ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУРГСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ
им. проф. М. А. БОНЧ-БРУЕВИЧА»
(СПбГУТ)**

Е. В. Давыдова, М. В. Котлова

ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА ИНФОРМАЦИОННЫХ СИСТЕМ

УЧЕБНОЕ ПОСОБИЕ

СПб ГУТ)))

**САНКТ-ПЕТЕРБУРГ
2017**

УДК 681.518(075.8)
ББК 32.965я73
Д13

Рецензенты:
кандидат технических наук, заведующий кафедрой робототехники
и автоматизации производственных систем
Санкт-Петербургского государственного
электротехнического университета «ЛЭТИ» им. В. И. Ульянова (Ленина)
М. П. Белов;
кандидат технических наук, доцент кафедры конструирования
и производства радиоэлектронных средств СПбГУТ
Т. В. Матюхина

Утверждено редакционно-издательским советом СПбГУТ
в качестве учебного пособия

Давыдова, Е. В.
Д13 Инструментальные средства информационных систем : учебное пособие / Е. В.Давыдова, М. В. Котлова ; СПбГУТ. – СПб., 2017. – 72 с.
ISBN 978-5-89160-138-3

Рассмотрены основные понятия процесса проектирования автоматизированных информационных систем, представлены методологии и технологии проектирования информационных систем, дан анализ инструментальных средств, используемых при проектировании информационных систем, приведены примеры работы в них.

Предназначено для подготовки студентов, обучающихся по направлению подготовки 09.03.02 «Информационные системы и технологии», а также для студентов, обучающихся по другим направлениям подготовки, изучающих дисциплины «Инструментальные средства информационных систем» и «Методы и средства проектирования информационных систем и технологий».

УДК 681.518(075.8)
ББК 32.965я73

ISBN 978-5-89160-138-3

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Содержание пособия отражает разделы учебной дисциплины «Инструментальные средства информационных систем», входящей в состав образовательной программы высшего образования направления подготовки 09.03.02 «Информационные системы и технологии (уровень бакалавриата)».

В учебном пособии приведены основные понятия процесса проектирования автоматизированных информационных систем, представлены методологии и технологии проектирования информационных систем, инструментальные средства, используемые при проектировании информационных систем, и показаны примеры работы в них.

В первом разделе дана общая характеристика информационных систем: понятие информационных систем, требования, предъявляемые к ним; понятие обеспечивающих подсистем и их функции.

Во втором разделе приведены основные подходы к проектированию информационных систем. Рассмотрено понятие реинжиниринга бизнес-процессов, даны основные сведения о методологиях и технологиях проектирования информационных систем, а также структурный подход к их проектированию.

В третьем разделе представлены инструментальные средства, используемые для проектирования информационных систем.

В четвертом разделе рассмотрена реализация методов моделирования данных и проектирования информационных систем с использованием приведенных во втором разделе инструментальных средств.

1. ОБЩАЯ ХАРАКТЕРИСТИКА ИНФОРМАЦИОННЫХ СИСТЕМ

1.1. Определение информационных систем

Сегодня в научно-технической литературе в связи с динамично протекающими процессами накопления знаний в области информационных технологий пока отсутствует устоявшееся, однозначное универсальное определение понятия «информационная система».

Федеральный закон РФ от 27 июля 2006 г. № 149-ФЗ «Об информации, информационных технологиях и о защите информации» трактует понятие информационной системы (ИС) как совокупность содержащейся в базах данных информации и обеспечивающих ее обработку информационных и технических средств. То есть ИС понимают как программно-аппаратную систему, обеспечивающую в соответствии с заложенной в нее логикой получение, обработку, хранение и вывод информации. Также под инструментальными средствами ИС понимают совокупность аппаратных и программных средств, обеспечивающих функционирование (рис. 1).

Основные обязательные *признаки* современной информационной системы следующие:

– единство системы (общая файловая система, единые стандарты и протоколы, единое управление (администрирование) и т. п.);

– возможность композиции и декомпозиции объектов системы при выполнении заданной функции.

В связи с этим будем использовать следующие определения:

информационная система (ИС) – совокупность информационных, экономико-математических методов и моделей, технических программных, технологических средств и специалистов, предназначенная для сбора, хранения, обработки и выдачи информации и принятия управленческих решений;

автоматизированная информационная система (АИС) – это система, состоящая их персонала и комплекса средств автоматизации его деятельности, реализующая информационную технологию установленных функций.

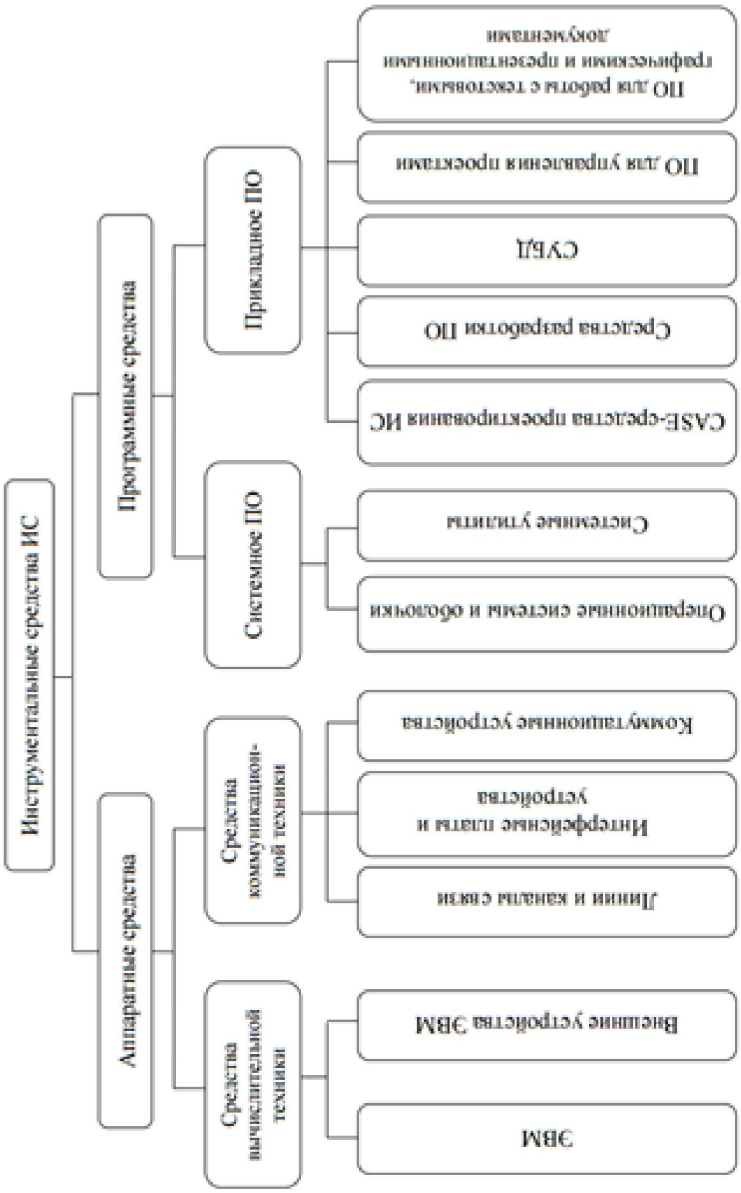

Рис. 1. Классификация инструментальных средств

6

Другими словами, ***информационная система*** – это вся инфраструктура предприятия (организации), задействованная в процессе управления всеми информационно-документальными потоками, включающая в себя следующие основные элементы:

– информационную модель, представляющую собой совокупность правил и алгоритмов функционирования ИС. Информационная модель включает в себя все формы документов, структуру справочников и данных и т. д.;

– регламент развития информационной модели и правил внесения в нее изменений;

– программный комплекс, конфигурация которого соответствует требованиям информационной модели;

– аппаратно-техническую базу (компьютеры, периферийные устройства, каналы связи, системное программное обеспечение (ПО), системы управления базами данных (СУБД)).

Информационную систему можно представить в виде модели, состоящей из нескольких взаимодействующих уровней иерархии (рис. 2).

Рис. 2. Иерархическая модель информационной системы

В основании модели лежит слой *различных типов компьютеров*, являющихся средствами хранения и обработки данных. Компьютеры определяют аппаратную платформу информационной системы.

Транспортная система состоит из активных и пассивных сетевых устройств, объединяющих компьютеры в локальные и глобальные сети и обеспечивающих обмен данными. Активными сетевыми устройствами являются сетевые адаптеры и модемы компьютеров, концентраторы,

коммутаторы, маршрутизаторы и другие подобные устройства. Среда передачи данных и элементы кабельной сети составляют пассивную часть транспортной системы.

Слой *сетевых операционных систем* обеспечивает выполнение приложений пользователей и посредством транспортной системы организует доступ к ресурсам других компьютеров и предоставляет свои ресурсы в общее пользование. Операционные системы компьютеров определяют программную платформу информационной системы. Ряд активных сетевых устройств, таких как коммутаторы и маршрутизаторы, как правило работают под управлением собственных операционных систем, называемых *операционными системами межсетевого взаимодействия*. Над слоем операционных систем работают слои различных приложений. Системные сервисы служат для обработки и преобразования информации, полученной от систем управления базами данных (СУБД) и других ресурсов, в вид, удобный для восприятия конечным пользователем или прикладной программой. СУБД иногда выделяются в отдельный слой. Этим подчеркивается их высокая значимость как средства хранения в упорядоченном виде данных и выполнения базовых операций поиска и извлечения нужной информации.

Верхний слой информационной системы составляют *приложения предметной области*, специфические для конкретного предприятия (организации) или определенного типа предприятия. Это могут быть программные системы автоматизации бухгалтерского учета, проектирования, управления производством, агрегатами, технологическими процессами и др.

Информационная система предприятия создается для работы прикладных программ. Именно эти программы обеспечивают сотрудников необходимой информацией для принятия решений и автоматизируют деятельность различных служб. Поэтому при проектировании информационной системы сначала определяются требования к этим программам, а уже затем – какие системные сервисы, базы данных, операционные системы, сетевые средства, компьютеры и серверы необходимы для их эффективного функционирования.

С точки зрения программных технологий информационная система – это не один и даже не несколько программных комплексов. Можно построить структурную модель информационной системы, выделив ее основные компоненты, которые содержат программные модули определенного класса (рис. 3).

Самым нижним уровнем ИС является *хранилище*, в котором содержится вся интеллектуальная собственность предприятия. Это могут быть документы, справочники, структурные таблицы, деловые правила, описание процессов. Прямого доступа к хранилищу быть не должно, как для

пользователей, так и для различных систем предприятия. Прямой доступ имеет лишь *система управления знаниями*, которая служит своего рода шлюзом для остальных систем и формирует информационное окружение предприятия. *Система управления знаниями* объединяет идеи, знания, содержание документов и деловые правила, автоматизируя процессы, базирующиеся на знаниях, как внутри предприятия, так и между разными организациями. Для этого нужен *шлюз*, позволяющий производить обмен данными с внешними системами. Это необходимое условие, так как современные процессы направлены на объединение предприятий в корпорации, и очевидно, что передача знаний очень важна. Например, системы планирования ресурсов предприятия (ERP – Enterprise Resource Planning) не могут работать независимо – процессы, связанные с управлением финансами, складами, человеческими ресурсами, используют уже накопленные знания и приносят новые.

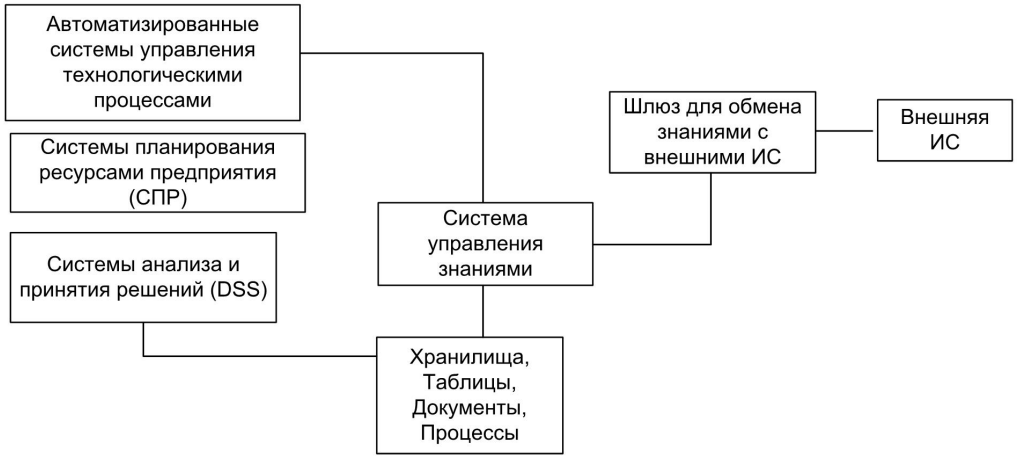

Рис. 3. Структурная модель информационной системы

Также важно выделить *класс систем анализа и принятия решений (DSS-decision support system)*, без которого жизненный цикл информации не будет завершен. На современных предприятиях интеллектуальный анализ данных становится все более важной задачей. Связано это с необходимостью аналитической обработки больших объемов информации, накопившейся в хранилищах. Такие системы помогают найти новые знания, выявить недостатки и слабые места информационной системы, оценить эффективность тех или иных процессов, установить новые информационные взаимосвязи.

В связи с развитием и широким внедрением коммуникационных и сетевых компьютерных технологий изменились и принципы построения информационных систем. Результатом такого развития стало появление

корпоративных информационных систем (КИС). Такие системы представляют целый комплекс программно-аппаратных средств, который позволяет автоматизировать бизнес-процессы и информационные потоки на предприятии или в организации в целях адекватного информационного обеспечения для повышения эффективности процесса управления.

1.2. Требования, предъявляемые к информационным системам

Рассмотрим перечень требований, предъявляемых к корпоративным информационным системам.

Функциональная полнота системы. Учитывая, что методологические подходы всех разработчиков ПО к структуризации предметной области и названию формируемых приложений различаются, общей характеристикой функциональной полноты корпоративной информационной системы является количество однократно учитываемых параметров деятельности предприятия. Считается, что для КИС значение этих параметров должно быть примерно следующим:

– количество учитываемых параметров 2–10 тыс.;

– количество таблиц баз данных 800–3000.

ИС должна обеспечивать не только формирование отчетов, но и ведение учета одновременно по российским и международным стандартам (ISA и GAAP).

Обязательным условием является локализация информационной системы: учет национального законодательства и системы расчетов; интерфейс и система помощи на национальном языке.

Система должна обеспечивать разграничение доступа к данным и функциям, предупреждать попытки несанкционированного доступа к информации.

КИС – система постоянно развивающаяся как в силу влияния внешних факторов (например, постоянных изменений в законодательстве), так и из-за изменения бизнес-функций предприятия, поэтому необходимо наличие инструментальных средств адаптации и сопровождения системы, таких как:

– управление структурой и функциями бизнес-процессов;

– изменение информационного пространства (редактирование БД, модификация структуры, полей таблиц, связей, индексов и т. п.);

– модификация интерфейсов ввода, просмотра и корректировки информации;

– изменение организационного и функционального наполнения рабочего места пользователя;

– генерация произвольных отчетов, сложных хозяйственных операций и форм.

Учитывая важность хранимых в системе данных, следует обеспечить: авторизацию информации, регистрацию времени ввода и модификации данных, ведение протокола удалений данных.

Как правило, большинство предприятий, для которых разрабатываются КИС, уже имеют установленные автоматизированные системы: АСУ ТП, САПР и т. п. Важно обеспечить обмен данными между КИС и другими программными продуктами, функционирующими на предприятии.

Для пользователей КИС большое значение имеет возможность консолидации информации: на уровне предприятий – для объединения информации филиалов, дочерних компаний, предприятий, входящих в холдинг и т. п.; на уровне отдельных задач; на уровне временных периодов – для выполнения анализа изменения тех или иных показателей за период, превышающий отчетный.

Очевидно, что КИ – это сложная система, и для обеспечения ее надежности требуются специальные средства анализа состояния системы в процессе эксплуатации:

– анализ архитектуры баз данных;

– анализ алгоритмов;

– анализ статистики: количество записей, документов, проводок, объем дисковой памяти;

– журнал выполненных операций;

– список работающих станций, внутрисистемная почта.

Некоторые специалисты рассматривают вложение средств в создание КИС как долговременные инвестиции, при этом большое значение приобретают уровень и качество обслуживания, предоставляемого разработчиком. Для заказчика оптимальной является ситуация, когда он, обратившись к одному поставщику, получает весь спектр услуг, это:

– постановка системы управления предприятием;

– консалтинг;

– решение вопросов постановки учета и документооборота;

– обучение персонала заказчика;

– внедрение КИС в опытную и промышленную эксплуатацию;

– сопровождение системы на протяжении всего ее жизненного цикла;

– проведение тематических семинаров как по проблемам методологии и организации учета, так и по вопросам использования КИС.

Из приведенного перечня требований видно, что создание корпоративной информационной системы – задача очень сложная, требующая немалых затрат.

Рис. 4. Обобщенная структура ИТ предприятия

В заключение данного раздела представим обобщенную структуру информационных технологий предприятия (рис. 4):

– САПР (CAD/CAM) – система автоматизированного проектирования-изготовления;

– АС ТПП (CAE) – автоматизированная система технологической подготовки производства;

– АСУ ТП (SCADA) – автоматизированные системы управления технологическими процессами (от эффективности которых зависит эффективность производства);

– КИС (MRP, ERP) – корпоративные информационные системы;

– ERP II – расширение ERP-системы за контуры производства (т. е. ERP +: CRM, B2B, DSS, SCM, PLM и др.);

– CRM – управление отношениями с клиентами (WF – частный случай CRM);

– B2B – электронная торговая площадка («онлайновый бизнес»);

– DSS – поддержка принятия управленческих решений;

– SPSS – статистический анализ данных;

– OLAP – анализ многомерных данных;

– MIS – автоматизированное рабочее место (АРМ) руководителя;

– SCM – управление цепочками поставок;

– PLM – управление жизненным циклом продукции;

– HR – управление персоналом, можно рассматривать как самостоятельную задачу, так и входящую в состав ERP;
– WF – электронный документооборот (электронная почта);
– УП – управление производством;
– ПУ – первичный учет;
– ПФ – планирование и бюджетирование;
– КУ – коммерческий учет;
– СУ – складской учет;
– БУ – бухгалтерский учет;
– СТ – системы телекоммуникаций.

