УДК 004.2

Г.П. Токмаков

ПРЕДСТАВЛЕНИЕ И ОБРАБОТКА ИНФОРМАЦИОННЫХ РЕСУРСОВ В ФУНКЦИЯХ УПРАВЛЕНИЯ АВТОМАТИЗИРОВАННЫХ СИСТЕМ. АНАЛИЗ ОБЩЕЙ СТРУКТУРЫ ФУНКЦИИ УПРАВЛЕНИЯ И ПОСТАНОВКА ЗАДАЧИ

Токмаков Геннадий Петрович, *доктор технических наук, окончил радиотехнический факультет Ульяновского политехнического института. Главный научный сотрудник ФНПЦ ОАО «НПО «Марс». Профессор кафедры «Вычислительная техника» Ульяновского государственного технического университета. Имеет монографию, учебные пособия, статьи и изобретения в области разработки моделей данных и систем искусственного интеллекта. [e-mail: mars@mv.ru].*

Аннотация

В статье рассматриваются проблемы представления и обработки информационных ресурсов (ИР) автоматизированных систем (АС) и формализации компонентов функций управления современных АС. Цель этой формализации – устранение тиражирования практически одинаковых программных модулей, обеспечивающих доступ к данным, путем их унификации. Рассматриваемая проблема является довольно объемной, и ее решение предлагается изложить в серии статей.

В данной статье, которая является первой из серии, выполнены общий анализ и формальное описание компонентов функций управления на трех уровнях: обработки данных, приложений и пользовательского интерфейса. В ходе этого анализа выявляются формализованные и неформализованные составляющие функций управления и формулируется постановка задачи решения проблемы унификации программных модулей, обеспечивающих доступ к данным ИР.

В следующих статьях этой серии будет реализована формализация данных и приложений на всех уровнях функций управления и на ее основе будет разработана концептуальная модель данных, обеспечивающая высокоуровневое представление ИР в терминах предметной области. Процедурная составляющая этой модели должна обеспечить унификацию процессов формирования выражений запросов для выполнения операций доступа к данным ИР.

Ключевые слова: функция управления, информационные ресурсы, уровень управления данными, уровень приложений, уровень представления, базы данных, базы метаданных, унификация приложений.

REPRESENTATION AND PROCESSING OF INFORMATION RESOURCES IN AUTOMATED SYSTEM CONTROL FUNCTIONS

Gennady Petrovich Tokmakov, *Doctor of Engineering; graduated from the Faculty of Radio Engineering of Ulyanovsk Polytechnic Institute; Chief Staff Scientist of FRPC OJSC 'RPA 'Mars'; Professor at the Department of Computer Science of Ulyanovsk State Technical University; an author of a monograph, text-books, articles, and inventions in the field of development of data models and artificial intelligence systems. e-mail: mars@mv.ru.*

Abstract

The article examines issues of representation and processing of information resources of an automated system and issues of formalization of up-to-date automated system control function components. This formalization is intended to eliminate replication of almost identical software modules that ensure data access through its unification. The issue under study is rather big, and its solution is offered to be discussed in a series of articles.

In this article, first of the series, we performed common analysis and formal description of control function components at three levels: data, application and user interface processing. While analyzing, we find out formalized and nonformalized control function components, and we formulate a statement to solve the problem of unification of software modules ensuring access to information resource data.

The following articles of this series will have data and application formalization implemented at all control function levels. Based on this formalization, we will develop a concept data model ensuring high-level information resource representation in terms of the field area. The procedure component of this model will ensure unification of processes to generate call expressions to perform information resource data access.

Key words: control function, information resources, data control level, application level, representation level, databases, metadata bases, application unification.

