Министерство образования и науки Российской Федерации

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

О. И. Жуковский

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И АНАЛИЗ ДАННЫХ

Учебное пособие

Томск
«Эль Контент»
2014

УДК 004.6(075.8)
ББК 32.973.233-018я73
Ж 864

Рецензенты:
Колупаева С. Н., докт. физ.-мат. наук, профессор, заведующая кафедрой прикладной математики Томского государственного архитектурно-строительного университета;
Рюмкин В. И., канд. физ.-мат. наук, доцент, доцент кафедры математических методов и информационных технологий в экономике экономического факультета Национального исследовательского Томского государственного университета.

Жуковский О. И.
Ж 864 Информационные технологии и анализ данных : учебное пособие / О. И. Жуковский. — Томск : Эль Контент, 2014. — 130 с.

ISBN 978-5-4332-0158-3

Учебное пособие представляет современные методы и средства, используемые в сфере информационных технологий. В пособии представлены основные понятия и концепции таких направлений информационных технологий, как структурная разметка документов, построение систем сбора данных и их оперативного анализа, CASE-технологии и геоинформационные системы.

Пособие подготовлено в соответствии с требованиями Государственного образовательного стандарта высшего профессионального образования.

Учебное пособие предназначено для студентов факультета дистанционного обучения ТУСУРа.

УДК 004.6(075.8)
ББК 32.973.233-018я73

ISBN 978-5-4332-0158-3

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Информатика как научное направление приобретает в подготовке специалистов разных уровней фундаментальный характер, являясь основой изучения ряда общепрофессиональных и специальных дисциплин. Информационные технологии являются составной частью научного направления «Информатика» и базируются на ее достижениях. Информатизация как процесс перехода к информационному обществу сопровождается возникновением новых и интенсивным развитием существующих информационных технологий.

Внедрение информационных технологий требует подготовки как пользователей, так и разработчиков, но для всех обучаемых необходимо [2]:

- знать базовые информационные процессы, структуру, модели, методы и средства базовых и прикладных информационных технологий, методику создания, проектирования и сопровождения систем на базе информационной технологии;
- уметь применять информационные технологии при решении функциональных задач в различных предметных областях, а также при разработке и проектировании информационных систем;
- иметь представление об областях применения информационных технологий и их перспективах в условиях перехода к информационному обществу.

В первой главе пособия даны определения основных понятий, используемых в сфере информационных технологий, рассмотрено понятие информации и особенности построения сообщений человеком. Во второй главе определены основные понятия и задачи информационной технологии. Раскрыты базовые информационные процессы, входящие в состав информационных технологий.

Непосредственно содержанию информационных технологий посвящены остальные главы пособия. В третьей главе рассматриваются современные технологии обработки текстовых документов, в четвертой главе затронуты вопросы построения систем обработки и анализа данных. В пятой главе рассмотрены основные понятия CASE технологий. Шестая глава знакомит с основными понятиями и идеями, используемыми геоинформационными технологиями, дает представление о присущих им инструментах анализа пространственных данных.

Соглашения, принятые в книге

Для улучшения восприятия материала в данной книге используются пиктограммы и специальное выделение важной информации.

Этот блок означает определение или новое понятие.

Этот блок означает внимание. Здесь выделена важная информация, требующая акцента на ней. Автор здесь может поделиться с читателем опытом, чтобы помочь избежать некоторых ошибок.

Пример

Эта пиктограмма означает пример. В данном блоке автор может привести практический пример для пояснения и разбора основных моментов, отраженных в теоретическом материале.

Контрольные вопросы по главе

Глава 1

СООБЩЕНИЕ, ИНФОРМАЦИЯ, ДАННЫЕ

1.1 Основные понятия

«Сообщение» и «информация» — это основные понятия информатики, техническое значение которых не вполне соответствует употреблению этих двух слов в обиходной речи. Необходимое в связи с этим уточнение содержания указанных понятий не может быть достигнуто с помощью определения, так как последнее лишь сводило бы эти понятия к другим не определённым основным понятиям. Поэтому целесообразно ввести ***сообщение*** и ***информацию*** как неопределяемые основные понятия и рассмотреть их использование на ряде примеров [1].

При разграничении понятий сообщения и информации будем исходить из распространённых оборотов речи типа «это сообщение не даёт мне никакой информации», что приводит к следующему отношению между этими понятиями: *(абстрактная) информация передаётся посредством (конкретного) сообщения*.

Соответствие между сообщением и информацией не является взаимно-однозначным. Для одной и той же информации могут существовать различные передающие её сообщения, например сообщения на разных языках или сообщения, которые получаются добавлением *неважного* сообщения, не несущего никакой дополнительной информации. Сообщения, передающие одну и ту же информацию, образуют класс эквивалентных сообщений. С другой стороны, одно и то же сообщение может передавать совершенно различную информацию: сообщение о падении самолета для близких родственников погибшего имеет совсем иной смысл, нежели для авиакомпании; разные читатели из одной и той же газетной статьи черпают совершенно различную информацию, соответствующую кругу их интересов.

Таким образом, одно и то же сообщение, по-разному ***интерпретированное***, может передавать разную информацию. Обобщая, можно сказать, что решающим

для связи между сообщением N и информацией I является некое отображение α, представляющее собой результат договорённости между отправителем и получателем сообщения или предписанное им обоим и называемое ***правилом интерпретации***. Символически мы будем записывать правило интерпретации в следующей форме: $N \xrightarrow{\alpha} I$.

Правило интерпретации α для данного сообщения часто получается как частный случай некоторого общего правила, применимого к целому множеству $\mathfrak{R}$ сообщений, которые построены по одинаковым законам. Если мы формулируем сообщения на некотором ***языке***, то высказывание «X понимает язык $\mathfrak{R}$» выражает тот факт, что лицо X знает правило интерпретации α для всех (или по крайней мере для большинства) сообщений, формулируемых на данном языке.

Иногда правило интерпретации известно лишь ограниченному кругу лиц; сюда относятся правила интерпретации для специальных языков, в частности для различных профессиональных и научных языков (жаргонов).

Связь между сообщением и информацией особенно отчётливо видна в криптографии: здесь никто посторонний не должен суметь извлечь информацию из передаваемого сообщения, иначе это означало бы, что он располагает «ключом».

Часто встречаются и такие сообщения, которые могут интерпретироваться поразному, причём различные интерпретации основываются одна на другой. Так, сообщение «идёт дождь» может нести дополнительную информацию «нужно взять с собой зонтик». В этом случае говорят об информациях различной ***степени отвлечённости***.

Для сообщений, которыми обмениваются люди, в большинстве случаев имеются соглашения относительно их формы. О таких сообщениях будем говорить, что они передаются в ***языковой форме***, что они составлены на некотором ***языке***. При этом слово «язык» используется в существенно более широком смысле, чем в случае связанного с ним понятия «говорить». Мы знаем разговорный и письменный языки, язык глухонемых, построенный на жестах и мимике, печать для слепых, воспринимаемую осязанием. Два последних примера показывают, что высокоразвитое языковое общение не ограничивается устной и письменной речью.

Хотя многое говорит в пользу того, что именно разговорный язык знаменует начало истории человека, всё же и в современном обществе остаётся язык жестов, дополненный специфическими звуками, такими как шипение, мычание, свист и щелчки, — хоть и примитивное, но иногда решающее вспомогательное средство, обеспечивающее взаимопонимание.

Когда мы говорим о ***языковых сообщениях***, мы имеем в виду то общее, что присуще каждому из этих случаев; способ передачи — письменно, устно, посредством осязания или ещё как-то — не имеет здесь никакого значения. При этом, однако, следует иметь в виду, что, например, информация, содержащаяся в устном сообщении, не всегда полностью воспроизводится соответствующим письменным сообщением. Такие настроения, как гнев, радость, горечь, искренность, находят своё полное выражение только в устной речи. Ударения и паузы также несут информацию.

В случае, когда мы говорим о немецком языке, английском языке и т. д., лучше использовать термин «***язык-речь***». Для различия между языком и языком-речью характерно, что внутри данного языка-речи можно говорить на высоком языке, на

разговорном языке, на блатном языке (сленге) или на профессиональном языке (жаргоне). Существуют также языки, которые не принадлежат и вообще не могут быть причислены ни к какому языку-речи, например искусственные языки вроде эсперанто или некоторые специальные языки, в том числе язык формул математики и языки программирования.

Наконец, понятие языка не ограничивается случаем общения между людьми, оно используется и в случае сравнительно высокоразвитых форм общения между другими живыми существами. Примером может служить открытый К. Фришем язык ориентации пчел.

Для нашего предмета особенно важны языки, в которых для передачи сообщений используются ***долговременные носители информации***.

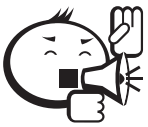
При этом передача освобождается от гнёта реального времени, и становятся даже возможными сообщения человека самому себе, ***заметки*** на память, чем уменьшается нагрузка на человеческую память за счёт использования «инструмента».

Представление сообщений на долговременных носителях будем называть ***письмом****, а сам долговременный носитель —* ***носителем письма***.

Прежде всего следует упомянуть зрительно воспринимаемое письмо, которое создается вручную (рукопись) или механически (машинопись, типографская печать). Письмо, воспринимаемое на слух, стало реальностью после того, как Эдисон изобрел фонограф. Письмом, воспринимаемым осязанием, является письмо слепых, которое пишется вручную (посредством наколов иголкой), а также механически. Фиксация изображений (например, в кино) также представляет собой письмо, как и географическая карта.

1.2 Роль органов чувств в восприятии сообщений

Выше уже были упомянуты органы чувств, которые могут служить для передачи языковых сообщений. Некоторые из воспринимающих органов чувств служат также и для односторонней неязыковой связи с окружающим миром. У высших живых существ только слуховые, зрительные и тактильные восприятия достаточно дифференцированы, чтобы естественным образом служить для передачи языковых сообщений. Наряду с языками непосредственного общения: разговорной речью, призывающими и предостерегающими звуками (слуховое восприятие), языком глухонемых (зрительное восприятие) и воспринимаемым рукой языком слепых (тактильное восприятие) выступают языки, в которых используются инструменты: барабанный бой, стук, свистки, звуки рожка, трубы, сирена — сигнал тревоги (слуховое восприятие), световые сигналы и сигналы флажками (зрительное восприятие).



Определенные знания о свойствах и работе органов чувств человека оказываются существенными, когда мы хотим рационально включить человека в цепочку обработки и передачи информации (в её начало или конец).

Функциональная способность органов чувств лежит в определенных пределах. Здесь прежде всего следует назвать ***время реакции*** (латентное, или скрытое, время). Для акустических (звуковой импульс) и оптических (загорание лампочки) сигналов оно составляет для человека 140–250 мс до ответа, состоящего в том, что испытуемый нажимает кнопку. Для более сложных заданий время реакции заметно увеличивается (прочитать указанное слово: 350–550 мс, назвать указанный предмет домашнего обихода: 600–800 мс). Это уже говорит о том, что процесс восприятия — не только функция рецепторов. Сюда примыкают проведение раздражения по нервным путям, переработка его в мозге, а также проведение ответа к эффектору. При этом на глаз как на воспринимающий орган приходится около 40 мс, а на срабатывание мышц руки как передающего органа — около 50 мс.

Для того чтобы орган вообще смог что-либо воспринять, интенсивность раздражения должна превосходить определённое ***пороговое значение***. Для слуха пороговое значение составляет около $2 \cdot 10^{-7}$ мбар.

Знакомство с физиологией и психологией чувств учит нас многому, прежде всего тому, что обработка информации (а не только её передача) есть обязательная составная часть нашего чувственного восприятия.

Когда мы ниже говорим об искусственной, т. е. изобретённой людьми и выполняемой машинами обработке информации, это должно напоминать нам, что обработка информации не является чем-то принципиально новым. Следует отклонить, как чересчур поспешные, выводы, которые наивно делаются посредством (недопустимой) редукции («человека можно объяснить с помощью законов физики», «умственная деятельность — не что иное, как приём, обработка, накопление и выдача информации»): уже сфера сенсорного восприятия говорит о столь сложных процессах в мозге, что такие утверждения не могут служить даже рабочей гипотезой [1].

1.3 Устройства связи и передача сообщений

Письмо и газета относятся к самым старым и до сих пор не устаревшим примерам (случайной и регулярной) передачи сообщений посредством записи на ***долговременном*** носителе сообщений. В случае передачи информации с помощью ***недолговременного*** носителя сообщений человек также использует различные физические устройства в соответствии с уровнем развития техники на данный мо-

мент. Примерами таких устройств связи служат телефон, радио и телевидение, предназначенные как для случайной, так и для регулярной передачи сообщений.

Внешне ***устройство связи***, или, точнее, «приёмо-передающее устройство», состоит из ***приёмника*** (получателя) и ***передатчика*** (отправителя). О внутреннем строении устройств связи никаких общих утверждений сделать нельзя, разве что при более внимательном рассмотрении многие из них оказываются составленными из нескольких более мелких устройств связи.

Может случиться, что для сообщений на входе и на выходе используется один и тот же носитель. Такие устройства служат лишь для возможного ***усиления*** или ***регенерации*** сообщения, связанных с устранением помех, и называются ***релейными линиями***. Примерами таких устройств являются рупор, слуховая трубка, а также их современные электронные варианты — мегафон и слуховой аппарат. Если для сообщений на входе и выходе устройства используются различные физические носители, то устройство связи называют ***преобразователем***.

Если устройство предназначено для связи между людьми, то сообщения на входе или выходе должны быть производимы или соответственно воспринимаемы людьми, т. е. носители должны соответствовать человеческим эффекторам и рецепторам. В качестве примеров назовём музыкальный инструмент, на котором играют, нажимая на клавиши (физический носитель на входе — давление, на выходе — звуковые волны), и осциллограф, управляемый через микрофон (на входе — звуковые волны, на выходе — световые волны).

Как уже упоминалось, два или более устройств связи можно соединить друг с другом таким образом, что результирующее устройство снова будет устройством связи. Приёмником составного устройства является приёмник первого устройства, участвующего в соединении, а передатчиком — передатчик последнего устройства. В этом случае между передатчиком одного устройства связи и приёмником другого могут использоваться и такие носители, которые недоступны человеческим эффекторам и рецепторам. Примером могут служить телефонная связь по проводам или радио.

Исходя из подобных примеров протяжённую в пространстве среду, «через» которую носитель сообщения передаётся от передатчика к приёмнику, называют ***каналом связи***.

Помимо бумаги, используемой в письме и чтении человеком, в качестве долговременных носителей текстовых сообщений в современной технике чаще всего используются намагничиваемые и светочувствительные плёнки, а также перфорируемая бумага (перфокарты, перфоленты).

Аналогии между рецепторами и эффекторами живых существ и техническими приёмо-передающими устройствами служат предметом исследований в ***кибернетике***. Кибернетика занимается главным образом аспектами, общими для человека и технических устройств с точки зрения передачи и переработки сообщений.

Следует особо отметить то, что передача сообщений происходит во времени. Поэтому в качестве носителей заслуживают внимания только такие физические величины, которые могут изменяться во времени.

Изменение некоторой физической величины во времени, обеспечивающее передачу сообщения (а тем самым и информации), называется ***сигналом****.*

При этом для воспроизведения сообщения могут использоваться различные свойства сигнала. Та характеристика сигнала, которая служит для представления сообщения, называется ***параметром сигнала***.

В качестве примера рассмотрим радио. Сигналами здесь являются электромагнитные колебания. В диапазоне средних волн сообщение воспроизводится амплитудой колебаний, а в диапазоне ультракоротких волн — частотой колебаний (***амплитудная*** и ***частотная модуляции*** соответственно). Таким образом, в первом случае параметром сигнала является амплитуда, а во втором — частота колебаний.

Если для передачи сообщений используются импульсы, то параметром сигнала может быть либо амплитуда импульса, либо интервал между импульсами (***амплитудно-импульсная*** и ***частотно-импульсная модуляции*** соответственно).

1.4 Дискретные сообщения, знаки и кодирование

Сигнал называется ***дискретным***, если параметр сигнала может принимать лишь конечное число значений и существен лишь в конечном числе моментов времени (возможно, периодически повторяющихся).

Дискретными сообщениями *называются такие сообщения, которые могут быть переданы с помощью дискретных сигналов.*

Языковые сообщения в письменной форме строят обычно, записывая знаки письма (***графемы***) друг за другом. Хотя длинные сообщения могут размещаться на многих строчках и страницах, это разбиение не имеет, вообще говоря, никакого значения; оно не несёт важной информации. По существу такие сообщения являются последовательностями знаков. Это оказывается справедливым и для устных языковых сообщений, если разложить устный текст на элементарные составные части, так называемые ***фонемы***, и под знаками понимать фонемы. Чтобы можно было воспроизводить фонемы и письменно, принято соглашение о международных письменных знаках для отдельных фонем. Точка зрения, что сообщение есть последовательность знаков, не ограничивается тем случаем, когда знаки — это фонемы или графемы (например, знаки букв и цифр, знаки препинания). Знаки планет или знаки зодиака и даже кивок и покачивание головой также могут пониматься как знаки. В силу этого определим понятие знака следующим образом.

Знак *— это элемент некоторого конечного множества отличимых друг от друга «вещей»,* ***набора знаков****. Набор знаков, в котором определён (линейный) порядок знаков, называется* ***алфавитом****.*

Пример 1.1

Вот некоторые примеры алфавитов (порядок в них — это порядок перечисления):

1) алфавит десятичных цифр
   {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};
2) алфавит заглавных кириллических букв
   {А, Б, В, Г, Д, Е, Ж, З, И, Й, К, Л, М, Н, О, П, Р, С, Т, У, Ф, Х, Ц, Ч, Ш, Щ, Ъ, Ы, Ь, Э, Ю, Я}.

Вот некоторые наборы знаков, для которых нет какого-либо общепринятого порядка знаков:

1) набор знаков клавиатуры пишущей машинки;
2) набор знаков планет.

Особенно важное значение имеют наборы, состоящие всего из двух знаков. Такие наборы называют ***двоичными наборами знаков***, а сами знаки — ***двоичными знаками***. Вместо термина «двоичный знак» часто употребляют сокращение ***бит*** (от английского *binary digit*).

Если N — предложение некоторого естественного языка, то N можно рассматривать как последовательность знаков по крайней мере тремя разными способами.

Прежде всего N представляет собой последовательность букв, цифр, знаков препинания и т. д.; далее, N — это последовательность слов, которые в другом контексте могут сами рассматриваться как знаки; наконец, и всё предложение целиком можно рассматривать как один знак.

Первое понимание используется, например, когда имеется правило для нанесения сообщения N на перфокарты; второе понимание лежит в основе стенографических сокращений; крайнее третье понимание бывает уместным при переводе на другой естественный язык, когда пословица одного языка переводится соответствующей по смыслу пословицей другого языка.

Дискретные сообщения представляют собой (конечные или бесконечные) *последовательности знаков*. При этом исходя из соображений, связанных с физиологией органов чувств, или из чисто технических соображений, их обычно разбивают на конечные последовательности знаков, называемые ***словами***. На более высоком уровне каждое слово можно снова рассматривать как знак, при этом соответствующий набор знаков будет, вообще говоря, шире первоначального. Обратно, данный набор знаков можно получить с помощью составления слов исходя из некоторого набора с меньшим числом знаков, в частности из двоичного набора знаков.

Слова над двоичным набором знаков называются ***двоичными словами***. Они не обязаны иметь постоянную длину (см. азбуку Морзе), если это всё же так, то говорят об n-разрядных двоичных знаках и n-разрядных двоичных кодах (например, 2-й международный телеграфный код (5-разрядные двоичные знаки)).

Будем пользоваться следующим определением:

***Кодом** называется правило, описывающее отображение одного набора знаков в другой набор знаков (или слов); кодом также называют и множество образов при этом отображении.*

Помимо основного значения слова «code» — «кодекс», «свод законов» — начиная с середины XIX-го в. оно означало книгу, в которой словам естественного языка сопоставлены группы цифр или букв. Употребление таких кодов приобрело значение скорее в связи со стремлением сэкономить на стоимости телеграмм, чем в связи с соображениями конспиративности.

Если каждый образ при кодировании является отдельным знаком, то такое отображение мы называем ***шифровкой***, а образы — ***шифрами*** (англ, *cipher*). Поскольку здесь имеется криптографический аспект, обращение этого отображения — когда оно однозначно — называется ***декодированием или дешифровкой***.

В коммерческих и криптографических кодах слова, фразы и понятия естественных языков кодируются в большинстве случаев словами над некоторым буквенным или цифровым алфавитом, обычно пятерками. В технических кодах буквы, цифры и другие знаки почти всегда кодируются двоичными словами. У большинства используемых в технике кодов все слова имеют одинаковую длину. Коды со словами разной длины встречаются в технике довольно редко. Исключением является код Морзе. Грубо говоря, это двоичный код с набором знаков {«точка», «тире»} и словами длины не более 5 для кодирования букв и цифр. Более точно, следует ещё добавить в качестве третьего знака знак «пропуск», который помечает стыки между кодовыми словами (слова нельзя отделить друг от друга по их длине).

В двоичных кодах с постоянной длиной кодовых слов слова могут следовать друг за другом непосредственно (***последовательная передача***), так что получается единая последовательность двоичных знаков. Расположение стыков и тем самым исходная группировка кодовых слов устанавливаются с помощью отсчёта, и, таким образом, сообщения, составленные из кодовых слов, однозначно декодируемы. Правда, при отсчёте кодовой длины нельзя просчитаться, нельзя «сбиться с ритма», а это ведёт к усложнению с технической точки зрения (параллельность, синхронизация).

Напротив, для кодов с переменной длиной кодовых слов расположение стыков, вообще говоря, восстановить нельзя. При определённых условиях сообщение, состоящее из нескольких кодовых слов, либо вовсе не декодируется, либо декодируется неоднозначно. Однако декодируемость будет обеспечена, если соблюдается следующее *Условие Фано*. Никакое кодовое слово не является началом другого кодового слова («свойство префиксности»). Тогда, очевидно, стык между кодовыми словами определяется тем моментом, когда «дальше не читается». Очевидно также, что код удовлетворяет условию Фано тогда и только тогда, когда кодовое дерево не содержит ни одного языка во «внутренних» вершинах (***дерево с размеченными листьями***).

Условие Фано является достаточным, но не необходимым условием однозначной декодируемости. Тривиальная возможность обеспечить выполнение условия Фано состоит в том, чтобы каждое кодовое слово начинать специальным знаком (или группой знаков), называемым ***разделителем***. Это, очевидно, имеет место

в случае кода Морзе, а именно пропуск является разделителем для последовательности точек и тире, а группа знаков ООО — разделителем при двойном кодировании кода Морзе. С технической точки зрения при передаче по телеграфу также передается разделитель (синхронизирующий «такт разбивки»).

При ***параллельной передаче*** мы, в отличие от последовательной, ограничены кодами со словами постоянной длины: для n-разрядного двоичного кода используется n параллельных двоичных каналов передачи. В случае оптического, электростатического, электролитического и электромагнитного телеграфа путь технического прогресса шёл прежде всего от параллельной к последовательной передаче.

Вопрос о том, какие коды являются оптимальными с точки зрения передачи, изучается в теории информации.

Следует различать собственно знак и его смысл. Знак вместе с его смыслом называется ***символом***.

В соответствии с целью употребления один и тот же знак часто имеет разный смысл. Знак ♀ применяется в астрономии как символ планеты Венера, а в биологии — как символ женской особи. К несчастью, часто бывает также, что разные знаки имеют одинаковый смысл; например, знаки $\cdot$ и $\times$, а в последнее время и $*$ — все понимаются как символы умножения.

Обычно всякое сообщение имеет смысл, т. е. уже является символом. Очевидно, что этот символ получается в результате присоединения к сообщению той информации, которая им передается.

1.5 Обработка сообщений и обработка информации. Данные

Всякое правило обработки сообщений можно понимать как отображение (функцию) ν

$$\mathfrak{N} \xrightarrow{\nu} \mathfrak{N}',$$

которое сообщениям N из некоторого множества сообщений $\mathfrak{N}$ ставит в соответствие новые сообщения N' из множества сообщений $\mathfrak{N}'$. Каждое из сообщений N и N' — это последовательность знаков.

Большая свобода в понимании сообщения как последовательности знаков, просматриваемая в обсуждавшихся выше примерах, позволяет констатировать: ***всякую обработку сообщений можно рассматривать как кодирование***.

Конечно, это соображение является важным и для изучения процессов обработки сообщений у живых существ, но прежде всего оно лежит в основе всякой машинной обработки дискретных сообщений.

Сообщения N вместе с сопоставленной им информацией J будем называть ***данными****.*

Примером могут служить сообщения, записываемые арабскими цифрами в позиционной системе счисления и связанная с ними информация, которую называют «натуральными числами», а также символы.

Итак, данные есть пары типа (N,J) с $N \xrightarrow{\alpha} J$, при этом информацию J называют ***значением*** данных, а сообщение N — ***обозначением*** данных. Говорят, что обозначение N обладает значением J при интерпретации α.

Например, обозначение 4 обладает значением «четыре», обозначение 004 — значением «четыре», обозначение 3.14 — значением «три целых и четырнадцать сотых». При этом обозначение определяет значение, которым оно обладает, однозначно. Поэтому для краткости говорят просто «данные x» вместо «данные с обозначением x».

Различные обозначения могут обладать одним и тем же значением — отображение α обычно не является обратимым [1].

Контрольные вопросы по главе 1

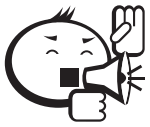
1. Каким образом связаны понятия «сообщение» и «информация»?
2. Что определяет информацию, которая предается конкретным сообщением?
3. Дайте характеристику роли органов чувств в восприятии сообщений человеком.
4. Чем отличается знак от символа?
5. Что называется кодом?
6. Приведите примеры наборов знаков, которые не являются алфавитом.
7. Как мы будем понимать обработку сообщений?

Глава 2

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

2.1 Определение и задачи информационной технологии

Термин «**технология**» имеет множество толкований. В широком смысле под технологией понимают науку о законах производства материальных благ, вкладывая в нее три основные части: идеологию, т. е. принципы производства; орудия труда, т. е. станки, машины, агрегаты; кадры, владеющие профессиональными навыками. Эти составляющие называют соответственно информационной, инструментальной и социальной. Для конкретного производства технологию понимают в узком смысле как совокупность приемов и методов, определяющих последовательность действий для реализации производственного процесса. Уровень технологий связан с научно-техническим прогрессом общества и влияет на его социальную структуру, культуру и идеологию [2].



Для любой технологии могут быть выделены цель, предмет и средства. Целью технологии в промышленном производстве является повышение качества продукции, сокращение сроков ее изготовления и снижение себестоимости.

Методология любой технологии включает в себя: декомпозицию производственного процесса на отдельные взаимосвязанные и подчиненные составляющие (стадии, этапы, фазы, операции); реализацию определенной последовательности выполнения операций, фаз, этапов и стадий производственного процесса в соответствии с целью технологии; технологическую документацию, формализующую выполнение всех составляющих.

Производство информации направлено на целесообразное использование информационных ресурсов и снабжение ими всех элементов организационной струк-

туры и реализуется путем создания информационной системы. Информационные ресурсы являются исходным «сырьем» для системы управления любой организационной структурой, конечным продуктом является принятое решение.

Принятие решения в большинстве случаев осуществляется в условиях недостатка информации, поэтому степень использования информационных ресурсов во многом определяет эффективность работы организации.

В своем становлении любая отрасль, в том числе и информационная, проходила стадии от кустарного ремесленного производства к производству, основанному на высоких технологиях.

Информационные технологии обеспечивают переход к промышленным методам и средствам работы с информацией в различных сферах человеческой деятельности, обеспечивая ее рациональное и эффективное использование.

В развитии технологии выделяют два принципиально разных этапа: один характеризуется непрерывным совершенствованием установившейся базисной технологии и достижением верхнего предельного уровня, когда дальнейшее улучшение является неоправданным из-за больших экономических вложений; другой отличается отказом от существующей технологии и переходом к принципиально иной, развивающейся по законам первого этапа.

Информационная технология *— совокупность методов и способов получения, обработки, представления информации, направленных на изменение ее состояния, свойств, формы, содержания и осуществляемых в интересах пользователей.*

Выделяют три уровня рассмотрения информационных технологий:

- *Первый уровень* — теоретический. Основная задача — создание комплекса взаимосвязанных моделей информационных процессов, совместимых параметрически и критериально.
- *Второй уровень* — исследовательский. Основная задача — разработка методов, позволяющих автоматизированно конструировать оптимальные конкретные информационные технологии.
- *Третий уровень* — прикладной, который целесообразно разделить на две страты: инструментальную и предметную.

Инструментальная страта (аналог — оборудование, станки, инструмент) определяет пути и средства реализации информационных технологий, которые можно разделить на методические, информационные, математические, алгоритмические, технические и программные.

Предметная страта связана со спецификой конкретной предметной области и находит отражение в ***специализированных информационных технологиях***, например, организационное управление, управление технологическими процессами, автоматизированное проектирование, обучение и другие.

Успешное внедрение информационных технологий связано с возможностью их типизации. ***Конкретная информационная технология*** обладает комплексным составом компонентов, поэтому целесообразно определить ее структуру и состав.

Конкретная информационная технология определяется в результате компиляции и синтеза базовых технологических операций, специализированных технологий и средств реализации.

Технологический процесс — часть информационного процесса, содержащая действия (физические, механические и др.) по изменению состояния информации.

Информационная технология базируется на реализации информационных процессов, разнообразие которых требует выделения базовых, характерных для любой информационной технологии.

Базовый технологический процесс основан на использовании стандартных моделей и инструментальных средств и может быть использован в качестве составной части информационной технологии.

К их числу можно отнести: операции ***извлечения, транспортировки, хранения, обработки и представления информации***.

Среди базовых технологических процессов выделим:

- извлечение информации;
- транспортирование информации;
- обработку информации;
- хранение информации;
- представление и использование информации.

Процесс *извлечения* информации связан с переходом от реального представления предметной области к его описанию в формальном виде и в виде данных, которые отражают это представление.

В процессе транспортирования осуществляют передачу информации на расстояние для ускоренного обмена и организации преобразования.

Процесс обработки информации состоит в получении одних «информационных объектов» из других «информационных объектов» путем выполнения некоторых алгоритмов; он является одной из основных операций, выполняемых над информацией и главным средством увеличения ее объема и разнообразия.

Процесс хранения связан с необходимостью накопления и долговременного хранения данных, обеспечением их актуальности, целостности, безопасности, доступности.

Процесс представления и использования информации направлен на решение задачи доступа к информации в удобной для пользователя форме.

Базовые информационные технологии строятся на основе базовых технологических операций, но кроме этого включают ряд специфических моделей и инструментальных средств. Этот вид технологий ориентирован на решение определенного класса задач и используется в конкретных технологиях в виде отдельной компоненты.

Среди них можно выделить:

- мультимедиа-технологии;
- геоинформационные технологии;

- технологии обработки тестовых сообщений;
- технологии защиты информации;
- CASE-технологии;
- телекоммуникационные технологии;
- технологии искусственного интеллекта.

Специфика конкретной предметной области находит отражение в ***специализированных информационных технологиях***, например организационное управление, управление технологическими процессами, автоматизированное проектирование, обучение и др. Среди них наиболее продвинутыми являются следующие информационные технологии:

- организационного управления (корпоративные информационные технологии);
- в промышленности и экономике;
- в образовании;
- автоматизированного проектирования.

Аналогом инструментальной базы (оборудование, станки, инструмент) являются средства реализации информационных технологий, которые можно разделить на *методические, информационные, математические, алгоритмические, технические и программные*.

Методические средства определяют требования при разработке, внедрении и эксплуатации информационных технологий, обеспечивая информационную, программную и техническую совместимость. Наиболее важными из них являются требования по стандартизации.

Информационные средства обеспечивают эффективное представление предметной области — к их числу относятся информационные модели, системы классификации и кодирования информации (общероссийские, отраслевые) и др.

Математические средства включают в себя модели решения функциональных задач и модели организации информационных процессов, обеспечивающие эффективное принятие решения. Математические средства автоматически переходят в алгоритмические, которые обеспечивают их реализацию.

Технические и программные средства задают уровень реализации информационных технологий как при их создании, так и при их реализации.

Таким образом, ***конкретная информационная технология*** определяется в результате компиляции и синтеза базовых технологических операций, «отраслевых технологий» и средств реализации.

Эволюция информационных технологий наиболее ярко прослеживается на процессах хранения, транспортирования и обработки информации. В процессе управления данными, объединяющем задачи их получения, хранения, обработки, анализа и визуализации, выделяют шесть временных фаз (поколений).

Вначале данные обрабатывали вручную. На следующем шаге использовали оборудование с перфокартами и электромеханические машины для сортировки и табулирования миллионов записей. В третьей фазе данные хранились на магнитных лентах, и сохраняемые программы выполняли пакетную обработку последовательных файлов. Четвертая фаза связана с введением понятия схемы базы данных и оперативного навигационного доступа к ним. В пятой фазе был обеспечен авто-

матический доступ к реляционным базам данных и была внедрена распределенная и клиент-серверная обработка.

Теперь мы находимся в сфере действия шестого поколения систем, которые хранят более разнообразные типы данных (документы, графические, звуковые и видеообразы). Эти системы шестого поколения представляют собой базовые средства хранения для приложений Интернета и Интранета.

2.2 Базовые информационные процессы, их характеристика и модели

Информационные технологии основаны на реализации информационных процессов, разнообразие которых требует выделения ***базовых***. К ним можно отнести ***извлечение***, ***транспортирование***, ***обработку***, ***хранение***, ***представление и использование информации***. На логическом уровне строятся математические модели, обеспечивающие параметрическую и критериальную совместимость информационных процессов в системе информационных технологий.

В процессе извлечения информации основной акцент сделан на формы и методы исследования данных, позволяющих формализовать и абстрагировано описать предметную область. Процесс транспортирования информации рассматривается в рамках эталонной семиуровневой модели, известной как модель OSI. Большое внимание уделено протоколам различных уровней, обеспечивающих необходимый уровень стандартизации. Процессы обработки информации излагаются в аспекте поддержки принятия решений с выделением типовых компонентов. Хранение информации представляется, с одной стороны, как совокупность моделей концептуального, логического и физического уровней, с другой — как набор методов и способов практической реализации. Большое внимание уделено эргономическим и психологическим факторам при распределении функции между человеком и техническими устройствами в процессе представления и использования информации.

2.2.1 Извлечение информации

Источниками данных в любой предметной области являются объекты и их свойства, процессы и функции, выполняемые этими объектами или для них. Любая предметная область рассматривается в виде трех представлений:

- реальное представление предметной области;
- формальное представление предметной области;
- информационное представление предметной области.