1.3. Обеспечение информационных систем

Различают девять обеспечивающих подсистем или так называемое обеспечение АИС. Ниже приведены определения каждого вида обеспечения, его компоненты и особенности.

Информационное обеспечение – совокупность форм документов, классификаторов, нормативной базы и реализованных решений по объемам, размещению и формам существования информации, применяемой в АИС при ее функционировании.

Информационное обеспечение включает:
− описание технологических процессов;
− описание организации информационной базы;
− описание входных потоков;
− описание выходных сообщений;
− описание систем классификации и кодирования;
− формы документов;
− описание структуры массивов.

Системы классификации позволяют группировать объекты, выделяя определенные классы, которые характеризуются рядом общих свойств. *Классификаторы* представляют собой систематизированные своды, перечни классифицируемых объектов и имеют определенное (обычно числовое) обозначение. Применяются государственные, отраслевые, региональные классификаторы.

Назначение классификаторов:
− систематизация наименований кодируемых объектов;
− однозначная интерпретация одних и тех же объектов в различных задачах;
− возможность обобщения информации по заданной совокупности признаков;

− возможность сопоставления одних и тех же показателей, содержащихся в формах статистической отчетности;

− возможность поиска и обмена информацией между подсистемами и внешними АИС;

− оптимизация использования ресурсов вычислительной техники при работе с кодируемой информацией.

Используются три метода классификации объектов:

− иерархический;

− фасетный;

− дескрипторный.

В *иерархической системе* классификации каждый объект на любом уровне должен быть отнесен к одному классу, характеризуемому конкретным значением выбранного классификационного признака. Количество уровней классификации, соответствующее числу признаков, выбранных в качестве основания деления, характеризует глубину классификации.

Достоинства иерархической системы классификации: простота построения и использование независимых классификационных признаков в различных ветвях иерархической структуры.

Недостатками этой системы являются: жесткая структура, осложняющая внесение изменений и невозможность группировать объекты по заранее не предусмотренным сочетаниям признаков.

При использовании *фасетного* метода классификации допустимо выбирать признаки классификации независимо как друг от друга, так и от семантического содержания классифицируемого объекта. Признаки классификации называются фасетами (facet – рамка). Каждый фасет содержит совокупность однородных значений данного классификационного признака, причем значения в фасете могут располагаться в произвольном порядке. Схема построения фасетной системы классификации представляется в виде таблицы. Названия столбцов соответствуют выделенным классификационным признакам (фасетам). В каждой клетке таблицы хранится конкретное значение фасета. Процедура классификации состоит в присвоении каждому объекту соответствующих значений из фасетов.

Достоинствами фасетной системы классификации являются: возможность создания большой емкости классификации, т. е. использование большого числа признаков классификации и их значений для создания группировок; возможность простой модификации всей системы классификации без изменения структуры существующих группировок.

Недостаток системы – сложность ее построения, которая связана с необходимостью использовать все многообразие классификационных признаков.

Для организации поиска информации, для ведения тезаурусов (словарей) эффективно используется *дескрипторная* (описательная) система классификации, язык которой приближается к естественному языку описания информационных объектов. Особенно широко она используется в библиотечной системе поиска.

Системы классификации принципиально отличаются от систем кодирования в соответствии с определением.

Система кодирования – совокупность правил кодового обозначения объектов. Код строится на базе алфавита, состоящего из букв, цифр и других символов. Код характеризуется длиной (число позиций в коде) и структурой (порядок расположения в коде символов, используемых для обозначения классификационного признака).

Кодирование применяется для замены названия объекта на условное обозначение (код) в целях обеспечения удобной и более эффективной обработки информации.

Лингвистическое обеспечение – совокупность средств и правил для формализации естественного языка, используемых при общении пользователей и эксплуатационного персонала АИС с комплексом средств автоматизации при функционировании АИС.

Языковые средства лингвистического обеспечения делятся на две группы: традиционные языки (естественные, математические, алгоритмические, моделирования) и языки, предназначенные для диалога с ЭВМ.

Математическое обеспечение – совокупность математических методов, моделей и алгоритмов, применяемых в АИС.

В состав математического обеспечения входят:

− средства математического обеспечения (средства моделирования типовых задач управления, методы многокритериальной оптимизации, математической статистики, теории массового обслуживания и др.);

− техническая документация (описание задач, алгоритмы решения задач, экономико-математические модели);

− методы выбора математического обеспечения (методы определения типов задач, методы оценки вычислительной сложности алгоритмов, методы оценки достоверности результатов).

Методическое обеспечение – совокупность документов, описывающих технологию функционирования АИС, методы выбора и применения пользователями технологических приемов для получения конкретных результатов при функционировании АИС.

Организационное обеспечение – совокупность документов, устанавливающих организационную структуру, права и обязанности пользователей и эксплуатационного персонала АИС в условиях функционирования, проверки и обеспечения работоспособности АИС.

Организационное обеспечение реализует следующие функции:

− анализ существующей системы управления предприятием (организацией), где используется АИС, выявление задач, подлежащих автоматизации;

− подготовку задач к автоматизации, включая подготовку технических заданий и технико-экономических обоснований эффективности;

− разработку управленческих решений по изменению структуры организации и методологий решения задач, направленных на повышение эффективности системы управления.

Организационное обеспечение включает:

− методические материалы, регламентирующие процесс создания и функционирования АИС;

− совокупность средств для эффективного проектирования и функционирования АИС;

− техническую документацию, получаемую в процессе обследования предприятия, проектирования, внедрения и сопровождения системы;

− персонал (организационно-штатные структуры предприятия), проектирующий, внедряющий, сопровождающий и использующий ИС.

Правовое обеспечение – совокупность правовых норм, регламентирующих правовые отношения при функционировании АИС и юридический статус результатов ее функционирования. Правовое обеспечение реализуется в организационном обеспечении АИС.

Программное обеспечение – совокупность программ на носителях данных и программных документов, предназначенных для отладки, функционирования и проверки работоспособности АИС.

Техническое обеспечение – совокупность всех технических средств, используемых при функционировании АИС.

Выбор технических средств, организация их эксплуатации, технологический процесс обработки данных, технологическое оснащение документально оформляются.

Эргономическое обеспечение – совокупность реализованных решений в АИС по согласованию психологических, психофизиологических, антропометрических, физиологических характеристик и возможностей пользователей АИС с техническими характеристиками комплекса средств автоматизации АИС и параметрами рабочей среды на рабочих местах персонала АИС.

2. ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ

2.1. Реинжиниринг бизнес-процессов

Первые классические модели создания КИС опирались, как правило, на комплексную автоматизацию ключевых производственных, административно-управленческих и организационных функций предприятий или организаций путем использования соответствующей компьютерной инфраструктуры и программного обеспечения. Со временем стало ясно, что такое сужение функций КИС не соответствует возможностям системы. Особенность создания и внедрения КИС состоит не столько в автоматизации функциональных подсистем предприятия (организации), сколько в решении задач оптимизации бизнес-процессов и совершенствования процессной организации управления.

Методическим направлением, изучающим вопросы процессной организации систем управления предприятиями и дающим решения по их построению, является реинжиниринг бизнес-процессов РБП (BPR – Business Process Reengineering).

Автоматизация управления предприятием базируется на реинжиниринге бизнес-процессов.

Бизнес-процесс – совокупность взаимосвязанных операций (работ) по изготовлению готовой продукции или выполнению услуг на основе потребления ресурсов.

Целью реинжиниринга бизнес-процессов является системная реорганизация материальных, финансовых и информационных потоков, направленная на упрощение организационной структуры, перераспределение и минимизацию использования различных ресурсов, сокращение сроков реализации потребностей клиентов, повышение качества их обслуживания.

При этом используются следующие положения:

- несколько работ объединяются в одну;
- исполнителям делегируется право по принятию решений;
- этапы процесса выполняются в естественном порядке;
- реализуются различные версии процесса;
- работа выполняется там, где ее целесообразно делать (выход работы за границы организационных структур);
- снижаются доли работ по проверке и контролю;
- минимизируется количество согласований;
- ответственный менеджер является единственной точкой контакта с клиентом процесса;
- используются и централизованные и децентрализованные операции.

Реинжиниринг бизнес-процессов решает следующие задачи:

1) определение оптимальной последовательности выполняемых функций, которое приводит к сокращению длительности цикла изготовления и продажи товаров и услуг; обслуживание клиентов, следствием чего служат повышение оборачиваемости капитала и рост всех экономических показателей фирмы;

2) оптимизация использования ресурсов в различных бизнес-процессах, в результате которой минимизируются издержки и обеспечивается оптимальное сочетание различных видов деятельности;

3) построение адаптивных бизнес-процессов, нацеленных на быструю адаптацию к изменениям потребностей конечных потребителей продукции, производственных технологий, поведение конкурентов на рынке и, следовательно, повышение качества обслуживания клиентов в условиях динамичности внешней среды;

4) определение рациональных схем взаимодействия с партнерами и клиентами и, как следствие, рост прибыли, оптимизация финансовых потоков.

Реинжиниринг бизнес-процессов предполагает изменение архитектуры корпоративной экономической информационной системы, которая призвана:

– *на оперативном уровне* обеспечить ускорение информационных потоков, связывающих участников деловых процессов, и улучшить синхронизацию одновременно выполняемых деловых процессов;

– *на тактическом уровне* способствовать повышению качества принимаемых управленческих решений, позволяющих адаптировать деловые процессы к изменению внешней среды;

– *на стратегическом уровне* обеспечивать процесс принятия решений относительно проектирования новых и перепроектирования существующих бизнес-процессов.

Организационный анализ компании при таком подходе проводится по определенной схеме с помощью *полной бизнес-модели* компании.

На рис. 5 представлена обобщенная схема организационной бизнес-модели.

Построение бизнес-модели компании начинается с моделирования ее взаимодействия с внешней средой, т. е. с определения миссии компании.

Миссия – это:

1) деятельность, осуществляемая предприятием для того, чтобы выполнить функцию, для которой оно было учреждено, предоставление заказчикам продукта или услуги;

2) механизм, с помощью которого предприятие реализует свои цели и задачи.

Миссия компании по удовлетворению социально значимых потребностей рынка определяется как компромисс интересов рынка и компании.

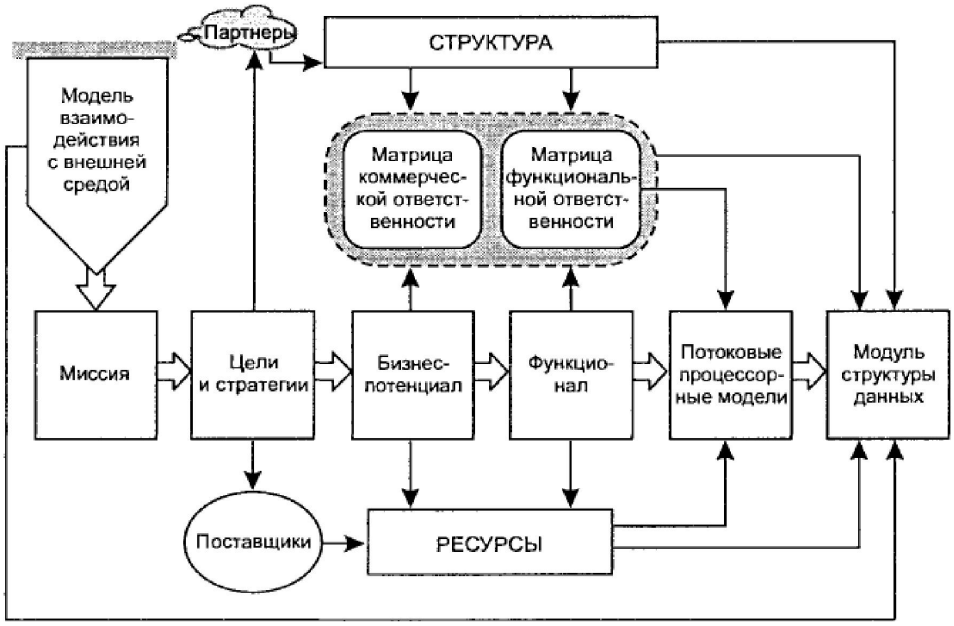

Рис. 5. Обобщенная схема организационного бизнес-моделирования

При этом миссия как атрибут открытой системы разрабатывается, с одной стороны, исходя из рыночной конъюнктуры и позиционирования компании относительно других участников внешней среды, а с другой – исходя из объективных возможностей компании и ее субъективных ценностей, ожиданий и принципов.

Определение миссии позволяет сформировать *дерево целей* компании – иерархические списки уточнения и детализации миссии.

Дерево целей формирует *дерево стратегий* – иерархические списки уточнения и детализации целей. При этом на корпоративном уровне разрабатываются стратегии роста, интеграции и инвестиции бизнесов.

Блок *бизнес-стратегий* определяет продуктовые и конкурентные стратегии, а также стратегии сегментации и продвижения.

Ресурсные стратегии определяют стратегии привлечения материальных, финансовых, человеческих и информационных ресурсов.

Функциональные стратегии определяют стратегии в организации компонентов управления и этапов жизненного цикла продукции. (Это позволяет обеспечить заказчикам необходимый продукт требуемого качества, в нужном количестве, в нужном месте, в нужное время и по приемлемой цене.)

Бизнес-потенциал определяет функционал компании – перечень бизнес-функций, функций менеджмента и функций обеспечения, требуемых для поддержания на регулярной основе указанных видов коммерческой деятельности.

Кроме того, уточняются необходимые для этого ресурсы (материальные, человеческие, информационные) и структура компании.

Построение бизнес-потенциала и функционала компании позволяет с помощью матрицы проекций определить зоны ответственности менеджмента.

Матрица проекций – модель, представленная в виде матрицы, задающей систему отношений между классификаторами в любой их комбинации.

Матрица коммерческой ответственности закрепляет ответственность структурных подразделений за получение дохода в компании от реализации коммерческой деятельности.

Ее дальнейшая детализация (путем выделения центров финансовой ответственности) обеспечивает построение финансовой модели компании, что позволяет внедрить систему бюджетного управления.

Матрица функциональной ответственности закрепляет ответственность структурных звеньев (и отдельных специалистов) за выполнение бизнес-функций при реализации процессов коммерческой деятельности (закупка, производство, сбыт и пр.), а также функций менеджмента, связанных с управлением этими процессами (планирование, учет, контроль в области маркетинга, финансов, управления персоналом и пр.).

Дальнейшая детализация матрицы (до уровня ответственности отдельных сотрудников) позволит получить функциональные обязанности персонала, что в совокупности с описанием прав, обязанностей, полномочий обеспечит разработку пакета должностных инструкций.

Описание бизнес-потенциала, функционала и соответствующих матриц ответственности представляет собой *статическое описание компании.* При этом процессы, протекающие в компании, идентифицируются, классифицируются и закрепляются за исполнителями (будущими хозяевами этих процессов). На этом этапе бизнес-моделирования формируется общепризнанный набор основополагающих внутрифирменных регламентов:

– базовое положение об организационно-функциональной структуре компании;

– пакет положений об отдельных видах деятельности (финансовой, маркетинговой и т. д.);

– пакет положений о структурных подразделениях (цехах, отделах, секторах, группах и т. п.);

– должностные инструкции.

Дальнейшее развитие (детализация) бизнес-модели происходит на этапе *динамического описания компании* на уровне *процессных потоковых моделей.*

Процессная потоковая модель – это модель, описывающая процесс последовательного во времени преобразования материальных и информационных потоков компании в ходе реализации какой-либо бизнес-функции или функции менеджмента (рис. 6).

Сначала (на верхнем уровне) описывается логика взаимодействия участников процесса, а затем (на нижнем уровне) – технология работы отдельных специалистов на своих рабочих местах.

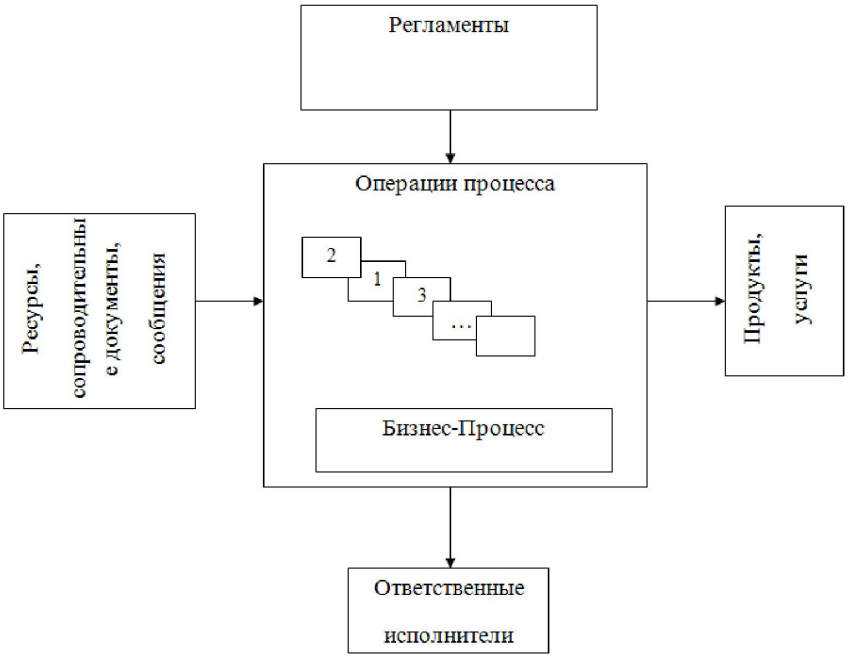

Рис. 6. Потоковая процессная модель

Организационное бизнес-моделирование завершается разработкой ***модели структур данных***, которая определяет перечень и форматы документов, сопровождающих процессы в компании, а также задает форматы описания объектов внешней среды, компонентов и регламентов самой компании. При этом создается система справочников, на основании которых получают пакеты необходимых документов и отчетов.

Такой подход позволяет описать деятельность компании с помощью универсального множества управленческих регистров (цели, стратегии, продукты, функции, организационные звенья и т. д.).

Полная бизнес-модель компании – это совокупность функционально ориентированных информационных моделей, обеспечивающая взаимосвязанные ответы на перечисленные выше вопросы (рис. 7).