Введение

Информационные ресурсы (ИР) современных автоматизированных систем (АС) преимущественно организованы в базы данных (БД), реализованные на основе реляционной модели и управляемые СУБД. Технология БД, основанная на реляционной модели, привела к резкому снижению трудоемкости разработок приложений и дала мощный импульс развитию информационных систем. Такое снижение трудоемкости разработок объясняется тем, что в дореляционный период для манипуляции с данными приходилось писать процедуры доступа, «жестко привязанные» к обрабатываемым данным (см. рис. 1а), а с появлением СУБД эти процедуры были заменены на выражения SQL-запросов, обеспечивающих выполнение соответствующих операций с данными (см. рис. 1б). Эти запросы вызывают унифицированную процедуру доступа из состава СУБД, которая, используя метаданные, осуществляет доступ к данным в физической памяти машины.

Использование СУБД для обработки данных привело к изменению состава и структуры программного обеспечения (ПО) АС, основным назначением которых является автоматизация выполнения функций управления (ФУ) в различных предметных областях (ПрО). Под ФУ в АС понимается деятельность должностного лица (ДЛ) по использованию и формированию ИР для выработки управляющего воздействия на изменение состояния системы или определения состояния системы [1]. При этом под ИР системы будем понимать совокупность сведений об элементах самой системы, внешней среде и внешних системах, необходимых ДЛ для выполнения конкретных ФУ.

Характерной особенностью программ, реализующих ФУ, ИР которых управляются с помощью СУБД, является тесное взаимодействие с БД в ходе их функционирования путем выполнения стандартных действий по считыванию и записи данных [2]. При этом в приложениях, реализующих ФУ, можно выделить фрагменты кода, связанные с формированием выражений запросов для:

- извлечения из базы используемых данных ИР;
- записи в базу сформированных данных ИР.

Эти фрагменты также можно рассматривать как процедуры доступа к данным ИР, но реализованные, в отличие от процедур доступа дореляционного периода (см. рис. 1а), на более высоком уровне абстракции с использованием языка SQL [3].

Согласно особенностям описанной технологии, для каждого формируемого ИР АС, реализующего заданное управляющее воздействие на состояние ПрО или информирующего о ее состоянии, требуется создавать отдельные процедуры, которые функционируют по схожим, но неидентичным алгоритмам. Различие алгоритмов функционирования процедур доступа к данным ФУ вызваны, прежде всего, характером выполняемой операции доступа и структурой данных ИР.

В течение ряда последних десятилетий технология создания АС на основе БД доказала свою эффективность и на сегодняшний день стала де факто стандартной. Это привело к широкому распространению описанной технологии разработки приложений АС, в результате чего существенно выросло число автоматизируемых ФУ. Соответственно выросло и количество типов ИР, которые в современных АС уже исчисляются сотнями и тысячами.

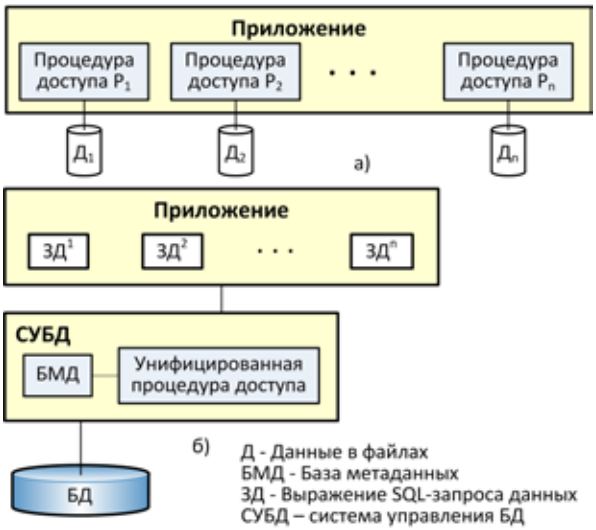

Рис. 1. Схемы доступа к данным без (а) и с помощью (б) СУБД

В этих условиях практика разработки ПО АС, приводящая к тиражированию сотен и тысяч практически одинаковых программных модулей, обеспечивающих доступ к данным ИР, требует слишком много трудозатрат на реализацию и усложняет перспективы дальнейшего развития АС в целом.

Таким образом, на новом витке развития информационных систем мы пришли к той же проблеме, связанной с необходимостью унификации процедур доступа к данным ИР, но уже на другом, более высоком уровне представления данных.