По аналогии с добычей полезных ископаемых процесс извлечения информации направлен на получение ее наибольшей концентрации. В связи с этим процесс извлечения можно представить, как прохождение информации через многослойный фильтр, в котором осуществляется оценка синтаксической ценности (правильность представления), семантической (смысловой) ценности, прагматической (потребительской) ценности.

При извлечении информации важное место занимают различные формы и методы исследования данных:

- поиск ассоциаций, связанных с привязкой к какому-либо событию;
- обнаружение последовательностей событий во времени;
- выявление скрытых закономерностей по наборам данных, путем определения причинно-следственных связей между значениями определенных косвенных параметров исследуемого объекта (ситуации, процесса);
- оценка важности (влияния) параметров на развитие ситуации;
- классификация, осуществляемая путем поиска критериев, по которым можно было бы относить объект (события, ситуации, процессы) к той или иной категории;
- кластеризация, основанная на группировании объектов по каким-либо признакам;
- прогнозирование событий и ситуаций.

Обратим внимание на неоднородность (разнородность) информационных ресурсов, характерную для многих предметных областей. Одним из путей решения данной проблемы является объектно-ориентированный подход, наиболее распространенный в настоящее время. Кратко рассмотрим его основные положения.

Декомпозиция на основе объектно-ориентированного подхода основана на выделении следующих основных понятий: объект, класс, экземпляр.

Объект — это абстракция множества предметов реального мира, обладающих одинаковыми характеристиками и законами поведения. Объект характеризует собой типичный неопределенный элемент такого множества. Основной характеристикой объекта является состав его атрибутов (свойств).

Атрибуты — это специальные объекты, посредством которых можно задать правила описания свойств других объектов.

Экземпляр объекта — это конкретный элемент множества. Например, объектом может являться государственный номер автомобиля, а экземпляром этого объекта — конкретный номер В 010 УХ.

Класс — это множество предметов реального мира, связанных общностью структуры и поведением. **Элемент класса** — это конкретный элемент данного множества. Например, класс регистрационных номеров автомобиля.

Обобщая эти определения, можно сказать, что объект — это типичный представитель класса, а термины «экземпляр объекта» и «элемент класса» равнозначны.

Важная особенность объектно-ориентированного подхода связана с понятием инкапсуляции, обозначающим сокрытие данных и методов (действий с объектом) в качестве собственных ресурсов объекта.

Понятия полиморфизма и наследования определяют эволюцию объектно-ориентированной системы, что подразумевает определение новых классов объектов на основе базовых.

Полиморфизм интерпретируется как способность объекта принадлежать более чем одному типу.

Наследование выражает возможность определения новых классов на основе существующих с возможностью добавления или переопределения данных и методов.

Для уменьшения избыточности используется процесс обогащения информации, например при хранении в компьютере списка сотрудников организации иногда достаточно использовать первые 3–4 буквы их фамилий.

Среди методов обогащения информации различают структурное, статистическое, семантическое и прагматическое обогащения.

Структурное обогащение предполагает изменение параметров сообщения, отображающего информацию в зависимости от частотного спектра исследуемого процесса, скорости обслуживания источников информации и требуемой точности.

При статистическом обогащении осуществляют накопление статистических данных и обработку выборок из генеральных совокупностей накопленных данных.

Семантическое обогащение означает минимизацию логической формы, исчислений и высказываний, выделение и классификацию понятий, содержания информации, переход от частных понятий к более общим. В итоге семантического обогащения удается обобщенно представить обрабатываемую либо передаваемую информацию и устранить логическую противоречивость в ней.

Прагматическое обогащение является важной ступенью при использовании информации для принятия решения, при котором из полученной информации отбирается наиболее ценная, отвечающая целям и задачам пользователя [2].

2.2.2 Транспортирование информации

Основным физическим способом реализации операции транспортировки в настоящее время является использование локальных сетей и сетей передачи данных. При разработке и использовании сетей для обеспечения совместимости используется ряд стандартов, объединенных в семиуровневую модель открытых систем, принятую во всем мире и определяющую правила взаимодействия компонентов сети на данном уровне (протокол уровня) и правила взаимодействия компонентов различных уровней (межуровневый интерфейс). Международные стандарты в области сетевого информационного обмена нашли отражение в эталонной семиуровневой модели, известной как модель OSI (Open Systems Interconnection — связь открытых систем). Данная модель разработана международной организацией по стандартизации (International Standards Organization — OSI). Большинство производителей сетевых программно-аппаратных средств стремятся придерживаться модели OSI.

В современной литературе наиболее часто принято начинать описание уровней модели OSI с 7-го уровня, называемого прикладным, на котором пользовательские приложения обращаются к сети [3]. Модель OSI заканчивается 1-м уровнем — физическим, на котором определены стандарты, предъявляемые независимыми производителями к средам передачи данных:

- тип передающей среды (медный кабель, оптоволокно, радиоэфир и др.),
- тип модуляции сигнала,
- сигнальные уровни логических дискретных состояний (нуля и единицы).

Любой протокол модели OSI должен взаимодействовать либо с протоколами своего уровня, либо с протоколами на единицу выше и/или ниже своего уровня. Взаимодействия с протоколами своего уровня называются горизонтальными, а с уровнями на единицу выше или ниже — вертикальными. Любой протокол модели OSI может выполнять только функции своего уровня и не может выполнять функций другого уровня, что не выполняется в протоколах альтернативных моделей.

Прикладной уровень (уровень приложений; англ. *application layer*) — верхний уровень модели, обеспечивающий взаимодействие пользовательских приложений с сетью:

- позволяет приложениям использовать сетевые службы: удалённый доступ к файлам и базам данных, пересылка электронной почты;
- отвечает за передачу служебной информации;
- предоставляет приложениям информацию об ошибках;
- формирует запросы к уровню представления.

Протоколы прикладного уровня: RDP (Remote Desktop Protocol), HTTP (HyperText Transfer Protocol), SMTP (Simple Mail Transfer Protocol), SNMP (Simple Network Management Protocol), POP3 (Post Office Protocol Version 3), FTP (File Transfer Protocol), XMPP (Extensible Messaging and Presence Protocol), OSCAR (Open System for CommunicAtion in Realtime), Modbus, SIP (Session Initiation Protocol), TELNET и другие [3].

Представительский уровень (уровень представления; англ. *presentation layer*) обеспечивает преобразование протоколов и шифрование/дешифрование данных. Запросы приложений, полученные с прикладного уровня, на уровне представления преобразуются в формат для передачи по сети, а полученные из сети данные преобразуются в формат приложений. На этом уровне может осуществляться сжатие/распаковка или кодирование/декодирование данных, а также перенаправление запросов другому сетевому ресурсу, если они не могут быть обработаны локально.

Уровень представлений обычно представляет собой промежуточный протокол для преобразования информации из соседних уровней. Это позволяет осуществлять обмен между приложениями на разнородных компьютерных системах прозрачным для приложений образом. Уровень представлений обеспечивает форматирование и преобразование кода. Форматирование кода используется для того, чтобы гарантировать приложению поступление информации для обработки, которая имела бы для него смысл. При необходимости этот уровень может выполнять перевод из одного формата данных в другой.

Уровень представлений имеет дело не только с форматами и представлением данных, он также занимается структурами данных, которые используются программами. Таким образом, уровень 6 обеспечивает организацию данных при их пересылке.

Чтобы понять, как это работает, представим, что имеются две системы. Одна использует для представления данных расширенный двоичный код обмена информацией EBCDIC, например, это может быть мейнфрейм компании IBM, а другая — американский стандартный код обмена информацией ASCII (его используют большинство других производителей компьютеров). Если этим двум системам необхо-

димо обменяться информацией, то нужен уровень представлений, который выполнит преобразование и осуществит перевод между двумя различными форматами.

Другой функцией, выполняемой на уровне представлений, является шифрование данных, которое применяется в тех случаях, когда необходимо защитить передаваемую информацию от приема несанкционированными получателями. Чтобы решить эту задачу, процессы и коды, находящиеся на уровне представлений, должны выполнить преобразование данных. На этом уровне существуют и другие подпрограммы, которые сжимают тексты и преобразовывают графические изображения в битовые потоки, так что они могут передаваться по сети.

Стандарты уровня представлений также определяют способы представления графических изображений. Для этих целей может использоваться формат PICT — формат изображений, применяемый для передачи графики QuickDraw между программами.

Другим форматом представлений является формат файлов изображений TIFF, который обычно используется для растровых изображений с высоким разрешением. Следующим стандартом уровня представлений, который может использоваться для графических изображений, является стандарт, разработанный Объединенной экспертной группой по фотографии (Joint Photographic Expert Group); в повседневном пользовании этот стандарт называют просто JPEG.

Существует другая группа стандартов уровня представлений, которая определяет представление звука и кинофрагментов. Сюда входят интерфейс электронных музыкальных инструментов (англ. Musical Instrument Digital Interface, MIDI) для цифрового представления музыки, разработанный Экспертной группой по кинематографии стандарт MPEG, используемый для сжатия и кодирования видеороликов на компакт-дисках, хранения в оцифрованном виде и передачи со скоростями до 1,5 Мбит/с, и QuickTime — стандарт, описывающий звуковые и видеоэлементы для программ, выполняемых на компьютерах Macintosh и PowerPC.

Протоколы уровня представления: AFP — Apple Filing Protocol, ICA — Independent Computing Architecture, LPP — Lightweight Presentation Protocol, NCP — NetWare Core Protocol, NDR — Network Data Representation, XDR — eXternal Data Representation, X.25 PAD — Packet Assembler/Disassembler Protocol [3].

Сеансовый уровень (англ. *session layer*) модели обеспечивает поддержание сеанса связи, позволяя приложениям взаимодействовать между собой длительное время. Уровень управляет созданием/завершением сеанса, обменом информацией, синхронизацией задач, определением права на передачу данных и поддержанием сеанса в периоды неактивности приложений.

Протоколы сеансового уровня: ADSP (AppleTalk Data Stream Protocol), ASP (AppleTalk Session Protocol), H.245 (Call Control Protocol for Multimedia Communication), ISO-SP (OSI Session Layer Protocol (X.225, ISO 8327)), iSNS (Internet Storage Name Service), L2F (Layer 2 Forwarding Protocol), L2TP (Layer 2 Tunneling Protocol), NetBIOS (Network Basic Input Output System), PAP (Password Authentication Protocol), ZIP (Zone Information Protocol), SDP (Sockets Direct Protocol).

Транспортный уровень (англ. *transport layer*) модели предназначен для обеспечения надёжной передачи данных от отправителя к получателю. При этом уровень надёжности может варьироваться в широких пределах. Существует множество классов протоколов транспортного уровня, начиная от протоколов, предостав-

ляющих только основные транспортные функции (например, функции передачи данных без подтверждения приема), и заканчивая протоколами, которые гарантируют доставку в пункт назначения нескольких пакетов данных в надлежащей последовательности, мультиплексируют несколько потоков данных, обеспечивают механизм управления потоками данных и гарантируют достоверность принятых данных. Например, UDP ограничивается контролем целостности данных в рамках одной датаграммы и не исключает возможности потери пакета целиком или дублирования пакетов, нарушения порядка получения пакетов данных; TCP обеспечивает надёжную непрерывную передачу данных, исключающую потерю данных или нарушение порядка их поступления или дублирования, может перераспределять данные, разбивая большие порции данных на фрагменты и, наоборот, склеивая фрагменты в один пакет.

Протоколы транспортного уровня: ATP (AppleTalk Transaction Protocol), CUDP (Cyclic UDP), DCCP (Datagram Congestion Control Protocol), FCP (Fiber Channel Protocol), IL (IL Protocol), NBF (NetBIOS Frames protocol), NCP (NetWare Core Protocol), RTP (Real-time Transport Protocol), SCTP (Stream Control Transmission Protocol), SPX (Sequenced Packet Exchange), SST (Structured Stream Transport), TCP (Transmission Control Protocol), UDP (User Datagram Protocol).

Сетевой уровень (англ. *network layer*) модели предназначен для определения пути передачи данных. Отвечает за трансляцию логических адресов и имён в физические, определение кратчайших маршрутов, коммутацию и маршрутизацию, отслеживание неполадок и «заторов» в сети.

Протоколы сетевого уровня маршрутизируют данные от источника к получателю. Работающие на этом уровне устройства (маршрутизаторы) условно называют устройствами третьего уровня (по номеру уровня в модели OSI). Протоколы сетевого уровня: IP/IPv4/IPv6 (Internet Protocol), IPX (Internetwork Packet Exchange, протокол межсетевого обмена), X.25 (частично этот протокол реализован на уровне 2), CLNP (сетевой протокол без организации соединений), IPsec (Internet Protocol Security). Протоколы маршрутизации — RIP (Routing Information Protocol), OSPF (Open Shortest Path First).

Канальный уровень (англ. *data link layer*) предназначен для обеспечения взаимодействия сетей по физическому уровню и контроля за ошибками, которые могут возникнуть. Полученные с физического уровня данные, представленные в битах, он упаковывает в кадры, проверяет их на целостность и, если нужно, исправляет ошибки (формирует повторный запрос поврежденного кадра) и отправляет на сетевой уровень. Канальный уровень может взаимодействовать с одним или несколькими физическими уровнями, контролируя и управляя этим взаимодействием.

Спецификация IEEE 802 разделяет этот уровень на два подуровня: MAC (англ. *media access control*) регулирует доступ к разделяемой физической среде, LLC (англ. *logical link control*) обеспечивает обслуживание сетевого уровня.

На этом уровне работают коммутаторы, мосты и другие устройства. Эти устройства используют адресацию второго уровня (по номеру уровня в модели OSI).

К протоколам канального уровня относятся — ARCnet, ATM, Controller Area Network (CAN), Econet, Ethernet, Ethernet Automatic Protection Switching (EAPS), Fiber Distributed Data Interface (FDDI), IEEE 802.2 (provides LLC functions to IEEE 802 MAC layers), Link Access Procedures, D channel (LAPD), IEEE 802.11 wireless LAN,

LocalTalk, Multiprotocol Label Switching (MPLS), Point-to-Point Protocol (PPP), Point-to-Point Protocol over Ethernet (PPPoE), Token ring, x.25 [3].

В программировании этот уровень представляет драйвер сетевой платы, в операционных системах имеется программный интерфейс взаимодействия канального и сетевого уровней между собой. Это не новый уровень, а просто реализация модели для конкретной ОС. Примеры таких интерфейсов: ODI, NDIS, UDI.

Физический уровень (англ. *physical layer*) — нижний уровень модели, который определяет метод передачи данных, представленных в двоичном виде, от одного устройства (компьютера) к другому. Составлением таких методов занимаются разные организации, в том числе: Институт инженеров по электротехнике и электронике, Альянс электронной промышленности, Европейский институт телекоммуникационных стандартов и другие. Осуществляют передачу электрических или оптических сигналов в кабель или в радиоэфир и соответственно их приём и преобразование в биты данных в соответствии с методами кодирования цифровых сигналов.

На этом уровне также работают концентраторы, повторители сигнала и медиаконвертеры.

Функции физического уровня реализуются на всех устройствах, подключенных к сети. Со стороны компьютера функции физического уровня выполняются сетевым адаптером или последовательным портом. К физическому уровню относятся физические, электрические и механические интерфейсы между двумя системами. Физический уровень определяет такие виды сред передачи данных как оптоволокно, витая пара, коаксиальный кабель, спутниковый канал передач данных и т. п. Стандартными типами сетевых интерфейсов, относящимися к физическому уровню, являются: V.35, RS-232, RS-485, RJ-11, RJ-45, разъемы AUI и BNC.

Протоколы физического уровня: IEEE 802.15 (Bluetooth), IRDA, EIA RS-232, EIA-422, EIA-423, RS-449, RS-485, DSL, ISDN, SONET/SDH, 802.11 Wi-Fi, Etherloop, GSM Um radio interface, ITU и ITU-T, TransferJet, G.hn/G.9960.

Каждому уровню с некоторой долей условности соответствует свой операнд — логически неделимый элемент данных, которым на отдельном уровне можно оперировать в рамках модели и используемых протоколов. На физическом уровне мельчайшая единица это бит, на канальном уровне информация объединена в кадры, на сетевом уровне в пакеты (дейтаграммы), на транспортном уровне в сегменты. Любой фрагмент данных, логически объединённых для передачи, — кадр, пакет, дейтаграмма — считается сообщением. Именно сообщения в общем виде являются операндами сеансового, представительского и прикладного уровней.

К базовым сетевым технологиям относятся физический и канальный уровни.

2.2.3 Обработка информации

Обработка информации состоит в получении одних «информационных объектов» из других «информационных объектов» путем выполнения некоторых алгоритмов и является одной из основных операций, осуществляемых над информацией, и главным средством увеличения ее объема и разнообразия.

На самом верхнем уровне можно выделить числовую и нечисловую обработку. В указанные виды обработки вкладывается различная трактовка содержания понятия «данные». При числовой обработке используются такие объекты, как перемен-

ные, векторы, матрицы, многомерные массивы, константы и т. д. При нечисловой обработке объектами могут быть файлы, записи, поля, иерархии, сети, отношения и т. д. Другое отличие заключается в том, что при числовой обработке содержание данных не имеет большого значения, в то время как при нечисловой обработке нас интересуют непосредственные сведения об объектах, а не их совокупность в целом.

С точки зрения реализации на основе современных достижений вычислительной техники выделяют следующие виды обработки информации:

- последовательная обработка, применяемая в традиционной фон-неймановской архитектуре ЭВМ, располагающей одним процессором;
- параллельная обработка, применяемая при наличии нескольких процессоров в ЭВМ;
- конвейерная обработка, связанная с использованием в архитектуре ЭВМ одних и тех же ресурсов для решения разных задач, причем если эти задачи тождественны, то это последовательный конвейер, если задачи одинаковые — векторный конвейер.

Основные процедуры обработки данных представлены на рис. 2.1. Создание данных как процесс обработки предусматривает их образование в результате выполнения некоторого алгоритма и дальнейшее использование для преобразований на более высоком уровне.

Модификация данных связана с отображением изменений в реальной предметной области, осуществляемых путем включения новых данных и удаления ненужных.

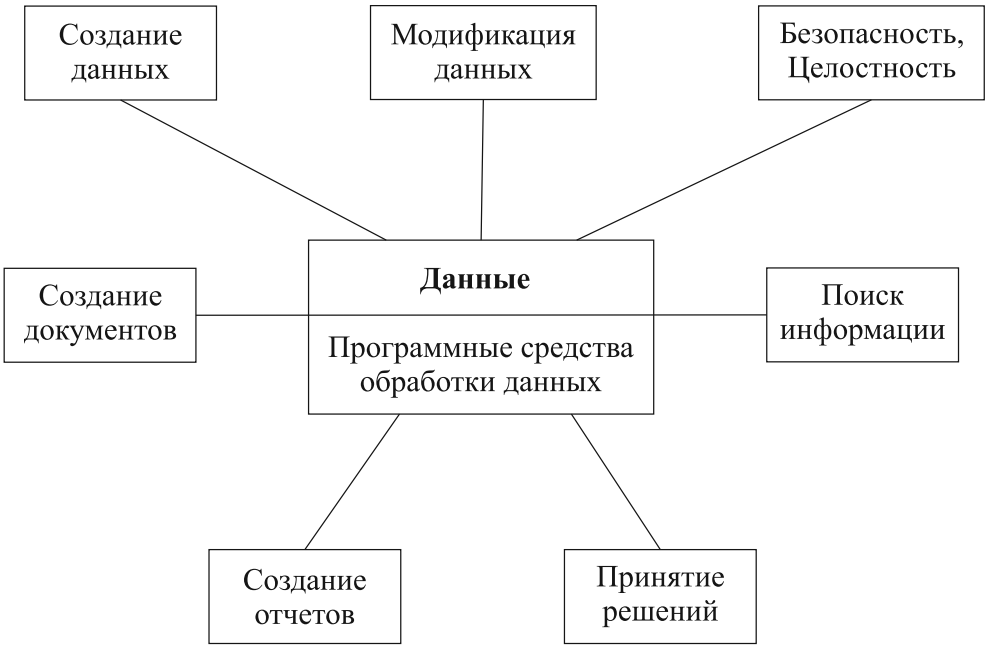


Рис. 2.1 – Основные процедуры обработки данных

Контроль, безопасность и целостность направлены на адекватное отображение реального состояния предметной области в информационной модели и обеспечивают защиту информации от несанкционированного доступа (безопасность) и от сбоев и повреждений технических и программных средств.

Поиск информации, хранимой в памяти компьютера, осуществляется как самостоятельное действие при выполнении ответов на различные запросы и как вспомогательная операция при обработке информации.

Поддержка принятия решения является наиболее важным действием, выполняемым при обработке информации. Широкая альтернатива принимаемых решений приводит к необходимости использования разнообразных математических моделей. Создание документов, сводок, отчетов заключается в преобразовании информации в формы, пригодные для чтения как человеком, так и компьютером. С этим действием связаны и такие операции, как обработка, считывание, сканирование и сортировка документов.

При преобразовании информации осуществляется ее перевод из одной формы представления или существования в другую, что определяется потребностями, возникающими в процессе реализации информационных технологий.

Реализация всех действий, выполняемых в процессе обработки информации, осуществляется с помощью разнообразных программных средств. Наиболее распространенной областью применения технологической операции обработки информации является принятие решений.

В зависимости от степени информированности о состоянии управляемого процесса, полноты и точности моделей объекта и системы управления, взаимодействия с окружающей средой, процесс принятия решения протекает в различных условиях.

Принятие решений в условиях определенности. В этой задаче модели объекта и системы управления считаются заданными, а влияние внешней среды — несущественным. Поэтому между выбранной стратегией использования ресурсов и конечным результатом существует однозначная связь, откуда следует, что в условиях определенности достаточно использовать решающее правило для оценки полезности вариантов решений, принимая в качестве оптимального то, которое приводит к наибольшему эффекту. Если таких стратегий несколько, то все они считаются эквивалентными. Для поиска решений в условиях определенности используют методы математического программирования.

Принятие решений в условиях риска. В отличие от предыдущего случая для принятия решений в условиях риска необходимо учитывать влияние внешней среды, которое не поддается точному прогнозу, а известно только вероятностное распределение ее состояний. В этих условиях использование одной и той же стратегии может привести к различным исходам, вероятности появления которых считаются заданными или могут быть определены. Оценку и выбор стратегий проводят с помощью решающего правила, учитывающего вероятность достижения конечного результата.

Принятие решений в условиях неопределенности. Как и в предыдущей задаче, между выбором стратегии и конечным результатом отсутствует однозначная связь. Кроме того, неизвестны также значения вероятностей появления конечных результатов, которые либо не могут быть определены, либо не имеют в контексте содержательного смысла. Каждой паре «стратегия — конечный результат» соответствует некоторая внешняя оценка в виде выигрыша. Наиболее распространенным является использование критерия получения максимального гарантированного выигрыша.

Принятие решений в условиях многокритериальности. В любой из перечисленных выше задач многокритериальность возникает в случае наличия нескольких самостоятельных, не сводимых одна к другой целей. Наличие большого числа решений усложняет оценку и выбор оптимальной стратегии. Одним из возможных путей решения является использование методов моделирования.

Решение задач с помощью искусственного интеллекта заключается в сокращении перебора вариантов при поиске решения, при этом программы реализуют те же принципы, которыми пользуется в процессе мышления человек.

Экспертная система пользуется знаниями, которыми она обладает в своей узкой области, чтобы ограничить поиск на пути к решению задачи путем постепенного сужения круга вариантов.

Для решения задач в экспертных системах используют:

- метод логического вывода, основанный на технике доказательств, называемой резолюцией и использующей опровержение отрицания (доказательство «от противного»);
- метод структурной индукции, основанный на построении дерева принятия решений для определения объектов из большого числа данных на входе;
- метод эвристических правил, основанных на использовании опыта экспертов, а не на абстрактных правилах формальной логики;
- метод машинной аналогии, основанный на представлении информации о сравниваемых объектах в удобном виде, например в виде структур данных, называемых фреймами.

Источники «интеллекта», проявляющегося при решении задачи, могут оказаться бесполезными либо полезными или экономичными в зависимости от определенных свойств области, в которой поставлена задача. Исходя из этого может быть осуществлен выбор метода построения экспертной системы или использования готового программного продукта.

Процесс выработки решения на основе первичных данных, схема которого представлена на рис. 2.2, можно разбить на два этапа: выработка допустимых вариантов решений путем математической формализации с использованием разнообразных моделей и выбор оптимального решения на основе субъективных факторов.

Информационные потребности лиц, принимающих решение, во многих случаях ориентированы на интегральные технико-экономические показатели, которые могут быть получены в результате обработки первичных данных, отражающих текущую деятельность предприятия. Анализируя функциональные взаимосвязи между итоговыми и первичными данными, можно построить так называемую информационную схему, которая отражает процессы агрегирования информации. Первичные данные, как правило, чрезвычайно разнообразны, интенсивность их поступления высока, а общий объем на интересующем интервале велик. С другой стороны, состав интегральных показателей относительно мал, а требуемый период их актуализации может быть значительно короче периода изменения первичных данных — аргументов.

Для поддержки принятия решений обязательным является наличие следующих компонент:

- обобщающего анализа;

- прогнозирования;
- ситуационного моделирования.

Рис. 2.2 – Процесс выработки решения на основе первичных данных

В настоящее время принято выделять два типа информационных систем поддержки принятия решений.

Системы поддержки принятия решений DSS (Decision Support System) осуществляют отбор и анализ данных по различным характеристикам и включают средства:

- доступа к базам данных;
- извлечения данных из разнородных источников;
- моделирования правил и стратегии деловой деятельности;
- деловой графики для представления результатов анализа;
- анализа «если что»;
- искусственного интеллекта на уровне экспертных систем.

Системы оперативной аналитической обработки OLAP (Online Analysis Processing) для принятия решений используют следующие средства:

- мощную многопроцессорную вычислительную технику в виде специальных OLAP-серверов;
- специальные методы многомерного анализа;
- специальные хранилища данных Data Warehouse.

Реализация процесса принятия решений заключается в построении информационных приложений. Выделим в информационном приложении типовые функциональные компоненты, достаточные для формирования любого приложения на основе БД [2].

PS (Presentation Services) — средства представления. Обеспечиваются устройствами, принимающими ввод от пользователя и отображающими то, что сообщает ему компонент логики представления PL, плюс соответствующая программная поддержка. Может быть текстовым терминалом или X-терминалом, а также персональным компьютером или рабочей станцией в режиме программной эмуляции терминала или X-терминала.

PL (Presentation Logic) — логика представления. Управляет взаимодействием между пользователем и ЭВМ. Обрабатывает действия пользователя по выбору альтернативы меню, по нажатию кнопки или выбору элемента из списка.

BL (Business or Application Logic) — прикладная логика. Набор правил для принятия решений, вычислений и операций, которые должно выполнить приложение.

DL (Data Logic) — логика управления данными. Операции с базой данных (SQL-операторы SELECT, UPDATE и INSERT), которые нужно выполнить для реализации прикладной логики управления данными.

DS (Data Services) — операции с базой данных. Действия СУБД, вызываемые для выполнения логики управления данными, такие как манипулирование данными, определение данных, фиксация или откат транзакций и т. п. СУБД обычно компилирует SQL-приложения.

FS (File Services) — файловые операции. Дисковые операции чтения и записи данных для СУБД и других компонент. Обычно являются функциями ОС.

Среди средств разработки информационных приложений можно выделить следующие основные группы:

- традиционные системы программирования;
- инструменты для создания файл-серверных приложений;
- средства разработки приложений «клиент–сервер»;
- средства автоматизации делопроизводства и документооборота;
- средства разработки Интернет/Интранет-приложений;
- средства автоматизации проектирования приложений.

2.2.4 Хранение информации

Хранение и накопление являются одними из основных действий, осуществляемых над информацией, и главным средством обеспечения ее доступности в течение некоторого промежутка времени. В настоящее время определяющим направлением реализации этой операции является концепция базы данных, склада (хранилища) данных.

База данных может быть определена как совокупность взаимосвязанных данных, используемых несколькими пользователями и хранящихся с регулируемой избыточностью. Хранимые данные не зависят от программ пользователей, для модификации и внесения изменений применяется общий управляющий метод.

Банк данных — система, представляющая определенные услуги по хранению и поиску данных определенной группе пользователей по определенной тематике.

Система баз данных — совокупность управляющей системы, прикладного программного обеспечения, базы данных, операционной системы и технических средств, обеспечивающих информационное обслуживание пользователей.

Хранилище данных (ХД — используют также термины Data Warehouse, «склад данных», «информационное хранилище») — это база, хранящая данные, агрегированные по многим измерениям. Основные отличия ХД от БД: агрегирование данных; данные из ХД никогда не удаляются; пополнение ХД происходит на периодической основе; формирование новых агрегатов данных, зависящих от старых, — автоматическое; доступ к ХД осуществляется, как правило, на основе многомерного куба или гиперкуба.

Альтернативой хранилищу данных является концепция витрин данных (Data Mart). **Витрины данных** — множество тематических БД, содержащих информацию, относящуюся к отдельным информационным аспектам предметной области.

Еще одним важным направлением развития баз данных являются репозитарии. **Репозитарий**, в упрощенном виде, можно рассматривать просто как базу данных, предназначенную для хранения не пользовательских, а системных данных. Технология репозитариев проистекает из словарей данных, которые по мере обогащения новыми функциями и возможностями приобретали черты инструмента для управления метаданными.

Каждый из участников действия (пользователь, группа пользователей, «физическая память») имеет свое представление об информации.

По отношению к пользователям применяют трехуровневое представление для описания предметной области: концептуальное, логическое и внутреннее (физическое).

Концептуальный уровень связан с частным представлением данных группы пользователей в виде внешней схемы, объединяемых общностью используемой информации. Каждый конкретный пользователь работает с частью БД и представляет ее в виде внешней модели. Этот уровень характеризуется разнообразием используемых моделей — модель «сущность–связь» (ER-модель, модель Чена), бинарные и инфологические модели, семантические сети.

Логический уровень является обобщенным представлением данных всех пользователей в абстрактной форме. Используются три вида моделей: иерархические, сетевые и реляционные.

Иерархическая модель является моделью объектов-связей, допускающей только бинарные связи «многие к одному», и использует для описания модель ориентированных графов.

Сетевая модель является разновидностью иерархической, являющейся совокупностью деревьев (лесом).

Реляционная модель использует представление данных в виде таблиц (реляций), в ее основе лежит математическое понятие теоретико-множественного отношения, она базируется на реляционной алгебре и теории отношений.

Физический (внутренний) уровень связан со способом фактического хранения данных в физической памяти ЭВМ. Во многом определяется конкретным методом управления. Основными компонентами физического уровня являются хранимые

записи, объединяемые в блоки; указатели, необходимые для поиска данных; данные переполнения; промежутки между блоками; служебная информация.

По наиболее характерным признакам БД можно классифицировать следующим образом:

- по способу хранения информации:
  - интегрированные;
  - распределенные;
- по типу пользователя:
  - однопользовательские;
  - многопользовательские;
- по характеру использования данных:
  - прикладные;
  - предметные.

В настоящее время при проектировании БД используют два подхода. Первый из них основан на стабильности данных, что обеспечивает наибольшую гибкость и адаптируемость к используемым приложениям. Применение такого подхода целесообразно в тех случаях, когда не предъявляются жесткие требования к эффективности функционирования (объему памяти и продолжительности поиска), существует большое число разнообразных задач с изменяемыми и непредсказуемыми запросами.

Второй подход базируется на стабильности процедур запросов к БД и является предпочтительным при жестких требованиях к эффективности функционирования, особенно это касается быстродействия.

Другим важным аспектом проектирования БД является проблема интеграции и распределения данных. Господствовавшая до недавнего времени концепция интеграции данных при резком увеличении их объема оказалась несостоятельной. Этот факт, а также увеличение объемов памяти внешних запоминающих устройств при их удешевлении, широкое внедрение сетей передачи данных способствовало внедрению распределенных БД. Распределение данных по месту их использования может осуществляться различными способами:

- **Копируемые данные**. Одинаковые копии данных хранятся в различных местах использования, так как это дешевле передачи данных. Модификация данных контролируется централизованно.
- **Подмножество данных**. Группы данных, совместимые с исходной базой данных, хранятся отдельно для местной обработки.
- **Реорганизованные данные**. Данные в системе интегрируются при передаче на более высокий уровень.
- **Секционированные данные**. На различных объектах используются одинаковые структуры, но хранятся разные данные.
- **Данные с отдельной подсхемой**. На различных объектах используются различные структуры данных, объединяемые в интегрированную систему.

- **Несовместимые данные**. Независимые базы данных, спроектированные без координации, требующие объединения.

Важное влияние на процесс создания БД оказывает внутреннее содержание информации. Существует два направления:

- прикладные БД, ориентированные на конкретные приложения, например может быть создана БД для учета и контроля поступления материалов;
- предметные БД, ориентированные на конкретный класс данных, например предметная БД «Материалы», которая может быть использована для различных приложений.

Конкретная реализация системы баз данных, с одной стороны, определяется спецификой данных предметной области, отраженной в концептуальной модели, а с другой стороны — типом конкретной СУБД, устанавливающей логическую и физическую организацию.

Для работы с БД используется специальный обобщенный инструментарий в виде СУБД, предназначенный для управления БД и обеспечения интерфейса пользователя.

Основные стандарты СУБД:

- независимость данных на концептуальном, логическом, физическом уровнях;
- универсальность (по отношению к концептуальному и логическому уровням, типу ЭВМ);
- совместимость, безызбыточность;
- безопасность и целостность данных;
- актуальность и управляемость.

Предназначение склада данных — информационная поддержка принятия решений, а не оперативная обработка данных. Потому база данных и склад данных не являются одинаковыми понятиями. Основные принципы организации хранилищ данных будут рассмотрены в последующих главах.

Рассмотрим кратко основные направления научных исследований в области баз данных:

- развитие теории реляционных баз данных;
- моделирование данных и разработка конкретных моделей разнообразного назначения;
- отображение моделей данных, направленных на создание методов их преобразования и конструирования коммутативных отображений, разработку архитектурных аспектов отображения моделей данных и спецификаций определения отображений для конкретных моделей данных;
- создание СУБД с мультимодельным внешним уровнем, обеспечивающих возможности отображения широко распространенных моделей;
- разработка, выбор и оценка методов доступа;
- создание самоописываемых баз данных, позволяющих применять единые методы доступа для данных и метаданных;
- управление конкурентным доступом;

- развитие системы программирования баз данных и знаний, которые обеспечивали бы единую эффективную среду как для разработки приложений, так и для управления данными;
- совершенствование машины баз данных;
- разработка дедуктивных баз данных, основанных на применении аппарата математической логики и средств логического программирования, а также пространственно-временных баз данных;
- интеграция неоднородных информационных ресурсов.