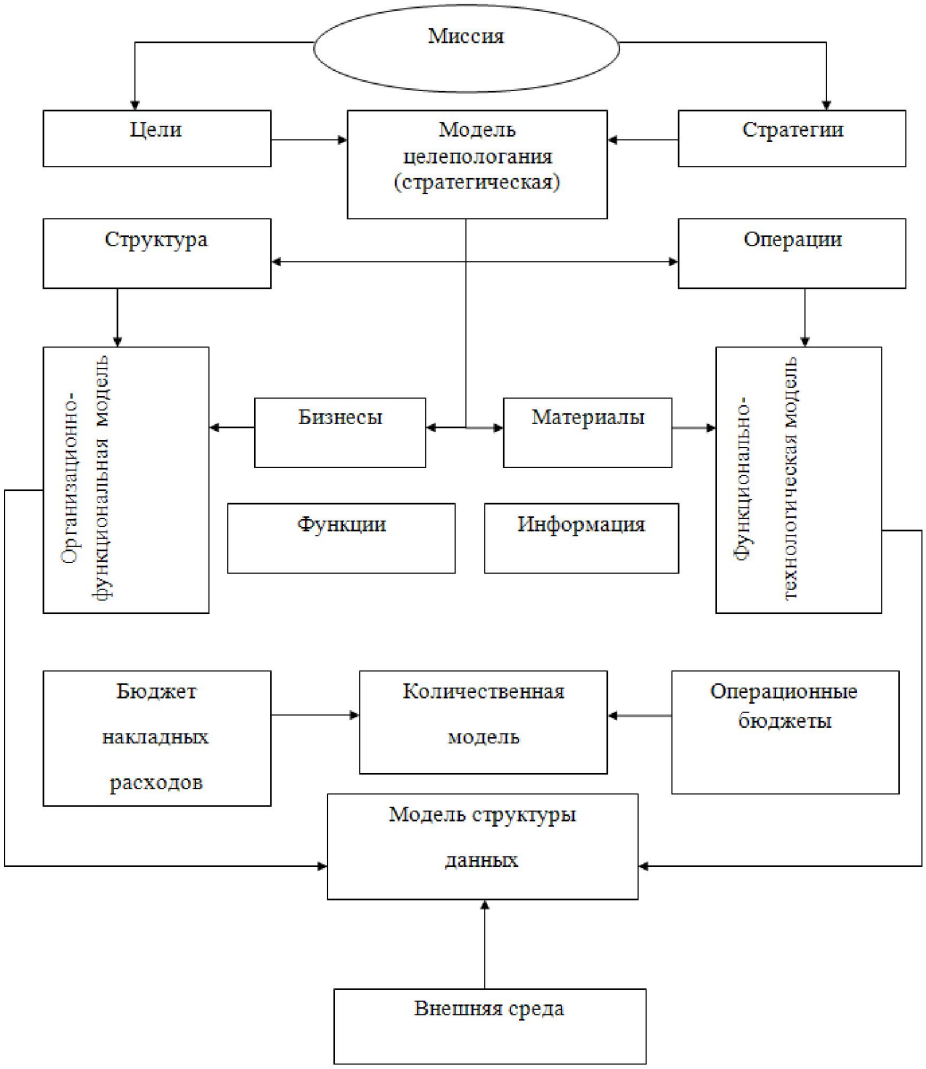

Рис. 7. Полная бизнес-модель компании

Таким образом, организационный анализ предполагает построение комплекса взаимосвязанных информационных моделей компании, который включает:

– стратегическую модель целеполагания (отвечает на вопросы: зачем компания занимается именно этим бизнесом, почему предполагает быть конкурентоспособной, какие цели и стратегии для этого необходимо реализовать);

– организационно-функциональную модель (отвечает на вопрос: кто и что делает в компании, и кто и за что отвечает);

– функционально-технологическую модель (отвечает на вопрос: что и как реализуется в компании);

– процессно-ролевую модель (отвечает на вопросы: кто?, что?, как?, кому?);

– количественную модель (отвечает на вопрос: сколько необходимо ресурсов);

– модель структуры данных (отвечает на вопрос: в каком виде описываются регламенты компании и объекты внешнего окружения).

Представленная совокупность моделей обеспечивает необходимую полноту и точность описания компании и позволяет вырабатывать понятные требования к проектируемой информационной системе.

2.2. Методологии и технологии проектирования информационных систем

Методологии, технологии и инструментальные средства проектирования (CASE-средства) составляют основу проекта любой ИС.

Методология реализуется через конкретные технологии и поддерживающие их стандарты, методики и инструментальные средства, которые обеспечивают выполнение процессов жизненного цикла (ЖЦ).

Технология проектирования определяется как совокупность трех составляющих:

– пошаговой процедуры, определяющей последовательность технологических операций проектирования;

– критериев и правил, используемых для оценки результатов выполнения технологических операций;

– нотаций (графических и текстовых средств), используемых для описания проектируемой системы.

Технологические инструкции, составляющие основное содержание технологии, должны состоять из описания последовательности технологических операций, условий, в зависимости от которых выполняется та или иная операция, и описаний самих операций (рис. 8).

Технология проектирования, разработки и сопровождения ИС должна поддерживаться комплексом согласованных CASE-средств, обеспечивающих автоматизацию процессов, выполняемых на всех стадиях ЖЦ и удовлетворять следующим общим требованиям:

– поддерживать полный ЖЦ ПО;

– гарантировать достижение целей разработки ИС с заданным качеством и в установленное время;

– обеспечивать возможность выполнения крупных проектов в виде подсистем (т. е. возможность декомпозиции проекта на составные части, разрабатываемые группами исполнителей ограниченной численности с последующей интеграцией составных частей);

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

– обеспечивать минимальное время получения работоспособной ИС;

– предусматривать возможность управления конфигурацией проекта, ведения версий проекта и его составляющих, возможность автоматического выпуска проектной документации и синхронизацию ее версий с версиями проекта;

– обеспечивать независимость выполняемых проектных решений от средств реализации ИС (СУБД, операционных систем, языков и систем программирования).

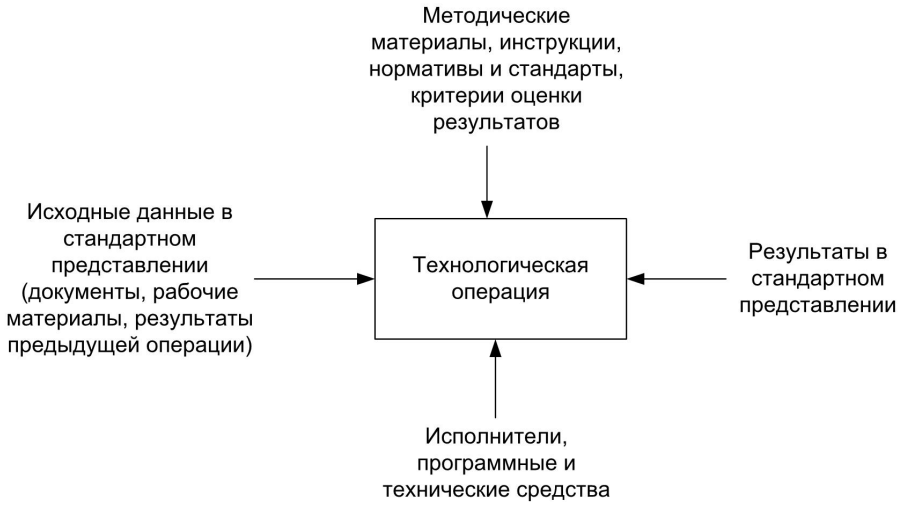

Рис. 8. Технологическая операция проектирования

2.3. Структурный подход к проектированию информационных систем

Сущность структурного подхода к разработке ИС заключается в ее декомпозиции (разбиении) на автоматизируемые функции: система разбивается на функциональные подсистемы, которые в свою очередь делятся на подфункции, подразделяемые на задачи и т. д. Процесс разбиения про-

должается вплоть до конкретных процедур. При этом автоматизируемая система сохраняет целостное представление, в котором все составляющие компоненты взаимоувязаны.

При разработке системы «снизу-вверх» от отдельных задач ко всей системе целостность теряется, возникают проблемы при информационной стыковке отдельных компонентов.

Все наиболее распространенные методологии структурного подхода базируются на ряде общих принципов. Основными из них являются следующие принципы:

– абстрагирования – заключается в выделении существенных аспектов системы;

– формализации – заключается в необходимости строгого методического подхода к решению проблемы;

– непротиворечивости – заключается в обоснованности и согласованности элементов;

– структурирования данных – заключается в том, что данные должны быть структурированы и иерархически организованы.

В структурном анализе используются в основном две группы средств, иллюстрирующих функции, выполняемые системой, и отношения между данными.

Каждой группе средств соответствуют определенные виды моделей (диаграмм), наиболее распространенными среди которых являются следующие:

– SADT (Structured Analysis and Design Technique) – модели и соответствующие функциональные диаграммы;

– DFD (Data Flow Diagrams) – диаграммы потоков данных;

– ERD (Entity-Relationship Diagrams) – диаграммы «сущность – связь».

На стадии проектирования ИС модели расширяются, уточняются и дополняются диаграммами, отражающими структуру программного обеспечения: архитектуру ПО, структурные схемы программ и диаграммы экранных форм.

Перечисленные модели в совокупности дают полное описание ИС независимо от того, является ли она существующей или вновь разрабатываемой. Состав диаграмм в каждом конкретном случае зависит от необходимой полноты описания системы.

2.4. Методологии проектирования информационных систем

Мировой опыт свидетельствует, что внедрению на предприятии ИС должно предшествовать серьезное функционально-информационное обследование предприятия в целях определения оптимальности бизнес-процессов, распределения ресурсов между функциями и т. д.

Понятие «моделирование бизнес-процессов» пришло в быт большинства аналитиков одновременно с появлением на рынке сложных программных продуктов, предназначенных для систем комплексной автоматизации управления предприятием. Подобные системы всегда подразумевают проведение глубокого предпроектного обследования деятельности. Результатом этого обследования является экспертное заключение, в котором отдельными пунктами выносятся рекомендации по устранению «узких мест» в управлении деятельностью. На основании данного заключения непосредственно перед проектом внедрения системы автоматизации проводится так называемая реорганизация бизнес-процессов. Подобные комплексные обследования предприятий являются сложными и существенно отличающимися от случая к случаю задачами.

Для решения подобных задач моделирования сложных систем существуют определенные методологии и стандарты.

Взаимосвязанная совокупность методик концептуального проектирования IDEF (Integrated Definition) разработана по программе Integrated Computer-Aided Manufacturing в США. В этой совокупности имеются методики функционального, информационного и поведенческого моделирования и проектирования, в ее состав в настоящее время входят IDEF-методики, отмеченные в табл. 1.

Таблица 1

IDEF-методики

Название	Назначение
IDEFO	Функциональное моделирование (Function Modeling Method)
IDEF1 и IDEF1X	Информационное моделирование (Information and Data Modeling Methods)
IDEF2	Поведенческое моделирование (Simulation Modeling Method)
IDEF3	Моделирование процессов (Process Flow and Object State Description Capture Method)
IDEF4	Объектно-ориентированное проектирование (Object-Oriented Design Method)
IDEF5	Систематизации объектов приложения (Ontology Description Capture Method)
IDEF6	Использование рационального опыта проектирования (Design Rationale Capture Method)
IDEF8	Взаимодействие человека и системы (Human-System Interaction Design)
IDEF9	Учет условий и ограничений (Business Constraint Discovery)
IDEF14	Моделирование вычислительных сетей (Network Design)

IDEF0 – методология функционального моделирования (Руководящий документ Госстандарта РФ «Методология функционального моделирования IDEF0»). Метод IDEF0 предназначен для функционального моделирования, т. е. моделирования выполнения функций объекта путем создания

описательной графической модели, показывающей, что, как и кем делается в рамках функционирования предприятия. Функциональная модель представляет собой структурированное изображение функций производственной системы или среды, информации и объектов, связывающих эти функции.

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

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

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

IDEF3 – методология документирования процессов, происходящих в системе, которая используется, например, при исследовании технологических процессов на предприятиях. С помощью IDEF3 описываются сценарий и последовательность операций для каждого процесса. IDEF3 имеет прямую взаимосвязь с методологией IDEF0 – каждая функция (функциональный блок) может быть представлена в виде отдельного процесса средствами IDEF3.

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

IDEF5 – методология онтологического исследования сложных систем. С помощью методологии IDEF5 онтология системы может быть описана при помощи определенного словаря терминов и правил, на основании которых могут быть сформированы достоверные утверждения о состоянии рассматриваемой системы в некоторый момент времени. На основе этих утверждений формируются выводы о дальнейшем развитии системы и производится ее оптимизация.

Развитие BPR-методик продолжается по программе *IICE* (Information Integration for Concurrent Engineering). Разработаны методики:

• **IDEF6**, направленная на сохранение рационального опыта проектирования информационных систем, что способствует предотвращению повторных ошибок;

• **IDEF8** для проектирования диалога человека с технической системой;

• **IDEF9** для анализа имеющихся условий и ограничений (в том числе физических, юридических, политических) и их влияния на принимаемые решения в процессе реинжиниринга;

• **IDEF14** для представления и анализа данных при проектировании вычислительных сетей на графическом языке с описанием конфигураций, очередей, сетевых компонентов, требований к надежности и т. п.

Основные положения стандартов IDEF0 и IDEF1X использованы также при создании комплекса стандартов ISO 10303, задающих технологию STEP для представления в компьютерных средах информации, относящейся к промышленному производству. В свою очередь стандарты STEP, совокупность языков, таких как Express и SGML, а также стандарты P-LIB и MANDATE составляют основу технологии CALS информационного обеспечения всех этапов жизненного цикла промышленных изделий.

Технология CALS призвана разрешить проблему согласования содержания и формы представления данных о промышленной продукции в территориально распределенной сети проектных и производственных узлов на основе совокупности международных стандартов и телекоммуникационных технологий. Только в этих условиях станет возможной оптимальная специализация предприятий, распределенное проектирование, минимизация затрат на освоение и эксплуатацию созданных систем.

3. ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА ПРОЕКТИРОВАНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

3.1. Инструментальные средства проектирования корпоративных информационных систем

Технология создания крупных информационных систем предъявляет особые требования к методикам реализации и программным инструментальным средствам.

Перед созданием ИС необходимо понять и описать бизнес-логику предметной области. Реализацию крупных проектов необходимо разбивать на стадии анализа, проектирования, при котором необходимо определить модули и архитектуру будущей системы, кодирования, тестирования и сопровождения. Ошибки, допущенные на предыдущей стадии, обходятся во много раз дороже, чем на текущей, следовательно, наиболее критичными являются первые стадии проекта. В связи с этим крайне важно иметь эффективные средства автоматизации ранних этапов реализации проекта.

Крупный проект невозможно реализовать без коллективной работы, которая существенно отличается от индивидуальной, поэтому при реализации крупных проектов необходимо иметь средства координации деятельности коллектива разработчиков и управления им.

Жизненный цикл создания сложной ИС сопоставим с предполагаемым временем ее эксплуатации. В современных условиях компании перестраивают свои бизнес-процессы примерно раз в два года, столько же времени требуется для создания ИС. Таким образом, может оказаться, что к моменту завершения работы над ИС она уже не соответствует новым требованиям или современным технологиям. Для создания крупной ИС необходим инструмент, в несколько раз уменьшающий время разработки ИС.

Вследствие значительного длительного жизненного цикла оказывается, что в процессе создания системы внешние условия изменились. Обычно внесение изменений в проект на поздних этапах создании ИС – весьма трудоемкий и дорогостоящий процесс. Поэтому для успешной реализации крупного проекта необходимо, чтобы инструментальные средства, на которых он реализуются, были достаточно гибкими к изменяющимся требованиям.

На современном рынке достаточно много средств разработки ИС, удовлетворяющих всем необходимым требованиям.

Для проведения анализа и реорганизации бизнес-процессов можно использовать CASE-средство верхнего уровня – BPwin, поддерживающее методологии IDEF0 (функциональная модель), IDEF3 (Work Flow Diagram) и DFD (Data Flow Diagram). Функциональная модель предназначена для описания существующих бизнес-процессов на предприятии (модель AS-IS

и модель TO-BE). Методология IDEF0 предписывает построение иерархической системы диаграмм – единичных описаний фрагментов системы.

Сначала необходимо построить контекстную диаграмму, т. е. привести описание системы в целом и ее взаимодействия с окружающим миром, после чего проводится функциональная декомпозиция. При построении диаграммы декомпозиции система разбивается на подсистемы, и каждая подсистема описывается отдельно. Затем каждая подсистема разбивается на более мелкие и так далее до достижения нужной степени подробности. После каждого сеанса декомпозиции проводится сеанс экспертизы, каждая диаграмма проверяется экспертами предметной области, непосредственно участвующими в бизнес-процессе. Такая технология создания модели позволяет построить модель, адекватную предметной области на всех уровнях абстрагирования. Если в процессе моделирования нужно осветить специфические стороны технологии предприятия, BPwin позволяет переключиться на любой ветви модели на нотацию IDEF3 или DFD и создать смешанную модель.

Нотация DFD включает такие понятия, как внешняя ссылка и хранилище данных, что делает ее более удобной (по сравнению с IDEF0) для моделирования документооборота. Методология IDEF3 включает элемент «перекресток», что позволяет описать логику взаимодействия компонентов системы.

На основе модели, разработанной с помощью средства BPwin, можно построить модель данных. Для построения модели данных используется удобный инструмент – ERwin. Процесс преобразования модели BPwin в модель данных плохо формализуется и поэтому полностью не автоматизирован, инструмент для облегчения построения модели данных на основе функциональной модели представляет собой механизм двунаправленной связи BPwin – ERwin. ERwin имеет два уровня представления модели – логический и физический. Физический уровень данных – это отображение системного каталога, который зависит от конкретной реализации СУБД.

ERwin позволяет проводить процессы прямого и обратного проектирования БД. По модели данных можно сгенерировать схему БД или автоматически создать модель данных на основе информации системного каталога. ERwin позволяет выравнивать модель и содержимое системного каталога после редактирования. Интеграция ERwin со средствами разработки клиентской части (Power Builder, SQL Windows, Visua lBasic, Delphi) позволяет автоматически генерировать код приложения, который готов к компиляции и выполнению. Создание современных информационных систем, основанных на широком использовании распределенных вычислений, объединении традиционных и новейших информационных технологий, требует тесного взаимодействия всех участников проекта: менеджеров,

бизнес и системных аналитиков, администраторов баз данных, разработчиков. Использующиеся на разных этапах и разными специалистами средства моделирования и разработки должны быть объединены общей системой организации совместной работы.

