAPIs sehr kreativ fьr ihre Internet-Anwendungen” , Computerzeitung, 44(1997) 10, p. 14.

Zimmermann, Produktionsfaktor Information 1972
Zimmermann, D.: Produktionsfaktor Information, Neuwied-Berlin 1972.

Zucker/Schmitz, Knowledge Flow Management 1994
Zucker, B.; Schmitz, C.: Knowledge Flow Management: Wissen nutzen statt verspielen, in: Gablers Magazin 8(1994)11–12, pp. 62–65.

Предметный указатель

А

Авторизация, *111*; *114*; *158*; *159*
Агрегация, *120*
  внедрение, *96*
  спецификация дизайна, *88*
Ассоциация, *119*
Атрибут, *80*; *148*
  ключевой, *81*
  корректировка, *87*

Б

База данных, *141*
  активная, *146*
  объектно-ориентированная, *88*; *149*
  распределенная, *161*
  реляционная, *88*; *149*
  СУБД, *96*
  схема, *96*; *142*
  транзакции, *145*
Бизнес-объект, *115*; *174*; *177*; *178*; *195*
Бизнес-приложение, *173*; *174*
Бизнес-процессы, *173; 187*
  диаграмма цепочки, *31*
  и корпоративные цели, *41*
  ключевые и вспомогательные, *25*
  ключевые процессы, *24; 117*; *139*; *174*
  описание, *15*
  событийно-управляемые, *31*
  стратегическая метамодель, *26*
  стратегический анализ, *22*
  управление, *15*

В

Внедрение
  аспект данных, *96*
  организационный аспект, *74*
  управление, *150*
  функциональный аспект, *63*
Выходной поток, *152*; *166*
  структура, *152*

Д

Диаграмма, *111*; *202*
  взаимодействия, *135*; *166*
  Де Марко, *122*
  действий, *154*; *202*
  классов, *20*; *80*
  окружения функции, *121*
  состояний, *133*
  цепочки процесса, *31*; *173*

И

Иерархическая структура
  организации, *65*
  мета-модель, *67*
  органиграммы, *66*; *190*
Информационные услуги, *155*; *169*; *171*
Информационный объект, *32*; *77*; *121*; *130*; *147*; *170*
Информационный продукт, *100*; *154*

К

Ключевые процессы, *24*
Компьютерная сеть, *72*
  конфигурация сети, *72*
  локальная сеть, *72*
  сетевые протоколы, *72*
  согласование сетей, *73*
  типы компонентов, *73*
  топология сети, *71*
  широкомасштабная сеть, *72*
Конкурентоспособность, *22*; *24*

Конфигурация, *19*
данных, *85*
результата, *105*
управление, *113*; *139*; *155*; *158*; *167*; *171*; *175*
функций, *53*
Конфигурирация
организации, *70*
Концепция ARIS применительно к бизнес-инжинирингу (HOBE), *15*; *17*; *18*; *192*
уровни конфигурации, *17*
Критерии эффективности, *23*
Критические факторы успеха, *25*
и параметры эффективности, *30*

М

Материальный продукт, *169*
Метамодель
описание, *20*
Метод PROMET, *33*
Мини-спецификация
исполняемые команды, *60*
команды вызова, *60*
проектирование экранов и листингов, *62*
управляющие структуры, *60*
Модели принятия решений, *49*
Моделирование, *36*
метод, *15*
объекты, *101*
стратегическое, *22*
управления, *108*
уровень, *49*; *101*
эталонная модель, *184*
Модель, *20*; *80; 117; 129; 132*
Модель бизнес-процесса в методологии BAAN, *132*
Модель линейного программирования, *49*
Модель организационного распределения результата, *157*; *166*

Н

Наследование, *118*
Непрерывное усовершенствование процессов, *17*; *177*; *191*

О

Общие функции, *42*
Объект, *118*; *134; 77*
макрообъект, *79*
микрообъект, *80*
Объектно-ориентированное моделирование, *147*
Объектно-ориентированный анализ, *118*
Операционно-ориентированный учет затрат, *85*; *166*
Описание внедрения, *20*
Описание требований, *20*
аспект данных, *77*
управление, *156*
управление, *151*
организационный аспект, *65*
результат, *101*
управление, *109*; *118*; *165*; *169*; *173*
функциональный аспект, *37*
Организационный аспект
внедрение, *74*
описание требований, *65*
спецификация дизайна, *71*
Организация, основанная на бизнес-процессах, *22*
Отношение, *88*; *143*; *163*
создание отношений, *88*

П

Позиционирование, *46*
Поток выходных данных, *152*
Поток данных, *122*
Поток затрат, *152*
Поток управления, *32*; *139*; *202*
Программа
определение, *60*