2.2.5 Представление и использование информации

В условиях использования информационных технологий функции распределены между человеком и техническими устройствами. При анализе деятельности человека наибольшее значение имеют эргономические (инженерно-психологические) и психологические (социально-психологические) факторы.

Эргономические факторы позволяют, во-первых, определить рациональный набор функций человека, во-вторых, обеспечить рациональное сопряжение человека с техническими средствами и информационной средой.

Психологические факторы имеют большое значение, так как внедрение информационных технологий в корне изменяет деятельность человека. Наряду с положительными моментами, связанными с рационализацией деятельности, предоставлением новых возможностей, возникают и негативные явления. Это может быть вызвано различными факторами: психологическим барьером, усложнением функций, другими субъективными факторами (условиями и организацией труда, уровнем заработной платы, результативностью труда, изменением квалификации).

При работе в среде информационных технологий человек воспринимает не сам объект, а некоторую его обобщенную информационную модель, что накладывает особые требования на совместимость пользователя с различными компонентами информационных технологий.

Важным признаком, который необходимо учитывать при разработке и внедрении информационных технологий, является отношение человека к информации. Оно может быть пассивным, когда пользователю предоставляется информация по жесткому алгоритму, и активным, когда пользователь создает необходимые ему данные.

Основной задачей операции представления информации пользователю является создание эффективного интерфейса в системе «человек — компьютер». При этом осуществляется преобразование информации в форму, удобную для восприятия пользователя.

Среди существующих вариантов интерфейса в системе «человек — компьютер» можно выделить два основных типа: на основе меню («смотри и выбирай») и на основе языка команд («вспоминай и набирай»).

Интерфейсы типа меню облегчают взаимодействие пользователя с компьютером, так как не требуют предварительного изучения языка общения с системой. На каждом шаге диалога пользователю предъявляются все возможные в данный момент команды в виде наборов пунктов меню, из которого пользователь должен

выбрать нужный. Такой способ общения удобен для начинающих и непрофессиональных пользователей.

Интерфейс на основе языка команд требует знания пользователем синтаксиса языка общения с компьютером. Достоинством командного языка является его гибкость и мощность.

Указанные два способа реализации интерфейса представляют собой крайние случаи, между которыми возможно существование различных промежуточных вариантов.

Технология представления информации должна давать дополнительные возможности для понимания данных пользователями, поэтому целесообразно использование графики, диаграмм, карт.

Пользовательский интерфейс целесообразно строить на основе концептуальной модели предметной области, которая представляется совокупностью взаимосвязанных объектов со своей структурой. Однако доступ к объектам и их экземплярам возможен только через систему окон различных типов. Ряд окон связан с конкретным объектом. В соответствии с этим предложением в сценарии работы пользователя при информационном наполнении понятий предметной области выделяем две фазы:

- выбор окон;
- работа с окнами.

Для упрощения работы окна можно группировать в соответствии с функциональными потребностями. С этой целью вводится механизм разделов, который предоставляет возможность создания иерархии функционально ориентированных разделов, в каждый из которых включается необходимый набор других разделов и окон. Посредством спецификации окон для каждого из объектов возможно указать допустимые режимы работы с экземплярами и состав видимых атрибутов с режимами работы с ними. Возможно отобрать несколько разделов и несколько окон в них одновременно.

Таким образом, фаза выбора объектов должна поддерживаться следующими функциями:

1) работой с общим каталогом окон в главном разделе;
2) созданием нового раздела;
3) удалением раздела;
4) редактированием описания раздела;
5) передачей определений и окон между разделами;
6) движением по иерархии разделов;
7) отбором разделов для работы;
8) отбором окон для работы.

Позиции окон могут быть связаны с другими окнами через соответствующие команды из типового набора. По существу спецификация окон задает сценарий работы с экземплярами.

Окно — средство взаимосвязи пользователя с системой. Окно представляется как специальный объект. Проектирование пользовательского интерфейса представляет собой процесс спецификации окон.

Примером оконного интерфейса является интерфейс MS Windows, использующий метафору рабочего стола и включающий ряд понятий, близких к естественным (окна, кнопки, меню и т. д.). Пользователь информационной системы большей частью вынужден использовать данные из самых разных источников: файлов, баз данных, электронных таблиц, электронной почты и т. д. При этом данные имеют самую различную форму: текст, таблицы, графика, аудио- и видеоданные и др. В связи с этим возникает проблема интеграции источников информации, заключающаяся в том, что, во-первых, пользователю должны предоставляться не данные, а информация в форме, максимально удобной для восприятия, во-вторых, он должен использовать единственный универсальный интерфейс, позволяющий единообразно работать с подготовленной информацией.

Пассивные пользователи, называемые иногда *потребителями* (юзерами), обладают рядом специфических качеств, связанных с отсутствием времени, желания и квалификации для более глубокого изучения используемых инструментальных средств. В этом случае алгоритм общения с системой должен быть предельно простым. Другая часть пользователей требует предоставления достаточно широкого круга средств активного влияния на выполняемые информационные процессы.

Этим требованиям удовлетворяет Web-технология. Развитие средств вычислительной техники привело к ситуации, когда вместо традиционных параметров — производительность, пропускная способность, объем памяти, узким местом стал интерфейс с пользователем. Первым шагом на пути преодоления кризисной ситуации стала концепция гипертекста, впервые предложенная Теодором Хольмом Нельсоном. По своей сути **гипертекст** — это обычный текст, содержащий ссылки на собственные фрагменты и другие тексты.

Аналогом гипертекста можно считать книгу, оглавление которой по своей сути представляет ссылки на главы, разделы, страницы. Внутри книги содержатся ссылки на другие источники. Дальнейшее развитие гипертекст получил с появлением сети Интернет, позволившей размещать тексты на различных, территориально удаленных компьютерах. Гипертекст стал уже пониматься как набор информационных фрагментов разной природы, объединенных в сеть. При этом потребовалось дальнейшее совершенствование интерфейса, так как имеющийся не позволял представить разнообразную информацию разной природы, был ограничен и затруднен для восприятия, отсутствовал доступ множества потребителей к единому массиву структурированной информации. В результате была предложена и реализована концепция браузера Web. Web-сервер выступает в качестве информационного концентратора, получающего информацию из разных источников и в однородном виде представляющего ее пользователю. Средства Web обеспечивают также представление информации с нужной степенью детализации с помощью Web-браузера. Таким образом, Web — это инфраструктурный интерфейс для пользователей различных уровней.

Несомненным преимуществом Web-технологии является удобная форма предоставления информационных услуг потребителям, имеющая следующие особенности:

- информация предоставляется потребителю в виде публикаций;
- публикация может объединять информационные источники различной природы и географического расположения;

- изменения в информационных источниках мгновенно отражаются в публикациях;
- в публикациях могут содержаться ссылки на другие публикации без ограничения на местоположение и источники последних (гипертекстовые ссылки);
- потребительские качества публикаций соответствуют современным стандартам мультимедиа (доступны текст, графика, звук, видео, анимация);
- публикатор не заботится о процессе доставки информации к потребителю;
- число потенциальных потребителей информации практически не ограничено;
- публикации отражают текущую информацию, время запаздывания определяется исключительно скоростью подготовки электронного документа;
- информация, предоставленная в публикации, легкодоступна благодаря гипертекстовым ссылкам и средствам контекстного поиска;
- информация легко усваивается потребителем благодаря широкому спектру изобразительных возможностей, предоставляемых Web-технологией;
- технология не предъявляет особых требований к типам и источникам информации;
- технология допускает масштабируемые решения: увеличение числа одновременно обслуживаемых потребителей не требует радикальной перестройки системы.

2.3 Классификация информационных технологий

Для того чтобы правильно понять, оценить, грамотно разработать и использовать информационные технологии в различных сферах жизни общества, необходима их предварительная классификация. Классификация информационных технологий зависит от критерия классификации. В качестве критерия может выступать показатель или совокупность признаков, влияющих на выбор той или иной информационной технологии. Как правило, выделяют следующие классификационные признаки информационных технологий:

1. По назначению и характеру использования.
2. По пользовательскому интерфейсу.
3. По способу организации сетевого взаимодействия.
4. По принципу построения.
5. По степени охвата задач управления.
6. По участию ТС в диалоге с пользователем.
7. По способу управления производственной технологией.

1. **По назначению** выделяют следующие два основных класса информационных технологий:

- обеспечивающие информационные технологии;
- функциональные информационные технологии.

Обеспечивающие информационные технологии — это технологии обработки информации, которые могут использоваться как инструменты в различных предметных областях для решения специализированных задач. Они представляют собой способы организации отдельных технологических операций информационных процессов и связаны с представлением, преобразованием, хранением, обработкой или передачей определенных видов информации.

К ним относятся технологии текстовой обработки, технологии работы с базами данных, мультимедиатехнологии, технологии распознавания символов, телекоммуникационные технологии, технологии защиты информации, технологии разработки программного обеспечения, технологии искусственного интеллекта и т. д.

Функциональные информационные технологии — это технологии, реализующие типовые процедуры обработки информации в определенной предметной области. Они строятся на основе обеспечивающих информационных технологий и направлены на обеспечение автоматизированного решения задач специалистов данной области. Модификация обеспечивающих технологий в функциональную может быть сделана как профессиональным разработчиком, так и самим пользователем, что зависит от квалификации пользователя и от сложности модификации.

К функциональным информационным технологиям относятся офисные технологии, финансовые технологии, информационные технологии в образовании, в промышленности, корпоративные информационные технологии, информационные технологии автоматизированного проектирования и т. д.

2. **Информационные технологии можно рассматривать с точки зрения пользовательского интерфейса**, т. е. возможностей доступа пользователя к информационным и вычислительным ресурсам в процессе обработки информации. По этому признаку выделяют:

- пакетные информационные технологии;
- диалоговые информационные технологии;
- сетевые информационные технологии.

Пакетные информационные технологии характеризуются тем, что операции по обработке информации производятся в заранее определенной последовательности и не требуют вмешательства пользователя. В этом случае задания или накопленные заранее данные по определенным критериям объединяются в пакет для последующей автоматической обработки в соответствии с заданными приоритетами. Пользователь не может влиять на ход выполнения заданий, пока продолжается обработка пакета, его функции ограничиваются подготовкой исходных данных по комплексу задач и передачей их в центр обработки. В настоящее время пакетный режим реализуется применительно к электронной почте и формированию отчетности.

Диалоговые информационные технологии предоставляют пользователям неограниченную возможность взаимодействовать с хранящимися в системе информационными ресурсами в режиме реального времени, получая при этом всю необходимую информацию для решения функциональных задач и принятия решений. Эти технологии предполагают отсутствие жестко закрепленной последовательности операций преобразования данных и активное участие пользователя, который анализирует промежуточные результаты и вырабатывает управляющие команды в процессе обработки информации.

Сетевые информационные технологии обеспечивают пользователю доступ к территориально распределенным информационным и вычислительным ресурсам с помощью специальных средств связи. В этом случае появляется возможность использования данных, накопленных на рабочих местах других пользователей, перераспределения вычислительных мощностей между процессами решения различных функциональных задач, а также возможность совместного решения одной задачи несколькими пользователями.

3. **По способу организации сетевого взаимодействия** выделяют:

- информационные технологии на базе локальных вычислительных сетей;
- информационные технологии на базе многоуровневых сетей;
- информационные технологии на базе распределенных сетей.

Информационные технологии на базе локальных вычислительных сетей представляют собой систему взаимосвязанных и распределенных на ограниченной территории средств передачи, хранения и обработки информации, ориентированных на коллективное использование общесетевых ресурсов — аппаратных, программных, информационных. Они позволяют перераспределять вычислительные мощности между пользователями сети в зависимости от изменения их потребностей и сложности решаемых задач и обеспечивают надежный и быстрый доступ пользователей к информационным ресурсам сети.

Построение информационных технологий на базе многоуровневых сетей заключается в представлении архитектуры создаваемой сети в виде иерархических уровней, каждый из которых решает определенные функциональные задачи. Такие технологии строятся с учетом организационно-функциональной структуры соответствующего многоуровневого экономического объекта и позволяют разграничить доступ к информационным и вычислительным ресурсам в зависимости от степени важности решаемых задач и реализуемых функций управления на каждом уровне.

Информационные технологии на базе распределенных сетей обеспечивают надежную передачу разнообразной информации между территориально удаленными узлами сети с использованием единой информационной инфраструктуры. Этот способ организации сетевого взаимодействия ориентирован на реализацию коммуникационных информационных связей между территориально удаленными пользователями и ресурсами сети.

4. **По принципу построения** информационные технологии делятся на следующие виды:

- функционально ориентированные технологии;
- объектно-ориентированные технологии.

При построении функционально ориентированных информационных технологий деятельность специалистов в рассматриваемой предметной области разбивается на множество иерархически подчиненных функций, выполняемых ими в процессе решения профессиональных задач. Для каждой функции разрабатывается технология ее реализации на рабочем месте пользователя, в рамках которой определяются исходные данные, процессы их преобразования в результатную информацию, а также выделяются информационные потоки, отражающие передачу данных между различными функциями.

Построение объектно-ориентированных информационных технологий заключается в проектировании системы в виде совокупности классов и объектов предметной области. При этом иерархический характер сложной системы отражается в виде иерархии классов, ее функционирование рассматривается как совокупность взаимодействующих во времени объектов, а конкретный процесс обработки информации формируется в виде последовательности взаимодействий. В качестве объектов могут выступать пользователи, программы, клиенты, документы, базы данных и т. д. Такой подход характерен тем, что используемые процедуры и данные заменяются понятием «объект», что позволяет динамически отражать поведение моделируемой предметной области в зависимости от возникающих событий.

Сравнительная характеристика функционально ориентированных и объектно-ориентированных технологий приведена в табл. 2.1.

Таблица 2.1 – Сравнительная характеристика функционально ориентированных и объектно-ориентированных технологий

	Функционально-ориентированная технология	**Объектно-ориентированная технология**
Рассматриваемая задача	Учет товаров на складе	
Представление системы	В виде функций: прием товара, отпуск товара, инвентарный контроль и т. д.	В форме классов объектов: товары, клиенты, поставщики, заказы и т. д.
Принцип построения	Разрабатываются технологии для каждой функции и определяются процессы передачи информации от одной функции к другой	Определяются состав и структура каждого класса объектов и процессы информационного взаимодействия этих классов друг с другом и с внешней средой

5. **По степени охвата задач управления** выделяют следующие виды:

- информационные технологии обработки данных;
- информационные технологии управления;
- информационные технологии автоматизации офисной деятельности;
- информационные технологии поддержки принятия решений;
- информационные технологии экспертных систем.

Информационные технологии обработки данных предназначены для решения функциональных задач, по которым имеются необходимые входные данные и известны алгоритмы, а также стандартные процедуры их обработки. Эти технологии применяются в целях автоматизации некоторых рутинных, постоянно повторяющихся операций управленческой деятельности, что позволяет существенно повысить производительность труда персонала. Характерной особенностью этого класса технологий является их построение без пересмотра методологии и организации процессов управления.

Целью информационной технологии управления является удовлетворение информационных потребностей сотрудников, имеющих дело с принятием решений. Эти технологии ориентированы на комплексное решение функциональных задач, формирование регулярной отчетности и работы в информационно-справочном режиме для подготовки управленческих решений. Они решают следующие задачи обработки данных:

- оценка планируемого состояния объекта управления;
- оценка отклонений от планируемых состояний;
- выявление причин отклонений;
- анализ возможных решений и действий.

Информационные технологии автоматизации офисной деятельности направлены на организацию и поддержку коммуникационных процессов как внутри организации, так и с внешней средой на базе компьютерных сетей и других современных средств передачи и работы с информацией. В них реализуются типовые процедуры делопроизводства и контроля управления:

- обработка входящей и исходящей информации;
- сбор и последующее составление отчетности за определенные периоды времени в соответствии с различными критериями выбора;
- хранение поступившей информации и обеспечение быстрого доступа к информации и поиск необходимых данных.

Эти технологии предусматривают наличие интегрированных пакетов прикладных программ: текстовый процессор, табличный процессор, электронная почта, телеконференции, специализированные программы реализации электронного документооборота и т. д.

Информационные технологии поддержки принятия решений предусматривают широкое использование экономико-математических методов, моделей и пакетов прикладных программ для аналитической работы и формирования прогнозов, составления бизнес-планов и обоснованных выводов по изучаемым процессам и явлениям производственно-хозяйственной практики. Отличительными характеристиками этих технологий является ориентация на решение слабо формализованных задач, генерация возможных вариантов решений, их оценка, выбор и предоставление пользователю лучшего из них и анализ последствий принятого решения. Информационные технологии поддержки принятия решений могут использоваться на любом уровне управления и обеспечивают координацию лиц, принимающих решение, как на разных уровнях управления, так и на одном уровне.

Информационные технологии экспертных систем составляют основу автоматизации труда специалистов-аналитиков. Эти работники, кроме аналитических методов и моделей для исследования складывающихся в рыночных условиях ситуаций, могут использовать накопленный и сохраняемый в системе опыт оценки ситуаций, т. е. сведения, составляющие базу знаний в конкретной предметной области. Обработанные по определенным правилам такие сведения позволяют подготавливать обоснованные решения и вырабатывать стратегии управления и развития. Отличие информационных технологий экспертных систем от технологии поддержки принятия решения состоит в том, что они предлагают пользователю

принять решение, превосходящее его возможности, и способны пояснять свои рассуждения в процессе получения решения.

6. **По характеру участия технических средств в диалоге с пользователем** выделяют следующие виды:

- информационно-справочные технологии;
- информационно-советующие технологии.

Информационно-справочные (пассивные) технологии поставляют информацию пользователю после его связи с системой по соответствующему запросу. Технические средства в таких технологиях используются только для сбора и обработки информации об управляемом объекте. На основе обработанной и представленной в удобной для восприятия форме информации оператор принимает решения относительно способа управления объектом.

Информационно-советующие (активные) технологии характеризуются тем, что сами выдают абоненту предназначенную для него информацию периодически или через определенные промежутки времени. В этих системах наряду со сбором и обработкой информации выполняются следующие функции:

- определение рационального технологического режима функционирования по отдельным технологическим параметрам процесса;
- определение управляющих воздействий по всем или отдельным управляемым параметрам процесса и т. д.

7. **По способу управления технологией промышленного производства выделяют**:

- децентрализованные информационные технологии;
- централизованные информационные технологии;
- централизованные рассредоточенные информационные технологии;
- иерархические информационные технологии.

Использование децентрализованных информационных технологий эффективно при автоматизации технологически независимых объектов управления по материальным, энергетическим, информационным и другим ресурсам. Такая технология представляет собой совокупность нескольких независимых технологий со своей информационной и алгоритмической базой. Для выработки управляющего воздействия на каждый объект управления необходима информация о состоянии только этого объекта.

В централизованной информационной технологии осуществляется реализация всех процессов управления объектами в едином органе управления, который осуществляет сбор и обработку информации об управляемых объектах и на основе их анализа в соответствии с критериями системы вырабатывает управляющие сигналы. Основной особенностью **централизованной рассредоточенной информационной технологии** является сохранение принципа централизованного управления, т. е. выработка управляющих воздействий на каждый объект управления на основе информации о состоянии совокупности объектов управления, но при этом некоторые функциональные устройства технологии управления являются общими для всех каналов системы. Для реализации функции управления каждый

локальный орган по мере необходимости вступает в процесс информационного взаимодействия с другими органами управления.

Иерархическая информационная технология построена по принципу разделения функций управления на несколько взаимосвязанных уровней, на каждом из которых реализуются свои процедуры обработки данных и выработка управляющих воздействий. Необходимость использования такой технологии вызвана тем, что с ростом числа задач управления в сложных системах значительно увеличивается объем переработанной информации и повышается сложность алгоритмов управления. Разделение функций управления позволяет справиться с информационными трудностями для каждого уровня управления и обеспечить согласование принимаемых этими органами решений. Иерархическая информационная технология содержит обычно три уровня:

- уровень управления работой оборудования и технологическими процессами;
- уровень оперативного управления ходом производственного процесса;
- уровень планирования работ.

Контрольные вопросы по главе 2

1. Какие информационные процессы являются базовыми?
2. В каких представлениях рассматривается предметная область?
3. Перечислите формы исследования данных.
4. Объясните суть декомпозиции на основе объектно-ориентированного подхода.
5. Что такое инкапсуляция, полиформизм и наследование?
6. Какие существуют методы обогащения информации?
7. Что собой представляет модель OSI?
8. Какие существуют протоколы сетевого взаимодействия?
9. Охарактеризуйте виды обработки информации.
10. Определите содержание основных процедур обработки данных.
11. Поясните особенности принятия решений в различных условиях.
12. Укажите основные компоненты поддержки принятия решений.
13. Какие существуют системы поддержки принятия решений?
14. Дайте характеристику способов организации данных.
15. Что такое СУБД и каковы ее стандарты?
16. Укажите способы реализации СУБД.
17. Опишите содержание процесса проектирования баз данных.
18. Какие существуют критерии оценки баз данных?
19. Что такое интерфейс и какова его роль в процессе представления и использования информации?
20. Какие существуют виды интерфейсов?
21. На чем основана концепция гипертекста?

Глава 3

СОВРЕМЕННЫЕ ТЕХНОЛОГИИ ОБРАБОТКИ ТЕКСТОВЫХ СООБЩЕНИЙ

3.1 Текст и документ

Каждый пользователь компьютера встречается с необходимостью подготовки, редактирования, той или иной текстовой информации. Одними из первых программ, созданных для компьютера, были системы обработки текстов, или, как их стали называть, ***текстовые редакторы***.

Компьютеры, оснащенные текстовыми редакторами, должны были заменить печатные машинки. Соответственно первые подобные программы имели функции ввода символов, простейшего редактирования текста (стирания, переноса, копирования и вставки) и распечатки полученного документа. Отличие от печатной машинки состояло в возможности сохранения готового текста и его последующего многократного использования. Однако реализация функций печатной машинки не могла удовлетворить пользователей компьютеров. Развитие текстовых редакторов шло очень быстро. К тому же параллельно началась разработка полиграфических (издательских) компьютерных программ. Идеи и находки разработчиков различных систем взаимно дополняли и «подпитывали» друг друга, в результате чего появились не только мощные полиграфические программы, но и «бытовые» текстовые редакторы, которые по своим возможностям лишь немногим уступают профессиональным. Возможности этих программ различны — от программ для подготовки небольших документов простой структуры до программ для набора, оформления и полной подготовки к типографическому издательству, изданию книг и журналов (издательские системы). Преимущества компьютера, оснащенного специальным текстовым процессором (редактором), перед печатающей машинкой были явными и заключались в том, что обеспечивали значительное повышение удобства, производительности выполнения работ и, самое главное, повышение качества получаемых при этом документов.

Разделение во времени этапов подготовки документа, таких, как ввод, редактирование, оформление, подготовка к печати и собственно сама печать, сделали процесс создания документа более простым и технологичным.

Существуют различные виды текстовых редакторов:

- Редакторы, предназначенные для подготовки сообщений, содержащих только текст (например, Блокнот). Размер созданного в таком редакторе документа в байтах равен числу символов (букв) в документе. Такие текстовые редакторы могут использоваться для редактирования текстов программ и для подготовки HTML-документов.
- Редакторы, с помощью которых можно редактировать и форматировать (оформлять) документы (например, WordPad). Документы, подготовленные в таких редакторах, содержат не только символы, текст, но и информацию об их формате, то есть форме представления (размере, выделении курсивом и подчеркиванием и т. д.).
- Редакторы, позволяющие готовить комплексные документы, то есть такие, которые содержат не только текст, но и другие объекты — картинки, диаграммы, звук и т. д. Такие редакторы часто называют текстовыми процессорами.

Редакторы текстов программ рассчитаны на редактирование программ на том или ином языке программирования. Редакторы текста и рассчитаны на тексты программ, и выполняют следующие функции:

- диалоговый просмотр текста;
- редактирование строк программы;
- копирование и перенос блоков текста из одного места в другое;
- копирование одной программы или её части в указанное место другой программы;
- контекстный поиск и замену подстрок текста;
- автоматический поиск строки, содержащей ошибку;
- распечатка программы или её необходимой части.

Редакторы документов — программы для обработки текстов, имеющих структуру. Такие тексты могут состоять из разделов, страниц, абзацев, предложений, слов и т. д. Следовательно, редакторы для обработки документов должны обеспечивать такие функции, как:

- возможность использования различных шрифтов, символов;
- задание произвольных межстрочных промежутков;
- автоматический перенос слов на следующую строку;
- автоматическая нумерация страниц;
- обработка и нумерация строк;
- печать верхних и нижних заголовков страниц (колонтитулов);
- выравнивание краев абзаца;
- набор текста в несколько столбцов;

- создание таблиц и построение диаграмм;
- проверка правописания и подбор символов.

Редакторы текстов программ можно использовать для создания и корректировки небольших текстовых сообщений. Однако при необходимости серьезной работы с документами (текстами, имеющими определенную структуру) лучше использовать редакторы, ориентированные на работу с документами.

Современные текстовые процессоры предоставляют пользователю широкие возможности по подготовке документов. Это и функции редактирования, допускающие возможность любого изменения, вставки, замены, копирования и перемещения фрагментов в рамках одного документа и между различными документами, контекстного поиска, функции форматирования символов, абзацев, страниц, разделов документа, верстки, проверки грамматики и орфографии, использования наряду с простыми текстовыми элементами списков, таблиц, рисунков, графиков и диаграмм.

Значительное сокращение времени подготовки документов обеспечивают такие средства автоматизации набора текста, как автотекст и автозамена, использование форм, шаблонов и мастеров типовых документов.

Наличие внешней памяти компьютера обеспечивает удобное длительное хранение подготовленных ранее документов, быстрый доступ к ним в любое время.

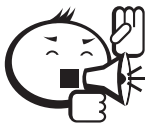
Существенно упрощают процедуру ввода данных сканеры и голосовые устройства. Существующие системы распознавания текстов, принимаемых со сканера, включают функцию экспорта документа в текстовые редакторы.

Широкий спектр печатающих устройств в сочетании с функциями подготовки документа к печати, предварительного просмотра обеспечивает получение высококачественных черно-белых и цветных копий на бумаге и прозрачной пленке. Таким образом, современные программы предусматривают множество функций, позволяющих готовить текстовую часть документа на типографском уровне. Кроме того, современные программы позволяют включать в текст графические объекты: рисунки, диаграммы, фотографии.

Благодаря этим возможностям файл, представляющий собой текстовый документ, может содержать, помимо алфавитно-цифровых символов, обширную двоичную информацию о форматировании текста, а также графические объекты.

3.2 Разметка документа

Каждый документ имеет три составляющие — содержание (смысловое наполнение), структуру и внешнее представление. Структура документа позволяет правильно определить составляющие его части и взаимоотношения между ними. Внешнее представление направлено на повышение эффективности восприятия информации читателем, что достигается за счет выделения смысловых частей документа теми или иными средствами, доступными для данной формы представления.



В документе, помимо смыслового наполнения, должна содержаться некоторая метаинформация, позволяющая определить его структуру и внешнее представление. Такая метаинформация называется ***разметкой документа***.

Исторически слово разметка (markup) использовалось для описания аннотаций или других отметок в тексте, предназначенных для указания машинистке или наборщику, как именно должна быть напечатана или набрана определенная фраза. Примеры включают волнистое подчеркивание для обозначения жирного шрифта, специальные символы для обозначения пропуска отдельных предложений или их печати определенным шрифтом и т. п. С автоматизацией форматирования и печати текстов термин был расширен, охватывая сейчас всяческие коды разметки (markup codes), вставляемые в электронные тексты для управления форматированием, печатью или иной обработкой.

*Обобщая, определим **разметку или кодирование (encoding)**, как любой метод выявления интерпретации текста.*

На примитивном уровне все печатные тексты кодированы в этом смысле: знаки пунктуации, использование заглавных букв, размещение букв на странице, даже пробелы между словами можно считать своеобразной разметкой, функция которой — помочь читателю определить, где заканчивается одно слово и начинается другое, или как отделить структурные элементы, например, заголовки, или элементы локальной структуры, например, подчиненные предложения.

Кодирование текста для компьютерной обработки — процесс выявления того, что неявно или предположительно, процесс указания пользователю, как интерпретировать содержимое текста.

*Под **языком разметки** будем понимать набор соглашений о разметке, используемых в комплексе для кодирования текстов.*

Базовым средством современных технологий обработки текстовых сообщений является SGML — язык разметки текстовых сообщений.

Язык разметки должен специфицировать, ***какая разметка является допустимой, какая — необходимой, как различаются разметка и текст, и что разметка означает***.

Разметка документа преследует следующие две основные цели:

- выделение смысловых частей (логических элементов) документа и связей между ними;
- указание действий, которые должны быть осуществлены с этими элементами.

Для достижения первой цели предназначена ***структурная разметка***. Действия, направленные на получение внешнего представления, задаются ***разметкой представления***.

В качестве примеров ниже приведены два возможных способа разметки начала данного раздела пособия.

Пример 3.1

```
<div1 type=''Section''>
<head> Разметка документа </head>
<p> Каждый документ имеет три составляющие...</p>
</div1>
```

Пример 3.2

```
<font face=''Arial Bold'' size=16>1. Разметка документа <hspace size=20>
<tab size=5><font face=''Times New Roman'' size=12>
Каждый документ имеет три составляющие...
```

В первом случае мы описываем раздел, который имеет заголовок и текст в виде абзаца, то есть определяем структуру документа. Структурная разметка говорит о том, как текст устроен, то есть из каких он частей состоит и как эти части друг с другом соотносятся.

Во втором случае мы показываем, каким образом данный текст должен быть отображен на бумаге или на мониторе — выделить шрифтом Arial Bold размера 16, отступить по вертикали 20, сделать табуляцию 5, выделить шрифтом Times New Roman размера 12. Здесь мы имеем дело с разметкой представления документа, которая говорит о том, что делать с текстом, как его отображать.

Исторически разметка представления появилась раньше, и в течение длительного времени разметка документа была ориентирована исключительно на внешний (бумажный) вид документа. Но в последнее время ситуация существенно меняется — быстрый рост числа документов, их создание, хранение и использование в электронном виде, автоматизированная обработка и обмен документами предъявляют новые требования к разметке. В числе этих требований — независимость от среды представления, возможность осуществления эффективного поиска, возможность повторного использования как документа целиком, так и отдельных его элементов.

Сейчас существует большое число устройств, с помощью которых можно отображать документы. Среди таких устройств — и дисплеи от компьютерных до мобильных, и принтеры от формата A1 до встроенных в кассовые аппараты, и раз-

личные синтезаторы речи, и многое другое. Для воспроизведения некоторого документа на всех этих устройствах требуется либо наличие огромного количества вариантов одного и того же документа, только размеченного разными способами, либо существование единой универсальной разметки и программных средств для корректного преобразования в соответствующее внешнее представление «на лету».

Быстрый рост количества документов привел к тому, что поиск нужной информации стал занимать все больше и больше времени. Например, если нам необходимо найти в Интернете информацию об авторе статей по фамилии Дуров, то простой контекстный поиск даст нам огромное количество ссылок на те места, где встречается данная фамилия. После чего нам придется либо просмотреть все полученные ссылки, либо задавать дополнительную информацию для сужения области поиска. Если бы мы могли сразу указать, что фамилию следует искать только среди авторов журнальных статей технического плана, это во много раз упростило бы поиск. Но для этого необходимо, чтобы документы, среди которых ведется поиск, были размечены должным образом с явным выделением элементов «автор», «тематика» и т. п.

Возможность повторного использования документов или отдельных его частей приводит к тому, что мы не составляем каждый раз заново отчет или деловое письмо, используем в своей работе шаблоны контрактов, изменяя лишь некоторую существенную для данного случая информацию. Но делаем мы это преимущественно вручную. Если говорить об автоматизированном формировании, связывании, повторном использовании документов, то это становится возможным только тогда, когда документы как информационные объекты являются структурированными, а используемая метаинформация полно и ясно описывает характеристики каждого элемента документа.

Все перечисленные задачи можно решить, используя исключительно структурный подход при разметке документов. Именно структурная разметка позволяет выделять смысловые элементы, определять их связи с другими элементами как в рамках одного документа, так и вне этих рамок. Далеко не всякая разметка настолько формализована, что можно говорить о языке разметки. Язык разметки должен определять ряд специальных инструкций, правил и соглашений для описания структуры элементов документа и отношений между элементами этой структуры. Специальные инструкции, их еще называют маркерами или тэгами, в структурированных документах должны определенным образом кодироваться, то есть выделяться среди основного текста. Их главное назначение — служить управляющими инструкциями для программных средств обработки структурированных текстов.

В данной главе мы остановимся на истории возникновения таких языков разметки, как SGML (Standard Generalized Markup Language, стандартный обобщенный язык разметки) и HTML (HyperText Markup Language, язык разметки гипертекстов), а также рассмотрим, что собой представляет XML (eXtensible Markup Language, расширяемый язык разметки).

3.3 Стандартный обобщенный язык разметки SGML

3.3.1 Основные положения SGML

Стандартный обобщенный язык разметки (Standard Generalized Markup Language, SGML) был утвержден международной организацией по стандартизации (International Standards Organisation, ISO) в качестве стандарта ISO 8879:1986 в 1986 году.

SGML — это метаязык, то есть средство формального описания прикладных языков разметки, предназначенных для кодирования структурированных документов.

Разметка, определяемая в рамках SGML, основывается на двух постулатах:

- разметка должна описывать структуру документа, а не указывать, что с документом или его частями должно происходить;
- разметка должна быть строгой, чтобы программы и базы данных могли быть использованы для хранения и обработки размеченных документов.

Структура документа с точки зрения SGML представляет собой граф компонентов, вершины которого являются компонентами, а ребра — связями между ними. Основным компонентом структурированного текста является элемент. Таким образом, можно сказать, что каждый структурированный документ состоит из некоторого набора семантических элементов, связанных друг с другом по определенным правилам.

Синтаксическое представление элемента документа показано на рис. 3.1.