ModelMart – это масштабируемая многопользовательская среда моделирования, которая предоставляет современные и простые в использовании сервисы и дает возможность разработчикам моделей эффективно работать вместе. ModelMart является интегрирующим ядром для средств моделирования ERwin и BPwin и улучшает взаимодействие между разработчиками за счет использования управляемой среды. В результате повышается качество разработок и производительность труда проектировщиков.

Система ModelMart представляет собой хранилище моделей, к которому открыт доступ для участников проекта создания информационной системы. ModelMart удовлетворяет всем требованиям, предъявляемым к средствам разработки крупных информационных систем. Каждый участник проекта имеет инструмент поиска и доступа к интересующей его модели в любое время. При совместной работе используются три режима:

– незащищенный;

– защищенный;

– режим просмотра.

В режиме просмотра запрещается любое изменение моделей. В защищенном режиме модель, с которой работает один пользователь не может быть изменена другими пользователями. В незащищенном режиме пользователи могут работать с общими моделями в реальном масштабе времени. Возникающие при этом конфликты разрешаются при помощи специального модуля – Intelligent Conflict Resolution (ICR). В дополнение к стандартным средствам организации совместной работы ModelMart позволяет сохранять множество версий, снабженных аннотациями, с последующим сравнением предыдущих и новых версий. При необходимости возможен возврат к предыдущим версиям.

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

Для каждого участника проекта определяются права доступа, в соответствии с которыми они получают возможность работать только с определенными моделями. Права доступа могут быть определены как для групп, так и для отдельных участников проекта. Роль специалистов, участвующих

в различных проектах, может меняться, поэтому в ModelMart можно определять и управлять правами доступа участников проекта к библиотекам, моделям и даже к специфическим областям модели.

ModelMart реализована на архитектуре клиент-сервер. В качестве платформы реализации хранилища выбраны СУБД Sybase, Microsoft SQL Server и Oracle. Клиентскими приложениями являются ERwin 3.x и BPwin 2.0x. В следующих версиях ModelMart предполагается открыть доступ к хранилищу моделей через API, что позволит постоянно наращивать возможности интегрированной среды путем включения новых инструментов моделирования и анализа. При разработке крупных проектов критичным становится время реализации проекта. Одним из решений проблемы может стать автоматическая генерация кода приложения (клиентской части) CASE-средствами на основе модели предметной области. Хотя ERwin решает эту задачу, код генерируется на основе модели IDEF1X, т. е. фактически на основе реляционной модели данных, которая непосредственно не содержит информацию о бизнес-процессах. Как следствие этого, сгенерированный код не может полностью обеспечить функциональность приложения со сложной бизнес-логикой. Существует альтернативная технология кодогенерации, которая лишена этого недостатка, – объектно-ориентированное проектирование, реализованное в RationalRose (RationalSoftware).

RationalRose позволяет строить объектные модели в различных нотациях (OMT, UML, Буч) и генерировать на основе полученной модели приложения на языках программирования C++, VisualBasic, PowerBuilder, Java, Ada, Smalltalk и др. Поскольку генерация кода реализована на основе знаний предметной области, а не на основе реляционной структуры данных, полученный код более полно отражает бизнес-логику. RationalRose поддерживает не только прямую генерацию кода, но и обратное проектирование, т. е. создание объектной модели по исходному коду приложения.

3.2. CASE-средства

Во многих случаях эффективную информационную систему не удается построить вручную. Это объясняется следующими причинами:

- не обеспечивается достаточно глубокий анализ требований к данным;
- большая длительность процесса структурирования;
- трудность учета и согласования изменений, сделанных в системе несколькими разработчиками;
- ограничения сроков на разработку системы;
- и др.

При разработке крупных информационных систем происходит концентрация сложности на начальных этапах (анализ требований и проекти-

рование спецификаций системы), в то время как сложность и трудоемкость последующих этапов остается относительно невысокой. Для преодоления сложностей начальных этапов разработки предназначен структурный анализ – метод исследования, которое начинается с общего обзора системы и затем детализируется, приобретая иерархическую структуру со все большим числом уровней. На каждом уровне рассматривается ограниченное число элементов (обычно от 3 до 6–8), каждый из которых в свою очередь может быть декомпозирован на составляющие детали на следующем уровне. При этом соблюдаются строгие формальные правила записи информации (обычно используются диаграммы различных типов).

Такая технология получила название CASE (Computer Aided Software Engeneering – создание программного обеспечения с помощью компьютера).

Под CASE-технологией будем понимать комплекс программных средств, поддерживающих процессы создания и сопровождения программного обеспечения, включая анализ и формулировку требований, проектирование, генерацию кода, тестирование, документирование, обеспечение качества, конфигурационное управление и управление проектом (CASE-средство может обеспечивать поддержку только в заданных функциональных областях или в широком диапазоне функциональных областей) [5].

Основные черты CASE-технологии:

− использование методологии структурного проектирования «сверху-вниз»;

− разработка прикладной системы представляется в виде последовательных этапов, представленных на рис. 9;

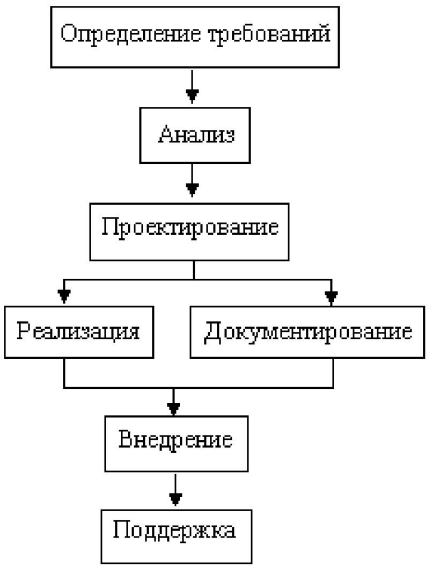

Рис. 9. Этапы разработки прикладной системы

− поддержка всех этапов жизненного цикла информационной системы, начиная с самых общих описаний предметной области до получения и сопровождения готового программного продукта;

− поддержка репозитория, хранящего спецификации проекта информационной системы на всех этапах ее разработки;

− возможность одновременной работы с репозиторием многих разработчиков;

− автоматизация различных стандартных действий по проектированию и реализации приложения.

Как правило, CASE-системы поддерживают следующие этапы процесса разработки.

Моделирование и анализ деятельности пользователей в рамках предметной области. Здесь осуществляется функциональная декомпозиция, определение иерархий (вложенности) функций, построение диаграмм потоков данных. Перечень информационных объектов, которыми манипулируют функции, передается на следующий этап проектирования.

Концептуальное моделирование – создание модели «сущность-связь» на основе перечня объектов, полученного на предыдущем этапе. Здесь уточняются характеристики каждого объекта (атрибуты), устанавливаются связи между объектами.

Реляционное моделирование – преобразование модели «сущность-связь» в соответствии с требованиями реляционной модели (реляционная модель допускает только бинарные связи, не разрешает существование атрибутов у связей, не поддерживает связи типа $n : m$).

Генерация схемы базы данных. Результатом выполнения данного этапа является набор SQL-операторов, описывающих создание схемы базы данных (CREATE TABLE, CREATE INDEX,...), с учетом особенностей целевой СУБД.

Генерация прототипов программных модулей по иерархии функций и потокам данных. Для каждого модуля автоматически подготавливается описание используемых им фрагментов данных (таблицы, атрибуты, индексы), а также создаются заготовки экранных форм или отчетов.

Современные CASE-средства поддерживают процессы инжиниринга и автоматизированного реинжиниринга.

Идеальное объектно-ориентированное CASE-средство (табл. 2) должно содержать четыре основных блока: анализ, проектирование, разработку и инфраструктуру.

Основные требования к блоку анализа:

− возможность выбора выводимой на экран информации из всей совокупности данных, описывающих модели;

− согласованность диаграмм при хранении их в репозитории;

– внесение комментариев в диаграммы и соответствующую документацию для фиксации проектных решений;

– возможность динамического моделирования в терминах событий;

– поддержка нескольких нотаций (не менее трех нотаций – Г. Буча, И. Джекобсона и OMT).

Таблица 2

Идеальное объектно-ориентированное CASE-средство

<table>
<tr><th colspan="2">Анализ</th><th>Проектирование</th><th>Реализация</th></tr>
<tr><td colspan="2">Возможность добавлять различные пояснительные надписи представления к диаграммам и скрывать в документацию не нужные в данный момент слои системы</td><td>Возможность просматривать и выбирать элементы и бизнес-объекты для использования в системе</td><td>Возможность генерировать заготовки программного кода на нескольких объектно-ориентированных языках</td></tr>
<tr><td colspan="2">Среда для создания диаграмм разнообразных моделей</td><td>Возможность создания пользовательского интерфейса (поддержка OLE, ActiveX, OpenDoc, HTML)</td><td>Возможность проверки кода на синтаксическую корректность</td></tr>
<tr><td>Поддержка различных нотаций</td><td>Возможность динамического моделирования событий в системе</td><td>Возможность определения бизнес-модели и бизнес-правил</td><td>Возможность генерировать код для 4GL и клиент-серверных продуктов (PowerBuilder, Forte, VisualAge, VisualWorks)</td></tr>
<tr><td>Возможность генерации документации для печати</td><td>Возможность динамической коррекции одной диаграммы из другой</td><td>Возможность связи с объектно-ориентированными базами данных и распределенными модулями (поддержка COBRA, DCOM, ПОР, HTML)</td><td></td></tr>
<tr><td colspan="4">Инфраструктура</td></tr>
<tr><td colspan="2">Контроль версий. Блокирование и согласование частей системы при групповой разработке</td><td>Репозиторий</td><td>Возможность реинжиниринга программного кода, 4GL, клиент-серверных систем в диаграммы моделей</td></tr>
</table>

Основные требования к блоку проектирования:

– поддержка всего процесса проектирования приложения;

– возможность работы с библиотеками, средствами поиска и выбора;

– возможность разработки пользовательского интерфейса;

– поддержка стандартов OLE, ActiveX и доступ к библиотекам HTML или Java;

– поддержка разработки распределенных или двух- и трехзвенных клиент-серверных систем (работа с CORBA, DCOM, Internet).

Основные требования к блоку реализации:

– генерация кода полностью из диаграмм;

− возможность доработки приложений в клиент-серверных CASE-средствах типа Power Builder;

− реинжиниринг кодов и внесение соответствующих изменений в модель системы;

− наличие средств контроля, которые позволяют выявлять несоответствие между диаграммами и генерируемыми кодами и обнаруживать ошибки как на стадии проектирования, так и на стадии реализации.

Основные требования к блоку инфраструктуры:

− наличие репозитория на основе базы данных, отвечающего за генерацию кода, реинжиниринг, отображение кода на диаграммах, а также обеспечивающего соответствие между моделями и программными кодами;

− обеспечение командной работы (многопользовательской работы и управление версиями) и реинжиниринга.

Выделим основные критерии оценки и выбора CASE-средств:

1) функциональные характеристики:

− среда функционирования: проектная среда, программное обеспечение/технические средства, технологическая среда;

− функции, ориентированные на фазы жизненного цикла: моделирование, реализация, тестирование;

− общие функции: документирование, управление конфигурацией, управление проектом;

2) надежность;

3) простота использования;

4) эффективность;

5) сопровождаемость;

6) переносимость;

7) общие критерии (стоимость, затраты, эффект внедрения, характеристики поставщика).

Данные критерии подробно изложены в стандартах IEEE: Std 1348–1995; IEEE recommended Practice for the Adoption of Computer – Aided Software Engineering (CASE) Tools; IEEE Std 1209–1992 Recommended Practice for the Evaluation and Selection of CASE Tools.

4. МОДЕЛИРОВАНИЕ ДАННЫХ

4.1. CASE-метод Баркера

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

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

Нотация ERD была впервые введена П. Ченом (Chen) и получила дальнейшее развитие в работах Баркера.

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

Сущность (Entity) – реальный либо воображаемый объект, имеющий существенное значение для рассматриваемой предметной области, информация о котором подлежит хранению (рис. 10).

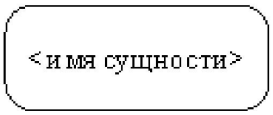

Рис. 10. Графическое изображение сущности

Каждый экземпляр сущности должен однозначно идентифицироваться и отличаться от всех других экземпляров данного типа сущности. Каждая сущность должна обладать некоторыми свойствами, например:

- уникальное имя, и к одному и тому же имени должна всегда применяться одна и та же интерпретация. Одна и та же интерпретация не может применяться к различным именам, если только они не являются псевдонимами;
- один или несколько атрибутов, которые либо принадлежат сущности, либо наследуются через связь;
- один или несколько атрибутов, которые однозначно идентифицируют каждый экземпляр сущности;
- любое количество связей с другими сущностями модели.

Легко заметить, что сущности, которые могут быть идентифицированы с главным менеджером, это автомашины и продавцы. Продавцу важны автомашины и связанные с их продажей данные. Для администратора важны покупатели, автомашины, продавцы и контракты. Исходя из этого выделяются 4 сущности (автомашина, продавец, покупатель, контракт), которые изображаются на диаграмме следующим образом (рис. 11).

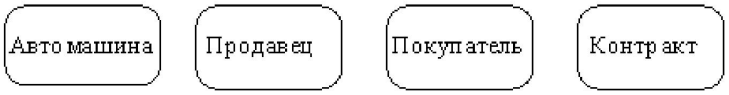

Рис. 11. Графическое изображение сущностей

Следующим шагом моделирования является идентификация связей.

Связь (Relationship) – поименованная ассоциация между двумя сущностями, значимая для рассматриваемой предметной области. Связь – это ассоциация между сущностями, при которой, как правило, каждый экземпляр одной сущности, называемой родительской сущностью, ассоциирован с произвольным (в том числе нулевым) количеством экземпляров второй сущности, называемой сущностью-потомком, а каждый экземпляр сущности-потомка ассоциирован в точности с одним экземпляром сущности-родителя. Таким образом, экземпляр сущности-потомка может существовать только при существовании сущности-родителя.

Связи может даваться имя, выражаемое грамматическим оборотом глагола и помещаемое возле линии связи. Имя каждой связи между двумя данными сущностями должно быть уникальным, но имена связей в модели не обязаны быть уникальными. Имя связи всегда формируется с точки зрения родителя, так что предложение может быть образовано соединением имени сущности-родителя, имени связи, выражения степени и имени сущности-потомка.

Например, связь продавца с контрактом может быть выражена следующим образом:

- продавец может получить вознаграждение за один или более контрактов;
- контракт должен быть инициирован одним продавцом.

Степень связи и обязательность графически изображены на рис. 12.

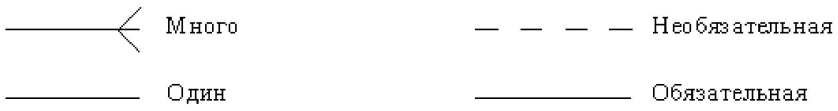

Рис. 12. Графическое изображение связей

Таким образом, два предложения, описывающие связь продавца с контрактом, графически выражены на рис. 13.

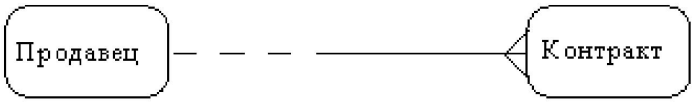

Рис. 13. Представление связи между сущностями

Описав также связи остальных сущностей, получим схему, представленную на рис. 14.

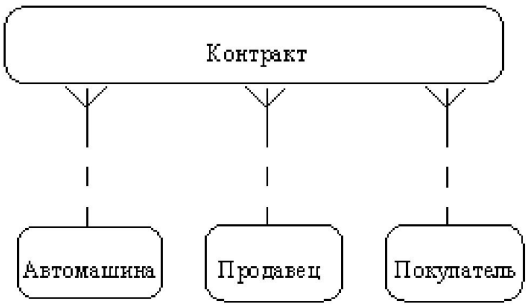

Рис. 14. Сущности с установленными связями

Последним шагом моделирования является идентификация атрибутов.

Атрибут – любая характеристика сущности, значимая для рассматриваемой предметной области и предназначенная для квалификации, идентификации, классификации, количественной характеристики или выражения состояния сущности. Атрибут представляет тип характеристик или свойств, ассоциированных со множеством реальных или абстрактных объектов (людей, мест, событий, состояний, идей, пар предметов и т. д.). Экземпляр атрибута – это определенная характеристика отдельного элемента множества. Экземпляр атрибута определяется типом характеристики и ее значением, называемым значением атрибута. В ER-модели атрибуты ассоциируются с конкретными сущностями. Таким образом, экземпляр сущности должен обладать единственным определенным значением для ассоциированного атрибута.

Атрибут может быть либо обязательным, либо необязательным (рис. 15). Обязательность означает, что атрибут не может принимать неопределенных значений (nullvalues). Атрибут может либо быть описательным (т. е. обычным дескриптором сущности), либо входить в состав уникального идентификатора (первичного ключа).

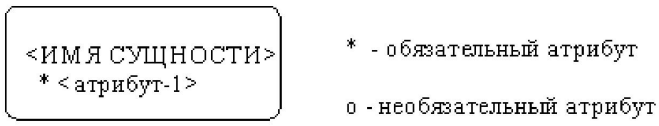

Рис. 15. Атрибуты сущностей

Уникальный идентификатор – это атрибут или совокупность атрибутов и/или связей, предназначенная для уникальной идентификации каждого экземпляра данного типа сущности. В случае полной идентификации каждый экземпляр данного типа сущности полностью идентифицируется

своими собственными ключевыми атрибутами, в противном случае в его идентификации участвуют также атрибуты другой сущности-родителя (рис. 16).

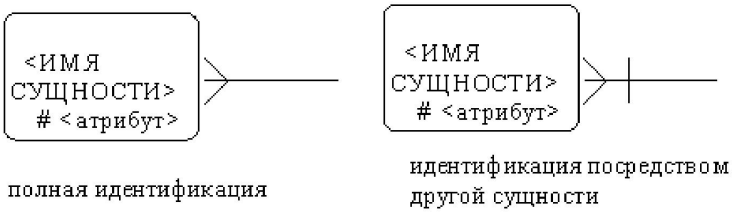

Рис. 16. Уникальный идентификатор

Каждый атрибут идентифицируется уникальным именем, выражаемым грамматическим оборотом существительного, описывающим представляемую атрибутом характеристику. Атрибуты изображаются в виде списка имен внутри блока ассоциированной сущности, причем каждый атрибут занимает отдельную строку. Атрибуты, определяющие первичный ключ, размещаются наверху списка и выделяются знаком « # ».