Программный дизайн
мини-спецификация, *60*
принцип абстракции, *55*
разработка модулей, *56*
этапы, *54*
Программный интерфейс BAPI, *174*
Продукт
информационный, *100*; *154*; *170*; *171*
материальный, *100*; *102*; *169*; *170*
моделирование, *101*
определение, *101;* , *102*
создание продукта, *102*
Процедуральные модели ARIS, *179*

P

Рабочее задание, *47*
Разработка модулей
классификация модулей, *59*
последовательность разработки, *56*
связь с функциями, *57*
структурная диаграмма, *58*
характеристики модуля, *56*
Разработка приложений
объектно-ориентированных, *199*
с использованием ARIS Framework, *192*
Результат
выходной поток, *35*
конфигурация, *105*
материальный продукт, *100*; *102*; *104*; *151*; *169*
описание требований, *101; 102*
типы продукта, *100*; *102 ; 104*; *151*; *169*
элементы выпуска, *26*
логические пути доступа, *91; 93; 94; 96*
Ролевая концепция, *69*
Ролевые имена, *43*; *73*

C

Сертификация качества, *179*
Сети SWOT, *33*
Система, *164*
Система ARIS, *15*
аспекты описания, *15*; *173*
информационные модели, *15*
концепция, *15*
концепция ARIS применительно к бизнес-инжинирингу, *15*; *17*
программная система ARIS Toolset, *15*; *181*; *183*; *184*; *187*; *195*
среда ARIS Framework, *176*; *192*
среда ARIS-Easy Design, *15*
фазы жизненного цикла модели, *15*
Система управления потоком, *19*; *20*; *130*; *140*; *187*
Событийно-управляемая цепочка процесса, *31*; *113*; *130*; *137*; *174*
Спецификация дизайна, *20; 88*
организационный аспект, *71*
управление, *115*; *141*; *159*; *177*
функциональный аспект, *54*
Спецификация материалов, *152*; *153*
Стандартные программные системы, *19*
Стратегическая метамодель бизнес-процесса, *26*
Стратегические цели, *22*
Стратегический анализ бизнес-процессов, *22*
Стратегическое моделирование бизнес-процессов, *17*
Структура выходного результата, *152*
Структура предприятия
организация бизнес-процессов, *23*
функциональный подход, *23*
Структурный анализ и проектирование, *117*

Т

Типы обработки
в пакетном режиме, *48*
в режиме онлайн, *48*
Триггер, *45*; *93*; *146*
Тэйлоризм, *22*

У

Унифицированный язык моделирования UML, *20*; *118*; *154*; *199*
Управление
внедрение, *150*
входные и выходные экраны, *124*
конфигурация, *113*; *139*; *155*; *158*; *167*; *171*; *175*
описание требований, *109*; *118*; *165*; *169*; *173*
организационное распределение функций, *109*
спецификация дизайна, *115*; *141*; *159*; *177*
Управление бизнес-процессами, *18*
Управление знаниями, *179*
Управление событиями, *128*; *159*; *128*
Управление сообщениями, *112*; *109*; *115*; *133*
Управляющие структуры, *60*; *143*
метамодель, *61*
описание, *60*
псевдокод, *60*
структограммы, *60*
Уровень абстракции, *49*
Услуги, *103*; *169*
информационные, *100*; *103*; *155*; *169*; *171*

Ф

Функции
активизация или блокирование, *54*
и цели, *37*
конфигурирование, *53*
обозначение, *37*
общие, *42*
параметры классификации, *40*
последовательность в процедурах, *45*
принцип, *54*
ручные, *48*
системные, *48*
структура, *38*
увязка с приложениями, *54*
Функционально-стоимостной анализ, *53*
Функциональный аспект, *36*
внедрение, *63*
описание требований, *37*
спецификация дизайна, *54*
Функциональный объект, *44*

Ц

Цели
доминирующие, *37*
и функции, *38*
корпоративные, *41*
подчиненные, *37*
Цепочка создания стоимости, *25*

Э

Эффективность
процессов, *23*
ресурсов, *22*; *24*
рынка, *22*; *24*

Я

Язык структурированных запросов SQL, *93*

Научно-популярное издание

Август-Вильгельм Шеер

ARIS — моделирование бизнес-процессов
3-е издание

Литературный редактор	*П.Н. Мачуга*
Верстка	*О.В. Романенко*
Художественныередакторы	*В.Г. Павлютин,Т.А. Тараброва*
Корректоры	*Л.А. Гордиенко, Л.В. Чернокозинская*

ООО “И. Д. Вильямс” 127055, г. Москва, ул. Лесная, д. 43, стр. 1

Подписано в печать 27.08.2008. Формат 70x100/16.
Гарнитура Petersburg. Печать офсетная.
Усл. печ. л. 18,06. Уч.-изд. л. 15,2.
Тираж 1000 экз. Заказ № 0000.

Отпечатано по технологии CtP
в ОАО “Печатный двор” им. А. М. Горького
197110, Санкт-Петербург, Чкаловский пр., 15.