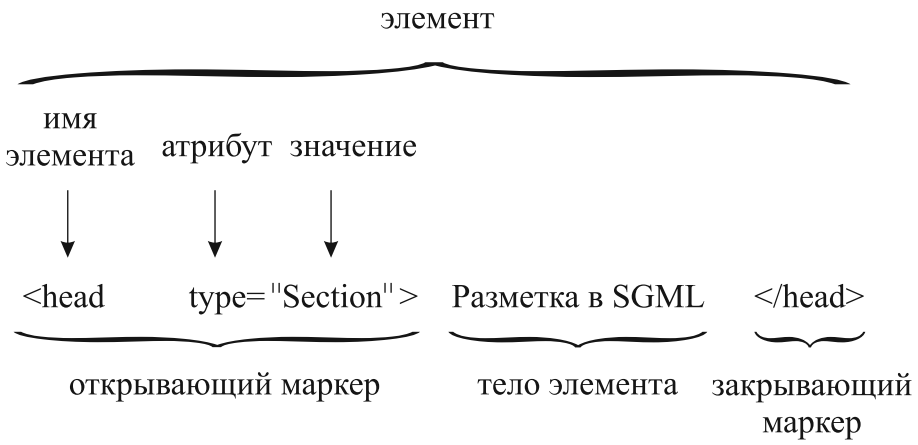


Рис. 3.1 – Пример SGML-элемента

Тело элемента (содержательный текст) обрамляется открывающим и закрывающим маркерами. Каждый маркер состоит из имени элемента, уникального для элементов одинаковой семантики, и может иметь некоторое количество атрибутов. Атрибуты предназначены для более детального описания текста среди семантически однородных элементов.

Важным достоинством SGML является то, что он не определяет заранее имена элементов и их атрибуты. Например, если автор документа считает, что семантически корректнее определить в тексте два типа списков: список фамилий и список компаний, то он может ввести два элемента listofpeople и listofcompanies. В дальнейшем эти элементы могут обрабатываться как различные семантические единицы.

Чтобы документ являлся синтаксически корректным с точки зрения SGML, необходимо, чтобы его разметка подчинялась некоторому набору правил, определяемых стандартом ISO. Одно из правил состоит в том, что допускается лишь полная вложенность одного элемента в другой. Таким образом, в каждом документе всегда будет один корневой элемент и некоторое количество иерархически вложенных элементов. (Вообще говоря, допускается наложение на документ двух независимых разметок, элементы одной из которых могут не являться вложенными в другую, но это предмет отдельного обсуждения.) Вложенность является одним из видов связей между вершинами графа компонентов.

Размеченный документ предназначен для дальнейшей обработки различными программами, каждая из которых может применять свои правила обработки к тем или иным элементам документа. Одна программа может преобразовывать текст к виду, пригодному для печати на бумаге, а другая — лишь извлекать некоторые данные (например, названия терминов) и помещать их в таблицу или базу данных.

3.3.2 Типы документов

Структурная разметка не предназначена для обеспечения удобочитаемости документов. Для этого существует разметка представления и соответствующие программные средства, преобразующие структурную разметку в разметку представления. Эти и другие программы, обрабатывающие документ, должны уметь распознавать элементы структуры и атрибуты элементов и применять необходимые операции к определенным элементам.

В SGML это достигается с помощью определений типов документов (Document Type Definition, DTD) посредством конструкций языка, называемых декларациями элементов. В то время как разметка документа занимается описанием семантических единиц, DTD определяет набор всех возможных разметок документов описываемого типа.

Тип документа формально определяется его составными частями и их структурой. Например, письмо можно определить как документ, имеющий реквизиты отправителя и получателя, заголовок, несколько абзацев и дату отправления. Если документ не имеет реквизитов отправителя, то, в соответствии с нашим определением, письмом он не является.

DTD определяет допустимые элементы для данного типа документа на любом из уровней вложенности, допустимое содержание каждого из элементов и набор допустимых атрибутов. При этом наличие DTD является обязательным для любого документа. Можно сказать, что в рамках SGML имеют право на существование информационные объекты, состоящие из размеченного документа и его DTD.

Декларация элементов в DTD определяет допустимое содержание как тела элемента, так и его атрибутов. Предположим, например, что необходимо дать определение элемента <list>, представляющего собой список. В этом случае декларации могут выглядеть так, как показано в Примере 3.3.

Пример 3.3

```
<!-- ELEMENT MIN CONTENT (EXEPTIONS) -->
<!ELEMENT list     - - (head?, item+)>
<!ELEMENT head     - 0 (#PCDATA)>
<!ELEMENT item     - 0 (p+)>
<!ELEMENT p      - 0 (#PCDATA)>
```

Первая декларация (вторая строка листинга, так как первая является комментарием) обозначает, что список может включать необязательный заголовок, но обязательно содержит один или несколько элементов списка.

Вторая декларация говорит, что заголовок содержит некоторое количество символов (текст).

Третья декларация указывает на то, что каждый элемент списка, в свою очередь, состоит из одного или более абзацев.

И, наконец, последняя декларация, как и вторая, говорит, что абзацы содержат символьный текст.

Символ «0» в колонке MIN обозначает, что закрывающий маркер в данном элементе может быть опущен без нарушения структуры документа. Следующий открывающий маркер такого же элемента или маркер внешнего элемента фактически будет выполнять ту же функцию.

Возможное использование списков приведено в Примере 3.4.

Пример 3.4

```
<list>
  <head>Перечень важных дел
  <item>
    <p>В 11-00 переговоры
    <p>Необходимо подготовить полный комплект документов
  <item>
    <p>В 14-00 совещание у руководства
</list>
<list>
  <head>Перечень важных дел
  <item>
    <p>В 11-00 переговоры
    <p>Необходимо подготовить полный комплект документов
  <item>
    <p>В 14-00 совещание у руководства
</list>
```

Одним из достоинств SGML является то, что он позволяет работать не только со структурированными текстами, но и с произвольными информационными объектами. Для этого вводится понятие объекта (entity).

Объектом может быть строка символов или файл (текстовый или бинарный). Для включения его в документ используется конструкция, известная в ряде языков программирования как ссылка на объект. Например, объявление

<!ENTITY SGML ''Standard Generalized Markup Language''>

определяет объект, называющийся SGML, значением которого является строка ''Standard Generalized Markup Language''. Это пример декларации объекта (entity declaration), которая содержит внутренний объект (internal entity). Следующее объявление, напротив, вводит системный объект (system entity):

<!ENTITY picture SYSTEM ''picture.gif''>

В этом случае определен объект, являющийся рисунком, а не структурированным текстом. При обработке документа некоторой программой файл picture.gif может быть, например, выведен на экран монитора для иллюстрации соответствующего текста.

SGML представляет собой достаточно емкий и, в то же время, сложный метаязык. На его основе создаются языки разметки, используемые в различных областях: подготовка книг, документации, построение систем визуализации данных и т. д. Такие языки, как HTML, XML, MathML, GML, KML и многие другие, созданы на основе SGML и полностью ему соответствуют.

Широта охвата порождает вместе с тем и ряд недостатков. Так, например, создание единого DTD для подготовки документации в рамках одной организации несомненно, имеет преимущества, такие, как унификация исходного кода, возможность автоматического индексирования данных, ведение единого словаря терминов, написание стандартных средств обработки документов, получение стандартного бумажного представления и т. п. Но как только мы выходим за рамки организации, проекта или отрасли, то все упирается в утверждение данного DTD в качестве общего стандарта. Кроме того, как только принимается стандарт на некоторый DTD, сразу начинается борьба за его расширение, и так может продолжаться до бесконечности.

Другой недостаток проявляется при создании программ (например, для редактирования SGML-документов), которые должны позволять работать с любыми возможными DTD и учитывать все возможности, предоставляемые стандартом SGML. К сожалению, это возможно лишь теоретически, так как объем таких программ будет чрезвычайно велик.

Вот почему со временем возникла тенденция создания языков разметки с более простым синтаксисом, которые, в то же время, подчинялись бы требованиям стандарта SGML.

3.4 HTML

Язык разметки HTML родился в Лаборатории физики высоких энергий (CERN) в Женеве в 1990 году. Первоначально HTML был предназначен для разметки научных документов и их последующего совместного использования сотрудниками разных институтов и лабораторий. HTML состоял из небольшого фиксированно-

го набора элементов — заголовков нескольких уровней, абзацев, списков и др., но главной его особенностью было использование гиперссылок и специальных меток (anchors) для указания точек перехода. Все вместе позволяло достаточно легко размечать простые документы и устанавливать связи как между ними, так и между компонентами одного документа. Человек всегда обрабатывает и анализирует информацию нелинейным образом. Поэтому возможности нелинейного хранения информации, простота использования языка разметки и широкие возможности применения привели к тому, что популярность HTML стала быстро расти и вне академических рамок. Как это часто бывает с любыми гениальными открытиями, успех превзошел все ожидания создателей.

В 1992 году HTML был формализован в качестве SGML DTD, при этом в его спецификацию была заложена возможность дальнейшего расширения. Простой синтаксис языка, в отличие от SGML, позволял создавать простые программы для анализа размеченного текста и его отображения. Начался бурный рост публикаций в HTML-формате и рост числа приложений, поддерживающих этот формат. Потребности пользователей, а также конкурентная борьба производителей программного обеспечения привели к тому, что в HTML стали добавляться неспецифицированные элементы разметки. Отсутствие строгих синтаксических правил и использование нестандартных элементов вынудили производителей программного обеспечения допускать использование синтаксически некорректных конструкций. Отметим, что в WWW найдется не так много документов, полностью удовлетворяющих общепринятым спецификациям.

В целях регулирования процесса роста и стандартизации предлагаемых решений для WWW в октябре 1994 года была создана координирующая рабочая группа — World Wide Web Consortium (W3C), которая объединяет представителей более чем 370 организаций. Основными задачами W3C являются накопление информации о WWW, необходимой как разработчикам, так и пользователям, подготовка и утверждение стандартов (технических спецификаций) на технологии, связанные с WWW, и создание прототипов и образцов приложений для демонстрации использования новых технологий.

Положительная роль W3C в судьбе HTML очевидна — этот язык удалось сохранить от разделения на несколько диалектов, правда, ценой постоянного принятия все новых и новых расширенных спецификаций, которые сменяют друг друга с периодичностью раз в два года. Но нельзя же до бесконечности расширять язык, изначально предназначенный совсем для других целей! Борьба за перетягивание одеяла на свою сторону двумя крупнейшими разработчиками Web-навигаторов в конце концов привела к тому, что стандарты начали плыть, а пользователям приходилось очень часто менять программное обеспечение. Сами же пользователи все больше и больше становились зависимыми от разработчиков программных продуктов — у них не было возможности добавлять собственные расширения в языки разметки.

За время своего существования HTML претерпел множество изменений, что весьма неприятно для создателей документов и разработчиков программ. Но гораздо большей неприятностью стало то, что, изначально задуманный как язык структурной разметки, в результате своего развития HTML превратился в язык разметки представления. Чего стоит, например, форматирование документа для

улучшения его внешнего вида с помощью таблиц! Исходный текст таких документов становится практически нечитаемым, а доля полезной информации составляет лишь несколько процентов.

К счастью, ситуация постепенно начинает улучшаться. В версии языка HTML 4.0 содержится около 80 элементов. Темп роста их числа заметно уменьшился. Этому способствовало, прежде всего, введение атрибута CLASS во все элементы. Используя этот атрибут, можно определить новые семантические единицы без изменения синтаксиса языка в целом (рис. 3.2). Кроме того, несомненным шагом вперед по направлению к структуризации языка стало удаление ряда элементов, отвечающих только за внешнее представление, и декларирование строгой необходимости использования таблиц стилей (style sheets) для целей внешнего представления.

```
<div CLASS="author">
   <div CLASS="name">   Дуров Илья     </div>
   <div CLASS="email">   durov@jet.msk.su     </div>
</div>
```

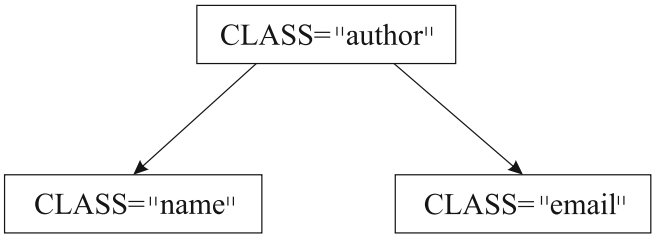


Рис. 3.2 – Пример использования атрибута CLASS

Несмотря на массовое признание и использование HTML, а также на ряд разумных шагов, предпринятых W3C, в HTML все еще имеются существенные недостатки.

Отсутствие жесткой иерархии элементов приводит к тому, что один и тот же документ может быть размечен и, соответственно, будет интерпретироваться программным обеспечением различными способами. Так, например, текст HTML-документа или любая его часть может предваряться заголовком любого уровня, что оставляет автору слишком большую свободу выбора, а читателю создает некоторые трудности при работе с документами разных авторов.

Далеко не всякая метаинформация может быть простым и корректным образом вставлена в документ, поэтому при преобразовании произвольного документа в формат HTML часть информации может быть потеряна. Использование атрибута CLASS только частично решает эту проблему.

Для некоторых областей деятельности HTML не предоставляет возможностей ни структурно размечать требуемые элементы, ни правильным образом выводить их на экран или принтер. Математикам необходима возможность работы с формулами, химикам нужно отображать структуру химических соединений, и, вместе с тем, всем разработчикам и пользователям WWW необходимо наличие единых принципов разметки документов, универсальность их обработки и отображения.

3.5 XML

Из предыдущих разделов видно, что, с одной стороны, максималистский подход при создании SGML привел к чрезмерной сложности языка и соответствующих программных продуктов, что неприемлемо для массового потребления. С другой стороны, простота и ограниченность HTML создавала трудности при описании сложных информационных объектов, поиске необходимой информации, создании приложений, обменивающихся данными через Интернет. Поэтому в 1996 году была сформирована рабочая группа W3C, основной задачей которой являлось создание нового языка разметки. Этот язык должен был включать в себя гораздо больше возможностей SGML, чем HTML, но, в то же время, оставаться подходящим для использования в WWW. Чуть позже этот язык стал известен как XML (eXtensible Markup Language, расширяемый язык разметки). Разработка нового языка разметки велась около двух лет, и в начале февраля 1998 года W3C утвердила в качестве рекомендации первую спецификацию XML — XML версии 1.0.

За сравнительно недолгий срок с момента своего появления на свет XML сумел завоевать огромную популярность среди разработчиков Интернет-технологий. Число созданных и разрабатываемых программных продуктов на основе XML, число компаний, включающих поддержку XML в свои уже готовые продукты, количество публикаций в компьютерной прессе уже весьма велико и продолжает расти. Что это — дань моде или естественное желание сохранить конкурентоспособность, используя современные и прогрессивные технологии?

Как и SGML, XML является метаязыком для формального описания прикладных языков разметки, предназначенных для кодирования структурированных документов. Спецификация XML определяет, как стандартным способом разметить документ, выделяя все семантически значимые компоненты.

При разработке нового языка разметки учитывались достоинства и недостатки уже существующих языков, а также то, что основным местом применения XML является Интернет. Основные требования к создаваемому языку были сформулированы следующим образом:

- XML должен быть годен к непосредственному применению в Интернете.
- XML должен быть совместимым с SGML (XML-документ должен одновременно являться и SGML-документом без внесения каких-либо изменений или дополнений).

  Число необязательных свойств в XML должно быть минимальным, в идеале нулевым (любая XML-программа должна уметь читать любой XML-документ).
- XML-документы должны быть легко читаемы с помощью простейших текстовых процессоров.
- XML-разметка должна быть простой для понимания.

Формальное описание нового языка разметки состоит из нескольких взаимосвязанных частей:

- Спецификации eXtensible Markup Language (XML) 1.0, которая определяет синтаксис языка.

- Спецификаций XML Pointer Language (XPointer) и XML Linking Language (XLink), которые определяют стандартные механизмы установления связей между компонентами XML-документов.
- Спецификации eXtensible Style Language (XSL), которая определяет механизмы для внешнего представления XML-документов.

По своей структуре XML-документ очень похож на SGML- или HTML-документ. В качестве примера 3.5 приведено уже знакомое нам начало раздела.

Существует несколько основных правил составления XML-документа.

Пример 3.5

```
<?xml version=''1.0''?>
<Section>
 <head-of-section> Разметка документа </head-of-section>
 <paragraph> Каждый документ имеет три составляющие. . .</paragraph>
</Section>
```

Каждый документ начинается с пролога. В данном случае — это инструкция <?xml version=''1.0''?>, которая является XML-декларацией. Ее наличие идентифицирует XML-документ и указывает, какой версии XML он соответствует.

В данном листинге нет указания на используемое определение типа документа (DTD), так как, в отличие от SGML, XML не требует обязательного определения DTD для каждого документа. При необходимости описание или указание на месторасположение DTD также помещается в прологе документа.

За прологом следует тело документа, которое представляет собой жесткую структуру элементов, подчиняющихся принципу вложенности. Именование элементов либо соответствует объявленному DTD, либо произвольно. Обязательным является наличие как открывающего, так и закрывающего маркера в каждом элементе, ибо без этого при отсутствии DTD определить структуру документа невозможно.

Каждый из элементов может по аналогии с SGML содержать атрибуты, предназначенные для более детального описания семантически однородных элементов.

Возможно наличие пустых элементов, то есть элементов без содержимого. Такие элементы обозначаются с помощью символа «/» перед закрывающей угловой скобкой, например:

< Empty-Marker/>

В общем случае XML-документ может иметь шесть типов компонент:

- элементы;
- ссылки на текстовые или бинарные объекты (entity references);
- комментарии;
- инструкции обработки;
- отмеченные разделы данных (CDATA sections);
- декларация типа документов.

Мы не будем подробно останавливаться на всех типах компонентов. Отметим лишь, что инструкции обработки, в соответствии со своим названием, предназначены для предоставления информации программам, которые будут в дальнейшем обрабатывать документ. Тип документа определяется тем же способом, что и в SGML, а отмеченные разделы данных позволяют передавать размещенные в них данные или текст «как есть», без анализа его структуры.

Что можно сказать про структурную и семантическую корректность разметки? Необязательность определения DTD, с одной стороны, существенно облегчает XML-разметку, но, с другой стороны, может значительно усложнить программы обработки. Каким образом определить в данном случае корректность XML-документа?

Чтобы определить класс правильно составленных (с точки зрения XML) документов, вводятся понятия структурной и синтаксической корректности. XML-документ является структурно корректным, если он отвечает следующим требованиям:

- Конструкция документа должна отвечать общим правилам составления документа, определенным в спецификации. В частности, некоторые конструкции (например, инструкция <?xml version=''1.0''?>) могут присутствовать только в определенных местах документа.
- Никакой атрибут не используется более одного раза в одном маркере элемента.
- Значения атрибутов не ссылаются на внешние объекты.
- Все непустые элементы удовлетворяют принципу вложенности.
- Все используемые объекты продекларированы.
- Нет ссылок на бинарные объекты непосредственно из текста. Такие ссылки возможны лишь в момент декларации объекта.
- Текстовые объекты не являются рекурсивными.

По определению, если документ не является структурно корректным, то он не является и XML-документом. При наличии у документа DTD возможна его проверка на синтаксическую корректность. При этом XML-документ считается синтаксически корректным, если он является структурно корректным и полностью соответствует всем правилам, изложенным в соответствующем DTD.

Ссылки в XML-документах. Для языка разметки с непредопределенными названиями элементов и даже отсутствующим иногда DTD невозможно определить стандарт на механизм связывания через элементы. Напротив, ссылающиеся и указуемые объекты должны иметь специальные атрибуты, которые идентифицируют их в этом качестве.

Все элементы XML имеют специально зарезервированный атрибут XML-LINK. Присутствие этого атрибута в элементе определяет наличие ссылки, а значение атрибута указывает, какой тип ссылки в данном месте используется. В XML, в отличие от HTML, возможно создание не только однонаправленных гипертекстовых ссылок по типу «один-к-одному», но и двунаправленных ссылок. Используя HTML и перейдя по стандартной гипертекстовой ссылке на новую страницу, пользователь имеет только одну возможность перехода назад — нажатием кнопки «Back» в Web-браузере. При использовании двунаправленных ссылок пользователь не только может вернуться по ссылке в то место, откуда пришел, но и перейти на те страницы, которые ссылаются на указуемый объект.

То, что произойдет при переходе по ссылке, определяется атрибутом SHOW, который может иметь одно из следующих значений: EMBED, REPLACE, NEW.

В первом случае указуемый объект будет импортирован в то место, откуда идет ссылка. Это произойдет либо при показе документа, либо при его обработке. Такой подход может быть полезен при вставке некоторого текста из другого файла или для вставки картинки внутрь текста. При этом возможна как автоматическая подстановка объекта, так и ручная, требующая от пользователя некоторых действий.

Во втором случае ссылающийся объект будет заменен на указуемый. Это может быть полезным, например, при наличии двух вариантов некоторого компонента. При помощи этого механизма возможен просмотр обеих версий или обработка по выбору, в зависимости от наличия тех или иных инструкций обработки.

В последнем случае исходный объект исчезает и происходит полный переход к указуемому объекту. Такой механизм реализован в обычных гипертекстовых ссылках, когда при переходе по ссылке на экране отображается новая HTML-страница.

Механизмы ссылок и адресации в XML описываются в трех спецификациях W3C: XPath, XPointer и Xlink. Xlink описывает механизмы связывания: организацию многонаправленных и однонаправленных ссылок между ресурсами, аннотированных ссылок и внешних наборов ссылок.

Отображение документов. Используя XML, автор документа может самостоятельно определять тот набор элементов, который наиболее точным образом будет соответствовать его структурным компонентам. Но свобода выбора имеет свою цену — набор используемых элементов не обладает предопределенной семантикой. Для совместной работы с XML-документами необходим стандартный механизм получения внешнего представления. Таким механизмом для XML является XSL (eXtensible Style Language, расширяемый язык стилей).

Обычные таблицы стилей, используемые, например, для работы с HTML, содержат набор инструкций, которые говорят программе (Web-навигатору, текстовому редактору или процессору печати), каким образом преобразовывать структуру документа во внешнее представление. При этом таблицы стилей, как правило, содержат такие инструкции, как:

- отображать гипертекстовые связи синим цветом;
- начинать главу с новой страницы;
- вести сквозную нумерацию рисунков по всему документу.

Необходимо понимать, что использование или наложение стиля — это не что иное, как преобразование исходного документа к требуемому виду. Документ, отображаемый на экране, и документ, написанный и размеченный автором, — это совсем не одно и то же. Степень трансформации может меняться в зависимости от презентационных целей — страница документа для публикации в Интернете и для высококачественной полноцветной полиграфической печати должна обрабатываться по-разному, но в любом случае требуется некоторое преобразование.

Использование языков разметки с предопределенной семантикой позволяет существенно упростить реализацию таблицы стилей. Программа, обрабатывающая, например, размеченную таблицу, может отобразить ее различным способом, но она заранее, даже без использования таблицы стилей, знает, что обрабатываемый объект является таблицей.

В случае использования XML-разметки XSL не только должен определять, каким образом тот или иной элемент будет отображаться, скажем, на экране, но и каким объектом он будет в итоге являться. Для того чтобы передать содержание XML-документа наиболее эффективным образом, необходимы две вещи: стандартный язык, описывающий требуемую разметку на выходе (в XSL это форматирующие объекты — formatting objects), и средство для преобразования исходного документа к требуемой разметке (в XSL это язык трансформации — transformation language). XSL включает стандартный словарь форматирующих объектов с хорошо определенными свойствами для осуществления контроля. Эти форматирующие объекты, такие, как страница, блок текста, таблица, список и другие, позволяют авторам стилей получать высококачественное внешнее представление.

Работа с XML начинается с обработки исходного текста программой-анализатором (parser). Эта программа проверяет структурную и синтаксическую корректность XML-документа и создает дерево элементов исходного документа. Далее вступает в действие XSL-процессор, который в качестве исходных данных берет построенное дерево и соответствующий стиль. Шаг за шагом, начиная с корневого элемента, XML-процессор по шаблону, определенному в таблице стилей, обрабатывает всю структуру документа. Получающееся в результате дерево элементов может состоять из форматирующих объектов, которые и описывают внешнее представление документа. Форматирующие объекты представляют собой описание, независимое от устройства представления, и, следовательно, конечный документ может быть использован различными устройствами вывода.

Возможна и альтернатива форматирующим объектам. Так, в случае необходимости преобразования к HTML-виду, вместо форматирующих объектов будут использованы элементы языка разметки HTML. При этом результирующий документ будет выглядеть очень похожим на HTML-документ и обрабатываться стандартными Web-навигаторами. Однако следует понимать, что любое XSL-преобразование XML-документа в результате даст тоже XML-документ.

Основными преимуществами XSL над другими механизмами наложения стилей, помимо возможности работы с элементами непредопределенной семантики, являются:

- возможность изменения порядка следования элементов в результирующем документе;
- возможность сортировки и сравнения элементов текста (список используемых терминов, упомянутых авторов);
- повторная обработка некоторых элементов (например, для печати разными стилями названия главы в начале страницы, в колонтитуле, оглавлении);
- возможность генерации вспомогательного текста («Глава», «Оглавление», «Список иллюстраций» и т. п.);
- возможность подавления вывода некоторого текста (удаление редакторских примечаний или вывод только предисловия, а не полного документа).

Контрольные вопросы по главе 3

1. Что понимается под разметкой документа?
2. Вспомните особенности структурной разметки документа.
3. Что необходимо иметь для эффективной работы с языками разметки?
4. На каких основных положениях основывается разметка, определяемая в SGML?
5. Приведите достоинства SGML.
6. Что определяет DTD?
7. Чем отличается HTML от SGML?
8. Что привело к созданию XML?
9. Чем определяется структура XML-документа?
10. Как определяются ссылки в XML-документах?
11. При решении каких задач целесообразно применять XML?

Глава 4

ИНФОРМАЦИОННЫЕ СИСТЕМЫ ОБРАБОТКИ ДАННЫХ

4.1 Основные классы информационных систем

В 60-х годах двадцатого столетия была осознана необходимость применения средств компьютерной обработки хранимой информации там, где были накоплены значительные объемы полезных данных, — в военной промышленности, в бизнесе. Появились автоматизированные информационные системы (АИС) — программно-аппаратные комплексы, предназначенные для хранения, обработки информации и обеспечения ею пользователей. Первые АИС работали преимущественно с информацией фактического характера, например с характеристиками объектов и их связей. По мере «интеллектуализации» АИС появилась возможность обрабатывать текстовые документы на естественном языке, изображения и другие виды и форматы представления данных.

Несмотря на то, что принципы хранения данных в системах обработки фактической и документальной (текстовой) информации схожи, алгоритмы обработки в них заметно различаются. Поэтому в зависимости от характера информационных ресурсов, которыми оперируют такие системы, принято различать два крупных их класса — документальные и фактографические.

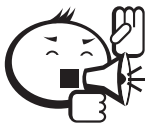
Документальные системы служат для работы с документами на естественном языке — монографиями, публикациями в периодике, сообщениями пресс-агентств, текстами законодательных актов. Они обеспечивают их смысловой анализ при неполном, приближенном представлении смысла. Наиболее распространенный тип документальных систем — информационно-поисковые системы (ИПС), предназна-

ченные для накопления и поиска по различным критериям документов на естественном языке.

Другой большой класс автоматизированных систем — фактографические системы. Они оперируют фактическими сведениями, представленными в виде специальным образом организованных совокупностей формализованных записей данных. Центральное функциональное звено фактографических информационных систем — системы управления базами данных (СУБД). Фактографические системы используются не только для реализации справочных функций, но и для решения задач обработки данных. Под обработкой данных понимается специальный класс решаемых на ЭВМ задач, связанных с вводом, хранением, сортировкой, отбором и группировкой записей данных однородной структуры. В большинстве случаев эти задачи предусматривают предоставление пользователям итоговых результатов обработки в виде отчетов табличной формы.

Задачи, связанные с обработкой данных, широко распространены в любой деятельности. На их основе ведут учет товаров в супермаркетах и на складах, начисляют зарплату в бухгалтериях и т. д. Невозможно представить себе деятельность современного предприятия или учреждения без использования АИС. Эти системы составляют фундамент информационной деятельности во всех сферах, начиная с производства, управления финансами и телекоммуникациями и заканчивая управлением семейным бюджетом.

Массивы информации, накопленные в АИС, должны быть оптимальным образом организованы для их компьютерного хранения и обработки, должна обеспечиваться их целостность и непротиворечивость. Используя функции стандартных файловых систем, невозможно добиться нужной производительности при решении подобных задач, поэтому все автоматизированные информационные системы опираются на СУБД — системы управления базами данных.



Среди фактографических систем важное место занимают два класса: системы операционной обработки данных и системы, ориентированные на анализ данных и поддержку принятия решений.

Первые рассчитаны на быстрое обслуживание относительно простых запросов большого числа пользователей. Системы операционной обработки работают с данными, которые требуют защиты от несанкционированного доступа, от нарушений целостности, от аппаратных и программных сбоев. Время ожидания выполнения типичных запросов в таких системах не должно превышать нескольких секунд. Сфера применения таких систем — это системы платежей, резервирования мест в поездах, самолетах, гостиницах, банковские и биржевые системы. Логическая единица функционирования систем операционной обработки данных — транзакция. Транзакция — это некоторое законченное, с точки зрения пользователя, действие над базой данных. В современной литературе для обозначения систем операционной обработки часто используют термин OLTP (On-Line Transaction Processing — оперативная обработка транзакций или выполнение транзакций в режиме реального времени). Ниже мы определим понятие транзакции, рассмотрим, как происходит выполнение транзакций в OLTP-системах, как в них поддержива-

ется целостность БД и какие средства используются для эффективного управления ресурсами в распределенных системах операционной обработки данных.

Другой класс информационных систем — системы поддержки принятия решений (аналитические системы). Эти системы ориентированы на выполнение более сложных запросов, требующих статистической обработки исторических (накопленных за некоторый промежуток времени) данных, моделирования процессов предметной области, прогнозирования развития тех или иных явлений. Аналитические системы также часто включают средства обработки информации на основе методов искусственного интеллекта, средства графического представления данных. Эти системы оперируют большими объемами исторических данных, позволяя выделить из них содержательную информацию — получить знания из данных.

Современные требования к скорости и качеству анализа привели к появлению систем оперативной аналитической обработки (OLAP — On-Line Analysis Processing). Оперативность обработки больших объемов данных в таких системах достигается за счет применения мощной, в том числе многопроцессорной, вычислительной техники, сложных методов анализа, а также специальных хранилищ данных, накапливающих информацию из различных источников за большой период времени и обеспечивающих быстрый доступ к ней.

Пример 4.1

Оба класса систем основаны на СУБД, но типы выполняемых ими запросов сильно различаются. Например, в OLTP-системе продажи железнодорожных билетов допустим такой запрос: «Есть ли свободные места в купе поезда Москва–Сочи, отправляющегося 20 августа в 23.15?». В аналитической системе запрос может быть таким: «Каким будет объем продажи железнодорожных билетов в денежном выражении в следующие три месяца с учетом сезонных колебаний?».

Принципиально отличаются и структуры баз данных для высокопроизводительных OLAP- и OLTP-систем.

Эти отличия, а также особенности обработки данных в OLTP- и OLAP-системах будут рассмотрены далее.

4.2 Особенности обработки данных в OLTP-системах

4.2.1 Обработка транзакций

Как уже было сказано выше, основной логической единицей функционирования систем операционной обработки данных является транзакция. Транзакцией называют неделимую с позиции воздействия на БД последовательность операции манипулирования данными. Транзакция может состоять из операции чтения, удаления, вставки, модификации данных. В OLTP-системах транзакция реализует

некоторое осмысленное, с точки зрения пользователя, действие, например перевод денег со счета на счет в платежной системе банка, резервирование места в поезде системой оформления железнодорожных билетов.

Традиционно понятие «обработка транзакций» использовалось применительно к крупномасштабным системам обработки данных — системам, осуществлявшим международные банковские операции, и др. Теперь ситуация меняется. Информационные системы в различных областях человеческой деятельности становятся все более распределенными и неоднородными, в них остро стоят проблемы сохранения целостности данных и разграничения доступа. Одно из направлений решения этих проблем — использование средств обслуживания транзакций в информационных системах.

Чтобы использование механизмов обработки транзакций позволило обеспечить целостность данных и изолированность пользователей, транзакция должна обладать четырьмя основными свойствами: атомарности (atomicity), согласованности (consistency), изолированности (isolation), долговечности (darability). Транзакции, обладающие перечисленными свойствами, иногда называют ACID-транзакциями по первым буквам их английских названий.

Свойство атомарности означает, что транзакция должна выполняться как единая операция доступа к БД. Она должна быть выполнена полностью либо не выполнена совсем. То есть должны быть выполнены все операции манипулирования данными, которые входят в транзакцию, либо, если по каким-то причинам выполнение части операций невозможно, ни одна из операций не должна выполняться. Свойство атомарности обычно коротко выражают фразой: «все или ничего».

Свойство согласованности гарантирует взаимную целостность данных, то есть выполнение ограничений целостности БД после окончания обработки транзакции. Следует отметить, что база данных может обладать такими ограничениями целостности, которые сложно не нарушить, выполняя только один оператор ее изменения. Например, если в отношении A хранится число кортежей отношения B, то добавить новый кортеж в отношение B, не нарушив ограничений целостности, невозможно. Поэтому такое нарушение внутри транзакции допускается, но к моменту ее завершения база данных должна быть в целостном состоянии. Несоблюдение в системах со средствами контроля целостности этого условия приводит к отмене всех операции транзакции.

В многопользовательских системах с одной БД одновременно могут работать несколько пользователей или прикладных программ. Поскольку каждая транзакция может изменять разделяемые данные, данные могут временно находиться в несогласованном состоянии. Доступ к этим данным другим транзакциям должен быть запрещен, пока изменения не будут завершены. Свойство изолированности транзакций гарантирует, что они будут выполняться отдельно друг от друга.

Свойство долговечности означает, что если транзакция выполнена успешно, то произведенные в ходе ее выполнения изменения в данных не будут потеряны ни при каких обстоятельствах.

Результатом выполнения транзакции может быть ее фиксация или откат. Фиксация транзакции — это действие, обеспечивающее запись в БД всех изменений, которые были произведены в процессе ее выполнения. До того как транзакция зафиксирована, возможна отмена всех сделанных изменений и возврат базы данных

в то состояние, в котором она была до начала выполнения транзакции. Фиксация транзакции означает, что все результаты ее выполнения становятся видимыми другим транзакциям. Для фиксации транзакции необходимо успешное выполнение всех ее операторов.

Если нормальное завершение транзакции невозможно, например нарушены ограничения целостности БД или пользователь выдал специальную команду, происходит откат транзакции. База данных возвращается в исходное состояние, все изменения аннулируются.

Механизм корректного отката и фиксации транзакций основан на использовании журнала транзакций. Для того чтобы иметь возможность сделать откат, СУБД должна сохранять все изменения, которые транзакция внесла в БД. Однако нет необходимости каждый раз сохранять всю информацию базы данных. Реляционные операции изменяют строки отношений БД, поэтому, чтобы обеспечить возможность отката, СУБД должна хранить те строки, которые были модифицированы. При выполнении любой операции, изменяющей базу данных, СУБД автоматически сохраняет в журнале транзакций состояние модифицируемых строк до операции и после нее. Только после этого изменения вносятся в БД. Если по окончании обработки транзакция фиксируется, то в журнале делается соответствующая отметка. Если же производится откат транзакции, то СУБД по журналу восстанавливает те строки отношений, которые были модифицированы, отменяя, таким образом, все изменения.