Каждая сущность должна обладать хотя бы одним возможным ключом. Возможный ключ сущности – это один или несколько атрибутов, чьи значения однозначно определяют каждый экземпляр сущности. При существовании нескольких возможных ключей один из них обозначается в качестве первичного ключа, а остальные – как альтернативные ключи.

С учетом имеющейся информации дополним построенную ранее диаграмму (рис. 17).

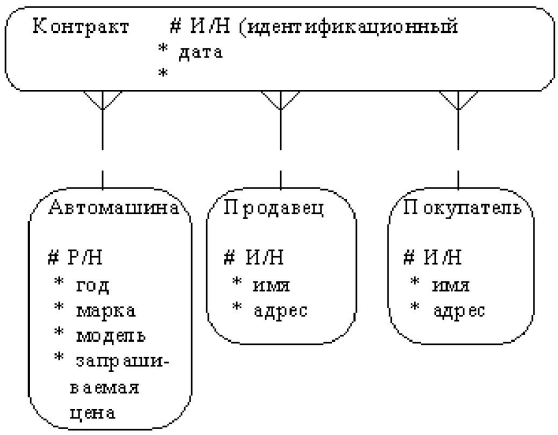

Рис. 17. Основная диаграмма

Помимо перечисленных основных конструкций модель данных может содержать ряд дополнительных.

Подтипы и супертипы. Одна сущность является обобщающим понятием для группы подобных сущностей (рис. 18).

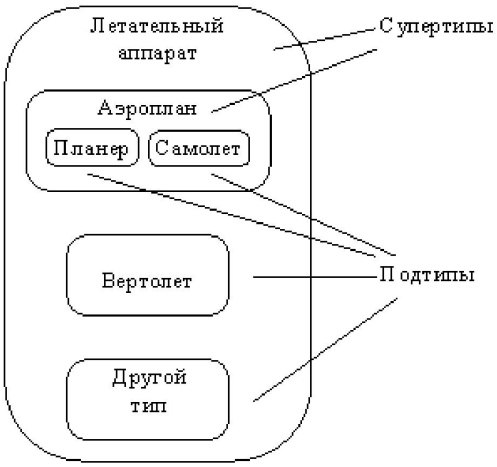

Рис. 18. Подтипы и супертипы

Взаимно исключающие связи. Каждый экземпляр сущности участвует только в одной связи из группы взаимно исключающих связей (рис. 19).

Рекурсивная связь. Сущность может быть связана сама с собой (рис. 20).

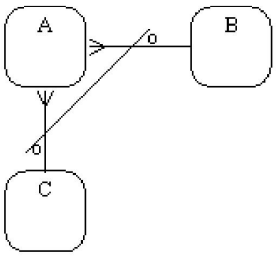

Рис. 19. Изображение взаимно исключающие связи

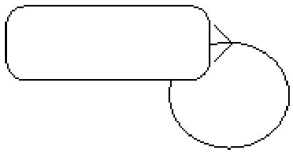

Рис. 20. Рекурсивная связь

Неперемещаемые (non-transferrable) связи. Экземпляр сущности не может быть перенесен из одного экземпляра связи в другой (рис. 21).

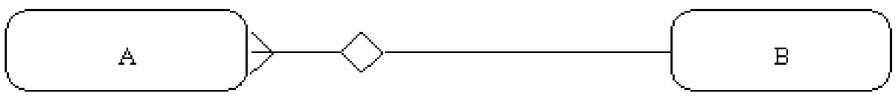

Рис. 21. Неперемещаемая связь

4.2. Методология IDEF1

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

Сущность в методологии IDEF1X является независимой от идентификаторов или просто независимой, если каждый экземпляр сущности может быть однозначно идентифицирован без определения его отношений с другими сущностями. Сущность называется зависимой от идентификаторов или просто зависимой, если однозначная идентификация экземпляра сущности зависит от его отношения к другой сущности (рис. 22).

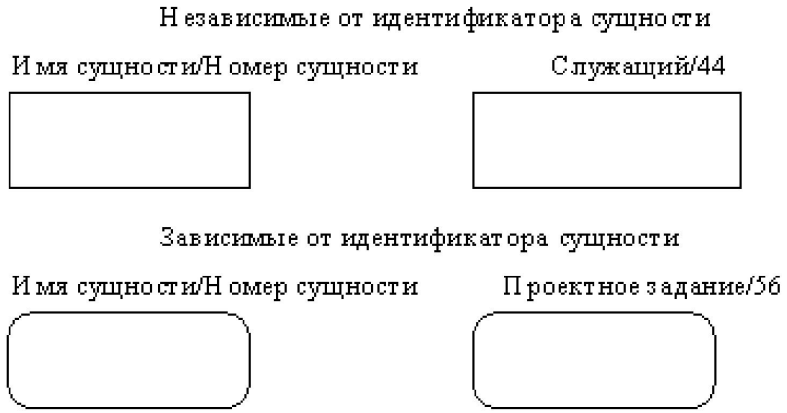

Рис. 22. Сущности

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

Связь может дополнительно определяться с помощью указания степени или мощности (количества экземпляров сущности-потомка, которое может существовать для каждого экземпляра сущности-родителя).

В IDEF1X могут быть выражены следующие мощности связей, когда каждый экземпляр сущности-родителя:

– может иметь ноль, один или более связанных с ним экземпляров сущности-потомка;

– должен иметь не менее одного связанного с ним экземпляра сущности-потомка;

– должен иметь не более одного связанного с ним экземпляра сущности-потомка;

– связан с некоторым фиксированным числом экземпляров сущности-потомка.

Если экземпляр сущности-потомка однозначно определяется своей связью с сущностью-родителем, то связь называется идентифицирующей, в противном случае – неидентифицирующей.

Связь изображается линией, проводимой между сущностью-родителем и сущностью-потомком с точкой на конце линии у сущности-потомка. Мощность связи обозначается, как показано на рис. 23 (мощность по умолчанию – N).

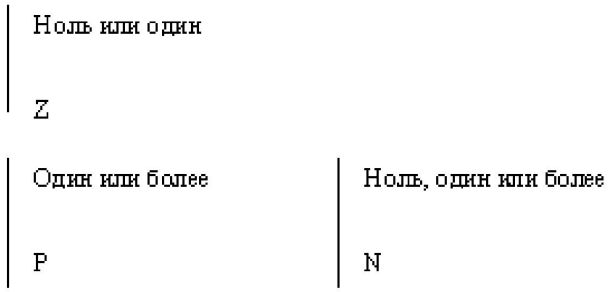

Рис. 23. Мощность связи

Идентифицирующая связь между сущностью-родителем и сущностью-потомком изображается сплошной линией (рис. 24). Сущность-потомок в идентифицирующей связи является зависимой от идентификатора сущностью. Сущность-родитель в идентифицирующей связи может быть как независимой, так и зависимой от идентификатора сущностью (это определяется ее связями с другими сущностями).

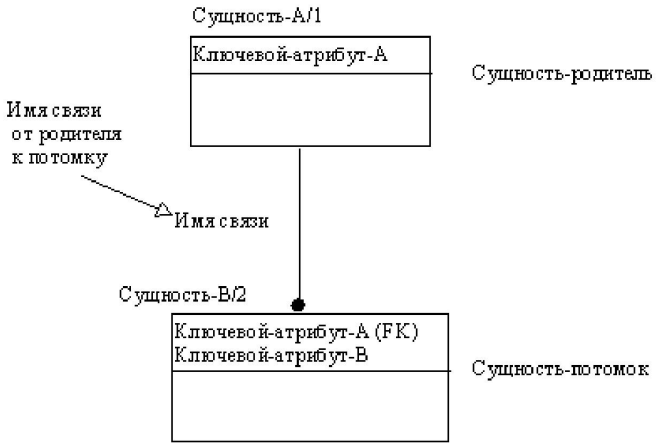

Рис. 24. Идентифицирующая связь

Пунктирная линия изображает неидентифицирующую связь (рис. 25). Сущность-потомок в неидентифицирующей связи будет независимой от идентификатора, если она не является также сущностью-потомком в какой-либо идентифицирующей связи.

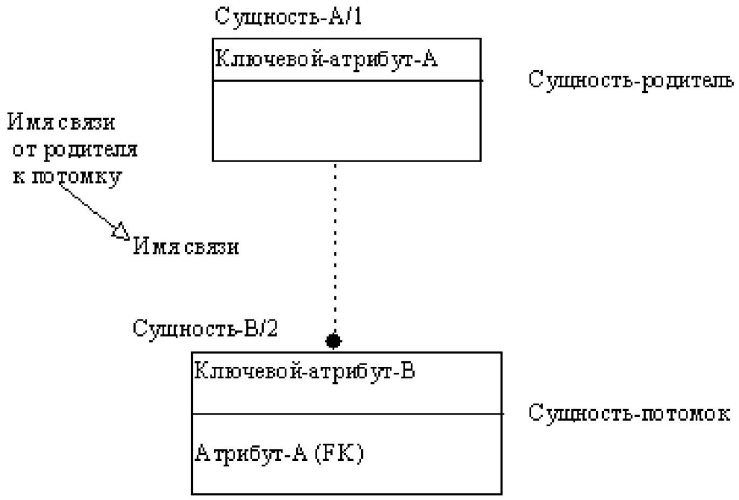

Рис. 25. Неидентифицирующая связь

Атрибуты изображаются в виде списка имен внутри блока сущности. Атрибуты, определяющие первичный ключ, размещаются наверху списка и отделяются от других атрибутов горизонтальной чертой (рис. 26).

Сущности могут иметь также внешние ключи (ForeignKey), которые используются в качестве части или целого первичного ключа или неключевого атрибута. Внешний ключ изображается путем размещения внутри блока сущности имен атрибутов, после которых следуют буквы FK в скобках (рис. 27).

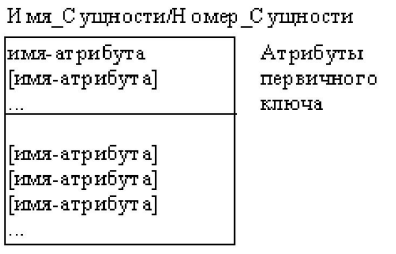

Рис. 26. Атрибуты и первичные ключи

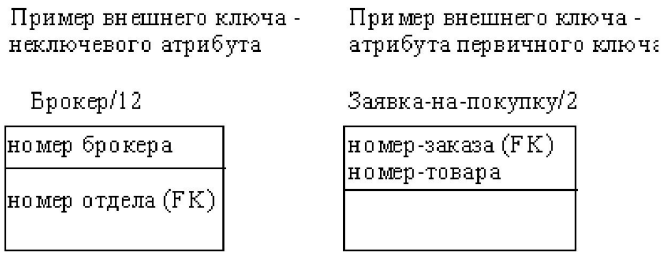

Рис. 27. Примеры внешних ключей

4.3. Подход, используемый в CASE-средстве Vantage Team Builder

В CASE-средстве VantageTeamBuilder используется один из вариантов нотации П. Чена. На ER-диаграммах сущность обозначается прямоугольником, содержащим имя сущности (рис. 28), а связь – ромбом, связанным линией с каждой из взаимодействующих сущностей. Числа над линиями означают степень связи.

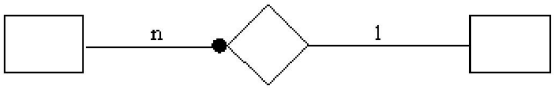

Рис. 28. Обозначение сущностей и связей

Связи являются многонаправленными и могут иметь атрибуты (за исключением ключевых). Выделяют два вида связей:

- необязательная (optional);
- слабая (weak).

В **необязательной связи** (рис. 29) могут участвовать не все экземпляры сущности.

Рис. 29. Необязательная связь

В отличие от необязательной связи в **полной (total)** связи участвуют все экземпляры хотя бы одной из сущностей. Это означает, что экземпляры такой связи существуют только при условии существования экземпляров другой сущности. Полная связь может иметь один из 4 видов: обязательная связь, слабая связь, связь «супертип-подтип» и ассоциативная связь.

Обязательная (mandatory) связь описывает связь между «независимой» и «зависимой» сущностями. Все экземпляры зависимой («обязательной») сущности могут существовать только при наличии экземпляров независимой («необязательной») сущности, т. е. экземпляр «обязательной» сущности может существовать только при условии существования определенного экземпляра «необязательной» сущности.

В примере, приведенном на рис. 30, подразумевается, что каждый автомобиль имеет по крайней мере одного водителя, но не каждый служащий управляет машиной.

Рис. 30. Обязательная связь

В **слабой связи** существование одной из сущностей, принадлежащей некоторому множеству («слабой»), зависит от существования определенной сущности, принадлежащей другому множеству («сильной»), т. е. экземпляр «слабой» сущности может быть идентифицирован только посредством экземпляра «сильной» сущности. Ключ «сильной» сущности является частью составного ключа «слабой» сущности.

Слабая связь всегда является бинарной и подразумевает обязательную связь для «слабой» сущности. Сущность может быть «слабой» в одной связи и «сильной» в другой, но не может быть «слабой» более чем в одной связи. Слабая связь может не иметь атрибутов.

В примере на рис. 31 показано, что ключ (номер) строки в документе может не быть уникальным и должен быть дополнен ключом документа.

Связь «супертип-подтип» представлена на рис. 32. Общие характеристики (атрибуты) типа определяются в сущности-супертипе, сущность-подтип наследует все характеристики супертипа. Экземпляр подтипа существует только при условии существования определенного экземпляра супертипа. Подтип не может иметь ключ (он импортирует ключ из супертипа). Сущность, являющаяся супертипом в одной связи, может быть подтипом в другой связи. Связь супертипа не может иметь атрибутов.

Рис. 31. Слабая связь

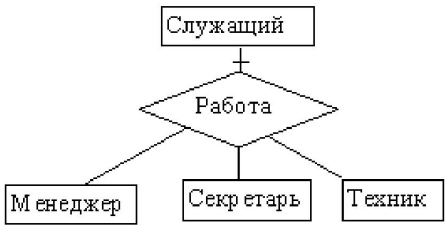

Рис. 32. Связь «супертип-подтип»

В **ассоциативной связи** каждый экземпляр связи (ассоциативный объект) может существовать только при условии существования определенных экземпляров каждой из взаимосвязанных сущностей. Ассоциативный объект – объект, являющийся одновременно сущностью и связью. Ассоциативная связь – это связь между несколькими «независимыми» сущностями и одной «зависимой» сущностью. Связь между независимыми сущностями имеет атрибуты, которые определяются в зависимой сущности. Таким образом, зависимая сущность определяется в терминах атрибутов связи между остальными сущностями.

В примере на рис. 33 самолет выполняет посадку на взлетную полосу в заданное время при определенных скорости и направлении ветра. Поскольку эти характеристики применимы только к конкретной посадке, они

являются атрибутами посадки, а не самолета или взлетной полосы. Пилот, выполняющий посадку, связан гораздо сильнее с конкретной посадкой, чем с самолетом или взлетной полосой.

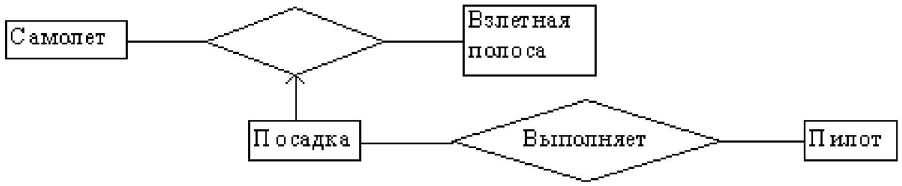

Рис. 33. Ассоциативная связь

Первичный ключ каждого типа сущности помечается звездочкой « * ».

ER-диаграмма должна подчиняться следующим правилам:

− каждая сущность, каждый атрибут и каждая связь должны иметь имя (однако связь супертипа или ассоциативная связь могут не иметь имени);

− имя сущности должно быть уникально в рамках модели данных;

− имя атрибута должно быть уникально в рамках сущности;

− имя связи должно быть уникально, если для нее генерируется таблица БД;

− каждый атрибут должен иметь определение типа данных;

− сущность в необязательной связи должна иметь ключевой атрибут. То же самое относится к сильной сущности в слабой связи, супертипу в связи «супертип-подтип» и необязательной сущности в обязательной (полной) связи;

− подтип в связи «супертип-подтип» не может иметь ключевой атрибут;

− в ассоциативной или слабой связи может быть только одна ассоциативная (слабая) сущность;

− связь не может быть одновременно обязательной, «супертип-подтип» или ассоциативной.

4.4. Пример использования структурного подхода

4.4.1. *Описание предметной области*

В данном примере используется методология Yourdon, реализованная в CASE-средстве VantageTeamBuilder.

В качестве предметной области используется описание работы видеобиблиотеки. Запросы рассматриваются администрацией видеобиблиотеки с использованием информации о клиентах, фильмах и лентах. При этом проверяется и обновляется список арендованных лент, а также проверяются записи о членстве в библиотеке. Администрация контролирует возвраты

лент, используя информацию о фильмах, лентах и список арендованных лент, который обновляется. Обработка запросов на фильмы и возвратов лент включает следующие действия: если клиент не является членом библиотеки, он не имеет права на аренду. Если требуемый фильм имеется в наличии, администрация информирует клиента об арендной плате. Однако если клиент просрочил срок возврата имеющихся у него лент, ему не разрешается брать новые фильмы. Когда лента возвращается, администрация рассчитывает арендную плату плюс пени за несвоевременный возврат.

Видеобиблиотека получает новые ленты от своих поставщиков. Когда новые ленты поступают в библиотеку, необходимая информация о них фиксируется. Информация о членстве в библиотеке содержится отдельно от записей об аренде лент.

Администрация библиотеки регулярно готовит отчеты за определенный период времени о членах библиотеки, поставщиках лент, выдаче определенных лент и лентах, приобретенных библиотекой.

4.4.2. *Организация проекта*

Весь проект разделяется на 4 фазы: анализ, глобальное проектирование (проектирование архитектуры системы), детальное проектирование и реализация (программирование).

На фазе анализа строится модель среды (EnvironmentalModel). Построение модели среды включает:

- анализ поведения системы (определение назначения ИС, построение начальной контекстной диаграммы потоков данных (DFD) и формирование матрицы списка событий (ELM), построение контекстных диаграмм);
- анализ данных (определение состава потоков данных и построение диаграмм структур данных (DSD), конструирование глобальной модели данных в виде ER-диаграммы).