Для решения этой проблемы в данной статье проанализируем особенности реализации ФУ на современном этапе развития технологии разработки АС и сформулируем постановку задачи, направленной на унификацию процедур доступа к данным ИР на уровне ФУ АС.

1 Общая структура функций управления современных АС

Как показывает многолетний опыт управления предприятиями и организациями, существует универсальный и конечный перечень типовых ФУ, необходимых для реализации свойства управляемости любой сложной организационно-технической системы любого назначения [1]. Каждая такая ФУ направлена на формирование управляющего воздействия на ПрО или информирование о ее состоянии путем использования и выработки соответствующих ИР. Поэтому будем считать, что они определены на данных, составляющих ИР решения данной задачи управления, а множество ФУ, реализуемых АС, опишем выражением

$$F=f_1(r_1),\ldots,f_\varphi(r_\varphi),\ldots,f_\Phi(r_\Phi),$$

где r_φ – обозначает данные ИР, используемые и формируемые в ФУ $f_\varphi(\ )$, для всех $\varphi=1,\ldots,\Phi$. Следует отметить, что символом r_φ обозначается тип ИР, представленный множеством экземпляров $r_\varphi^{j(\varphi)}, j(\varphi)=1(\varphi),\ldots,J(\varphi)$.

Таким образом, ФУ АС представлены приложениями, входными и выходными аргументами которых являются данные ИР r_φ, $\varphi=1,\ldots,\Phi$. ИР АС в целом представлены подмножествами данных r_φ, в совокупности описывающих всю ПрО в БД D и удовлетворяющих условию $D=r_1\cup\ldots\cup r_\varphi\cup\ldots\cup r_\Phi$, причем $r_1\cap\ldots\cap r_\varphi\cap\ldots\cap r_\Phi\neq\varnothing$.

В современных АС, реализованных на основе последних достижений в области SQL- и Web-технологий, приложения, рассматриваемые ранее как монолитные, разделяются на логические части, представленные следующими уровнями:

- уровнем управления данными, осуществляющим доступ к данным физической БД;
- уровнем приложений или бизнес-логики, реализующим алгоритм выполнения задачи;
- уровнем представления или клиента, содержащим логику взаимодействия ДЛ с АС.

Под влиянием этих технологий сформировалась трехзвенная архитектура АС, в которой каждая ФУ $f_\varphi(r_\varphi)$, $\varphi=1,\ldots,\Phi$ представляется на трех уровнях, что можно записать как

$$f_\varphi(r_\varphi)=f_c^\varphi(r_\varphi),f_p^\varphi(r_\varphi),f_d^\varphi(r_\varphi),$$

где $f_d^\varphi(r_\varphi)$ – фрагмент ФУ $f_\varphi(r_\varphi)$, осуществляющий управление данными;

$f_c^\varphi(r_\varphi)$ – клиентская составляющая, реализующая пользовательский интерфейс;

$f_p^\varphi(r_\varphi)$ – составляющая бизнес-логики, в которой реализуется алгоритм выполнения ФУ.

В данной архитектуре множество ФУ, реализуемых АС, можно представить выражением

$$F=F_c,F_p,F_d,$$

где $F_c=\left\{f_c^\varphi(r_\varphi)\right\}$, $F_p=\left\{f_p^\varphi(r_\varphi)\right\}$, $F_d=\left\{f_d^\varphi(r_\varphi)\right\}$, для всех $\varphi=1,\ldots,\Phi$.

Для определения степени формализации перечисленных уровней ФУ АС проанализируем каждую ее составляющую по отдельности.