Для того чтобы оперировать транзакцией как единой логической единицей, СУБД должна уметь определять ее границы, то есть первую и последнюю входящую в нее операции. Стандарт языка SQL предусматривает следующий принцип выделения транзакции как некоторой законченной последовательности действий. Предполагается, что транзакция начинается с первого SQL-оператора, вводимого пользователем или содержащегося в прикладной программе. Все следующие далее операторы составляют тело транзакции. Тело транзакции завершается SQL-операторами COMMIT WORK или ROLLBACK WORK. Выполнение транзакции заканчивается также при завершении программы, которая сгенерировала транзакцию. Транзакция фиксируется, если ее тело оканчивается оператором COMMIT WORK либо при успешном завершении программы, сформировавшей транзакцию. Откат транзакции производится при достижении оператора ROLLBACK WORK либо в случае, когда приложение, сгенерировавшее транзакцию, завершилось с ошибкой.

Пример 4.2

Рассмотрим пример транзакции, модифицирующей телефон (атрибут Phone) сотрудника с фамилией (атрибут Name) «Петров» в отношении Отдел (Department). Транзакция завершается фиксацией по достижении оператора COMMIT WORK.

```
UPDATE Department SET Phone = ''5388'' WHERE Name = ''Петров''
COMMIT WORK
```

Некоторые диалекты языка SQL, например диалект, принятый в СУБД Sybase, включают специальные операторы, позволяющие производить промежуточную фиксацию транзакции. В теле транзакции могут быть определены точки, в которых сохраняется состояние базы данных. Откат в этом случае может производиться как к одной из точек промежуточной фиксации, так и к состоянию до начала выполнения транзакции. Точки промежуточной фиксации применяются в «длинных» транзакциях. Они позволяют разделить ее на несколько отдельных фрагментов.

Применение транзакций — эффективное механизм организации многопользовательского доступа к БД. Однако при реализации этого механизма СУБД приходится сталкиваться с целым рядом проблем. Во-первых, необходимо избежать потери изменений БД в ситуации, когда несколько программ читают одни и те же данные, изменяют их и пытаются записать результат на прежнее место. В БД могут быть сохранены изменения, выполненные только одной программой, результаты работы всех остальных программ будут потеряны. Во-вторых, требуется исключить возможность чтения незафиксированных изменений. Это может произойти в случае, когда одна транзакция вносит изменения в БД, они тут же считываются в другой транзакции, но затем другая транзакция прерывается оператором ROLLBACK WORK.

Чтобы избежать этих проблем, должна быть использована специальная дисциплина совместной обработки (сериализации) транзакций. В ее основе лежат следующие принципы.

1. Транзакция не может получить доступ к незафиксированным данным, то есть к данным, в которых произведены изменения, но эти изменения еще не зафиксированы.
2. Результат совместного выполнения транзакций должен быть эквивалентен результату их последовательного выполнения. То есть если две транзакции выполняются параллельно, то полагается, что результат будет такой же, как если бы сначала выполнялась первая, а затем вторая транзакция или сначала вторая, а потом первая. В современных СУБД сериализация транзакций реализуется через механизм блокировок.

На время выполнения транзакции СУБД блокирует часть БД (отношение, строку или группу строк), к которой транзакция обращается. Блокировка сохраняется до момента фиксации транзакции. Если в процессе параллельной обработки другой транзакции делается попытка обратиться к блокированным данным, обработка транзакции приостанавливается и возобновляется только после завершения транзакции, заблокировавшей данные и снятия блокировки.

При выполнении транзакции современные СУБД могут блокировать всю БД, отношение, группу строк или отдельную строку. Очевидно, что чем больше блокируемый элемент данных, тем медленнее СУБД обрабатывает транзакции — велико время ожидания снятия блокировок. При работе в режиме оперативного доступа к БД, как правило, реализуется блокировка на уровне отдельных строк. В этом случае можно добиться максимальной производительности за счет того, что блокируемый объект — минимальная структурная единица БД.

Транзакции могут попасть в ситуацию взаимоблокировки. Для предотвращения таких ситуаций СУБД периодически проверяет блокировки, установленные выполняющимися транзакциями. Если обнаруживается ситуация взаимоблокировки, то одна из транзакций, вызвавших эту ситуацию, прерывается. Это разрешает

тупиковые ситуации. Программа, которая сгенерировала прерванную транзакцию, получает сообщение об ошибке. Для того чтобы избежать взаимоблокировок, стараются в каждой транзакции обновление отношений делать в одной и той же последовательности.

Современные информационные системы работают с распределенными БД, поэтому в одной транзакции могут модифицироваться отношения, физически хранящиеся на удаленных вычислительных системах. Транзакция, обновляющая данные на нескольких узлах сети, называется распределенной. Если транзакция работает с БД, расположенной на одном узле, то ее называют локальной. Таким образом, логически распределенная транзакция состоит из нескольких локальных.

С точки зрения пользователя, локальные и глобальные транзакции должны обрабатываться одинаково, то есть СУБД должна организовать процесс выполнения распределенной транзакции так, чтобы все локальные транзакции, входящие в нее, синхронно фиксировались на затрагиваемых ими узлах распределенной системы. Однако распределенная транзакция должна фиксироваться только в случае, когда зафиксированы все локальные транзакции, ее составляющие. Если прерывается хотя бы одна из локальных транзакций, должна быть прервана и распределенная транзакция.

Для практической реализации этих требований в СУБД используют механизм двухстадийной фиксации транзакций (two phase commit). При его использовании фиксация распределенных транзакций осуществляется в два этапа (стадии). На первой стадии сервер БД, фиксирующий распределенную транзакцию, посылает команду «приготовиться к фиксации» всем узлам сети (серверам БД), задействованным для выполнения локальных транзакций, инициированных распределенной транзакцией. Все серверы локальных БД должны в ответ сообщить, что они готовы к фиксации. Если хотя бы от одного из серверов ответ не получен, например если имела место программная или аппаратная ошибка при выполнении локальной транзакции, то сервер распределенной БД производит откат локальных транзакций на всех узлах, включая те, которые прислали оповещение о готовности к фиксации.

Вторая стадия начинается, когда все локальные СУБД готовы к фиксации. Сервер, обрабатывающий распределенную транзакцию, заканчивает ее фиксацию, посылая команду «зафиксировать транзакцию» всем локальным серверам.

Описанный механизм фиксации гарантирует синхронную фиксацию распределенной транзакции на всех узлах сети.

4.2.2 Тиражирование данных

Описанный подход выполнения транзакций в распределенных системах не единственно возможный. Альтернатива ему — технология **тиражирования данных**. Эта технология предполагает отказ от распределенности данных — во всех узлах вычислительной системы должна находиться своя копия БД. Средства тиражирования автоматически поддерживают согласованное состояние информации в нескольких БД посредством копирования изменений, вносимых в любую из них. Любая транзакция в такой системе выполняется локально, поэтому нет необходимости в сложной процедуре фиксации.

Узкое место такого подхода — обеспечение тождественности данных в узлах сети. Процесс переноса изменений исходной БД в базы, принадлежащие различным

узлам распределенной системы, принято называть тиражированием данных. Функции тиражирования данных выполняет специальный модуль СУБД — сервер тиражирования данных (репликатор). При любых изменениях в тиражируемых данных репликатор копирует их на все остальные узлы системы. Схема тиражирования может быть построена на полном обновлении содержимого таблицы на удаленных серверах (схема с полным обновлением) или же на обновлении только изменившихся записей (быстрое обновление). Если в системе нет необходимости поддерживать постоянную идентичности данных и БД узлов должны согласовываться лишь периодически, репликатор накапливает изменения и в нужные моменты времени копирует их на другие узлы. Процесс тиражирования данных скрыт от прикладных программ пользователей, репликатор автоматически поддерживает БД в согласованном состоянии.

При использовании технологии тиражирования уменьшается график, так как все запросы обрабатываются локальной СУБД, а на другие узлы передаются только изменения в данных, увеличивается скорость доступа к данным. Кроме того, обрыв связи между узлами не останавливает обработку данных. Однако эта технология не лишена недостатков. Так, невозможно полностью исключить конфликты, возникающие при одновременном изменении одних и тех же данных на разных узлах. При переносе этих изменений в узлах вычислительной системы могут оказаться несогласованные копии БД, в результате чего пользователи различных узлов распределенной БД будут получать разные ответы на одни и те же запросы.

4.2.3 Надежность хранения данных

Одно из основных требований к современным OLTP-системам — надежность хранения данных. СУБД должна уметь восстанавливать согласованное состояние базы данных после любых аппаратных и программных сбоев. Для восстановления после сбоев СУБД использует журнал транзакций, который содержит последовательность записей, описывающих изменения в БД.

Общий принцип восстановления после сбоя таков — результаты выполнения транзакций, зафиксированных до сбоя, должны присутствовать в восстановленной БД, результаты незафиксированных транзакций в ней должны отсутствовать. То есть восстанавливается последнее до сбоя согласованное состояние базы данных. Процесс восстановления основан на механизме отката незавершенных транзакций, который описан ранее.

Конечно, журнал транзакций не поможет, если содержимое внешней памяти системы физически уничтожено, утеряна вся информация БД. Для того чтобы избежать подобных ситуаций, реализуют дублированное хранение данных, например зеркалирование дискового пространства — запись данных одновременно на несколько устройств. После сбоя копируется содержимое БД, а затем, как и в первом случае, на основе журнала откатываются все незавершенные транзакции.

4.2.4 Мониторы транзакций

С ростом сложности распределенных вычислительных систем возникают проблемы эффективного использования их ресурсов. Для решения этих проблем в состав распределенных OLTP-систем вводят дополнительный компонент — монитор

транзакций (TPM — transaction processing monitor). Мониторы транзакций выполняют две основные функции: динамическое распределение запросов в системе (выравнивание нагрузки) и оптимизация числа выполняющихся серверных приложений. Кратко рассмотрим эти функции.

Если в системе функционирует несколько серверов, предоставляющих одинаковый сервис, например доступ к БД, то для оптимизации пропускной способности системы (числа обрабатываемых запросов в единицу времени) необходимо добиться сбалансированной их загрузки. То есть необходимо обеспечить, чтобы на каждый из них поступало примерно равное число пользовательских запросов. При распределении запросов может учитываться также удаленность серверов, их готовность, содержимое запроса. Реализуется функция выравнивания нагрузки следующим образом (см. рис. 4.1).

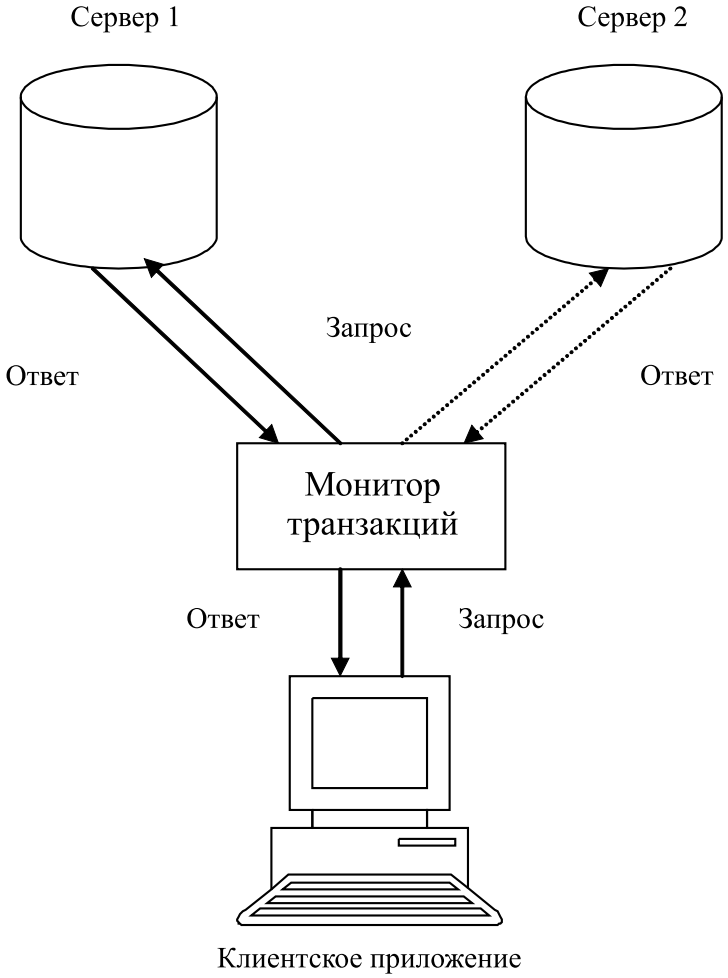


Рис. 4.1 – Упрощенная схема работы монитора транзакций

Запрос клиентского приложения поступает монитору транзакций, который, действуя от имени клиентского приложения, определяет получателя этого запроса. Для этого он обращается к динамической маршрутной таблице, по которой определяет систему, предоставляющую соответствующий сервис. Если нужный сервис предлагают несколько систем, то в зависимости от используемого алгоритма маршрутизации выбирается одна из них, после чего ей перенаправляется

запрос клиентского приложения. Маршрутизация может быть произвольной, когда система выбирается случайным образом, циклической, когда запросы посылаются системам по очереди, либо определяться содержимым запроса, если, например, серверы БД обслуживают разные подмножества данных. Результат выполнения запроса через монитор транзакций перенаправляется приложению, пославшему запрос. Клиентские приложения не знают о том, какой системе будут направлены их запросы, предлагается ли нужный им сервис одним или несколькими серверами, расположен ли нужный сервер локально, удаленно или одновременно локально и удаленно, — в любом случае их запрос будет обработан оптимальным образом. Подобную схему обработки запросов называют «прозрачность местонахождения серверов» (service location transparency).

Скорость обработки транзакций напрямую зависит от числа запущенных серверных приложений. Чем больше приложений одновременно обслуживает запросы, тем выше пропускная способность вычислительной системы. Это увеличение наиболее заметно на многопроцессорных системах, где каждое приложение может работать на отдельном процессоре. В идеале для эффективного использования системных ресурсов нужно по мере необходимости увеличивать или уменьшать число серверных приложений в зависимости от числа обрабатываемых запросов.

Для решения этой задачи мониторы транзакций периодически измеряют отношение числа запросов в очереди к числу работающих серверных приложений. Если это отношение превышает некоторое максимальное пороговое значение (maximum watermark), то запускается дополнительная копия серверного приложения. Если это отношение падает ниже минимального порогового значения (minimum watermark), то одна из копий завершается.

На рынке мониторов транзакций доступно довольно много продуктов. В числе наиболее известных: TUXEDO фирмы USL, TOP END фирмы NCR, CICS фирмы IBM, ENCINA фирмы Transarc, ACMS фирмы DEC.

4.3 Системы многомерного анализа данных

4.3.1 Хранилища данных

К середине 80-х годов в развитых странах мира завершился первый этап оснащения бизнеса и органов государственного управления средствами вычислительной техники. Военные ведомства и крупные корпорации установили распределенные вычислительные системы, состоявшие из мощных мейнфреймов. С появлением персональных компьютеров ЭВМ стали доступны множеству средних фирм и организаций. Исторически эти системы в первую очередь реализовывали потребности в операционной обработке данных — они обслуживали информационные архивы, телефонные сети, системы резервирования билетов, сбора метеоданных и др. Использование мощных средств вычислительной техники позволило накапливать большие объемы информации: документы, сведения о банковских операциях, клиентах, предоставленных услугах. Однако период хранения этой информации был относительно невелик — сохранялись только данные за текущий календарный период.

Вскоре возникло понимание, что сбор данных — не самоцель и накопленные информационные массивы могут быть полезны. Системы операционной обработки способны выполнять тривиальный анализ данных — вычислять максимальные, минимальные и средние значения атрибутов. Но из накопленных данных можно почерпнуть намного более глубокие сведения как о функционировании организации, которая обслуживается информационной системой, так и о сфере ее деятельности. В информационных массивах можно попытаться выявить скрытые, на первый взгляд, закономерности и вывести из них правила, которым подчиняется предметная область информационной системы. Впоследствии эти правила можно использовать для стратегического планирования, принятия решений и прогнозирования их последствий.

Осознание пользы накапливаемой информации и возможности использовать ее для решения аналитических задач привело к появлению нового класса вычислительных систем — систем поддержки принятия решений (СППР), ориентированных на аналитическую обработку данных. Под системой поддержки принятия решений понимают человеко-машинный вычислительный комплекс, ориентированный на анализ данных и обеспечивающий получение информации, необходимой для разработки решений в сфере управления. Следует заметить, что аналитические системы существовали и ранее, но именно возможность обработки больших объемов накапливаемых данных дала новый толчок их развитию и приходу на рынок. Также этому способствовали снижение стоимости высокопроизводительных компьютеров и расходов на хранение больших объемов данных, развитие математических методов обработки информации. К числу задач, которые традиционно решают системы поддержки принятия решений, относятся: оценка альтернатив решений, прогнозирование, классификация, кластеризация, выявление ассоциаций и др. Подробнее эти задачи и методы их решения рассмотрены в соответствующих главах настоящей книги.

Для получения интересующей их информации лица, принимающие решения (ЛПР), или аналитики обращаются к СППР с запросами. Эти запросы в большинстве случаев более сложные, чем те, которые применяются в системах операционной обработки данных. Например, в OLTP-системе банка запрос может сводиться к получению сведений о сумме на счету конкретного клиента. В аналитической системе запрос может быть таким: «Найти среднее значение промежутка времени между выставлением счета и оплатой его клиентом в текущем и прошедшем году отдельно для разных групп клиентов».

В большинстве случаев сложный аналитический запрос невозможно сформулировать в терминах языка SQL, поэтому для получения информации приходится применять специализированные языки, ориентированные на аналитическую обработку данных. К их числу можно отнести, например, язык Express 4GL фирмы Oracle. Также для выполнения аналитических запросов могут быть использованы приложения, написанные специально для решения тех или иных аналитических задач.

Для того чтобы можно было извлекать полезную информацию из данных, они должны быть организованы особым, отличным от принятого в OLTP-системах образом. Связано это со следующими факторами. Во-первых, для выполнения аналитических запросов необходима обработка больших информационных массивов.

Чем выше степень нормализации базы данных и чем больше в ней таблиц, тем медленнее выполняется анализ. Происходит это прежде всего потому, что увеличивается число операций соединения отношений. В системах обработки транзакций нормализация таблиц БД позволяет устранить избыточность данных, уменьшив тем самым объем действий, необходимых при обновлении информации. Поэтому в нормализованных БД нет необходимости менять одни и те же значения в различных отношениях. В аналитических системах данные практически не обновляются — в системе производится лишь их накопление и чтение. Поэтому проблема нормализации БД в них не столь актуальна, как в системах обработки транзакций. Во-вторых, выполнение некоторых аналитических запросов, например анализ тенденций и прогнозирование, требует хронологической упорядоченности данных. Реляционная модель не предполагает существования порядка записей в таблице. В-третьих, данные, используемые для целей анализа, как правило, отличаются от данных систем обработки транзакций. При обслуживании аналитических запросов чаще используются не детальные, а обобщенные (агрегированные) данные. Так, например, для прогнозирования объема продаж сети универмагов будет излишним иметь информацию о каждой сделанной покупке, достаточно знать значение прогнозируемой величины за несколько предыдущих лет.

Принципы, лежащие в основе систем поддержки принятия решений, не позволяют эффективно обрабатывать транзакции, поэтому данные, применяемые для анализа, стали выделять в отдельные базы данных. Впоследствии эти базы данных стали называть хранилищами данных (ХД) или информационными хранилищами. В литературе используется также англоязычный термин «Data Warehouse».

Отцом концепции использования хранилищ данных в аналитических системах считают Билла Инмона (Bill Inmon), технического директора компании «Призм Сольюшнс» (Prism Solutions). В начале 90-х годов он опубликовал ряд работ, которые стали отправной точкой для последующих исследований в области аналитических систем. Большое влияние на разработку концепции хранилищ данных оказала также американская корпорация «Ай Би Эм» (IBM).

Концепция хранилищ данных — это концепция подготовки данных для последующего анализа. Она предполагает выполнение следующих положений:

1) интеграции и согласования данных из различных источников: традиционных систем операционной обработки данных, информации из внутренних и внешних по отношению к организации электронных архивов;
2) разделения наборов данных, используемых системами обработки транзакций и системами поддержки принятия решений.

В работе «Создание хранилища данных» («Building the Data Warehouse») Билл Инмон определил ***хранилище данных*** *как «предметно-ориентированный, интегрированный, неизменяемый и поддерживающий хронологию набор данных, предназначенный для обеспечения принятия управленческих решений».*

Производство и реализация товаров имеют много общего с анализом данных: на предприятии из сырья получается готовая продукция, которая затем доставляет-

ся потребителю; в процессе анализа из накопленных данных добывается и предоставляется полезная специалистам информация, используемая для разработки решений. Любая продукция, прежде чем быть доставленной потребителю, должна быть изготовлена. Этим занимаются заводы. Произведенная продукция отправляется на склад, откуда поступает в магазины. Именно там она находит своего потребителя.

Подобная схема обработки и снабжения справедлива и для аналитической системы. Исходные данные для анализа производятся системами операционной обработки, поступают из электронных архивов и от поставщиков информации, например онлайновых информационных агентств. Эти источники слабо связаны между собой, поэтому и данные, которые они предоставляют, имеют различную структуру и форматы представления. Необходимо произвести согласование данных разных источников, чтобы ими было удобно оперировать при анализе. Это подразумевает приведение их к единому формату, устранение дублирующихся и некорректных значений.

Подготовленные данные загружаются в хранилище. Пользователи-аналитики осуществляют доступ к нему через клиентские приложения. Эти приложения могут осуществлять трансляцию запросов потребителей информации либо производить аналитическую обработку данных хранилища. В отличие от систем операционной обработки данных в СППР, использующих концепцию ХД, критерии поиска и состав выдаваемой в виде отчета информации не фиксируются при ее разработке, пользователи оперируют в основном заранее не регламентированными запросами (ad-hoc query).

Использование концепции хранилища данных в системе поддержки принятия решений преследует следующие цели:

1) своевременное обеспечение аналитиков всей информацией, необходимой для выработки решений;
2) создание единой модели данных организации;
3) создание интегрированного источника данных, предоставляющего удобный доступ к разнородной информации и гарантирующего получение одинаковых ответов на одинаковые запросы из различных аналитических подсистем (единый «источник истины»).

Вернемся к определению, данному Инмоном, чтобы подробнее рассмотреть свойства, присущие хранилищам данных.

Ориентация на предметную область. Хранилище должно разрабатываться с учетом специфики предметной области, а не приложений, оперирующих данными. Структура хранилища должна отражать представления аналитика об информации, с которой ему приходится работать. Например, если система операционной обработки поставщика товаров работает с понятиями «сделка» и «заявка», то хранилище должно использовать понятия «клиенты», «товары» и «производители».

Интегрированность. Информация загружается в хранилище из приложений, созданных разными разработчиками. Необходимо объединить данные этих приложений, приведя их к единому синтаксическому и семантическому виду. Например, в таблицах БД, полученных из разных источников, могут встречаться атрибуты, которые определены на разных доменах, но обозначают те же понятия. На-

пример, месяц года может быть задан полным наименованием (январь, февраль и т. д.), сокращенным наименованием (янв., фев. и т. д.) и номером (1, 2 и т. д.). В процессе загрузки хранилища требуется преобразовать эти атрибуты к единому представлению. Важно также провести проверку поступающих данных на целостность и непротиворечивость. Характерный для информационных хранилищ прием — хранение агрегированных данных. Аналитика редко интересует информация о конкретных днях и часах, ему более важны данные о месяцах, кварталах и даже годах. Чтобы при выполнении аналитических запросов избежать выполнения операций группирования, данные должны обобщаться (агрегироваться) при загрузке хранилища. Объем накопленных данных должен быть достаточным для решения аналитических задач с требуемым качеством. Используемые в настоящее время ХД содержат информацию, накопленную за годы и даже десятилетия.

Неизменяемость данных. Важное отличие аналитических систем от систем операционной обработки данных состоит в том, что данные после загрузки в них остаются неизменными, внесения каких-либо изменений, кроме добавления записей, не предполагается. Именно поэтому для СППР не столь актуальны средства для обеспечения отката транзакций, борьбы с взаимными блокировками процессов — разработчики подобных систем сосредоточивают основные усилия на достижении высокой скорости доступа к данным. Важное условие неизменности информации в хранилище — использование для его реализации надежного оборудования, которое обеспечивает защиту от сбоев.

Поддержка хронологии. Для выполнения большинства аналитических запросов необходим анализ тенденций развития явлений или характера изменения значений переменных во времени. Учет хронологии достигается введением ключевых атрибутов типа «ДАТА» и/или «ВРЕМЯ» в структуры хранилища данных. Время выполнения аналитических запросов можно уменьшить, если физически упорядочить записи по времени, то есть расположить записи по возрастанию значений атрибута «ДАТА/ВРЕМЯ».

В последнее время сформировался новый класс систем поддержки принятия решений — системы оперативной аналитической обработки (OnLine Analysis Processing — OLAP). Под OLAP-системой принято понимать СППР, как правило, основанную на концепции хранилища данных и обеспечивающую малое время выполнения аналитических запросов.

К числу основных задач, которые требуется решать при создании ХД, относятся:

1) выбор оптимальной структуры хранения данных с точки зрения обеспечения приемлемого времени отклика на аналитические запросы и требуемого объема памяти;
2) первоначальное заполнение и последующее пополнение хранилища данными;
3) обеспечение удобства доступа пользователей к данным. Рассмотрим пути решения этих задач более детально.

Модели данных, используемые для построения хранилищ.

Задачи, решаемые OLTP и аналитическими системами, существенно различаются, поэтому их БД тоже построены на разных принципах. Критерием эффективности для систем операционной обработки данных служит число транзакций, которое они способны выполнить в единицу времени. Для аналитических систем

важнее скорость выполнения сложных запросов и прозрачность структуры хранения информации для пользователей. Важная особенность СППР на основе ХД состоит в том, что загрузка данных выполняется сравнительно редко, но большими порциями (до нескольких миллионов записей за один раз), поэтому в таких системах обычно не предусматриваются развитые средства обеспечения целостности, восстановления, устранения взаимных блокировок. Это не только существенно облегчает и упрощает сами средства реализации, но и значительно снижает внутренние накладные расходы при доступе к информации и, следовательно, повышает производительность анализа.

В настоящее время существуют два в чем-то конкурирующих, а в чем-то взаимодополняющих друг друга подхода к построению хранилищ данных: подход, основанный на использовании многомерной модели БД (Multidimedsional OLAP — MOLAP), и подход, использующий реляционную модель БД (Relational OLAP — ROLAP). Прежде чем рассказать о каждом из них, попытаемся разобраться, какие данные могут находиться в хранилище и как они могут быть представлены. Чаще всего там содержатся сведения о значении некоторых параметров, характеризующих предметную область в определенные моменты или за определенные промежутки времени. Пусть, например, требуется создать хранилище, накапливающее информацию об изменении социально-экономической обстановки в России. Эта обстановка характеризуется многими параметрами, в числе которых: объем промышленного производства, индекс потребительских цен и др. Госкомстат России собирает их значения для различных субъектов Российской Федерации помесячно, поквартально или за год.

В хранилище должны попадать факты вида:

Название параметра в субъекте Российской Федерации в момент времени был равен {*значение*}.

Пример 4.3

Например, индекс потребительских цен в городе Москве в декабре 1996 года был равен 101%. В рассматриваемом примере каждое значение связано с точкой в трехмерном пространстве (N, S, T) с измерениями: N — название параметра; S — субъект федерации; T — момент времени. Число возможных параметров, субъектов РФ, а также рассматриваемых моментов времени конечно, поэтому все возможные значения можно представить в виде гиперкуба (см. рис. 4.2).

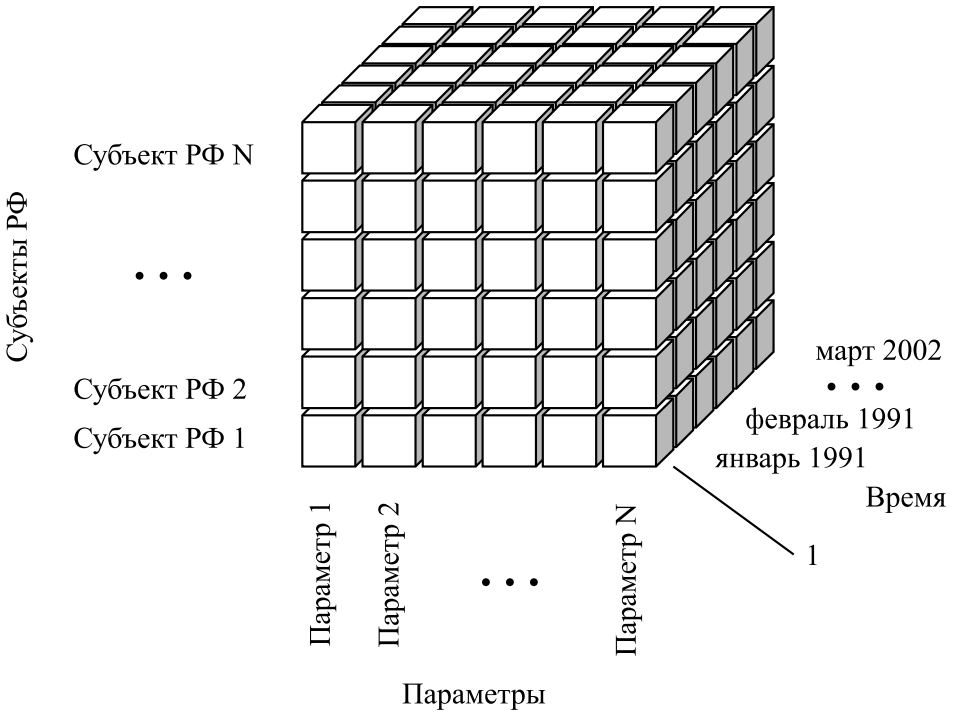


Рис. 4.2 – Представление данных в виде гиперкуба

В этом гиперкубе каждое значение находится в строго определенной ячейке, что значительно упрощает обращение к ней.

Представленный пример, конечно, упрощен, но он позволяет понять, что такое многомерный взгляд на данные. В реальной задаче число измерений может быть больше трех. Представление данных в виде гиперкуба более наглядно, чем совокупность нормализованных таблиц, оно понятно не только администратору БД, но и рядовым сотрудникам. Это дает им дополнительные возможности построения аналитических запросов к системе, использующей хранилище данных. Кроме того, использование многомерной модели данных позволяет резко уменьшить время поиска в ХД, обеспечивая выполнение аналитических запросов в реальном времени.

Гиперкуб может быть реализован в рамках реляционной модели или существовать как отдельная БД специальной многомерной структуры. В зависимости от этого и принято различать реляционный (ROLAP) и многомерный (MOLAP) подходы к построению ХД.

4.3.2 Многомерная модель

Многомерная модель БД появилась довольно давно, однако в силу присущих ей ограничений применение получила лишь в последнее время. При использовании этой модели данные хранятся не в виде плоских таблиц, как в реляционных БД, а в виде гиперкубов — упорядоченных многомерных массивов. То есть многомерное представление данных здесь реализуется физически. Конечно, такой подход требует большего объема памяти для хранения данных, при его использовании сложно модифицировать структуру данных. Например, добавление еще одно-

го измерения приводит к необходимости полной перестройки гиперкуба. Однако многомерные СУБД обеспечивают более быстрый по сравнению с реляционными системами поиск и чтение данных, избавляют от необходимости многократно соединять таблицы. Среднее время ответа на сложный аналитический запрос при использовании многомерных СУБД обычно в 10–100 раз меньше, чем в случае реляционной СУБД с нормализованной структурой.

Основные понятия многомерной модели — измерение и значение (ячейка). Измерение — это множество, образующее одну из граней гиперкуба (аналог домена в реляционной модели). Измерения играют роль индексов, используемых для идентификации конкретных значений в ячейках гиперкуба. Значения — это подвергаемые анализу количественные или качественные данные, которые находятся в ячейках гиперкуба (см. рис. 4.2).

В многомерной модели вводятся следующие основные операции манипулирования измерениями: 1) сечение; 2) вращение; 3) детализация; 4) свертка.

При выполнении операции сечения формируется подмножество гиперкуба, в котором значение одного или более измерений фиксировано. Например, если на рис. 4.2 зафиксировать значение измерения «Время» равным «январь 1991 года», то мы получим двухмерную таблицу с информацией о значениях всех параметров для всех субъектов РФ в январе 1991 года.

Операция вращения изменяет порядок представления измерений. Она обычно применяется к двухмерным таблицам, обеспечивая представление их в более удобной для восприятия форме. Если в исходной таблице по горизонтали были расположены субъекты РФ, а по вертикали параметры социально-экономической сферы, то после операции вращения параметры будут размещены по горизонтали, а названия субъектов РФ — по вертикали.

Для выполнения операций свертки и детализации должна существовать иерархия значений измерения, то есть некоторая подчиненность одних значений другим. Например, 12 месяцев образуют год, субъекты РФ образуют регионы. При выполнении операции свертки одно из значений измерения заменяется значением более высокого уровня иерархии. Например, аналитик, узнав значения параметров для января 1991 года, желает получить их значения за весь 1991 год. Чтобы это сделать, необходимо выполнить операцию свертки. Операция детализации — это операция, обратная свертке. Она обеспечивает переход от обобщенных к детализированным данным.

Основное назначение СУБД, поддерживающих многомерную модель, — реализация систем, ориентированных на аналитическую обработку. Многомерные СУБД лучше других справляются с задачами выполнения сложных нерегламентированных запросов.

Однако у многомерных БД имеются серьезные недостатки, сдерживающие их применение. Многомерные СУБД неэффективно по сравнению с реляционными используют память. В многомерной СУБД заранее резервируется место для всех значений, даже если часть из них заведомо будет отсутствовать. Другой недостаток состоит в том, что выбор высокого уровня детализации при реализации гиперкуба может очень сильно увеличить размер многомерной БД. В силу этих, а также некоторых других причин доступные на рынке многомерные СУБД не в состоянии оперировать данными большого объема. Объем, доступный им для хранения, ограничен 10–20 гигабайтами.

Целесообразно использовать многомерную модель, если объем БД невелик и гиперкуб использует стабильный во времени набор измерений.