Назначение ИС определяет соглашение между проектировщиками и заказчиками относительно назначения будущей ИС, общее описание ИС для самих проектировщиков и границы ИС. Назначение фиксируется как текстовый комментарий в «нулевом» процессе контекстной диаграммы.

Например, в данном случае назначение ИС формулируется следующим образом: ведение базы данных о членах библиотеки, фильмах, аренде и поставщиках. При этом руководство библиотеки должно иметь возможность получать различные виды отчетов для выполнения своих задач.

Перед построением контекстной DFD необходимо проанализировать внешние события (внешние объекты), оказывающие влияние на функционирование библиотеки. Эти объекты взаимодействуют с ИС путем информационного обмена с ней.

Из описания предметной области следует, что в процессе работы библиотеки участвуют следующие группы людей: клиенты, поставщики и руководство. Эти группы являются внешними объектами. Они не только взаимодействуют с системой, но также определяют ее границы и изображаются на начальной контекстной DFD как терминаторы (внешние сущности).

Начальная контекстная диаграмма изображена на рис. 34. В отличие от нотации Gane/Sarson внешние сущности обозначаются обычными прямоугольниками, а процессы – окружностями.

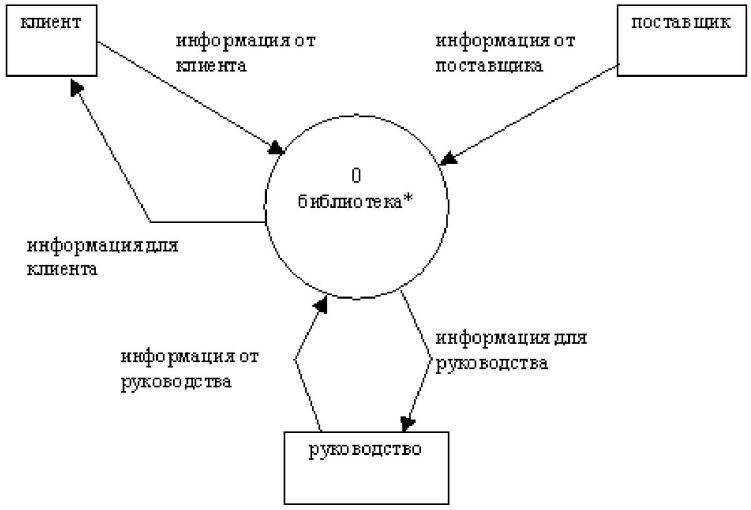

Рис. 34. Начальная контекстная диаграмма

Список событий строится в виде матрицы (ELM) и описывает различные действия внешних сущностей и реакцию ИС на них. Эти действия представляют собой внешние события, воздействующие на библиотеку. Различают следующие типы событий, приведенные в табл. 3.

Таблица 3

Типы сообщений

Аббревиатура	Тип
NC	Нормальное управление
ND	Нормальные данные
NCD	Нормальное управление/данные
TC	Временное управление
TD	Временные данные
TCD	Временное управление/данные

Все действия помечаются как нормальные данные. Эти данные являются событиями, которые ИС воспринимает непосредственно, например изменение адреса клиента, которое должно быть сразу зарегистрировано. Они появляются в DFD в качестве содержимого потоков данных.

Матрица списка событий имеет следующий вид, представленный в табл. 4.

Таблица 4

Матрица событий

Описание	Тип	Реакция
Клиент желает стать членом библиотеки	ND	Регистрация клиента в качестве члена библиотеки
Клиент сообщает об изменении адреса	ND	Регистрация измененного адреса клиента
Клиент запрашивает аренду фильма	ND	Рассмотрение запроса
Клиент возвращает фильм	ND	Регистрация возврата
Руководство предоставляет полномочия новому поставщику	ND	Регистрация поставщика
Поставщик сообщает об изменении адреса	ND	Регистрация измененного адреса поставщика
Поставщик направляет фильм в библиотеку	ND	Получение нового фильма
Руководство запрашивает новый отчет	ND	Формирование требуемого отчета для руководства

Для завершения анализа функционального аспекта поведения системы строится полная контекстная диаграмма, включающая диаграмму нулевого уровня. При этом процесс «библиотека» декомпозируется на 4 процесса, отражающие основные виды административной деятельности библиотеки. Существующие «абстрактные» потоки данных между терминаторами и процессами трансформируются в потоки, представляющие обмен данными на более конкретном уровне. Список событий показывает, какие потоки существуют на этом уровне: каждое событие из списка должно формировать некоторый поток (событие формирует входной поток, реакция – выходной поток). Один «абстрактный» поток может быть разделен на более чем один «конкретный» поток. В табл. 5 представлены основные типы потоков.

Таблица 5

Типы потоков

Потоки на диаграмме верхнего уровня	Потоки на диаграмме нулевого уровня
Информация от клиента	Данные о клиенте, Запрос об аренде
Информация для клиента	Членская карточка, Ответ на запрос об аренде
Информация от руководства	Запрос отчета о новых членах, Новый поставщик, Запрос отчета о поставщиках, Запрос отчета об аренде, Запрос отчета о фильмах
Информация для руководства	Отчет о новых членах, Отчет о поставщиках, Отчет об аренде, Отчет о фильмах
Информация от поставщика	Данные о поставщике, Новые фильмы

На приведенной DFD (рис. 35) накопитель данных «библиотека» является глобальным или абстрактным представлением хранилища данных.

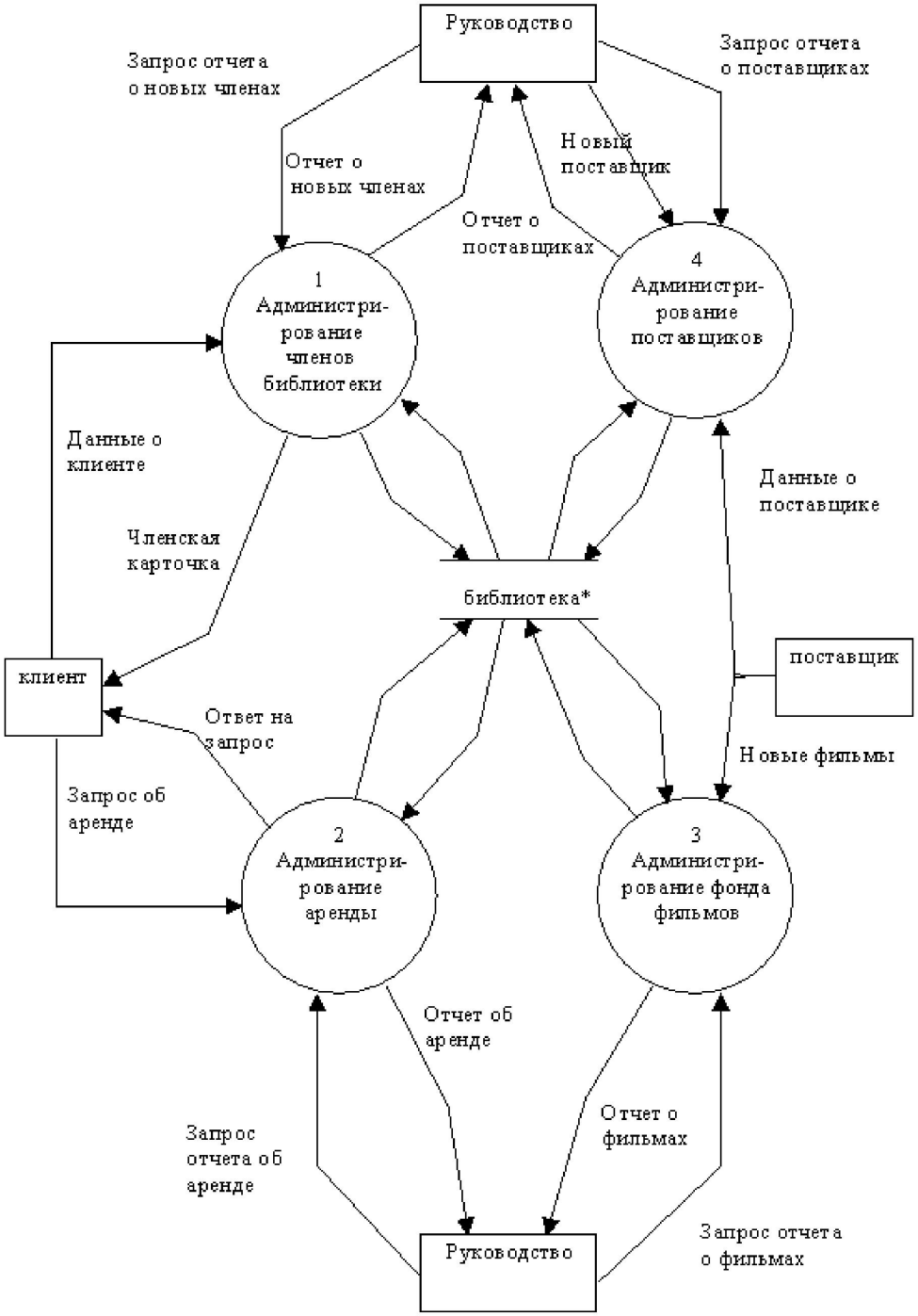

Рис. 35. Контекстная диаграмма

Анализ функционального аспекта поведения системы дает представление об обмене и преобразовании данных в системе. Взаимосвязь между «абстрактными» потоками данных и «конкретными» потоками данных на диаграмме нулевого уровня выражается в диаграммах структур данных (рис. 36).

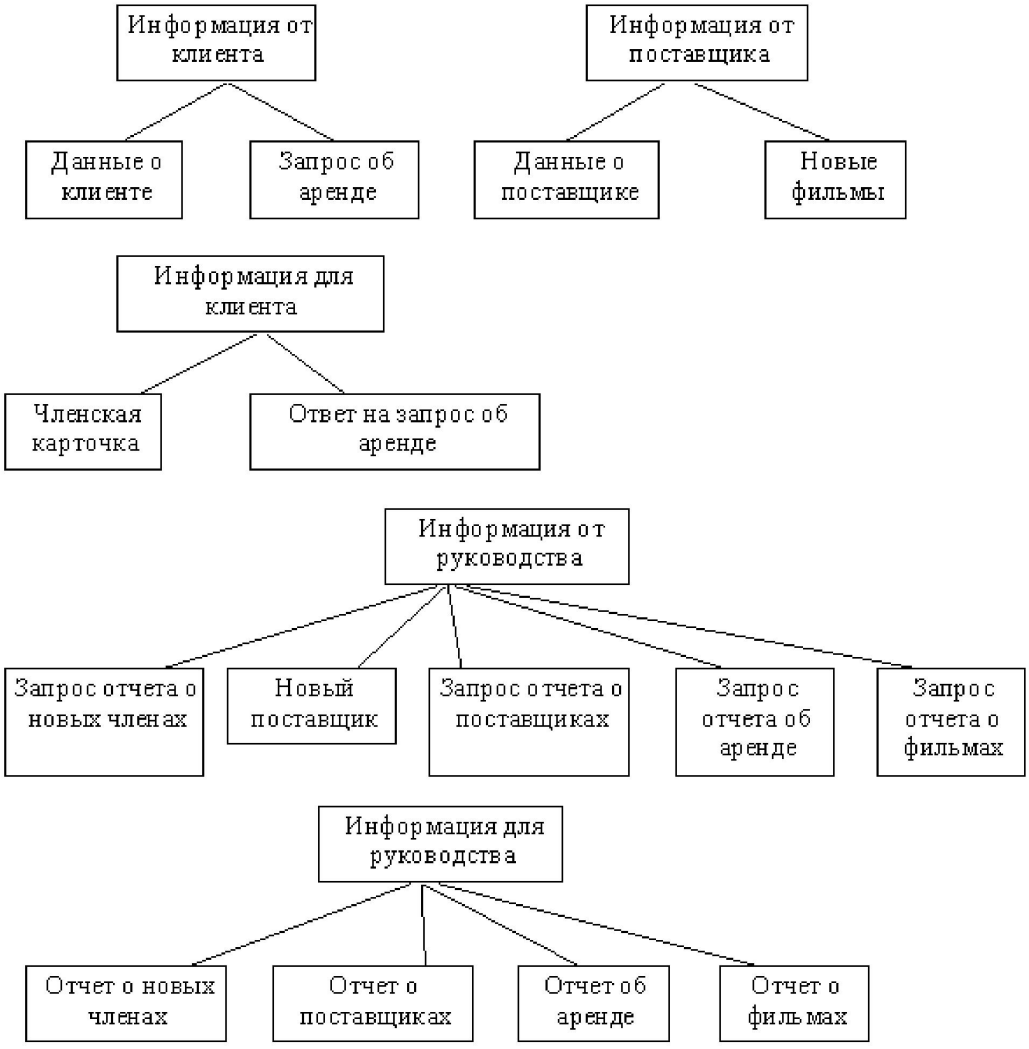

Рис. 36. Диаграмма структур данных

На фазе анализа строится глобальная модель данных, представляемая в виде диаграммы «сущность-связь» (рис. 37).

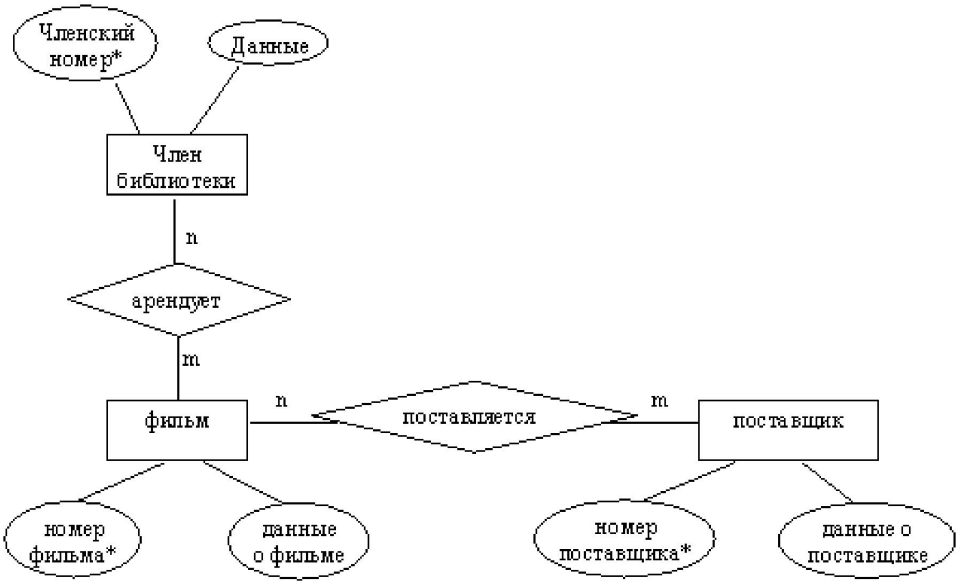

Рис. 37. Диаграмма «сущность-связь»

Между различными типами диаграмм существуют следующие взаимосвязи:

− ELM-DFD: события – входные потоки, реакции – выходные потоки;

− DFD-DSD: потоки данных – структуры данных верхнего уровня;

− DFD-ERD: накопители данных – ER-диаграммы;

− DSD-ERD: структуры данных нижнего уровня – атрибуты сущностей.

На фазе проектирования архитектуры строится предметная модель. Процесс построения предметной модели включает в себя:

− детальное описание функционирования системы;

− дальнейший анализ используемых данных и построение логической модели данных для последующего проектирования базы данных;

− определение структуры пользовательского интерфейса, спецификации форм и порядка их появления;

− уточнение диаграмм потоков данных и списка событий, выделение среди процессов нижнего уровня интерактивных и неинтерактивных, определение для них мини-спецификаций.

Результатами проектирования архитектуры являются модели:

• процессов (диаграммы архитектуры системы (SAD) и мини-спецификации на структурированном языке);

• данных (ERD и подсхемы ERD);

• пользовательского интерфейса (классификация процессов по интерактивным и неинтерактивным функциям, диаграмма последовательности форм (FSD – FormSequenceDiagram). Модель показывает, какие формы

появляются в приложении и в каком порядке. На FSD фиксируется набор и структура вызовов экранных форм. Диаграммы FSD образуют иерархию, на вершине которой находится главная форма приложения, реализующего подсистему. На втором уровне находятся формы, реализующие процессы нижнего уровня функциональной структуры, зафиксированной на диаграммах SAD.

На фазе детального проектирования строится модульная модель. Под модульной моделью понимается реальная модель проектируемой прикладной системы. Процесс ее построения включает в себя:

- уточнение модели базы данных для последующей генерации SQL-предложений;
- уточнение структуры пользовательского интерфейса;
- построение структурных схем, отражающих логику работы пользовательского интерфейса и модель бизнес-логики (StructureChartsDiagram – SCD), и привязка их к формам.

Результатами детального проектирования являются модели:

- процессов (структурные схемы интерактивных и неинтерактивных функций);
- данных (определение в ERD всех необходимых параметров для приложений);
- пользовательского интерфейса (диаграмма последовательности форм (FSD), показывающая, какие формы появляются в приложении и в каком порядке, взаимосвязь между каждой формой и определенной структурной схемой, взаимосвязь между каждой формой и одной или более сущностями в ERD).

На фазе реализации строится реализационная модель. Процесс ее построения включает в себя:

- генерацию SQL-предложений, определяющих структуру целевой БД (таблицы, индексы, ограничения целостности);
- уточнение структурных схем (SCD) и диаграмм последовательности форм (FSD) с последующей генерацией кода приложений.

На основе анализа потоков данных и взаимодействия процессов с хранилищами данных осуществляется окончательное выделение подсистем (предварительное должно было быть сделано и зафиксировано на этапе формулировки требований в техническом задании). При выделении подсистем необходимо руководствоваться принципом функциональной связанности и принципом минимизации информационной зависимости. Необходимо учитывать, что на основании таких элементов подсистемы, как процессы и данные на этапе разработки, должно быть создано приложение, способное функционировать самостоятельно. Кроме того, при группировке процессов и данных в подсистемы необходимо учитывать требования к конфигурированию продукта, если они были сформулированы на этапе анализа.

4.5. Методология DATARUN

Одной из наиболее распространенных в мире электронных методологий является DATARUN. В соответствии с методологией DATARUN ЖЦ ПО разбивается на стадии, которые связываются с результатами выполнения основных процессов, определяемых стандартом ISO 12207. Каждую стадию кроме ее результатов должен завершать план работ на следующую стадию.