4.3.3 Реляционная модель хранилища данных

Основой при построении хранилища данных может служить и традиционная реляционная модель данных. В этом случае гиперкуб эмулируется СУБД на логическом уровне. В отличие от многомерных реляционные СУБД способны хранить огромные объемы данных, однако они проигрывают по скорости выполнения аналитических запросов.

При использовании РСУБД для организации хранилища данные организуются специальным образом. Чаще всего используется так называемая радиальная схема. Другое ее название — «звезда» (star). В этой схеме используются два типа таблиц: таблица фактов (фактологическая таблица) и несколько справочных таблиц (таблицы измерений).

В таблице фактов обычно содержатся данные, наиболее интенсивно используемые для анализа. Если проводить аналогию с многомерной моделью, то запись фактологической таблицы соответствует ячейке гиперкуба. В справочной таблице перечислены возможные значения одного из измерений гиперкуба. Каждое измерение описывается своей собственной справочной таблицей. Фактологическая таблица индексируется по сложному ключу, скомпонованному из индивидуальных ключей справочных таблиц. Это обеспечивает связь справочных таблиц с фактологической по ключевым атрибутам. В качестве примера на рис. 4.3 приведена упрощенная схема структуры хранилища данных, используемого для накопления информации из рассмотренного ранее примера (см. рис. 4.2).

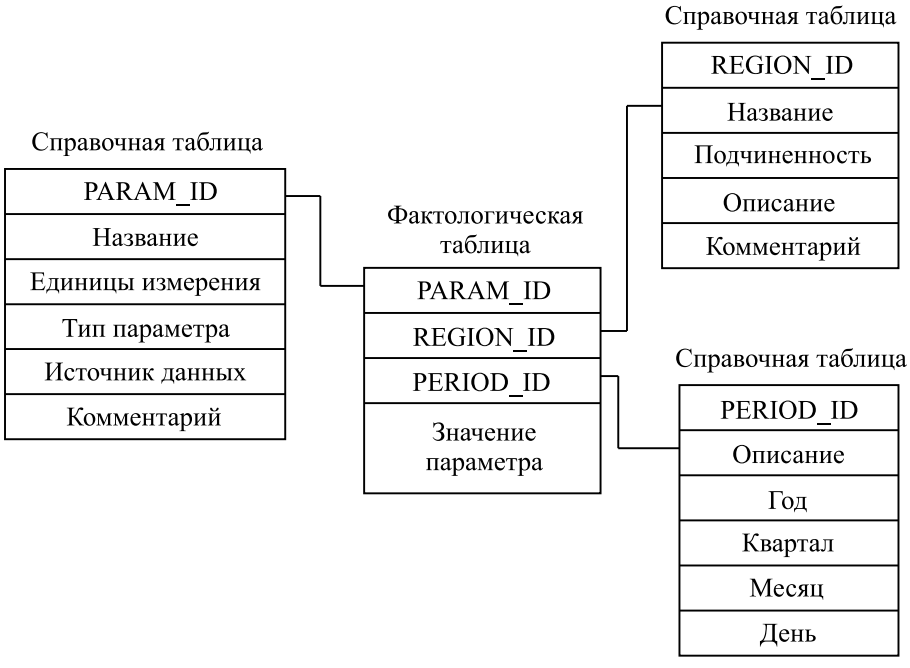


Рис. 4.3 – Пример базы данных со схемой «звезда»

В реальных системах количество строк в фактологической таблице может составлять десятки и сотни миллионов. Число справочных таблиц обычно не превышает двух десятков. Для увеличения производительности анализа в фактологической таблице могут храниться не только детализированные, но и предварительно вычисленные агрегированные данные.

Если БД включает большое число измерений, можно использовать схему «снежинка» (snowflake). В этой схеме атрибуты справочных таблиц могут быть детализированы в дополнительных справочных таблицах (см. рис. 4.4).

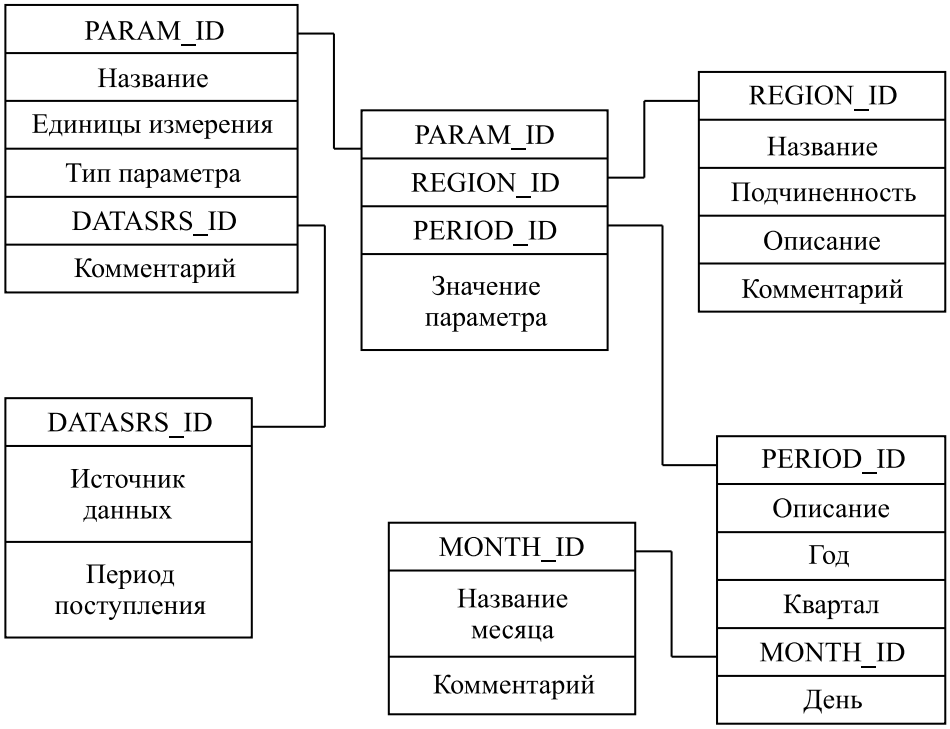


Рис. 4.4 – Пример базы данных со схемой «снежинка»

Для сокращения времени, требуемого для получения отклика от аналитической системы, можно использовать некоторые специальные средства. В состав мощных реляционных СУБД обычно входят оптимизаторы запросов. При создании хранилищ данных на основе РСУБД их наличие приобретает особую важность. Оптимизаторы анализируют запрос и определяют лучшую, с позиции некоторого критерия, последовательность операций обращения к БД для его выполнения. Например, может минимизироваться число физических обращений к дискам при выполнении запроса. Оптимизаторы запросов используют сложные алгоритмы статистической обработки, которые оперируют числом записей в таблицах, диапазонами ключей и т. д.

Каждая из описанных моделей имеет как преимущества, так и недостатки. Многомерная модель позволяет производить быстрый анализ данных, но не позволяет хранить большие объемы информации. Реляционная модель, напротив, практически не имеет ограничений по объему накапливаемых данных, однако СУБД на ее основе не обеспечивают такой скорости выполнения аналитических запросов, как МСУБД. Нельзя ли совместить два этих подхода так, чтобы скрыть их недо-

статки и сделать более заметными их достоинства? Удачные проекты реализации хранилищ данных, появившиеся в последнее время, показывают, что это возможно.

Ситуация, когда для анализа необходима вся информация, находящаяся в хранилище, возникает довольно редко. Обычно каждый аналитик или аналитический отдел обслуживает одно из направлений деятельности организации, поэтому в первую очередь ему необходимы данные, характеризующие именно это направление. Реальный объем этих данных не превосходит ограничений, присущих многомерным СУБД. Возникает идея выделить данные, которые реально нужны конкретным аналитическим приложениям, в отдельный набор. Такой набор мог бы быть реализован в многомерной БД. Источником данных для него должно быть центральное хранилище организации.

Если проводить аналогии с производством и реализацией продукции, то многомерные БД выполняют роль мелких складов. В концепции ХД их принято именовать киосками данных (Data Marts). Киоск данных — это специализированное тематическое хранилище, обслуживающее одно из направлений деятельности организации.

4.3.4 Обнаружение знаний в хранилищах данных

Data mining.

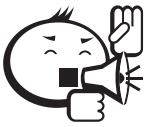
При использовании хранилищ остро встает проблема обнаружения в них знаний (KDD — knowledge discovery in databases). Основным шагом этого процесса является data mining (исследование данных, или, дословно, «**добыча данных**»). После применения традиционных методов увеличения доходов (маркетинговые исследования и действия на рынке, работа с конкурентами) или уменьшения расходов (изменение технологии, работа с поставщиками) перед менеджерами высшего звена встает задача по дальнейшему увеличению прибыли как основной цели деятельности любого коммерческого предприятия.

Для этого в последнее время был разработан ряд технологий, которые призваны извлекать из хранилищ данных большого объема новую информацию путем построения различных моделей. Они и получили название «data mining». Простой доступ пользователя к хранилищу данных обеспечивает только получение ответов на задаваемые вопросы, в то время как технология data mining позволяет увидеть («добыть») такие интересные взаимоотношения между данными, которые прежде даже не приходили пользователю в голову и применение которых может способствовать увеличению прибыли предприятия.

Как известно, большинство организаций накапливают за время своей деятельности огромные объемы данных, но единственное, что они хотят от них получить, — это информация. Как можно узнать из данных о том, что нужно наиболее предпочтительным для организации клиентам, как разместить ресурсы наиболее эффективным образом или как минимизировать потери? Новейшая технология, адресованная к решению этих проблем, — это технология data mining. Она использует сложный статистический анализ и моделирование для нахождения моделей и отношений, скрытых в базе данных, — таких моделей, которые не могут быть найдены обычными методами.

Модель, как и карта, — это абстрактное представление реальности. Карта может указывать на путь от аэропорта до дома, но она не может показать аварию,

которая создала пробку, или ремонтные работы, которые ведутся в настоящий момент и требуют объезда. До тех пор пока модель не соответствует существующим реально отношениям, невозможно получить успешные результаты.



Существуют два вида моделей: предсказательные и описательные. Первые используют один набор данных с известными результатами для построения моделей, которые явно предсказывают результаты для других наборов, а вторые описывают зависимости в существующих данных, которые в свою очередь используются для принятия управленческих решений или действий.

Конечно же, компания, которая долго находится на рынке и знает своих клиентов, уже осведомлена о многих моделях, которые наблюдались в течение нескольких последних периодов. Но технологии data mining могут не только подтвердить эти эмпирические наблюдения, но и найти новые, неизвестные ранее модели. Сначала это может дать пользователю лишь небольшое преимущество. Но такое преимущество, если его соединить за несколько по каждому товару и каждому клиенту, дает существенный отрыв от тех, кто не пользуется технологиями data mining. С другой стороны, с помощью методов data mining можно найти такую модель, которая приведет к радикальному улучшению в финансовом и рыночном положении компании.

В чем же разница между средствами data mining и средствами OLAP — средствами оперативной аналитической обработки?

OLAP — это часть технологий, направленных на поддержку принятия решения. Обычные средства формирования запросов и отчетов описывают саму базу данных. Технология OLAP используется для ответа на вопрос, почему некоторые вещи являются такими, какими они есть на самом деле. При этом пользователь сам формирует гипотезу о данных или отношениях между данными и после этого использует серию запросов к базе данных для подтверждения или отклонения этих гипотез. Средства data mining отличаются от средств OLAP тем, что вместо проверки предполагаемых взаимозависимостей, они на основе имеющихся данных могут производить модели, позволяющие количественно оценить степень влияния исследуемых факторов. Кроме того, средства data mining позволяют производить новые гипотезы о характере неизвестных, но реально существующих отношений в данных.

Средства OLAP обычно применяются на ранних стадиях процесса KDD потому, что они помогают нам понять данные, фокусируя внимание аналитика на важных переменных, определяя исключения или интересные значения переменных. Это приводит к лучшему пониманию данных, что, в свою очередь, ведет к более эффективному результату процесса KDD.

Наличие хранилища данных является необходимым условием для успешного проведения всего процесса KDD. Вспомним, что хранилище данных — это предметно-ориентированное, интегрированное, привязанное ко времени, неизменяемое собрание данных для поддержки процесса принятия управленческих решений. Предметная ориентация означает, что данные объединены в категории и хранятся

в соответствии с теми областями, которые они описывают, а не с приложениями, которые их используют. Интегрированность означает, что данные удовлетворяют требованиям всего предприятия (в его развитии), а не единственной функции бизнеса. Тем самым хранилище данных гарантирует, что одинаковые отчеты, сгенерированные для различных аналитиков, будут содержать одинаковые результаты. Привязанность ко времени означает, что хранилище можно рассматривать как совокупность «исторических» данных: можно восстановить картину на любой момент времени. Атрибут времени всегда явно присутствует в структурах хранилища данных. Неизменяемость означает, что, попав однажды в хранилище, данные уже не изменяются в отличие от оперативных систем, где данные обязаны присутствовать только в последней версии, поэтому постоянно меняются. В хранилище данные только добавляются.

Для решения перечисленного ряда задач, неизбежно возникающих при организации и эксплуатации информационного хранилища, существует специализированное программное обеспечение. Современные средства администрирования хранилища данных обеспечивают эффективное взаимодействие с инструментарием data mining. В качестве примера можно привести два продукта компании SAS Institute: SAS Warehouse Administrator и SAS Enterprise Miner, степень взаимной интеграции которых позволяет использовать при реализации проекта data mining также и метаданные из информационного хранилища.

Виды моделей.

Рассмотрим основные виды моделей, которые используются для нахождения нового знания на основе данных хранилища. Целью технологи data mining является производство нового знания, которое пользователь может в дальнейшем применить для улучшения результатов своей деятельности. Результат моделирования — это выявленные отношения в данных. Можно выделить по крайней мере шесть методов выявления и анализа знаний: классификация, регрессия, прогнозирование временных последовательностей (рядов), кластеризация, ассоциация, последовательность. Первые три используются главным образом для предсказания, в то время как последние удобны для описания существующих закономерностей в данных.

Вероятно, наиболее распространенной сегодня операцией интеллектуального анализа данных является **классификация**. С ее помощью выявляются признаки, характеризующие группу, к которой принадлежит тот или иной объект. Это делается посредством анализа уже классифицированных объектов и формулирования некоторого набора правил. Во многих видах бизнеса болезненной проблемой считается потеря постоянных клиентов. В разных сферах (таких, как сотовая телефонная связь, фармацевтический бизнес или деятельность, связанная с кредитными карточками) ее обозначают различными терминами — «переменой моды», «истощением спроса» или «покупательской изменой», — но суть при этом одна. Классификация поможет вам выявить характеристики «неустойчивых» покупателей и создать модель, способную предсказать, кто именно склонен уйти к другому поставщику. Используя ее, можно определить самые эффективные виды скидок и других выгодных предложений, которые будут наиболее действенны для тех или иных типов покупателей. Благодаря этому вам удастся удержать клиентов, потратив ровно столько денег, сколько необходимо. Однажды определенный эффективный клас-

сификатор используется для классификации новых записей в базе данных в уже существующие классы, и в этом случае он приобретает характер прогноза. Например, классификатор, который умеет идентифицировать риск выдачи займа, может быть использован для целей принятия решения, велик ли риск предоставления займа определенному клиенту. То есть классификатор используется для прогнозирования возможности возврата займа.

Регрессионный анализ используется в том случае, если отношения между переменными могут быть выражены количественно в виде некоторой комбинации этих переменных. Полученная комбинация далее используется для предсказания значения, которое может принимать целевая (зависимая) переменная, вычисляемая на заданном наборе значений входных (независимых) переменных. В простейшем случае для этого используются стандартные статистические методы, такие как линейная регрессия. К сожалению, большинство реальных моделей не укладываются в рамки линейной регрессии. Например, размеры продаж или фондовые цены очень сложны для предсказания, потому что могут зависеть от комплекса взаимоотношений множества переменных. Таким образом, необходимы комплексные методы для предсказания будущих значений.

Прогнозирование временных последовательностей позволяет на основе анализа поведения временных рядов оценить будущие значения прогнозируемых переменных. Конечно, эти модели должны включать в себя особые свойства времени: иерархия периодов (декада-месяц-год или месяц-квартал-год), особые отрезки времени (пяти- шести- или семидневная рабочая неделя, тринадцатый месяц), сезонность, праздники и др.

Кластеризация относится к проблеме сегментации. Этот подход распределяет записи в различные группы или сегменты. Кластеризация в чем-то аналогична классификации, но отличается от нее тем, что для проведения анализа не требуется иметь выделенную целевую переменную.

Ассоциация адресована, главным образом, к классу проблем, типичным примером которых является анализ структуры покупок. Классический анализ структуры покупок относится к представлению приобретения какого-либо количества товаров как одиночной экономической операции (транзакции). Так как большое количество покупок совершается в супермаркетах, а покупатели для удобства используют корзины или тележки, куда и складывается весь товар, то наиболее известным примером нахождения ассоциаций является анализ структуры покупки (market-basket analysis). Целью этого подхода является нахождение трендов среди большого числа транзакций, которые можно использовать для объяснения поведения покупателей. Эта информация может быть использована для регулирования запасов, изменения размещения товаров на территории магазина и принятия решения по проведению рекламной кампании для увеличения всех продаж или для продвижения определенного вида продукции, хотя этот подход пришел исключительно из розничной торговли, он может также хорошо применяться в финансовой сфере для анализа портфеля ценных бумаг и нахождения наборов финансовых услуг, которые клиенты часто приобретают вместе. Это, например, может использоваться для создания некоторого набора услуг, как части кампании по стимулированию продаж. Другими словами, ассоциация имеет место в том случае, если несколько событий связаны друг с другом. Например, исследование, проведенное

в супермаркете, может показать, что 65% купивших картофельные чипсы берут также и «кока-колу», а при наличии скидки за такой комплект «колу» приобретают в 85% случаев. Располагая этими сведениями, менеджерам легко оценить, насколько действенна предоставляемая скидка.

Последовательность. Традиционный анализ структуры покупок имеет дело с набором товаров, представляющим одну транзакцию. Вариант такого анализа встречается, когда существует дополнительная информация (номер кредитной карты клиента или номер его банковского счета) для связи различных покупок в единую временную серию. В такой ситуации важно не только сосуществование данных внутри одной транзакции, но и порядок, в котором эти данные появляются в различных транзакциях, и время между этими транзакциями. Правила, которые устанавливают эти отношения, могут быть использованы для определения типичного набора предшествующих продаж, которые могут повести за собой последующие продажи определенного товара. То есть если существует цепочка связанных во времени событий, то говорят о последовательности. После покупки дома в 45% случаев в течение месяца приобретается и новая кухонная плита, а в пределах двух недель 60% новоселов обзаводятся холодильником.

Эти основные типы моделей используются для нахождения нового знания в хранилище данных. Обратимся теперь к методам, которые используются для проведения интеллектуального анализа данных.

Методы анализа данных.

Интеллектуальные средства анализа данных используют следующие основные методы: нейронные сети; деревья решений; индукцию правил.

Кроме этих методов, существуют еще несколько дополнительных: системы рассуждения на основе аналогичных случаев; нечеткая логика; генетические алгоритмы; алгоритмы определения ассоциаций и последовательностей; эволюционное программирование; визуализация данных. Иногда применяется комбинация перечисленных методов.

Нейронные сети относятся к классу нелинейных адаптивных систем с архитектурой, условно имитирующей нервную ткань из нейронов. Математическая модель нейрона представляет собой некоторый универсальный нелинейный элемент с возможностью широкого изменения и настройки его характеристик. В одной из наиболее распространенных нейросетевых архитектур — многослойном персептроне с обратным распространением ошибки — эмулируется работа нейронов в составе иерархической сети, где каждый нейрон более высокого уровня соединен своими входами с выходами нейронов нижележащего слоя. На нейроны самого нижнего слоя подаются значения входных параметров, на основе которых производятся вычисления, необходимые для принятия решений, прогнозирования развития ситуации и т. д. Эти значения рассматриваются как сигналы, передающиеся в вышележащий слой, ослабляясь или усиливаясь в зависимости от числовых значений (весов), приписываемых межнейронным связям. В результате этого на выходе нейрона самого верхнего слоя вырабатывается некоторое значение, которое рассматривается как ответ, реакция всей сети на введенные значения входных параметров. Для того чтобы сеть можно было применять в дальнейшем, ее прежде надо «натренировать» на полученных ранее данных (примерах), для которых известны и значения входных параметров, и правильные ответы на них. Процесс «тренировки»

состоит в подборе весов межнейронных связей и модификации внутренних параметров активационной функции нейронов. Для каждого сочетания обучающих данных на входе выходные значения сравниваются с известным результатом. Если они различаются, то вычисляется корректирующее воздействие, учитываемое при обработке в узлах сети. Указанные шаги повторяются, пока не выполнится условие останова, например необходимая коррекция не будет превышать заданной величины.

Нейронные сети, по существу, представляют собой совокупность связанных друг с другом узлов, получающих входные данные, осуществляющих их обработку и генерирующих на выходе некий результат. Между узлами видимых входного и выходного уровней может находиться какое-то число скрытых уровней обработки. Нейронные сети реализуют непрозрачный процесс. Это означает, что построенная модель, как правило, не имеет четкой интерпретации. Некоторые алгоритмы могут транслировать модель нейронной сети в набор правил, помогающих уяснить, что именно она делает. Такую возможность предлагают некоторые оригинальные продукты, использующие технологию нейронной сети. Многие пакеты, реализующие принципы нейронных сетей, применяются не только в сфере обработки коммерческой информации. Нередко без них трудно обойтись при решении более общих задач распознавания образов, скажем, расшифровки рукописного текста или интерпретации кардиограмм.

Деревья решений — это метод, который пригоден не только для решения задач классификации, но и для вычислений, и поэтому довольно широко применяется в области финансов и бизнеса, где чаще встречаются задачи численного прогноза. В результате применения этого метода к обучающей выборке данных создается иерархическая структура классифицирующих правил типа «ЕСЛИ... ТО...», имеющая вид дерева (это похоже на определитель видов из ботаники или зоологии). Для того чтобы решить, к какому классу отнести некоторый объект или ситуацию, мы отвечаем на вопросы, стоящие в узлах этого дерева, начиная с его корня. Вопросы могут иметь вид «значение параметра *A* больше *x*?» для случая измеряемых переменных или вида «значение переменной *B* принадлежит подмножеству признаков *C*». Если ответ положительный, мы переходим к правому узлу следующего уровня, если отрицательный — то к левому узлу; затем снова отвечаем на вопрос, связанный с соответствующим узлом. Таким образом мы, в конце концов, доходим до одного из оконечных узлов — листьев, где стоит указание, к какому классу (сочетанию признаков) надо отнести рассматриваемый объект. Этот метод хорош тем, что такое представление правил наглядно и его легко понять.

Сегодня наблюдается всплеск интереса к продуктам, применяющим деревья решений. В основном это объясняется тем, что многие коммерческие проблемы решаются ими быстрее, чем алгоритмами нейронных сетей. К тому же они более просты и понятны для пользователей. В то же время нельзя сказать, что деревья решений всегда действуют безотказно: для определенных типов данных они могут оказаться неприемлемыми. В частности, методы дерева решений не очень эффективны, если целевая переменная зависит линейным образом от входных переменных, так как в этом случае дерево должно иметь большое число листьев. Иногда возникают проблемы при обработке непрерывных величин, скажем, данных о возрасте или объеме продаж. В этом случае их необходимо группировать

и ранжировать. Однако выбранный для ранжирования метод способен случайно скрыть выявляемую закономерность. Например, если группа объединяет людей в возрасте от 25 до 34 лет, то тот факт, что на рубеже 30 лет некий параметр испытывает существенный разрыв, может оказаться скрытым. Этого недостатка не имеет продукт SAS Enterprise Miner в силу того, что реализованные в нем методы построения дерева решений могут автоматически выявлять границу (численный критерий) разделения данных на более однородные подгруппы.

Для деревьев решений очень остро стоит проблема значимости. Дело в том, что отдельным узлам на каждом новом построенном уровне дерева соответствует все меньшее и меньшее число записей данных — дерево может сегментировать данные на большое количество частных случаев. Чем больше этих частных случаев, чем меньше обучающих примеров попадает в каждый такой частный случай, тем менее надежной становится их классификация. Если построенное дерево слишком «кустистое» — состоит из неоправданно большого числа мелких веточек — оно не будет давать статистически обоснованных ответов. Как показывает практика, в большинстве систем, использующих деревья решений, эта проблема не находит удовлетворительного решения. Исключением из этого ряда является упомянутый выше SAS Enterprise Miner, включающий в себя широкий спектр диагностических инструментов, с помощью которых аналитик может выбрать статистически наиболее обоснованную модель из производимого множества деревьев решений и более того — сравнить полученную модель дерева с принципиально другими типами моделей (регрессионной и нейросетевой). В данном продукте в качестве целевой переменной можно использовать как измеряемые, так и дискретные (не измеряемые переменные или признаки), что существенно расширяет область применения рассмотренных выше методов.

Индукция правил создает неиерархическое множество условий, которые могут перекрываться. Индукция правил осуществляется путем генерации неполных деревьев решений, а для того чтобы выбрать, какое из них будет применено к входным данным, используются статистические методы.

Идея **систем рассуждения на основе аналогичных случаев** крайне проста. Для того чтобы сделать прогноз на будущее или выбрать правильное решение, эти системы находят в прошлом близкие аналоги наличной ситуации и выбирают тот же ответ, который был для них правильным. Поэтому этот метод еще называют методом «ближайшего соседа» (nearest neighbour). Системы рассуждения на основе аналогичных случаев показывают очень хорошие результаты в самых разнообразных задачах. Главный их минус заключается в том, что они вообще не создают каких-либо моделей или правил, обобщающих предыдущий опыт, — в выборе решения они основываются на всем массиве доступных исторических данных, поэтому невозможно сказать, на основе каких конкретно факторов эти системы строят свои ответы.

Нечеткая логика применяется для таких наборов данных, где причисление данных к какой-либо группе является вероятностью, находящейся в интервале от 0 до 1, но не принимающей крайние значения. Четкая логика манипулирует результатами, которые могут быть либо истиной, либо ложью. Нечеткая логика применяется в тех случаях, когда необходимо манипулировать степенью «может быть» в дополнении к «да» и «нет».

Строго говоря, интеллектуальный анализ данных — далеко не основная область применения **генетических алгоритмов**, которые, скорее, нужно рассматривать как мощное средство решения разнообразных комбинаторных задач и задач оптимизации. Тем не менее генетические алгоритмы вошли сейчас в стандартный инструментарий методов data mining. Этот метод назван так потому, что в какой-то степени имитирует процесс естественного отбора в природе. Пусть нам надо найти решение задачи, наиболее оптимальное с точки зрения некоторого критерия. Пусть каждое решение полностью описывается некоторым набором чисел или величин нечисловой природы. Скажем, если нам надо выбрать совокупность фиксированного числа параметров рынка, наиболее выраженно влияющих на его динамику, это будет набор имен этих параметров. Об этом наборе можно говорить как о совокупности хромосом, определяющих качества индивида — данного решения поставленной задачи. Значения параметров, определяющих решение, будут тогда называться генами. Поиск оптимального решения при этом похож на эволюцию популяции индивидов, представленных их наборами хромосом. В этой эволюции действуют три механизма: во-первых, отбор сильнейших — наборов хромосом, которым соответствуют наиболее оптимальные решения; во-вторых, скрещивание — производство новых индивидов при помощи смешивания хромосомных наборов отобранных индивидов и, в-третьих, мутации — случайные изменения генов у некоторых индивидов популяции. В результате смены поколений вырабатывается такое решение поставленной задачи, которое уже не может быть далее улучшено.

Генетические алгоритмы имеют два слабых места. Во-первых, сама постановка задачи в их терминах не дает возможности проанализировать статистическую значимость получаемого с их помощью решения, и, во-вторых, эффективно сформулировать задачу, определить критерий отбора хромосом под силу только специалисту. В силу этих факторов сегодня генетические алгоритмы надо рассматривать скорее как инструмент научного исследования, чем как средство анализа данных для практического применения в бизнесе и финансах.

Алгоритмы выявления ассоциаций находят правила об отдельных предметах, которые появляются вместе в одной экономической операции, например в одной покупке. Последовательность — это тоже ассоциация, но зависящая от времени.

Ассоциация записывается как А(Б, где А называется левой частью или предпосылкой, Б — правой частью или следствием.

Частота появления каждого отдельного предмета или группы предметов определяется очень просто — считается количество появления этого предмета во всех событиях (покупках) и делится на общее количество событий. Эта величина измеряется в процентах и носит название «распространенность». Низкий уровень распространенности (менее одной тысячной процента) говорит о том, что такая ассоциация несущественна.

Для определения важности каждого полученного ассоциативного правила необходимо получить величину, которая носит название «доверительность А к Б» (или взаимосвязь А и Б). Эта величина показывает, как часто при появлении А появляется Б, и рассчитывается как отношение частоты появления (распространенности) А и Б вместе к распространенности А. То есть если доверительность А к Б равна 20%, то это значит, что при покупке товара А в каждом пятом случае

приобретается и товар Б. Необходимо заметить, что если распространенность А не равна распространенности Б, то и доверительность А к Б не равна доверительности Б к А. В самом деле, покупка компьютера чаще ведет к покупке дискет, чем покупка дискеты к покупке компьютера.

Ещё одной важной характеристикой ассоциации является мощность ассоциации. Чем больше мощность, тем сильнее влияние, которое появление А оказывает на появление Б. Мощность рассчитывается по формуле:

$$\frac{\text{доверительность А к Б}}{\text{распространенность Б}}.$$

Некоторые алгоритмы поиска ассоциаций сначала сортируют данные и только после этого определяют взаимосвязь и распространенность. Единственным различием таких алгоритмов является скорость или эффективность нахождения ассоциаций. Это особенно важно из-за огромного количества комбинаций, которые необходимо перебрать для нахождения наиболее значимых правил. Алгоритмы поиска ассоциаций могут создавать свои базы данных распространенности, доверительности и мощности, к которым можно обращаться по запросу. Например: «Найти все ассоциации, в которых для товара X доверительность более 50% и распространенность не менее 2,5%».

При нахождении последовательностей добавляется переменная времени, которая позволяет работать с серией событий для нахождения последовательных ассоциаций на протяжении некоторого периода времени.

Подводя итоги этому методу анализа, необходимо сказать, что случайно может возникнуть такая ситуация, когда товары в супермаркете будут сгруппированы при помощи найденных моделей, но это, вместо ожидаемой прибыли, даст обратный эффект. Это может получиться из-за того, что клиент не будет долго ходить по магазину в поисках желаемого товара, приобретая при этом ещё что-то, что попадается на глаза, и то, что он изначально не планировал приобрести.

Эволюционное программирование — сегодня самая молодая и наиболее перспективная ветвь data mining. Суть метода в том, что гипотезы о виде зависимости целевой переменной от других переменных формулируются системой в виде программ на некотором внутреннем языке программирования. Если это универсальный язык, то теоретически на нем можно выразить зависимость любого вида. Процесс построения этих программ строится как эволюция в мире программ (этим метод немного похож на генетические алгоритмы). Когда система находит программу, достаточно точно выражающую искомую зависимость, она начинает вносить в нее небольшие модификации и отбирает среди построенных таким образом дочерних программ те, которые повышают точность. Таким образом, система «выращивает» несколько генетических линий программ, которые конкурируют между собой в точности выражения искомой зависимости. Специальный транслирующий модуль переводит найденные зависимости с внутреннего языка системы на понятный пользователю язык (математические формулы, таблицы и пр.), делая их легкодоступными. Для того чтобы сделать полученные результаты еще понятнее для пользователя-нематематика, имеется богатый арсенал разнообразных средств визуализации обнаруживаемых зависимостей.

Поиск зависимости целевых переменных от остальных ведется в форме функций какого-то определенного вида. Например, в одном из наиболее удачных ал-

горитмов этого типа — методе группового учета аргументов (МГУА) зависимость ищут в форме полиномов. Причем сложные полиномы заменяются несколькими более простыми, учитывающими только некоторые признаки (групп аргументов). Обычно используются попарные объединения признаков. По всей видимости, этот метод не имеет существенных преимуществ по сравнению с нейронными сетями, с их готовым набором стандартных нелинейных функций, несмотря на то, что полученная формула зависимости, в принципе, поддается анализу и интерпретации (хотя на практике все же бывает слишком сложна для этого).

Комбинированные методы. Часто производители сочетают указанные подходы. Объединение в себе средств нейронных сетей и технологии деревьев решений должно способствовать построению более точной модели и повышению ее быстродействия. Программы визуализации данных в каком-то смысле не являются средством анализа информации, поскольку они только представляют ее пользователю. Тем не менее визуальное представление, скажем, сразу четырех переменных достаточно выразительно обобщает очень большие объемы данных. Некоторые производители понимают, что для решения каждой проблемы следует применять оптимальный метод. Например, Продукт SAS Enterprise Miner 3.0 включает в себя модуль автоматического построения результирующей гибридной модели, определенной на множестве моделей, созданных предварительно принципиально различными методами — методами дерева решений, нейронных сетей, обобщенной многофакторной регрессии. Другой продукт под названием Darwin, готовящийся к выпуску в первой половине этого года компанией Thinking Machines (Бедфорд, шт. Массачусетс), позволит не только строить модели на основе нейронных сетей или деревьев решений, но также использовать визуализацию и системы рассуждения на основе аналогичных случаев. Кроме того, продукт включает в себя своеобразный генетический алгоритм для оптимизации моделей.

Чрезвычайно активно работает в области анализа и интерпретации информации хранилищ данных и компания IBM. Многие из полученных в ее лабораториях результатов нашли применение в выпускаемых компанией инструментальных пакетах, которые можно отнести к четырем из пяти стандартных типов приложений «глубокой переработки» информации: классификации, кластеризации, выявлению последовательностей и ассоциаций. Выделение подмножества данных. Одной из наиболее серьезных проблем анализа и интерпретации информации является необходимость выделения подмножества данных (из соображений производительности). При построении своей модели вы можете искать компромисс между числом записей (строк) в выборке данных и количеством оцениваемых переменных. В SAS Enterprise Miner для преодоления такого рода трудностей имеется специальный модуль, позволяющий легко настроить процесс выборки из генеральной совокупности.

Контрольные вопросы по главе 4

1. Приведите характеристики основных классов информационных систем.
2. Что понимается под термином «АИС»?
3. Приведите основные особенности и назначение OLTP-систем.
4. Раскройте сущность процесса управления транзакциями.
5. При решении каких задач применяется двухстадийная фиксация транзакций?
6. Что такое «тиражирование данных»?
7. Для чего нужны «хранилища данных»?
8. Какие основные операции присущи многомерной базе данных?
9. Приведите основные характеристики процесса обнаружения знаний в хранилище данных.
10. Какие существуют основные методы анализа данных?

Глава 5

CASE-ТЕХНОЛОГИИ

5.1 Истоки возникновения CASE-технологий

Тенденции развития современных информационных технологий приводят к постоянному возрастанию сложности информационных систем (ИС), создаваемых в различных областях экономики. Современные крупные проекты ИС характеризуются, как правило, следующими особенностями:

- сложностью описания (достаточно большое количество функций, процессов, элементов данных и сложные взаимосвязи между ними), требующей тщательного моделирования и анализа данных и процессов;
- наличием совокупности тесно взаимодействующих компонентов (подсистем), имеющих свои локальные задачи и цели функционирования (например, традиционных приложений, связанных с обработкой транзакций и решением регламентных задач, и приложений аналитической обработки (поддержки принятия решений), использующих нерегламентированные запросы к данным большого объема);
- отсутствием прямых аналогов, ограничивающим возможность использования каких-либо типовых проектных решений и прикладных систем;
- необходимостью интеграции существующих и вновь разрабатываемых приложений;
- функционированием в неоднородной среде на нескольких аппаратных платформах;
- разобщенностью и разнородностью отдельных групп разработчиков по уровню квалификации и сложившимся традициям использования тех или иных инструментальных средств;
- существенной временной протяженностью проекта, обусловленной, с одной стороны, ограниченными возможностями коллектива разработчиков и, с другой стороны, масштабами организации-заказчика и различной степенью готовности отдельных ее подразделений к внедрению ИС.

Для успешной реализации проекта объект проектирования (ИС) должен быть прежде всего адекватно описан, должны быть построены полные и непротиворечивые функциональные и информационные модели ИС. Накопленный к настоящему времени опыт проектирования ИС показывает, что это логически сложная, трудоемкая и длительная по времени работа, требующая высокой квалификации участвующих в ней специалистов. Однако до недавнего времени проектирование ИС выполнялось в основном на интуитивном уровне с применением неформализованных методов, основанных на искусстве, практическом опыте, экспертных оценках и дорогостоящих экспериментальных проверках качества функционирования ИС. Кроме того, в процессе создания и функционирования ИС информационные потребности пользователей могут изменяться или уточняться, что еще более усложняет разработку и сопровождение таких систем.

В 70-х и 80-х годах при разработке ИС достаточно широко применялась структурная методология, предоставляющая в распоряжение разработчиков строгие формализованные методы описания ИС и принимаемых технических решений. Она основана на наглядной графической технике: для описания различного рода моделей ИС используются схемы и диаграммы. Наглядность и строгость средств структурного анализа позволяла разработчикам и будущим пользователям системы с самого начала неформально участвовать в ее создании, обсуждать и закреплять понимание основных технических решений. Однако широкое применение этой методологии и следование ее рекомендациям при разработке конкретных ИС встречалось достаточно редко, поскольку при неавтоматизированной (ручной) разработке это практически невозможно. Действительно, вручную очень трудно разработать и графически представить строгие формальные спецификации системы, проверить их на полноту и непротиворечивость и тем более изменить. Если все же удается создать строгую систему проектных документов, то ее переработка при появлении серьезных изменений практически неосуществима. Ручная разработка обычно порождала следующие проблемы:

- неадекватную спецификацию требований;
- неспособность обнаруживать ошибки в проектных решениях;
- низкое качество документации, снижающее эксплуатационные качества;
- затяжной цикл и неудовлетворительные результаты тестирования.

С другой стороны, разработчики ИС исторически всегда стояли последними в ряду тех, кто использовал компьютерные технологии для повышения качества, надежности и производительности в своей собственной работе (феномен «сапожника без сапог»).

Перечисленные факторы способствовали появлению программно-технологических средств специального класса — CASE-средств, реализующих CASE-технологию создания и сопровождения ИС. Термин CASE (Computer Aided Software Engineering) используется в настоящее время в весьма широком смысле. Первоначальное значение термина CASE, ограниченное вопросами автоматизации разработки только лишь программного обеспечения (ПО), в настоящее время приобрело новый смысл, охватывающий процесс разработки сложных ИС в целом. Теперь под термином CASE-средства понимаются программные средства, поддерживающие процессы создания и сопровождения ИС, включая анализ и формулировку требо-

ваний, проектирование прикладного ПО (приложений) и баз данных, генерацию кода, тестирование, документирование, обеспечение качества, конфигурационное управление и управление проектом, а также другие процессы. CASE-средства вместе с системным ПО и техническими средствами образуют полную среду разработки ИС.

Появлению CASE-технологии и CASE-средств предшествовали исследования в области методологии программирования. Программирование обрело черты системного подхода с разработкой и внедрением языков высокого уровня, методов структурного и модульного программирования, языков проектирования и средств их поддержки, формальных и неформальных языков описаний системных требований и спецификаций и т. д. Кроме того, появлению CASE-технологии способствовали и такие факторы, как:

- подготовка аналитиков и программистов, восприимчивых к концепциям модульного и структурного программирования;
- широкое внедрение и постоянный рост производительности компьютеров, позволившие использовать эффективные графические средства и автоматизировать большинство этапов проектирования;
- внедрение сетевой технологии, предоставившей возможность объединения усилий отдельных исполнителей в единый процесс проектирования путем использования разделяемой базы данных, содержащей необходимую информацию о проекте.

CASE-технология представляет собой методологию проектирования ИС, а также набор инструментальных средств, позволяющих в наглядной форме моделировать предметную область, анализировать эту модель на всех этапах разработки и сопровождения ИС и разрабатывать приложения в соответствии с информационными потребностями пользователей. Большинство существующих CASE-средств основано на методологиях структурного (в основном) или объектно-ориентированного анализа и проектирования, использующих спецификации в виде диаграмм или текстов для описания внешних требований, связей между моделями системы, динамики поведения системы и архитектуры программных средств.

Однако, несмотря на все потенциальные возможности CASE-средств, существует множество примеров их неудачного внедрения, в результате которых CASE-средства становятся «полочным» ПО (shelfware). В связи с этим необходимо отметить следующее:

- CASE-средства не обязательно дают немедленный эффект; он может быть получен только спустя какое-то время;
- реальные затраты на внедрение CASE-средств обычно намного превышают затраты на их приобретение;
- CASE-средства обеспечивают возможности для получения существенной выгоды только после успешного завершения процесса их внедрения.

Ввиду разнообразной природы CASE-средств было бы ошибочно делать какие-либо безоговорочные утверждения относительно реального удовлетворения тех или иных ожиданий от их внедрения. Можно перечислить следующие факторы, усложняющие определение возможного эффекта от использования CASE-средств:

- широкое разнообразие качества и возможностей CASE-средств;
- относительно небольшое время использования CASE-средств в различных организациях и недостаток опыта их применения;
- широкое разнообразие в практике внедрения различных организаций;
- отсутствие детальных метрик и данных для уже выполненных и текущих проектов;
- широкий диапазон предметных областей проектов;
- различная степень интеграции CASE-средств в различных проектах.

Вследствие этих сложностей доступная информация о реальных внедрениях крайне ограничена и противоречива. Она зависит от типа средств, характеристик проектов, уровня сопровождения и опыта пользователей. Некоторые аналитики полагают, что реальная выгода от использования некоторых типов CASE-средств может быть получена только после одно- или двухлетнего опыта. Другие полагают, что воздействие может реально проявиться в фазе эксплуатации жизненного цикла ИС, когда технологические улучшения могут привести к снижению эксплуатационных затрат.

Для успешного внедрения CASE-средств организация должна обладать следующими качествами:

- Технология. Понимание ограниченности существующих возможностей и способность принять новую технологию.
- Культура. Готовность к внедрению новых процессов и взаимоотношений между разработчиками и пользователями.
- Управление. Четкое руководство и организованность по отношению к наиболее важным этапам и процессам внедрения.

Если организация не обладает хотя бы одним из перечисленных качеств, то внедрение CASE-средств может закончиться неудачей независимо от степени тщательности следования различным рекомендациям по внедрению.

Для того, чтобы принять взвешенное решение относительно инвестиций в CASE-технологию, пользователи вынуждены производить оценку отдельных CASE-средств, опираясь на неполные и противоречивые данные. Эта проблема зачастую усугубляется недостаточным знанием всех возможных «подводных камней» использования CASE-средств. Среди наиболее важных проблем выделяются следующие:

- достоверная оценка отдачи от инвестиций в CASE-средства затруднительна ввиду отсутствия приемлемых метрик и данных по проектам и процессам разработки ПО;
- внедрение CASE-средств может представлять собой достаточно длительный процесс и может не принести немедленной отдачи. Возможно даже краткосрочное снижение продуктивности в результате усилий, затрачиваемых на внедрение. Вследствие этого руководство организации-пользователя может утратить интерес к CASE-средствам и прекратить поддержку их внедрения;
- отсутствие полного соответствия между теми процессами и методами, которые поддерживаются CASE-средствами, и теми, которые используются в данной организации, может привести к дополнительным трудностям;

- CASE-средства зачастую трудно использовать в комплексе с другими подобными средствами. Это объясняется как различными парадигмами, поддерживаемыми различными средствами, так и проблемами передачи данных и управления от одного средства к другому;
- некоторые CASE-средства требуют слишком много усилий для того, чтобы оправдать их использование в небольшом проекте, при этом тем не менее можно извлечь выгоду из той дисциплины, к которой обязывает их применение;
- негативное отношение персонала к внедрению новой CASE-технологии может быть главной причиной провала проекта.

Пользователи CASE-средств должны быть готовы к необходимости долгосрочных затрат на эксплуатацию, частому появлению новых версий и возможному быстрому моральному старению средств, а также постоянным затратам на обучение и повышение квалификации персонала.

Несмотря на все высказанные предостережения и некоторый пессимизм, грамотный и разумный подход к использованию CASE-средств может преодолеть все перечисленные трудности. Успешное внедрение CASE-средств должно обеспечить такие выгоды, как:

- высокий уровень технологической поддержки процессов разработки и сопровождения ПО;
- положительное воздействие на некоторые или все из перечисленных факторов: производительность, качество продукции, соблюдение стандартов, документирование;
- приемлемый уровень отдачи от инвестиций в CASE-средства.

5.2 Структурный подход к проектированию ИС

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

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

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

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

- принцип абстрагирования — заключается в выделении существенных аспектов системы и отвлечения от несущественных;
- принцип формализации — заключается в необходимости строгого методического подхода к решению проблемы;
- принцип непротиворечивости — заключается в обоснованности и согласованности элементов;
- принцип структурирования данных — заключается в том, что данные должны быть структурированы и иерархически организованы.

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

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

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

5.3 Методология функционального моделирования SADT

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

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

- графическое представление блочного моделирования. Графика блоков и дуг SADT-диаграммы отображает функцию в виде блока, а интерфейсы входа/выхода представляются дугами, соответственно входящими в блок и выходящими из него. Взаимодействие блоков друг с другом описывается посредством интерфейсных дуг, выражающих «ограничения», которые в свою очередь определяют, когда и каким образом функции выполняются и управляются;
- строгость и точность. Выполнение правил SADT требует достаточной строгости и точности, не накладывая в то же время чрезмерных ограничений на действия аналитика. Правила SADT включают:
  - ограничение количества блоков на каждом уровне декомпозиции (правило 3–6 блоков);
  - связность диаграмм (номера блоков);
  - уникальность меток и наименований (отсутствие повторяющихся имен);
  - синтаксические правила для графики (блоков и дуг);
  - разделение входов и управлений (правило определения роли данных);
  - отделение организации от функции, т. е. исключение влияния организационной структуры на функциональную модель.

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

Результатом применения методологии SADT является модель, которая состоит из диаграмм, фрагментов текстов и глоссария, имеющих ссылки друг на друга.

Диаграммы — главные компоненты модели, все функции ИС и интерфейсы на них представлены как блоки и дуги. Место соединения дуги с блоком определяет тип интерфейса. Управляющая информация входит в блок сверху, в то время как информация, которая подвергается обработке, показана с левой стороны блока, а результаты выхода показаны с правой стороны. Механизм (человек или автоматизированная система), который осуществляет операцию, представляется дугой, входящей в блок снизу (рисунок 5.1).

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

На рисунке 5.2, где приведены четыре диаграммы и их взаимосвязи, показана структура SADT-модели. Каждый компонент модели может быть декомпозирован на другой диаграмме. Каждая диаграмма иллюстрирует «внутреннее строение» блока на родительской диаграмме.

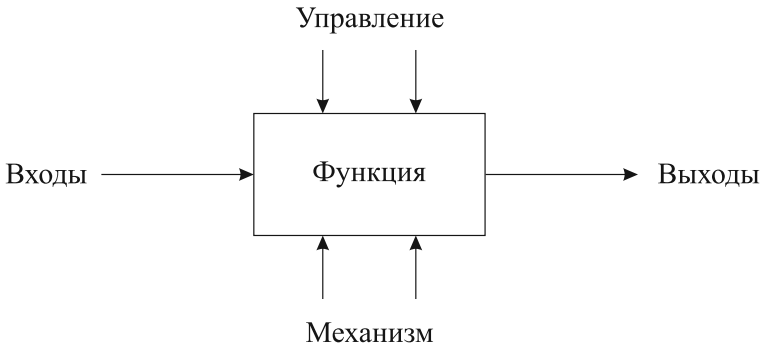


Рис. 5.1 – Функциональный блок и интерфейсные дуги

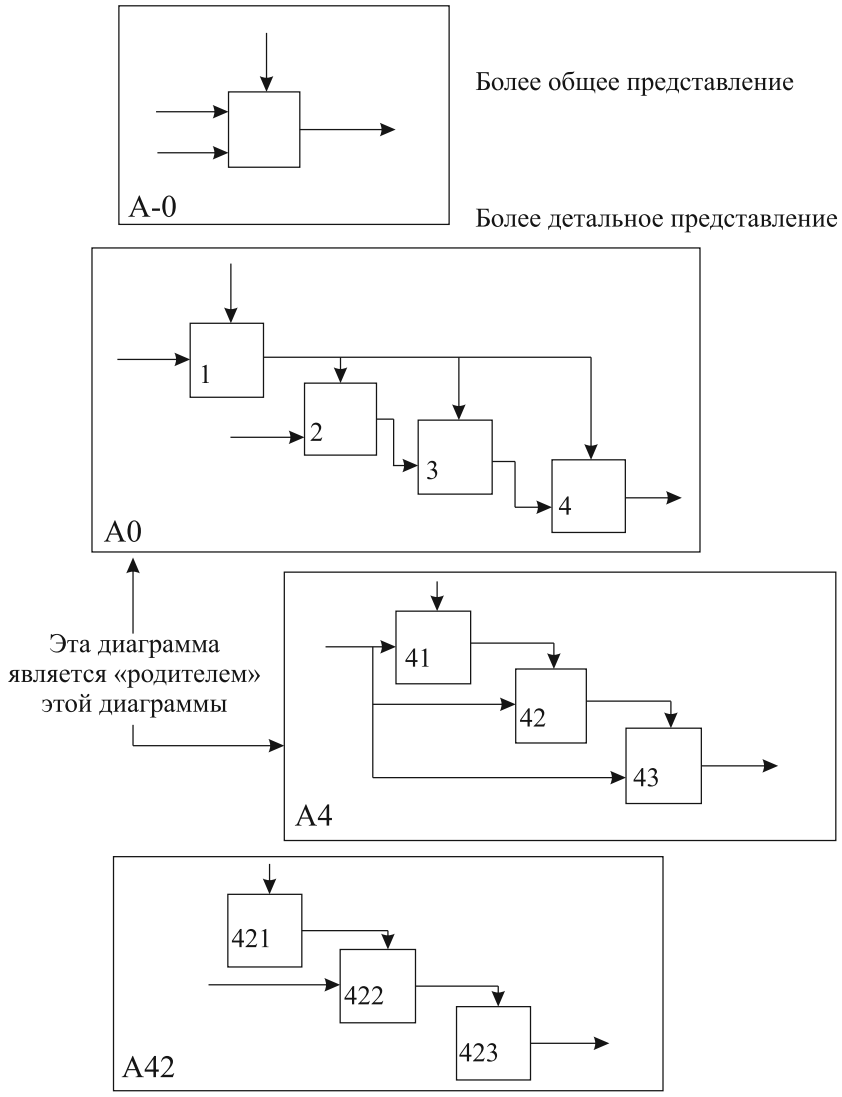


Рис. 5.2 – Структура SADT-модели. Декомпозиция диаграмм

Построение SADT-модели начинается с представления всей системы в виде простейшей компоненты — одного блока и дуг, изображающих интерфейсы с функциями вне системы. Поскольку единственный блок представляет всю систему как единое целое, имя, указанное в блоке, является общим. Это верно и для интерфейс-

ных дуг — они также представляют полный набор внешних интерфейсов системы в целом. Такая диаграмма, содержащая один блок, называется контекстной.

Затем блок, который представляет систему в качестве единого модуля, детализируется на другой диаграмме с помощью нескольких блоков, соединенных интерфейсными дугами. Эти блоки представляют основные подфункции исходной функции. Данная декомпозиция выявляет полный набор подфункций, каждая из которых представлена как блок, границы которого определены интерфейсными дугами. Каждая из этих подфункций может быть декомпозирована подобным образом для более детального представления. Число блоков на диаграмме не менее 3 и не более 6.

Во всех случаях каждая подфункция может содержать только те элементы, которые входят в исходную функцию. Кроме того, модель не может опустить какие-либо элементы, т. е., как уже отмечалось, родительский блок и его интерфейсы обеспечивают контекст. К нему нельзя ничего добавить, и из него не может быть ничего удалено.

Модель SADT представляет собой серию диаграмм с сопроводительной документацией, разбивающих сложный объект на составные части, которые представлены в виде блоков. Детали каждого из основных блоков показаны в виде блоков на других диаграммах. Каждая детальная диаграмма является декомпозицией блока из более общей диаграммы. На каждом шаге декомпозиции более общая диаграмма называется родительской для более детальной диаграммы.

Дуги, входящие в блок и выходящие из него на диаграмме верхнего уровня, являются точно теми же самыми, что и дуги, входящие в диаграмму нижнего уровня и выходящие из нее, потому что блок и диаграмма представляют одну и ту же часть системы.

Некоторые дуги присоединены к блокам диаграммы обоими концами, у других же один конец остается не присоединенным. Не присоединенные дуги соответствуют входам, управлениям и выходам родительского блока. Источник или получатель этих пограничных дуг может быть обнаружен только на родительской диаграмме, не присоединенные концы должны соответствовать дугам на исходной диаграмме. Все граничные дуги должны продолжаться на родительской диаграмме, чтобы она была полной и непротиворечивой. Каждый блок должен иметь дуги управления и выхода.

На SADT-диаграммах не указаны явно ни последовательность, ни время. Обратные связи, итерации, продолжающиеся процессы и перекрывающиеся (по времени) функции могут быть изображены с помощью дуг.

Как было отмечено, механизмы (дуги с нижней стороны) показывают средства, с помощью которых осуществляется выполнение функций. Механизм может быть человеком, компьютером или любым другим устройством, которое помогает выполнять данную функцию (рисунок 5.3).

Каждый блок на диаграмме имеет свой номер. Блок любой диаграммы может быть далее описан диаграммой нижнего уровня, которая в свою очередь может быть далее детализирована с помощью необходимого числа диаграмм. Таким образом, формируется иерархия диаграмм.

Для того, чтобы указать положение любой диаграммы или блока в иерархии, используются номера диаграмм. Например, A21 является диаграммой, которая де-

тализирует блок 1 на диаграмме А2. Аналогично, А2 детализирует блок 2 на диаграмме А0, которая является самой верхней диаграммой модели. На рисунке 5.4 показано типичное дерево диаграмм.

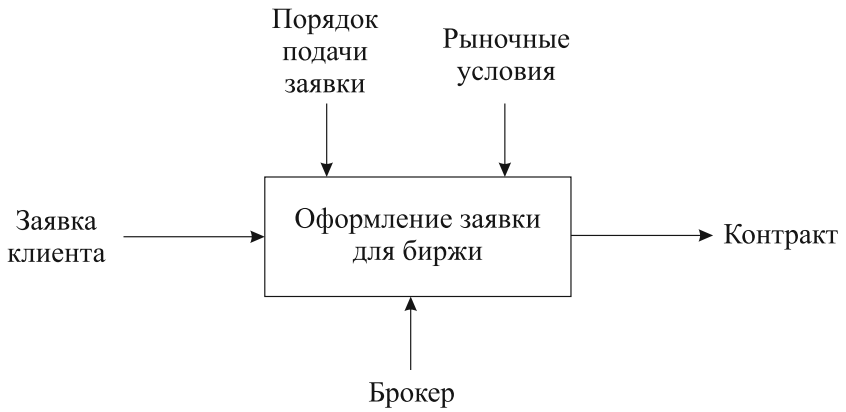


Рис. 5.3 – Пример механизма

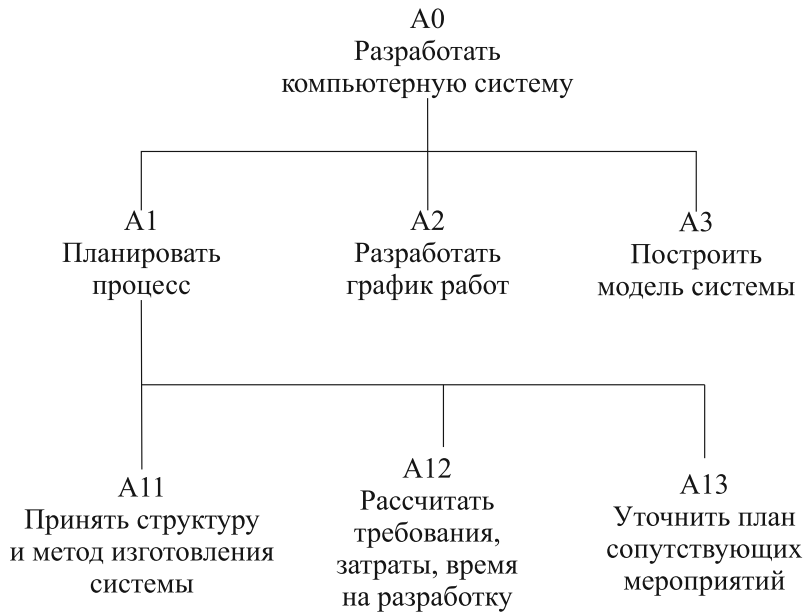


Рис. 5.4 – Иерархия диаграмм

5.4 Моделирование потоков данных (процессов)

Методология Гейна/Сарсона. В основе данной методологии (методологии Gane/Sarson) лежит построение модели анализируемой ИС — проектируемой или реально существующей. В соответствии с методологией модель системы определяется как иерархия диаграмм потоков данных (ДПД или DFD), описывающих асинхронный процесс преобразования информации от ее ввода в систему до выдачи пользователю. Диаграммы верхних уровней иерархии (контекстные диаграммы)

определяют основные процессы или подсистемы ИС с внешними входами и выходами. Они детализируются при помощи диаграмм нижнего уровня. Такая декомпозиция продолжается, создавая многоуровневую иерархию диаграмм, до тех пор, пока не будет достигнут такой уровень декомпозиции, на котором процессы становятся элементарными и детализировать их далее невозможно.

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

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

Внешняя сущность обозначается квадратом (рисунок 5.5), расположенным как бы «над» диаграммой и бросающим на нее тень, для того, чтобы можно было выделить этот символ среди других обозначений:

Рис. 5.5 – Внешняя сущность

При построении модели сложной ИС она может быть представлена в самом общем виде на так называемой контекстной диаграмме в виде одной системы как единого целого либо может быть декомпозирована на ряд подсистем. Подсистема (или система) на контекстной диаграмме изображается следующим образом (рисунок 5.6).

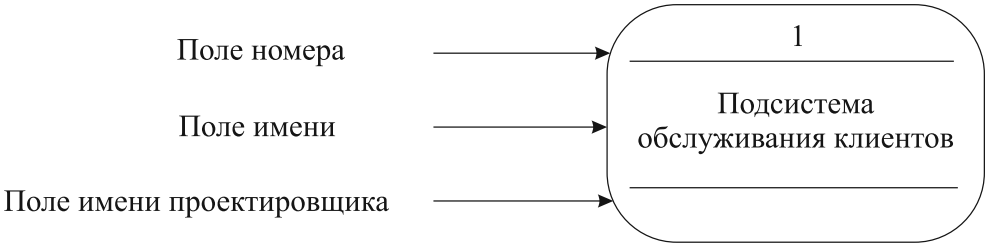


Рис. 5.6 – Изображение подсистемы

Номер подсистемы служит для ее идентификации. В поле имени вводится наименование подсистемы в виде предложения с подлежащим и соответствующими определениями и дополнениями.

Процесс представляет собой преобразование входных потоков данных в выходные в соответствии с определенным алгоритмом. Физически процесс может быть реализован различными способами: это может быть подразделение организации (отдел), выполняющее обработку входных документов и выпуск отчетов, программа, аппаратно реализованное логическое устройство и т. д. Процесс на диаграмме потоков данных изображается, как показано на рисунке 5.7.

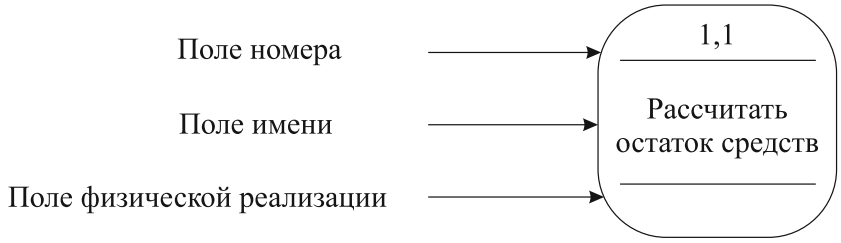


Рис. 5.7 – Изображение процесса

Номер процесса служит для его идентификации. В поле имени вводится наименование процесса в виде предложения с активным недвусмысленным глаголом в неопределенной форме (вычислить, рассчитать, проверить, определить, создать, получить), за которым следуют существительные в винительном падеже, например:

- «Ввести сведения о клиентах».
- «Выдать информацию о текущих расходах».
- «Проверить кредитоспособность клиента».

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

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

Накопитель данных представляет собой абстрактное устройство для хранения информации, которую можно в любой момент поместить в накопитель и через некоторое время извлечь, причем способы помещения и извлечения могут быть любыми. Накопитель данных может быть реализован физически в виде микрофиши, ящика в картотеке, таблицы в оперативной памяти, файла на магнитном носителе и т. д. Накопитель данных на диаграмме потоков данных изображается, как показано на рисунке 5.8.

Рис. 5.8 – Накопитель данных

Накопитель данных идентифицируется буквой «D» и произвольным числом. Имя накопителя выбирается из соображения наибольшей информативности для проектировщика.

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

Поток данных определяет информацию, передаваемую через некоторое соединение от источника к приемнику. Реальный поток данных может быть информацией, передаваемой по кабелю между двумя устройствами, пересылаемыми по почте письмами, магнитными лентами или дискетами, переносимыми с одного компьютера на другой и т. д.

Поток данных на диаграмме изображается линией, оканчивающейся стрелкой, которая показывает направление потока (рисунок 5.9). Каждый поток данных имеет имя, отражающее его содержание

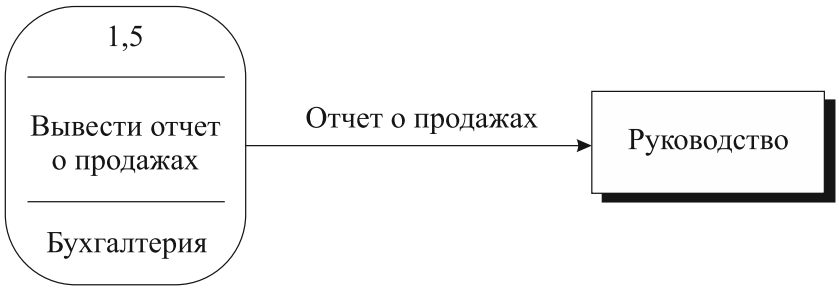


Рис. 5.9 – Поток данных

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

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

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

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

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

Разработка контекстных диаграмм решает проблему строгого определения функциональной структуры ИС на самой ранней стадии ее проектирования, что особенно важно для сложных многофункциональных систем, в разработке которых участвуют разные организации и коллективы разработчиков.

После построения контекстных диаграмм полученную модель следует проверить на полноту исходных данных об объектах системы и изолированность объектов (отсутствие информационных связей с другими объектами).

Для каждой подсистемы, присутствующей на контекстных диаграммах, выполняется ее детализация при помощи ДПД. Каждый процесс на ДПД, в свою очередь, может быть детализирован при помощи ДПД или миниспецификации. При детализации должны выполняться следующие правила:

- правило балансировки — означает, что при детализации подсистемы или процесса детализирующая диаграмма в качестве внешних источников/приемников данных может иметь только те компоненты (подсистемы, процессы, внешние сущности, накопители данных), с которыми имеет информационную связь детализируемая подсистема или процесс на родительской диаграмме;
- правило нумерации — означает, что при детализации процессов должна поддерживаться их иерархическая нумерация. Например, процессы, детализирующие процесс с номером 12, получают номера 12.1, 12.2, 12.3 и т. д.

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

Миниспецификация является конечной вершиной иерархии ДПД. Решение о завершении детализации процесса и использовании миниспецификации принимается аналитиком исходя из следующих критериев:

- наличия у процесса относительно небольшого количества входных и выходных потоков данных (2–3 потока);
- возможности описания преобразования данных процессом в виде последовательного алгоритма;
- выполнения процессом единственной логической функции преобразования входной информации в выходную;
- возможности описания логики процесса при помощи миниспецификации небольшого объема (не более 20–30 строк).

При построении иерархии ДПД переходить к детализации процессов следует только после определения содержания всех потоков и накопителей данных, которое описывается при помощи структур данных. Структуры данных конструируются из элементов данных и могут содержать альтернативы, условные вхождения и итерации. Условное вхождение означает, что данный компонент может отсутствовать в структуре. Альтернатива означает, что в структуру может входить один из перечисленных элементов. Итерация означает вхождение любого числа элементов в указанном диапазоне. Для каждого элемента данных может указываться его тип (непрерывные или дискретные данные). Для непрерывных данных может указы-

ваться единица измерения (кг, см и т. п.), диапазон значений, точность представления и форма физического кодирования. Для дискретных данных может указываться таблица допустимых значений.

После построения законченной модели системы ее необходимо верифицировать (проверить на полноту и согласованность). В полной модели все ее объекты (подсистемы, процессы, потоки данных) должны быть подробно описаны и детализированы. Выявленные недетализированные объекты следует детализировать, вернувшись на предыдущие шаги разработки. В согласованной модели для всех потоков данных и накопителей данных должно выполняться правило сохранения информации: все поступающие куда-либо данные должны быть считаны, а все считываемые данные должны быть записаны.

5.5 Моделирование данных

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

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

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

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

5.6 Общая характеристика и классификация CASE-средств

Современные CASE-средства охватывают обширную область поддержки многочисленных технологий проектирования ИС: от простых средств анализа и документирования до полномасштабных средств автоматизации, покрывающих весь жизненный цикл ПО.

Наиболее трудоемкими этапами разработки ИС являются этапы анализа и проектирования, в процессе которых CASE-средства обеспечивают качество принимаемых технических решений и подготовку проектной документации. При этом большую роль играют методы визуального представления информации. Это предполагает построение структурных или иных диаграмм в реальном масштабе времени,

использование многообразной цветовой палитры, сквозную проверку синтаксических правил. Графические средства моделирования предметной области позволяют разработчикам в наглядном виде изучать существующую ИС, перестраивать ее в соответствии с поставленными целями и имеющимися ограничениями.

В разряд CASE-средств попадают как относительно дешевые системы для персональных компьютеров с весьма ограниченными возможностями, так и дорогостоящие системы для неоднородных вычислительных платформ и операционных сред. Так, современный рынок программных средств насчитывает около 300 различных CASE-средств, наиболее мощные из которых так или иначе используются практически всеми ведущими западными фирмами.

Обычно к CASE-средствам относят любое программное средство, автоматизирующее ту или иную совокупность процессов жизненного цикла ПО и обладающее следующими основными характерными особенностями:

- мощные графические средства для описания и документирования ИС, обеспечивающие удобный интерфейс с разработчиком и развивающие его творческие возможности;
- интеграция отдельных компонент CASE-средств, обеспечивающая управляемость процессом разработки ИС;
- использование специальным образом организованного хранилища проектных метаданных (репозитория).

Интегрированное CASE-средство (или комплекс средств, поддерживающих полный ЖЦ ПО) содержит следующие компоненты:

- репозиторий, являющийся основой CASE-средства. Он должен обеспечивать хранение версий проекта и его отдельных компонентов, синхронизацию поступления информации от различных разработчиков при групповой разработке, контроль метаданных на полноту и непротиворечивость;
- графические средства анализа и проектирования, обеспечивающие создание и редактирование иерархически связанных диаграмм (DFD, ERD и др.), образующих модели ИС;
- средства разработки приложений, включая языки 4GL и генераторы кодов;
- средства конфигурационного управления;
- средства документирования;
- средства тестирования;
- средства управления проектом;
- средства реинжиниринга.

Все современные CASE-средства могут быть классифицированы в основном по типам и категориям. Классификация по типам отражает функциональную ориентацию CASE-средств на те или иные процессы ЖЦ. Классификация по категориям определяет степень интегрированности по выполняемым функциям и включает отдельные локальные средства, решающие небольшие автономные задачи (tools), набор частично интегрированных средств, охватывающих большинство этапов жизненного цикла ИС (toolkit), и полностью интегрированные средства, поддержива-

ющие весь жизненный цикл ИС и связанные общим репозиторием. Помимо этого, CASE-средства можно классифицировать по следующим признакам:

- применяемым методологиям и моделям систем и БД;
- степени интегрированности с СУБД;
- доступным платформам.

Классификация по типам в основном совпадает с компонентным составом CASE-средств и включает следующие основные типы:

- средства анализа (Upper CASE), предназначенные для построения и анализа моделей предметной области (Design/IDEF (Meta Software), BPwin (Logic Works));
- средства анализа и проектирования (Middle CASE), поддерживающие наиболее распространенные методологии проектирования и использующиеся для создания проектных спецификаций. Выходом таких средств являются спецификации компонентов и интерфейсов системы, архитектуры системы, алгоритмов и структур данных;
- средства проектирования баз данных, обеспечивающие моделирование данных и генерацию схем баз данных (как правило, на языке SQL) для наиболее распространенных СУБД. К ним относятся ERwin (Logic Works), S-Designor (SDP) и DataBase Designer (ORACLE);
- средства разработки приложений;
- средства реинжиниринга, обеспечивающие анализ программных кодов и схем баз данных и формирование на их основе различных моделей и проектных спецификаций. В области анализа программных кодов наибольшее распространение получают объектно-ориентированные CASE-средства, обеспечивающие реинжиниринг программ на языке C++ (Rational Rose (Rational Software), Object Team (Cayenne)).