Стадия формирования требований и планирования включает в себя действия по определению начальных оценок объема и стоимости проекта. Должны быть сформулированы требования и экономическое обоснование для разработки ИС, функциональные модели (модели бизнес-процессов организации) и исходная концептуальная модель данных, которые дают основу для оценки технической реализуемости проекта. Основными результатами этой стадии должны быть модели деятельности организации (исходные модели процессов и данных организации), требования к системе, включая требования по сопряжению с существующими ИС, исходный бизнес-план.

Стадия концептуального проектирования начинается с детального анализа первичных данных и уточнения концептуальной модели данных, после чего проектируется архитектура системы. Архитектура включает в себя разделение концептуальной модели на обозримые подмодели. Оценивается возможность использования существующих ИС и выбирается соответствующий метод их преобразования. После построения проекта уточняется исходный бизнес-план. Выходными компонентами этой стадии являются концептуальная модель данных, модель архитектуры системы и уточненный бизнес-план.

На стадии спецификации приложений продолжается процесс создания и детализации проекта. Концептуальная модель данных преобразуется в реляционную модель данных. Определяется структура приложения, необходимые интерфейсы приложения в виде экранов, отчетов и пакетных процессов вместе с логикой их вызова. Модель данных уточняется бизнес-правилами и методами для каждой таблицы. В конце этой стадии принимается окончательное решение о способе реализации приложений. По результатам стадии должен быть построен проект ИС, включающий модели архитектуры ИС, данных, функций, интерфейсов (с внешними системами и с пользователями), требований к разрабатываемым приложениям (модели данных, интерфейсов и функций), требований к доработкам существующих ИС, требований к интеграции приложений.

На стадии разработки, интеграции и тестирования должна быть создана тестовая база данных, частные и комплексные тесты. Проводится разработка, прототипирование и тестирование баз данных и приложений в соответствии

с проектом. Отлаживаются интерфейсы с существующими системами. Описывается конфигурация текущей версии ПО. На основе результатов тестирования проводится оптимизация базы данных и приложений. Приложения интегрируются в систему, проводится тестирование приложений в составе системы и испытания системы. Основными результатами стадии являются готовые приложения, проверенные в составе системы на комплексных тестах, текущее описание конфигурации ПО, скорректированная по результатам испытаний версия системы и эксплуатационная документация на систему.

Стадия внедрения включает в себя действия по установке и внедрению баз данных и приложений. Основными результатами стадии должны быть готовая к эксплуатации и перенесенная на программно-аппаратную платформу заказчика версия системы, документация сопровождения и акт приемочных испытаний по результатам опытной эксплуатации.

Стадии сопровождения и развития включают процессы и операции, связанные с регистрацией, диагностикой и локализацией ошибок, внесением изменений и тестированием, проведением доработок, тиражированием и распространением новых версий ПО с места его эксплуатации, переносом приложений на новую платформу и масштабированием системы. Стадия развития фактически является повторной итерацией стадии разработки.

Методология DATARUN опирается на две модели или на два представления:

- модель организации;
- модель ИС.

Методология DATARUN базируется на системном подходе к описанию деятельности организации. Построение моделей начинается с описания процессов, из которых затем извлекаются первичные данные (стабильное подмножество данных, которые организация должна использовать для своей деятельности). Первичные данные описывают продукты или услуги организации, выполняемые операции (транзакции) и потребляемые ресурсы. К первичным относятся данные, которые описывают внешние и внутренние сущности, такие как служащие, клиенты или агентства, а также данные, полученные в результате принятия решений, как, например, графики работ, цены на продукты.

Основной принцип DATARUN заключается в том, что первичные данные, если они должным образом организованы в модель данных, становятся основой для проектирования архитектуры ИС. Архитектура ИС будет более стабильной, если она основана на первичных данных, тесно связанных с основными деловыми операциями, определяющими природу бизнеса, а не на традиционной функциональной модели.

Любая ИС (рис. 38) представляет собой набор модулей, исполняемых процессорами и взаимодействующих с базами данных. Базы данных и

процессоры могут располагаться централизованно или быть распределенными. События в системе могут инициироваться внешними сущностями, такими как клиенты у банкоматов или временные события (конец месяца или квартала). Все транзакции осуществляются через объекты или модули интерфейса, которые взаимодействуют с одной или более базами данных.

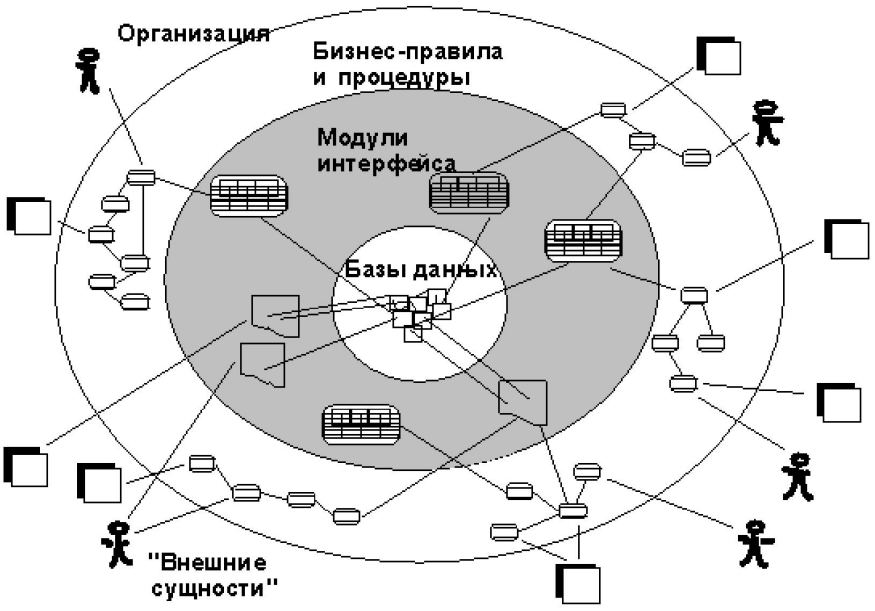

Рис. 38. Модель ИС

Подход DATARUN преследует две цели:

− определить стабильную структуру, на основе которой будет строиться ИС. Такой структурой является модель данных, полученная из первичных данных, представляющих фундаментальные процессы организации;

− спроектировать ИС на основании модели данных.

Объекты, формируемые на основании модели данных, являются объектами базы данных, обычно размещаемыми на серверах в среде клиент/сервер. Объекты интерфейса, определенные в архитектуре компьютерной системы, обычно размещаются на клиентской части. Модель данных, являющаяся основой для спецификации совместно используемых объектов базы данных и различных объектов интерфейса, обеспечивает сопровождаемость ИС. На рис. 39 представлена последовательность шагов проектирования ИС.

На рис. 40 определены модели, создаваемые в процессе разработки ИС. Для их создания используется CASE-средство Silverrun, которое обеспечивает автоматизацию проведения проектных работ в соответствии с методологией DATARUN. Предоставляемая этими средствами среда проектиро-

вания дает возможность руководителю проекта контролировать проведение работ, отслеживать выполнение работ, вовремя замечать отклонения от графика. Каждый участник проекта, подключившись к этой среде, может выяснить содержание и сроки выполнения порученной ему работы, детально изучить технику ее выполнения в гипертексте по технологиям, и вызвать инструмент (модуль Silverrun) для реального выполнения работы.

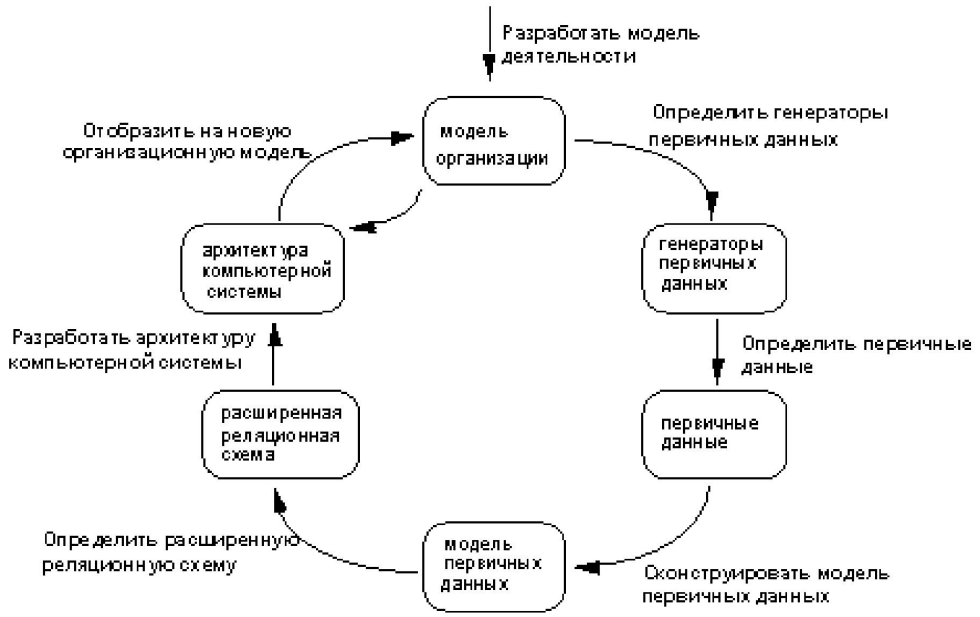

Рис. 39. Последовательность шагов проектирования системы

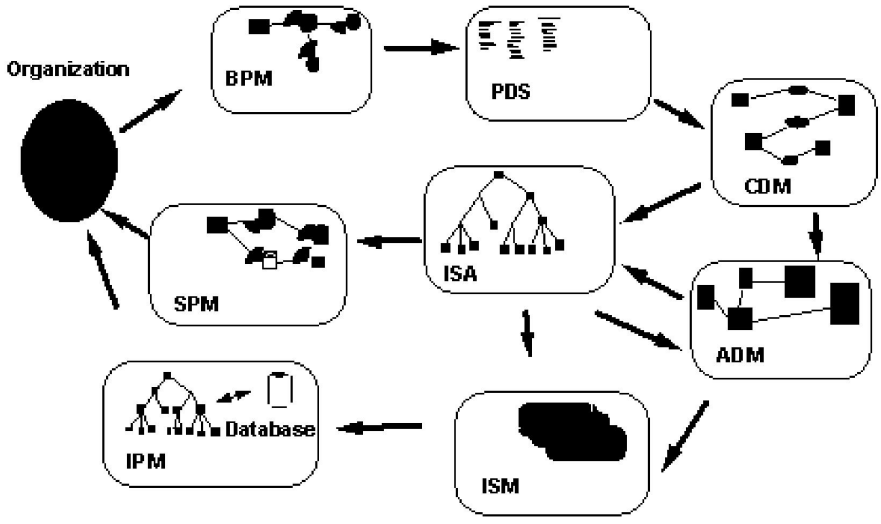

Рис. 40. Модели, создаваемые с помощью подхода DATARUN

Информационная система создается последовательным построением ряда моделей, начиная с модели бизнес-процессов и заканчивая моделью программы, автоматизирующей эти процессы:

BPM (Business Process Model) – модель бизнес-процессов;

PDS (Primary Data Structure) – структура первичных данных;

CDM (Conceptual Data Model) – концептуальная модель данных;

SPM (System Process Model) – модель процессов системы;

ISA (Information System Architecture) – архитектура информационной системы;

ADM (Application Data Model) – модель данных приложения;

IPM (Interface Presentation Model) – модель представления интерфейса;

ISM (Interface Specification Model) – модель спецификации интерфейса.

Создаваемая ИС должна основываться на функциях, выполняемых организацией. Поэтому первая создаваемая модель – это модель бизнес-процессов, построение которой осуществляется в модуле Silverrun BPM. Для этой модели используется специальная нотация BPM. В процессе анализа и спецификации бизнес-функций выявляются основные информационные объекты, которые документируются как структуры данных, связанные с потоками и хранилищами модели. Источниками для создания структур являются используемые в организации документы, должностные инструкции, описания производственных операций. Эти данные вводятся в том виде, как они существуют в деятельности организации. Нормализация и удаление избыточности производятся позже, при построении концептуальной модели данных в модуле Silverrun ERX. После создания модели бизнес-процессов информация сохраняется в репозитории проекта.

В процессе обследования работы организации выявляются и документируются структуры первичных данных. Эти структуры заносятся в репозиторий модуля BPM при описании циркулирующих в организации документов, сообщений, данных. В модели бизнес-процессов первичные структуры данных связаны с потоками и хранилищами информации.

На основе структур первичных данных в модуле Silverrun ERX создается концептуальная модель данных (ER-модель). От структур первичных данных концептуальная модель отличается удалением избыточности, стандартизацией наименований понятий и нормализацией. Эти операции в модуле ERX выполняются при помощи встроенной экспертной системы. Цель концептуальной модели данных – описать используемую информацию без деталей возможной реализации в базе данных, но в хорошо структурированном нормализованном виде.

На основе модели бизнес-процессов и концептуальной модели данных проектируется архитектура ИС. Определяются входящие в систему приложения, для каждого приложения специфицируются используемые данные

и реализуемые функции. Архитектура ИС создается в модуле Silverrun BPM с использованием специальной нотации ISA. Основное содержание этой модели – структурные компоненты системы и навигация между ними. Концептуальная модель данных разбивается на части, соответствующие входящим в состав системы приложениям.

Перед разработкой приложений должна быть спроектирована структура корпоративной базы данных. DATARUN предполагает использование базы данных, основанной на реляционной модели. Концептуальная модель данных после нормализации переносится в модуль реляционного моделирования Silverrun RDM с помощью специального моста ERX-RDM. Преобразование модели из формата ERX в формат RDM происходит автоматически без вмешательства пользователя. После преобразования форматов получается модель реляционной базы данных. Эта модель детализируется в модуле Silverrun RDM определением физической реализации (типов данных СУБД, ключей, индексов, триггеров, ограничений ссылочной целостности). Правила обработки данных можно задавать как непосредственно на языке программирования СУБД, так и в декларативной форме, не привязанной к реализации. Мосты Silverrun к реляционным СУБД переводят эти декларативные правила на язык требуемой системы, что снижает трудоемкость программирования процедур сервера базы данных, а также позволяет из одной спецификации генерировать приложения для разных СУБД.

С помощью модели системных процессов детально документируется поведение каждого приложения. В модуле BPM создается модель системных процессов, определяющая, каким образом реализуются бизнес-процессы. Эта модель создается отдельно для каждого приложения и тесно связана с моделью данных приложения.

Приложение состоит из интерфейсных объектов (экранных форм, отчетов, процедур обработки данных). Каждый интерфейс системы (экранная форма, отчет, процедура обработки данных) имеет дело с подмножеством базы данных. В модели данных приложения (созданной в модуле RDM) создается подсхема базы данных для каждого интерфейса этого приложения. Уточняются также правила обработки данных, специфичные для каждого интерфейса. Интерфейс работает с данными в ненормализованном виде, поэтому спецификация данных, как ее видит интерфейс, оформляется как отдельная подсхема модели данных интерфейса.

Модель представления интерфейса – это описание внешнего вида интерфейса, как его видит конечный пользователь системы. Это может быть как документ, показывающий внешний вид экрана или структуру отчета, так и сам экран (отчет), созданный с помощью одного из средств визуальной разработки приложений – так называемых языков четвертого поколения (4GL – Fourth Generation Languages). Так как большинство языков 4GL

позволяют быстро создавать работающие прототипы приложений, пользователь имеет возможность увидеть работающий прототип системы на ранних стадиях проектирования.

После создания подсхем реляционной модели для приложений проектируется детальная структура каждого приложения в виде схемы навигации экранов, отчетов, процедур пакетной обработки. На данном шаге эта структура детализируется до указания конкретных столбцов и таблиц базы данных, правил их обработки, вида экранных форм и отчетов. Полученная модель детально документирует приложение и непосредственно используется для программирования специфицированных интерфейсов.

Далее с помощью средств разработки приложений происходит физическое создание системы: приложения программируются и интегрируются в информационную систему.

4.6. Инструментальное средство SE Companion

Инструментальное средство SE Companion является средой, в которой реализован электронный вариант методологии DATARUN. Оно позволяет:

− создать гипертекстовое описание методологии в виде иерархии описания стадий, этапов и операций разработки;

− создать гипертекстовое описание всех методов и методик реализации процессов ЖЦ ПО;

− выделить из гипертекстового описания иерархию процессов ЖЦ ПО для планирования и управления процессом создания ПО (иерархию работ);

− изменить гипертекстовые описания ЖЦ и методов так, как это необходимо разработчику, иными словами, произвести авторизацию методологии и отследить эти изменения в иерархии работ, предназначенной для управления проектом;

− привязать к процессам ЖЦ инструментальные средства поддержки этих процессов и обеспечить вызов инструментальных средств из соответствующих экранов гипертекстового справочника;

− обеспечить просмотр гипертекстовых экранов описания используемых методов из инструментальных средств;

− обеспечить поддержку процесса управления разработкой, в частности за счет взаимодействия со средством планирования работ MS Project, оценивания трудоемкости проекта, отслеживания выполнения работ, создания графиков работ и др.

Особенно важными являются возможность авторизации методологии и интерактивный доступ любого разработчика к описанию любого метода или процесса в нужный ему момент времени. На современном этапе развития

технологии, в условиях быстрого изменения как программных и аппаратных средств, так и задач бизнеса, методология создания, сопровождения и развития ПО не должна быть неизменной; она должна иметь возможность изменяться и настраиваться на новые технологии, методы и инструментальные средства. Современные разработчики больших ИС приобретают одну или несколько методологий поставщика, а затем создают на их основе собственные методологии и технологии, адаптированные к конкретным условиям.

В SE Companion исходным документом, описывающим методологию (как процессы ЖЦ, так и все сопутствующие методы и методики), является файл в формате MS Word. Это обеспечивает возможности для описания методологии с любой степенью детализации, проведения разметки для создания гипертекста и авторизации методологии в принятом стандартном формате.

Гипертекстовое описание методологии и технологии создания ПО строится из описания процессов жизненного цикла, методов и методик, и представляет собой единый гипертекстовый документ в формате MS Help. Итоговое гипертекстовое описание получается в результате трансляции исходного документа. Все изменения и дополнения методологии производятся посредством корректировки и, возможно, дополнительной разметки исходного документа.

Описание методологии создания системы обычно состоит из раздела описания процессов ЖЦ и разделов описания методов и методик. В свою очередь, раздел описаний процессов состоит из иерархии описаний стадий, этапов и операций жизненного цикла с обязательным описанием выходных компонентов каждого процесса. Компоненты ПО создаются с применением методик и методов, описываемых в соответствующих разделах.