Вспомогательные типы включают:

- средства планирования и управления проектом (SE Companion, Microsoft Project и др.);
- средства конфигурационного управления;
- средства тестирования;
- средства документирования.

Контрольные вопросы по главе 5

1. Какие основные задачи призваны решать CASE-технологии?
2. В чем сущность структурного подхода к проектированию ИС?
3. Вспомните основные элементы методологии SADT.

4. Из чего состоит функциональная модель в методологии SADT?
5. Приведите основные положения методологии Гейна/Сарсона моделирования потоков данных.
6. Из каких шагов состоит построение иерархии диаграмм потоков данных?
7. Какие методологии моделирования данных Вам известны?

Глава 6

ГЕОИНФОРМАЦИОННАЯ ТЕХНОЛОГИЯ

6.1 История появления ГИС

Аббревиатура ГИС расшифровывается буквально как **географическая информационная система** или **геоинформационная система**. Можно рассматривать ГИС как набор аппаратных и программных инструментов, используемых для ввода, хранения, манипулирования, анализа и отображения пространственной информации. Термин «***геоинформационная***» стал сегодня обозначать уже нечто большее, чем его развернутый вариант.

Первой ГИС принято считать систему, созданную в 1962 году в Канаде Аланом Томлинсоном, которая так и называлась Канадская Географическая Информационная Система. Первые ГИС представляли собой целые комнаты, занятые вычислительной аппаратурой и множеством полок, заполненных перфокартами с пространственной и описательной информацией об объектах (координатами). Из-за высокой стоимости такие ГИС были немногочисленны и доступны только крупным государственным организациям, а также организациям, управляющим эксплуатацией природных ресурсов. Развитие ГИС в современном их понимании и роли как технологии несомненно, связано с бурным развитием информационных технологий в целом и, в первую очередь, с развитием аппаратной базы.

Три источника рождения ГИС-технологий. ГИС-технологии предназначены для работы с любыми данными, имеющими пространственно-временную привязку, что обусловило их быстрое распространение и широкое использование во многих отраслях науки и техники, и прежде всего в областях, связанных с применением карт и планов. Значение карты трудно переоценить в различных сферах деятельности человека и общества в целом. Цифровая геодезия и цифровая картография (**Automated Mapping, AM**) стали естественным продолжением традиционных наук и **первым** из трех источников ГИС-технологий. Они научились хорошо описывать, структурировать, хранить и обрабатывать пространственную геодезическую и картографическую информацию, решать задачи картографической алгебры. **Вто-**

рым источником стало развитие систем управления базами данных (**СУБД**), обеспечившее рациональные методы хранения всех видов информации и реальное время доступа к данным даже при условии их распределенного хранения, а иногда благодаря ему. Обычные (непространственные) данные, как-либо связанные с пространственными данными, называются в ГИС **атрибутивной информацией**. Уже эти два компонента имеют мощный потенциал, позволивший эффективно развиваться цифровой картографии и автоматизации управления инженерными сетями и коммуникациями (**Facilities Management, FM**). Пространственная информация FM-систем во многом строилась на информации о проектах инженерных сетей, построенных в системах автоматизированного проектирования (**CAD**). В конце 80-х годов в США появились первые природоохранные ГИС. Однако и эти ГИС все еще требовали довольно дорогих программных и аппаратных средств (высокопроизводительных рабочих станций) и не выходили на уровень массовых технологий. Сделать **третий** последний шаг для выхода на уровень массовой технологии позволило развитие вычислительных и сетевых возможностей массового персонального компьютера до уровня возможностей рабочей станции.

Первые общедоступные, полнофункциональные ГИС, способные работать на персональных компьютерах, появились в 1994 (ArcView 2.0). С этого времени и началось бурное развитие ГИС как массовой технологии. ГИС-технологии широко шагнули в жизнь и различные массовые задачи: управления; торговли, транспорта и складского хозяйства; сельского хозяйства; экологии и природопользования; здравоохранения; туризма; строительства; оптимального инвестирования и т. д.

Основу привлекательности ГИС-технологий составляют:

- наглядность пространственного представления результатов анализа баз данных;
- мощные возможности интеграции данных, в том числе возможности совместного исследования факторов атрибутивной информации, которые имеют пространственное пересечение;
- возможности изменения пространственной информации по результатам совместного анализа баз атрибутивных и пространственных данных.

Если же говорить о началах цифровой картографии, то первая в мире цифровая модель местности (ЦММ, DTM — Digital Terrain Model) была создана в 1957 году профессором Массачусетского технологического института Миллером. Она представляла собой цифровую модель рельефа и предназначалась для проектирования автодорог. В дальнейшем ЦММ стали применяться в других областях. Картографы и геодезисты осознали, что они могут служить основой автоматизации картографирования. В СССР первые попытки создания ЦММ были предприняты в 1960-х годах. Но уже в начале 70-х и в 80-х были запущены спутники, обеспечившие глобальное покрытие земного шара стереосъемкой для создания карт масштаба 1:50000 непревзойденного качества.

6.2 Общие функциональные компоненты ГИС

Функциональными составляющими ГИС как программно-технического комплекса являются: данные; программное обеспечение; аппаратное обеспечение; персонал; функциональные возможности.

Данные — любая пространственная информация и связанная с ней табличная (атрибутивная) информация. ГИС представляет собой одновременно средство по созданию данных и управлению ими.

Источниками данных *для ГИС* являются: существующие карты (в том числе в виде слайдов постоянного хранения); геодезические данные точного измерения координат и метрической информации поверхности: воздушные, наземные, подземные, водные, космические; аэрокосмическая фотосъемка и сканирование, стереофотосъемка; данные из архитектурно-строительных и инженерно-коммуникационных САПР (**CAD**).

Программное обеспечение — функции и инструменты, необходимые для управления, анализа и визуализации пространственной информации, а также управления ГИС в целом.

Аппаратное обеспечение — компьютер, на котором работает ГИС, а также средства ввода/вывода (сканеры, GPS-приемники, принтеры, плоттеры и т. д.). ГИС могут работать на различных типах компьютерных платформ, от централизованных серверов до отдельных или связанных сетью персональных компьютеров (ПК).

Персонал. Создание и управление ГИС невозможно без людей. Персоналом ГИС являются как технические специалисты, разрабатывающие и поддерживающие систему, создающие и поддерживающие в актуальном виде данные, так и непосредственные пользователи.

Функциональные возможности — методологический и алгоритмический аппарат, заложенный в ГИС. Современные ГИС включают средства разработки, позволяющие наращивать функциональность и превращать универсальные ГИС в специализированные системы для конкретных отраслей, сфер знания, производственных коллективов.

Основными функциями ГИС считаются следующие три широкие группы функций:

1) функции автоматизированного картографирования;
2) функции пространственного анализа;
3) функции управления данными.

Функции **автоматизированного картографирования** должны обеспечивать работу с пространственными данными ГИС с целью их отбора, обновления и преобразования для производства высококачественных карт и изображений. Функции автоматизированного картографирования должны включать векторно-растровые преобразования, преобразования координатной системы, картографических проекций и масштабов, «склейки» отдельных листов, осуществления картометрических измерений (вычисления площадей, расстояний), размещение текстовых надписей и внемасштабных картографических знаков, формирование макета печати.

Функции **пространственного анализа** должны обеспечивать совместное использование и обработку картографических и атрибутивных данных в интересах создания производных картографических данных. Функции пространственного

анализа должны включать анализ географической близости, анализ сетей, топологическое наложение полигонов, интерполяцию и изолинейное картографирование полей, вычисление буферных зон.

Функции **управления данными** должны обеспечивать работу с атрибутивными (неграфическими) данными ГИС с целью их отбора, обновления и преобразования для производства стандартных и рабочих отчетов. Функции управления данными должны включать пользовательские запросы, генерацию пользовательских документов, статистические вычисления, логические операции, поддержание информационной безопасности, стандартных форм запросов и представления их результатов.

В общем случае ГИС должна состоять из следующих четырех подсистем:

- сбора, подготовки и ввода данных;
- хранения, обновления и управления данными;
- обработки, моделирования и анализа данных;
- контроля, визуализации и вывода данных.

Задача подсистемы **сбора, подготовки и ввода данных** — формирование баз географических и атрибутивных данных ГИС.

Задача подсистемы **обработки, моделирования и анализа данных** — организация обработки данных, обеспечение процедур их преобразования, математического моделирования и сопряженного анализа.

Задача **подсистемы хранения**, обновления и управления данными — организация хранения данных, обеспечение их редактирования и обновления, обслуживание запросов на информационный поиск, поступающих в систему.

Основная задача **подсистемы контроля**, визуализации и вывода данных — генерация и оформление результатов работы системы в виде карт, графических изображений, таблиц, текстов на твердых или магнитных носителях.

Еще одной очень важной для прикладного развития, распространения и применения ГИС-технологий функцией ГИС является поддержка ***встроенной*** среды разработки (***engine***) дополнительных функций программного обеспечения или даже автономных, например Интернет-приложений. Разумеется, это свойство не только ГИС, но и любых развитых компьютерных технологий.

Программное обеспечение ГИС включает в себя обеспечение множества технологических аспектов ГИС-технологий: это и оцифровка (векторизация) бумажных и растровых карт; ввод и преобразование данных наземной и аэрокосмической топосъемки, GPS-приемников; восстановление рельефа по стереоснимкам методами фотограмметрии; построение топологических моделей по векторным; решение задач картографической алгебры и т. д. Ведущие производители программного обеспечения ГИС поддерживают сегодня практически весь спектр ПО.

Многообразие ПО ГИС одного производителя называется ГИС-платформой. Мировыми лидерами в области производства ГИС являются: Autodesk Inc с линейкой программных продуктов AutoCAD Map, AutoCAD Civil [3], MapGuide; компания ESRI (USA) с ГИС-платформой ArcGIS; компания MapInfo Corp с ГИС-платформой MapInfo. Стоимость платформ и отдельных систем колеблется от нескольких сотен до единиц тысяч долларов, а наиболее характерный диапазон — от

$1000 до $5000. Autodesk проводит очень выгодную для вузов академическую политику.

Среди российских производителей следует отметить линейку GeoGraph/GeoDraw/GeoConstructor производства ЦГИ ИГ РАН (Москва); ГИС Panorama (GeoSpectrum International, Москва); ГИС Terra (НИИПМК, Н. Новгород). Так, ГИС GeoGraph по своим возможностям весьма близка к ГИС ArcView-ArcGIS, а ЦГИ ИГ РАН проводит льготную ценовую политику в отношении вузов. Возможности программного обеспечения разных фирм постоянно сближаются, и сегодня более важна полнота базы данных и функциональность проекта прикладной системы [2].

В состав ГИС-платформы ArcGIS входит: настольная (локальная) ГИС; ГИС, встроенная в другую систему в виде некоего движка; серверная ГИС с серверной и вьюерной (Интернет/интранет) частью и даже мобильный вариант ГИС. Каждая из частей поддержана соответствующей библиотекой или группой сервисов.

6.3 Принципы организации ГИС

6.3.1 Слой, карта и проект

Рабочей средой при работе с ГИС-платформой является Проект. ***Проект*** может включать в себя все информационные компоненты, на которых строятся ГИС-технологии. ***Основной структурной единицей ГИС является тематический слой***, понятие которого тесно связано с более общим понятием покрытия, несущим в себе объектное содержание (например, единица административно-территориального деления).

Покрытие (Coverage) — цифровая модель единицы хранения базы векторных данных ГИС, хранит в виде записей все объекты первичного уровня (точки, дуги, полигоны) и вторичного уровня (координаты опорных точек, аннотации и т. д.) некоторого пространственного объекта и структуру отношений между ними, в том числе топологические. Пустое покрытие — покрытие, в котором отсутствуют какие-либо пространственные объекты.

Слой (Map Layer) — покрытие, рассматриваемое в контексте его содержательной определенности (растительность, рельеф, административное деление и т. п.) или его статуса в среде редактора (активный слой, пассивный слой).

Слой, как правило, однороден не только по тематике, но и по типам объектов (точечные, линейные, полигональные, растровые). Информационные компоненты могут быть внешними (векторные и растровые слои, таблицы, библиотеки символов) или внутренними (специальные типы слоев, запросы, макросы, карты, макеты печати и т. д.).

При создании нового проекта необходимо **подключить или создать новые слои**. Векторные слои (содержащие точечные, линейные, площадные объекты) могут быть созданы непосредственно в среде ГИС или в других программных средах (например, это может быть чертеж в обменном или двоичном формате AutoCAD). В качестве слоев могут быть загружены растровые изображения различных форматов (как правило, используемых в цифровой картографии). С каждым векторным слоем может быть связана таблица характеристик, хранимая с векторным слоем,

и **набор таблиц** с атрибутивными (тематическими) данными, хранимый во внешней СУБД.

Для каждого слоя можно определить следующие объекты базы данных:

- ***Запросы*** к атрибутивным таблицам;
- ***Темы*** (варианты тематического картографирования слоя);
- ***Формы*** представления справочной информации об объектах;
- ***Диаграммы*** (представления результатов в виде различных графиков);
- ***Макросы*** — внешние исполняемые программы или внутренние функции ГИС (задаются пользователем для карты в целом, для слоя или для отдельных объектов).

Слои (или покрытия) объединяются в цифровые карты. Карты могут не поддерживать в своей структуре покрытия, но в этом случае берут часть или все функции покрытий на себя. В рамках одного проекта может быть создано неограниченное количество карт. Карты могут быть связаны друг с другом как вертикально, так и горизонтально. Работая внутри карты, можно добавлять слои, создавать и редактировать пространственные объекты, в том числе с соблюдением топологии, осуществлять работу с таблицами (записывать в таблицы результаты измерений по карте, производить изменение структуры, сортировку, редактирование, выборки вручную, запросы с отображением результатов выборок на карте).

Дополнительные возможности управления дает панель **управления слоями (Легенда)** карты, на которой представлены все слои. Здесь можно:

- включать и выключать отображение слоя;
- присваивать слоям диапазон масштабов, при которых они будут видимыми;
- удалять слои из списка слоев, отнесенных к карте;
- перемещать слои в списке (и, одновременно, в порядке воспроизведения) вверх или вниз;
- изменять тематическую классификацию для слоев и т. д.

Карты могут быть подготовлены к печати в виде **макетов (Layouts) печати**. В состав макета печати можно включить любые карты и их легенды. В макет печати могут входить также тексты, таблицы, графики, растровые изображения и др.

Любую карту, макет печати, таблицы, темы, запросы, диаграммы, макросы — можно сохранить в проекте для последующего использования. Одним из важных для реализации роли ГИС, интегрирующей различные информационные среды, является **контекстная ориентированность** рабочей среды.

Это значит, что весь интерфейс ГИС (набор меню, панелей и инструментов, реакции по правой клавише мыши и т. д.) качественно меняется в зависимости от того, с каким объектом вы работаете в данный момент.

6.3.2 Пространственные объекты слоев и их модели

При оцифровке карт выделяется три типа объектов, к которым можно отнести любой имеющийся на карте:

- **Точечный объект**. Объект, обозначенный точкой, поскольку его размеры слишком малы, чтобы можно было отразить его форму (границы, площадь) в масштабе карты. Может также представлять некий условный объект, не имеющий размеров, например отметку высот.
- **Линейный объект**. Объект, локализованный в виде линии, поскольку его ширина не выражается в масштабе карты-источника, — река, дорога и т. д. Может также представлять некий условный объект, например, границу.
- **Полигональный объект**. Объект, имеющий площадь, выражающуюся в масштабе карты-источника. Определяется замкнутым контуром и его внутренней областью, например лес, озеро.

Возможности ГИС в значительной мере зависят от того, какими моделями она поддерживает примитивы пространственных слоев. Сложность модели должна соответствовать сложности реальных объектов и сложности решаемых задач. В ГИС, допускающих трехмерное моделирование, таких как, например, AutoCAD Map и AutoCAD Civil, кроме классических объектов, могут также использоваться 3D-поверхности и 3D-solid модели.

Векторные модели. Большая часть функций и задач моделирования пространственных объектов ГИС может быть реализована на основе векторных моделей (в виде точек, линий и полигонов). Векторные модели особенно удобны для представления и хранения дискретных объектов, таких как здания, трубопроводы или границы участков.

Точки — это пары координат (x, y) или тройки координат (x, y, z), где z — высота). ***Линии*** — наборы координат, определяющих совокупность отрезков. ***Полигоны*** — наборы координат, определяющих границы замкнутых областей. Значения координат зависят от географической системы координат, в которой хранятся данные.

ГИС могут хранить векторные данные в классах пространственных объектов и наборах топологически-связанных классов объектов. Во втором случае мы имеем дело уже с **векторной топологичекой моделью**. Атрибуты, связанные с объектами, хранятся в таблицах данных.

В разных векторных ГИС используются разные реализации векторной и векторно-топологической модели пространственных данных, например: ***покрытия***, ***шейп-файлы***, формат ***dwg***. Есть случаи, когда в ГИС в качестве основной используется векторная модель, но при необходимости решения специальных задач по векторной модели строится и затем используется топологическая модель. Так устроены, например, ГИС ArcView, AutoCAD Map, AutoCAD Civil. Бывает, что топологические модели хранятся в составе коллекции геоинформации разных типов в ***базе геоданных***.

Векторные топологические модели.

В топологическом слое в процессе его создания и редактирования создаются и фиксируются как сами пространственные объекты, так и пространственные отношения между указанными объектами — связность, соседство, смежность, вложенность. При этом объекты типа «полигон» создаются в результате сборки полигонов из дуг, образующих замкнутые контуры. Благодаря своим свойствам топологические модели обеспечивают решение пространственных задач. В ГИС применяются

узловые топологии (Node Topology), сетевые топологии (Network Topology) и полигональные топологии (Poligon Topology).

Элементом узловой топологии является узел. Каждый узел в узловой топологии может характеризоваться набором данных: $\{$ID, X,Y[,Z, w,]$\}$, где ID — идентификатор; X,Y — координаты; w — вес.

Элементом сетевой топологии является ребро (дуга). Дуга (линия) — упорядоченный набор связных отрезков (соединенных вершинами). Каждое ребро (дуга) сетевой топологии может характеризоваться следующим набором данных: $\{$ID, StartNode, EndNode[, LeftPol, RightPol, DirectWeight, BackWeight,]$\}$, где ID — идентификатор; StartNode, EndNode — начальный и конечный узлы дуги; LeftPol, RightPol — идентификаторы полигонов справа и слева от дуги (если одновременно построена полигональная топология); DirectWeight, BackWeight — вес дуги в прямом и обратном направлениях. В зависимости от того, сколько дуг объединено в одном узле, узлы могут обозначаться по-разному (рис. 6.1) и различаться как:

- △ — ***нормальные узлы*** (три и более дуг);
- ◇ — ***псевдоузлы*** (две дуги, в том числе разные концы одной дуги);
- □ — ***висячие узлы*** (одна дуга).

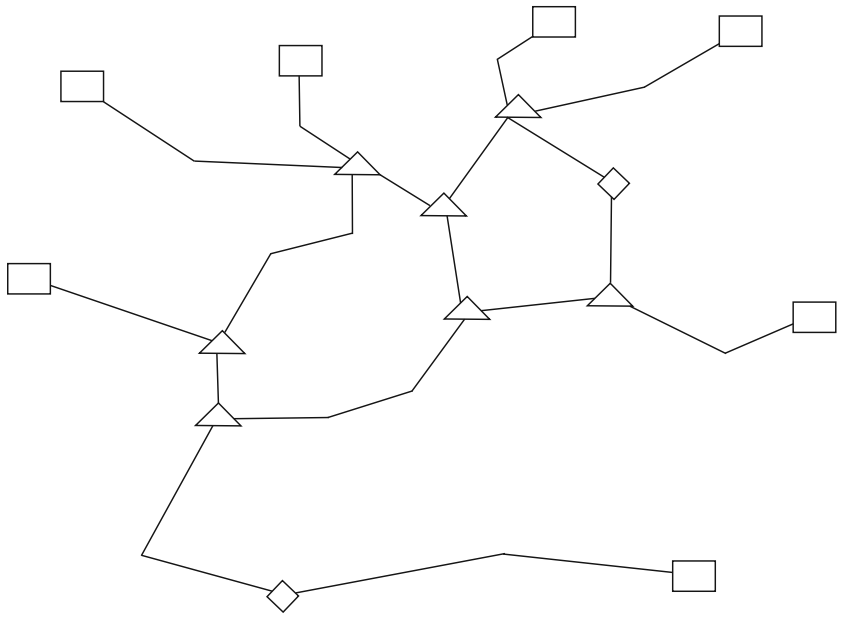


Рис. 6.1 – Пример разных узлов

Псевдоузлы не являются узлами ветвления, не являются необходимыми для решения топологических задач и поэтому могут быть удалены (подчистка псевдоузлов) с объединением каждой пары дуг, инцидентных псевдоузлу, в одну дугу в соответствующей вершине.

Элементом полигональной топологии является ***полигон***. При создании полигональной топологии создаются и сетевая, и узловая топологии.

Каждый полигон может характеризоваться следующим набором данных: {ID, Area, N, X,Y}, где ID — идентификатор полигона; Area — его площадь; N — число ребер, ограничивающих полигон; X,Y — координаты центроида полигона.

Понятие центроида полигона не является простым. В общем случае центроид — это точка, обязательно лежащая внутри полигона.

Существуют различные алгоритмы ее автоматического выбора. Однако после автоматического выбора центроид может быть вручную перенесен в другую внутреннюю точку. При автоматическом создании топологии, центроиды могут быть также назначены предварительно из числа (из слоя) точечных объектов (например, областной центр может быть назначен в качестве центроида области на карте России).

Растровые модели. В растровой модели пространственная информация представлена в виде таблицы, каждой ячейке которой соответствует заданный цвет. Часто растровая модель местности (например, данные аэро- или спутниковой съемки или сканированные карты) используется как исходный материал для построения векторных моделей (для векторизации) и/или как подложка для них (см. напр. GoogleEarth). Для точного размещения растра в пространстве модели (географическом пространстве) указываются координаты как минимум одного угла (или опорной точки) растра. Очень часто нужно совместить несколько растров, перекрывающих друг друга. Для точного совмещения оказывается необходимым подвергнуть растры согласованным аффинным, проективным, кусочно-аффинным или нелинейным преобразованиям по дополнительной информации о координатах набора опорных точек (тиков). При оцифровке наборов объектов карты в разные слои полезно использовать одни и те же тики, чтобы слои правильно совместились.

Растровые модели удобны для хранения и анализа данных, распределенных непрерывно на определенной площади. Каждая ячейка содержит значение, определяющее принадлежность к классу или категории, это может быть измерение или результат его интерпретации.

Кроме изображений к растровым данным относят также гриды (grids). Гриды содержат расчетные данные, что часто выгодно использовать для моделирования и анализа. Такие данные могут быть получены из точек замеров (например, грид химического состава почв) или основаны на классификации изображения, например грид землепользования. Гриды также можно создать из векторных данных.

GRID переводится как «сеть», «решетка». В отличие от модели TIN она сложена не треугольниками, а квадратиками (или прямоугольниками), является регулярной и, вообще говоря, плоской. Представьте себе шахматную доску. Она состоит из клеток (в нашем случае они называются ячейками), цвет которых может рассматриваться как характеристика клетки, ее атрибут. Естественно, что таких атрибутов может быть много. Если представить не шахматную доску, а топографическую карту, то у каждой ячейки может быть наличие зеленых насаждений, наличие водоема, высота и пр.

Для удобства практического понимания давайте рассмотрим рельеф. Таким образом, у нас у каждой ячейки есть высота. Естественно, что чем меньше ячейка, тем детальней описан рельеф, и очевидно также, что между ячейками высота точно не может быть определена. Такой способ представления информации о рельефе позволяет, используя высоты соседствующих ячеек, производить несложные расчеты для определения крутизны склона, его экспозиции или направления стока поверхностных вод.

Для того чтобы построить рельеф в виде GRIDa, можно оцифровать обычные горизонтали (об оцифровке мы поговорим ниже), разнести по этим горизонталям

точки, присвоив для них значение высоты той горизонтали, на которой они лежат, добавить точки с известными высотами от иных объектов (высотные отметки, отметки уреза воды), построить по этим точкам модель TIN, а затем уже по модели TIN построить модель GRID. Переход от TIN к GRID позволит получить значения высот для ячеек GRID, находящихся между горизонталями. Затем использовать GRID в решении своих задач. Вы правы, GRID очень похож на растр.

В гридах могут храниться как непрерывные данные (например, высота рельефа), так и категории (например, тип растительности) и дополнительные атрибуты категорий. Например, в гриде типов растительности для каждой категории может храниться код, название типа, пригодность для обитания определенных видов животных и код обобщенного типа. В этом отличие от векторных данных, где атрибуты соответствуют отдельным объектам.

Чем меньше размер ячейки растрового слоя, тем больше разрешение и подробнее данные. Однако поскольку ячейки равномерного грида покрывают всю поверхность, уменьшение размера ячейки может существенно увеличить объем хранимых данных.

ГИС распознает и может использовать растры из файлов изображений многих типов и из гридов, хранящихся в рабочих областях. Можно добавлять растровые наборы данных к карте так же, как векторные объекты.

Если мы рассматриваем растры как модель данных для некоторой ГИС, а не только как входные данные, то должны быть также определены еще и задачи, которые решаются на этой модели данных, например, после сшивки растров. Это могут быть задачи: восстановления рельефа по раскраске карты; классификации объектов карты (растра) по ее раскраске; решение задачи о близости или инцидентности объектов карты; идентификация и связывание объектов растровой карты с базой данных.

Модели TIN. Модель TIN относится к классу трехмерных векторных моделей и предсталяет собой триангуляционную нерегулярную сетку (TIN) на моделируемой поверхности.

TIN — эффективный способ хранения и анализа поверхностей, так как триангуляционная сеть позволяет более точно, чем растр, моделировать неоднородные поверхности, которые могут резко менять форму на одних участках и незначительно — на других. Это связано с тем, что можно поместить больше точек там, где значения меняются резко, и меньше точек там, где поверхность меняется плавно. Модель TIN применяется как способ хранения входных данных о поверхности и модель для решения задач на поверхностях в ГИС, допускающих работу с 3D-моделями.

TIN — это аббревиатура Triangular Irregular Network, что переводится как «нерегулярная триангуляционная сеть», а может треугольная, но если понятнее: «поверхность, сложенная из треугольников разного размера». Наверняка вы встречали трехмерные изображения, например рельефа, сформированные такими треугольниками. Иногда их можно видеть в компьютерных играх, если, конечно, кто-то из Ваших знакомых тратит на это время. Достаточно часто изображения TIN присутствуют в рекламных материалах, на практике встречаются не часто. Для построения такой поверхности нужен набор точек с тремя координатами и, скорее всего, достаточно мощные возможности по выводу на экран того, что Вы построите.

Особенностью такого объекта карты, как горизонталь рельефа, является то, что в природе его не существует. Горизонталь — это способ отобразить информацию, представленную в виде того, что часто называют полем. По сути, это не объект, а модель, которую именно в таком виде легко понимать с бумаги. При переводе этой модели отображения в поверхность TIN появляется новое качество.

6.4 Задачи пространственного анализа, решаемые современными ГИС

В данном разделе мы перечислим те задачи пространственного анализа, которые могут решаться на базе рассмотренных векторных топологических моделей (иногда на векторных моделях, о чем будет всякий раз указано).

Простейшей группой пространственных задач, доступной для решения и на векторных моделях, являются ***пространственные запросы***, позволяющие выбрать пространственные объекты как по значениям полей базы данных, так и по пространственным признакам положения (***Location***), таким как:

- принадлежность точки, линии, полигона прямоугольной или полигональной зоне, заданной координатами вершин, пересечение с границей зоны;
- попадание в буферную зону, заданную расстоянием от точки или линии.

Полигональные операции:

- Наложение (оверлей) полигонов, в том числе с сохранением результата в новый слой.
- Наложение (оверлей) полигонов и сетей, в том числе с сохранением результата (рассеченных полигонов) в новый слой.
- Снятие границ и слияние соседних полигонов (слияние собственности), слияние полигонов по признакам.

Анализ близости:

- Генерация диаграмм Вороного (полигонов < 0078 > *Tj*/Тиссена).
- Построение буферных зон со слиянием и без слияния: на множестве точек; на множестве кривых; на множестве полигонов.
- Построение буферных зон с взвешиванием факторов.

Анализ сетей:

- Поиск кратчайшего пути с вариантами взвешивания дуг и узлов, моделирующими, например, пробки на дорогах, таможенные платежи.
- Суммирование значений атрибутов (например, пробег) по элементам сети до момента достижения ограничения (растекание потока в сетях с взвешиванием дуг и узлов).
- Задача о максимальном потоке и минимальном разрезе (пропускная способность сетей и продуктопроводов).
- Размещение центров и распределение ресурсов в сети.
- Поиск пространственной смежности и ближайшего соседа.

- Геокодирование (вычисление пространственных координат зданий по их почтовому адресу).

Функции картографической алгебры (Geoprocessing):

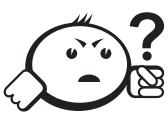
- Вычисление min, max и средних значений по множеству слоев (как правило, однородных по содержанию, но разных по времени), возведение в степень, дифференцирование.
- Переклассификация и пересборка полигонов по значениям полей базы данных (Dissolving).
- Сложение (вычитание, умножение, деление) слоев (тем) карты.
- Вырезание одного слоя другим.
- Логические комбинации слоев (тем). Пересечение — совместная встречаемость явлений.
- Слияние данных по топологической принадлежности.
- Анализ формы (вытянутость, фрагментированность).

Цифровое моделирование рельефа (возможно на векторной модели):

- Вычисление углов наклона рельефа (по TIN и регулярной сетке).
- Определение экспозиции склонов (под каким углом виден склон при заданном положении источника излучения или камеры).
- Интерполяция высот (для построения регулярной сетки по TIN).
- Определение зон видимости для точечных объектов, линейных объектов и полигонов.
- Генерация горизонталей с высотой заданной пользователем.
- Определение границ водораздела, расчет дренажной сети и оптимального пути по поверхности.
- Генерация профилей поперечных сечений по TIN и равномерной сетке.
- Вычисление объемов относительно заданной горизонтальной плоскости и минимизация вывоза грунта.

Прочие функции:

- Логические операции на множестве карт.
- Генерация случайной пространственной сети замеров явления.
- Работа с базами атрибутивной информации.
- Работа с базами геоданных.



Контрольные вопросы по главе 6

1. Какие типы объектов присутствуют на электронной карте?
2. Чем отличается полигональный объект от линейного?

3. В чем суть задачи геокодирования?
4. Что такое «буферная зона»?
5. Что такое «узел» и какие типы узлов используются?

ЗАКЛЮЧЕНИЕ

Представленный в данном пособии материал является базой для успешного изучения последующих специализированных курсов, поскольку все они неизбежно посвящены одной или ряду конкретных информационных технологий.

Автор надеется, что представленный в данном учебном пособии материал не только позволил ощутить дух и основные идеи современных информационных технологий, но и пробудил в читателях интерес к дальнейшему самосовершенствованию в данной, бурно развивающейся в настоящее время, области знаний.

ЛИТЕРАТУРА

[1] Бауэр Ф. Л. Информатика : пер. с немец. / Ф. Л. Бауэр, Г. Гооз. — М. : Мир, 1990.

[2] Советов Б. Я. Информационные технологии / Б. Я. Советов, В. В. Цехановский. — М. : Высш. шк., 2006. — 263 с.

[3] Турлапов В. Е. Геоинформационные системы в экономике : учеб.-метод. пособие / В. Е. Турлапов. — Нижний Новгород : НФ ГУ-ВШЭ, 2007. — 118 с.

ГЛОССАРИЙ

DTD — (Document Type Definition) определение типа документа, определяет набор всех возможных разметок документов описываемого типа.

HTML — (HyperText Markup Language) язык разметки гипертекстов.

OLAP — (On-Line Analysis Processing) автоматизированные системы оперативной аналитической обработки данных.

OLTP — (On-Line Transaction Processing) автоматизированные системы оперативной обработки транзакций.

SGML — (Standard Generalized Markup Language) стандартный обобщенный язык разметки.

XML — (eXtensible Markup Language) расширяемый язык разметки.

ГИС — географическая информационная система или геоинформационная система. Можно рассматривать ГИС как набор аппаратных и программных инструментов, используемых для ввода, хранения, манипулирования, анализа и отображения пространственной информации.

ГИС-платформа — многообразие ПО ГИС одного производителя.

Диаграммы потоков данных — (ДПД или DFD), технология описания асинхронного процесса преобразования информации от ее ввода в систему до выдачи пользователю.

Документальные системы — автоматизированные информационные системы, служащие для работы с документами на естественном языке.

Информационная технология — совокупность методов и способов получения, обработки, представления информации, направленных на изменение ее состояния, свойств, формы, содержания и осуществляемых в интересах пользователей.

Монитор транзакций — (TPM — transaction processing monitor). Мониторы транзакций выполняют две основные функции: динамическое распределение запросов в системе (выравнивание нагрузки) и оптимизация числа выполняющихся серверных приложений.

ПО — программное обеспечение.

Разметка документа — некоторая метаинформация, позволяющая определить структуру документа и его внешнее представление.

Разметка или кодирование (encoding) — любой метод выявления интерпретации текста.

Редакторы документов — программы для обработки текстов, имеющих структуру.

Редакторы текстов — программы рассчитанные на редактирование гладкого текста или программ на том или ином языке программирования.

Технология — наука о законах производства материальных благ, содержащая три такие основные части, как идеология (принципы производства), орудия труда (станки, машины, агрегаты) и кадры, владеющие профессиональными навыками.

Транзакция — неделимая с позиции воздействия на БД последовательность операции манипулирования данными.

Фактографические системы — автоматизированные информационные системы, оперирующие фактическими сведениями, представленными в виде специальным образом организованных совокупностей формализованных записей данных.

Язык разметки — набор соглашений о разметке, используемых в комплексе для кодирования текстов.

Учебное издание

Жуковский Олег Игоревич

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ
И АНАЛИЗ ДАННЫХ

Учебное пособие

Корректор Осипова Е. А.
Компьютерная верстка Перминова М. Ю.

Подписано в печать 19.02.14. Формат 60x84/8.
Усл. печ. л. 15,35. Тираж 100 экз. Заказ

Издано в ООО «Эль Контент»
634029, г. Томск, ул. Кузнецова д. 11 оф. 17
Отпечатано в Томском государственном университете
систем управления и радиоэлектроники.
634050, г. Томск, пр. Ленина, 40
Тел. (3822) 533018.