4.7. Инструментальные средства проектирования и разработки информационных систем

В современных информационных технологиях важное место отводится инструментальным средствам и средам разработки АИС, в частности системам разработки и сопровождения их программного обеспечения. Эти технологии и среды образуют системы, называемые CASE-системами.

Толкование аббревиатуры CASE, соответствующее двум направлениям использования CASE-систем. Первое из них – Computer Aided Software Engineering – переводится как автоматизированное проектирование программного обеспечения. Соответствующие CASE-системы часто называют инструментальными средами разработки ПО (RAD – Rapid Application

Development). Второе – Computer Aided System Engineering – подчеркивает направленность на поддержку концептуального проектирования сложных систем, преимущественно слабоструктурированных. Такие CASE-системы часто называют системами BPR (Business Process Reengineering).

Среди систем RAD различают интегрированные комплексы инструментальных средств для автоматизации всех этапов жизненного цикла ПО (такие системы называют Workbench) и специализированные инструментальные средства для выполнения отдельных функций (Tools). Средства CASE по своему функциональному назначению принадлежат к одной из следующих групп:

1) программирования;
2) управления программным проектом;
3) верификации (анализа) программ;
4) документирования.

4.8. Проектирование программного обеспечения с помощью CASE-систем

Проектирование ПО с помощью CASE-систем включает несколько этапов. Начальный этап – предварительное изучение проблемы. Результат представляется в виде исходной диаграммы потоков данных и согласуется с заказчиком. На следующем этапе выполняется детализация ограничений и функций программной системы, и полученная логическая модель вновь согласуется с заказчиком. Далее разрабатывается физическая модель, т. е. определяется модульная структура программы, выполняется инфологическое проектирование базы данных, детализируются граф-схемы программной системы и ее модулей, проектируется пользовательский интерфейс.

Примерами широко известных инструментальных сред RAD служат платформа .NET, Delphi, PowerBuilder соответственно фирм Microsoft, Borland, PowerSoft. Применение инструментальных сред существенно сокращает объем ручной работы программистов (особенно при разработке интерфейсных частей программ).

В средах быстрой разработки приложений обычно реализуется способ программирования, называемый управлением событиями. При этом достигается автоматическое создание каркасов программ, существенно сокращается объем ручного кодирования, особенно при разработке интерфейсных частей программ.

Создание сред RAD для сетевого программирования требует решения ряда дополнительных проблем, обусловливаемых много-платформенностью, обилием применяемых форматов данных и т. п. Решение этих

проблем, а также устранение некоторых особенностей языка C++, усложняющих программирование, достигнуто в языке программирования Java и платформе .NET.

4.9. Спецификации моделей информационных систем

Важное значение в процессе разработки информационных систем имеют средства спецификации их проектов. Средства спецификации в значительной мере определяют суть методов CASE.

Существует ряд способов представления моделей. Практически все способы функциональных спецификаций имеют следующие общие черты:

• модель имеет иерархическую структуру, представляемую в виде диаграмм нескольких уровней;

• элементарной частью диаграммы каждого уровня является конструкция «вход – функция – выход»;

• необходимая дополнительная информация содержится в файлах поясняющего текста.

В большинстве случаев функциональные диаграммы – это диаграммы потоков данных (DFD – Data Flow Diagram). В DFD блоки (прямоугольники) соответствуют функциям, дуги – входным и выходным потокам данных. Поясняющий текст дается в виде «словарей данных», в которых указываются компонентный состав потоков данных, число повторений циклов и т. п. Для описания структуры информационных потоков можно использовать нотацию Бэкуса–Наура.

Разработка DFD начинается с построения диаграммы верхнего уровня, отражающей связи программной системы, представленной в виде единого процесса, с внешней средой. Декомпозиция процесса проводится до уровня, где фигурируют элементарные процессы, которые могут быть представлены одностраничными описаниями алгоритмов (мини-спецификациями) на языке программирования.

Для описания информационных моделей наибольшее распространение получили диаграммы «сущность – связь» (ERD – Entity-RelationsDiagrams), фигурирующие, например в методике IDEF1X.

Поведенческие модели описывают процессы обработки информации. В системах CASE их представляют в виде граф-схем, диаграмм перехода состояний, таблиц решений, псевдокодов (языков спецификаций), языков программирования, в том числе языков четвертого поколения (4GL).

В граф-схемах блоки, как и в DFD, используют для задания процессов обработки, но дуги имеют иной смысл: они описывают последовательность передач управления (вместе со специальными блоками управления).

В диаграммах перехода состояний узлы соответствуют состояниям моделируемой системы, дуги – переходам из состояния в состояние, атрибуты дуг – условиям перехода и инициируемым при их выполнении действиям. Очевидно, что, как и в других конечно-автоматных моделях, кроме графической формы представления диаграмм перехода состояний можно использовать также табличные формы. Так, при изоморфном представлении с помощью таблиц перехода состояний каждому переходу соответствует строка таблицы, в которой указываются исходное состояние, условие перехода, инициируемое при этом действие и новое состояние после перехода.

Близкий по своему характеру способ описания процессов основан на таблицах (или деревьях) решений. Каждый столбец таблицы решений соответствует определенному сочетанию условий, при выполнении которых осуществляются действия, указанные в нижерасположенных клетках столбца.

В псевдокодах алгоритмы записываются с помощью как средств некоторого языка программирования (преимущественно для управляющих операторов), так и естественного языка (для выражения содержания вычислительных блоков). Используются конструкции (операторы) следования (условные) цикла.

Языки четвертого поколения направлены на описание программ как совокупностей заранее разработанных программных модулей, поэтому возможно соответствие одной команды языка 4GL значительному фрагменту программы на языке 3GL. Примерами языков 4GL могут служить Informix-4GL, JAM, NewEra.

Мини-спецификации процессов могут быть выражены с помощью псевдокодов (языков спецификаций), визуальных языков проектирования или языков программирования.

4.10. Методики функционального моделирования

Наиболее известная методика функционального моделирования сложных систем – методика SADT (Structured Analysisand Design Technique), предложенная в 1973 г. Россом и впоследствии ставшая основой стандарта IDEFO. Эта методика рекомендуется для начальных стадий проектирования сложных информационных систем управления, производства, бизнеса, включающих людей, оборудование, программное обеспечение.

Разработка SADT-модели начинается с формулировки вопросов, на которые модель должна давать ответы, т. е. формулируется цель моделирования. Далее выполняются этапы:

1) сбор информации. Источниками информации могут быть документы, наблюдение, анкетирование и т. п. Существуют специальные методики выбора экспертов и анкетирования;

2) создание модели. Используется нисходящий стиль: сначала разрабатываются верхние уровни, затем нижние;

3) рецензирование модели. Реализуется в итерационной процедуре рассылки модели на отзыв и ее доработки по замечаниям рецензентов

Поведенческое моделирование сложных систем используется для определения динамики функционирования сложных систем. В его основе лежат модели и методы имитационного моделирования систем массового обслуживания, сети Петри, возможно применение моделей конечных автоматов, описывающих поведение системы как последовательности смены состояний.

Поведенческие аспекты приложений отражает методика IDEF3. Если методика IDEFO связана с функциональными аспектами и позволяет отвечать на вопрос «что делает система», то в IDEF3 детализируются и конкретизируются IDEF0-функции, IDEF3-модель отвечает на вопрос: «Как система это делает?» В IDEF3 входят два типа описаний:

1) процессно-ориентированные в виде последовательности операций;

2) объектно-ориентированные, представляемые диаграммами перехода состояний, характерными для конечно-автоматных моделей, в этих диаграммах имеются средства для изображения состояний системы, активностей, переходов из состояния в состояние и условий перехода.

Системы информационного моделирования реализуют методики инфологического проектирования баз данных. Широко используются язык и методика IDEF1X создания информационных моделей приложений, развивающая более раннюю методику IDEF1. Кроме того, развитые коммерческие СУБД, как правило, имеют в своем составе совокупность CASE-средств проектирования приложений.

4.11. Этапы разработки информационной модели

В IDEF1X имеется ясный графический язык для описания объектов и отношений в приложениях. Этот язык есть язык диаграмм «сущность – связь» (ER). Разработка информационной модели по IDEF1X выполняется в несколько этапов.

Этап 1. Определение целей проекта. Планирование сбора информации. Обычно исходные положения для информационной модели вытекают из IDEF0-модели.

Этап 2. Выявление и определение сущностей. Это неформальная процедура.

Этап 3. Выявление и определение основных отношений. Результат представляется или графически, в виде ER-диаграмм, или в виде матрицы отношений, элемент которой $A_{ij} = 1$, если имеется связь между сущностями i и j, иначе $A_{ij} = 0$, транзитивные связи не указываются.

Этап 4. Детализация неспецифических отношений, определение ключевых атрибутов, установление внешних ключей. Детализация неспецифических отношений заключается в замене связей «многие ко многим» на связи «многие к одному» и «один ко многим» введением сущности-посредника.

Этап 5. Определение атрибутов и их принадлежности к сущностям.

Методика IDEF4 реализует объектно-ориентированное проектирование больших систем. Она предоставляет пользователю графический язык для изображения классов, диаграмм наследования, таксономии методов.

Методика IDEF5 направлена на представление онтологической информации приложения в удобном для пользователя виде. Для этого используются символические обозначения (дескрипторы) объектов, их ассоциаций, ситуаций и схемный язык описания отношений классификации, «часть – целое», перехода и т. п. В методике имеются правила связывания объектов (термов) в правильные предложения и аксиомы интерпретации термов.

Развитие BPR-методик продолжается по программе *IICE* (Information Integration for Concurrent Engineering). Разработаны методики:

• IDEF6, направленная на сохранение рационального опыта проектирования информационных систем, что способствует предотвращению повторных ошибок;

• IDEF8 для проектирования диалога человека с технической системой;

• IDEF9 для анализа имеющихся условий и ограничений (в том числе физических, юридических, политических) и их влияния на принимаемые решения в процессе реинжиниринга;

• IDEF14 для представления и анализа данных при проектировании вычислительных сетей на графическом языке с описанием конфигураций, очередей, сетевых компонентов, требований к надежности и т. п.

Основные положения стандартов IDEF0 и IDEF1X использованы также при создании комплекса стандартов ISO 10303, задающих технологию STEP для представления в компьютерных средах информации, относящейся к промышленному производству. В свою очередь, стандарты STEP, совокупность таких языков, как Express и SGML, а также стандарты P-LIB и MANDATE составляют основу технологии CALS информационного обеспечения всех этапов жизненного цикла промышленных изделий.

Технология CALS призвана разрешить проблему согласования содержания и формы представления данных о промышленной продукции в территориально распределенной сети проектных и производственных узлов

на основе совокупности международных стандартов и телекоммуникационных технологий. Только в этих условиях станет возможной оптимальная специализация предприятий, распределенное проектирование, минимизация затрат на освоение и эксплуатацию созданных систем.

4.12. Классическое проектирование информационных систем

Рассматриваемые методы в разной терминологии под различными названиями предусматривали последовательную организацию работ. За 20 лет и в разных «школах» проектирования разбиение работ на стадии и их названия менялись. Кроме того, наиболее разумно организованные методики и стандарты избегали жестко однозначного приписывания работ к конкретным стадиям. Вместе с тем при возможности неоднократного включения некоей работы в общую схему выделялись следующие проектные стадии (некоторые названия соответствующих этапов работ и (или) соответствующих документов в англоязычной литературе):

• запуск: организация основания для деятельности и запуск работ: приказ и (или) договор о разработке автоматизированной системы, задание на выполнение работ;

• обследование: предпроектное обследование, общий анализ ситуации на предприятии, разработка общего обоснования целесообразности создания ИС;

• концепция, ТЗ: исследования требований предприятия и пользователей, выработка рекомендаций по разработке ИС, разработка ТЗ на проектирование ИС в целом и частных ТЗ по подсистемам;

• эскизный проект: разработка архитектуры будущей ИС в рамках эскизного проекта;

• опытный вариант ИС: разработка упрощенного варианта, пилотного проекта будущей ИС;

• опытное использование пилот-проекта ИС: разработка исправлений и дополнений к ТЗ;

• технологическое проектирование: разработка технического проекта ИС;

• рабочее проектирование: разработка рабочей документации проекта (development, test, system implementation);

• ввод в действие: по-другому – «внедрение» ИС.

Одно из использовавшихся в западной литературе названий такой схемы организации работ – это «водопадная или каскадная модель». Схема обязана была включать итерационные процедуры уточнения требований к системе и рассмотрения вариантов проектных решений. Все же эти про-

цедуры и целые этапы работ носили в основном последовательный характер, а кроме того, предметом была проектируемая ИС целиком, в целостном ее представлении.

Положительные факторы применения данной схемы наблюдались в следующем:

• на каждой стадии формировался законченный, отвечающий критериям полноты и согласованности набор проектной, а затем и пользовательской документации, охватывающий все предусмотренные стандартами виды обеспечения ИС: организационное, методическое, информационное, программное, аппаратное и др.;

• выполняемые в логичной последовательности этапы работ достаточно очевидным образом позволяли планировать сроки завершения всех работ и соответствующие затраты. Структура ИС, как она формируется в ходе разработки, представлена в табл. 6.

Таблица 6

Структура формирования информационной системы

Стадии проекта	Виды обеспечения информационной системы				
	организационное	методическое	информационное	программное	аппаратное
Запуск	+ –				
Обследование	+ –	+ –	+ –		
Концепция ТЗ	+ –	+ –	+ –		
Эскизный проект	+ –	+ –	+ –	+ –	
ТП	+ –	+ +	+	+ –	+ –
РП	+ +	+ +	+ +	+ +	+
Ввод в действие	+ +	+ +	+ +	+ +	+ +
Символами «+», « + – » и « + +» показаны примерные оценки доли наличия каждого компонента на каждой стадии					

СПИСОК ЛИТЕРАТУРЫ

1. *Александров, Д. В*. Инструментальные средства информационного менеджмента. CASE-технологии и распределенные информационные системы : учеб. пособие / Д. В. Александров. – М. : Финансы и статистика, 2011. – 225 с.

2. *Белов, В. С*. Информационно-аналитические системы : основы проектирования и применения / В. С. Белов. – 2-е изд., перераб. и доп. – М. : Евразийский открытый институт, 2010. – 111 с.

3. *Баин, А. М*. Автоматизированные информационные системы электронных бизнес-отношений / А. М. Баин. – М. : Финансы и статистика, 2009. – 208 с.

4. *Вольфсон, М. Б*. Управление ИТ-сервисами и контентом [Электронный ресурс] : учеб. пособие : в 2 ч. / М. Б. Вольфсон, Ю. П. Левчук, Е. П. Охинченко ; рец.: А. А. Захаров, Н. Н. Беляева ; СПбГУТ. – СПб., 2014. – Ч. 2. – 64 с.

5. *Гагарина, Л. Г*. Разработка и эксплуатация автоматизированных информационных систем : учеб. пособие / Л. Г. Гагарина. – М. : ИД «Форум»: Инфра-М, 2013. – 384 с.

6. *Грекул, В. И*. Проектирование информационных систем : учеб. пособие / В. И. Грекул, Г. Н. Денищенко, Н. Л. Коровкина. – М. : Интернет-университет информационных технологий, 2008. – 486 с.

7. *Котлова, М. В*. Методы и средства проектирования информационных систем и технологий [Электронный ресурс] : учеб. пособие / М. В. Котлова, Е. В. Давыдова ; рец.: М. П. Белов, Т. В. Матюхина; СПбГУТ. – СПб., 2015. – 62 с.

8. *Маглинец, Ю. А*. Анализ требований к автоматизированным информационным системам [Электронный ресурс] : учеб. пособие / Ю. А. Маглинец. – М. : Интернет-университет информационных технологий (ИНТУИТ), БИНОМ. Лаборатория знаний, 2016. – 200 с.

9. *Советов, Б. Я*. Моделирование систем : учебник для вузов / Б. Я. Советов, С. А. Яковлев. – 4-е изд. стер. – М. : Высш. школа, 2005. – 343 с.

10. *Мишин, А. В*. Информационные технологии в профессиональной деятельности [Электронный ресурс] : учеб. пособие / А. В. Мишин, Л. Е. Мистров, Д. В. Картавцев. – М. : Российская академия правосудия, 2011. – 311 с.

11. *Пирогов, В. Ю*. Информационные системы и базы данных: организация и проектирование. – СПб. : БХВ – Петербург, 2010. – 528 с.

12. *Погонин, В. А.* Интегрированные системы проектирования и управления. Корпоративные информационные системы : учеб. пособие для студ. вузов, обуч. по направл. подгот. дипломир. специалистов «Автоматизированные технологии и пр-ва» / В. А. Погонин, А. Г. Схиртладзе ; Тамбовский гос. технический ун-т. – Тамбов : Изд-во ТГТУ, 2006. – 142 с.

13. *Советов, Б. Я*. Информационные технологии : учебник для вузов / Б. Я. Советов, В. В. Цехановский. – М. : Высш. школа, 2003. – 262 с.

14. *Советов, Б. Я*. Информационная технология / Б. Я. Советов. – М. : Высш. школа, 1994.

15. *Советов, Б. Я.*, Моделирование систем / Б. Я. Советов, С. А. Яковлев. – М. : Высш. школа, 1985.

16. *Степанова, Е. Е.* Информационное обеспечение управленческой деятельности : учеб. пособие для студ. образовательных учреждений сред. проф. образования / Е. Е. Степанова, Н. В. Хмелевская. – 2-е изд., испр. и доп. – М. : ФОРУМ, 2010. – 192 с.

17. *Татарникова, Т. М.* Управление данными : учеб. пособие / Т. М. Татарникова ; СПбГУТ. – СПб., 2006. – 84 с.

18. *Шелухин, О. И.* Моделирование информационных систем : учеб. пособие для вузов. – М. : Горячая линия – Телеком, 2012. – 516 с.

Давыдова Екатерина Викторовна
Котлова Мария Владимировна

ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА ИНФОРМАЦИОННЫХ СИСТЕМ

Учебное пособие

Редактор *И. И. Щенсняк*

План издания 2017 г., п. 85

Подписано к печати 28.02.2017
Объем 4,5 усл.-печ. л. Тираж 28 экз. Заказ 744

Редакционно-издательский отдел СПбГУТ
191186 СПб., наб. р. Мойки, 61

Отпечатано в СПбГУТ