Уровень управления данными. Использование СУБД или SQL-процессоров на уровне данных привело к унификации фрагментов ФУ, реализующих управление данными [4]. При этом множество ФУ данными $F_d=\left\{f_d^\varphi(r_\varphi)\right\}$, используемых в дореляционных АС, заменили SQL-процессором или СУБД, содержащей унифицированную процедуру доступа $F_d=f_d\left(q\left(r_\varphi^j,d_\mu\right)\right)$. В данном выражении:

$f_d(\ )$ обозначает СУБД, в которой реализованы унифицированные процедуры для всех типов операций доступа к данным, заменяющие множество процедур по доступу к данным соответствующих ИР $F_d=\left\{f_d^\varphi(r_\varphi)\right\}$, для всех $\varphi=1,\ldots,\Phi$;

r_φ^j – j-й экземпляр ИР r_φ для $j=1,\ldots,J$;

d_μ – метаданные, описывающие ИР r_φ (подробнее о БМД в целом см. в разделе 2.3);

$q\left(r_\varphi^j,d_\mu\right)$ – SQL-выражение запроса к данным ИР r_φ, выполняемого СУБД путем использования сведений из БМД d_μ.

Обратите внимание на то, что в выражении, обозначающем функцию управления данными, исчез верхний индекс, указывающий на зависимость от ФУ, т. е. от ИР, обрабатываемого ею. Вместо этого в числе аргументов функции появился дополнительный элемент d_μ, обозначающий метаданные БД, реализованные в СУБД. При этом был использован следующий прием: изменчивость структур данных различных типов, влияющая на алгоритмы функции их обработки, была «вынесена за скобку», т. е. реализована в виде метаданных и перешла в разряд аргументов, что и позволило унифицировать функции обработки данных.

Уровень приложений. В отличие от уровня данных, в силу творческого характера деятельности ДЛ по управлению, определить формальное описание бизнес-логики на сегодняшний день не представляется возможным [1].

С другой стороны, существенная часть алгоритмов ФУ связана с реализацией операций доступа к входным и выходным составляющим ИР. Эта часть бизнес-логики обеспечивает формирование выражений SQL-запросов для уровня данных, которые зависят от типа операции доступа и структур обрабатываемых данных, имеющих формальные спецификации. Поэтому справедливо предположить, что алгоритмы, реализующие формирование выражений запросов, имеют перспективу для формализации при соответствующем представлении сведений о спецификации синтаксиса операторов выражений SQL-запросов для всех видов операций доступа и структурах обрабатываемых данных.

Таким образом, если на сегодняшний день проблема формализации составляющей бизнес-логики в целом нерешаема, то можно рассмотреть возможность ее частичного решения. С этой целью бизнес-логику ФУ предлагается разбить на две части:

– первая часть связана с решением собственно задач управления и описывает последовательность действий ДЛ по выработке управляющих воздействий;

– вторая часть решает задачи по извлечению входных данных из и записи выходных данных в БД путем формирования выражений SQL-запросов для последующей реализации соответствующих операций доступа к данным в СУБД.

Результат такого разделения можно зафиксировать с помощью выражения

$$f_p^{\varphi}\left(r_{\varphi}\right)=\tilde{f}_p^{\varphi}\left(r_{\varphi},\hat{f}_p^{\varphi}\left(r_{\varphi}\right)\right), \quad (1)$$

где $\tilde{f}_p^{\varphi}\left(\ \right)$ – выражение для обозначения функций, реализующих алгоритмы ФУ, использующих данные r_{φ} из БД, а также формирующих данные для записи в БД. Диакритический знак '~' над символом f означает, что эту составляющую мы определяем как неформализуемую;

$\hat{f}_p^{\varphi}\left(r_{\varphi}\right)$ – выражение для обозначения функций, реализующих алгоритмы формирования SQL-запросов операций доступа для выполнения в СУБД, используемых в ФУ $\tilde{f}_p^{\varphi}\left(r_{\varphi}\right)$. Эта составляющая, называемая приложением баз данных (ПБД), далее будет проанализирована с целью определения путей ее формализации. Диакритический знак '^' над символом f означает, что в данной функции содержатся возможности формализации и последующей унификации, которые еще предстоит реализовать, т. е. представить в виде $f_p\left(r_{\varphi}\right)$.

Если составные части $f_p^{\varphi}\left(r_{\varphi}\right)$ определяются как отдельные процедуры, а для процедур $\hat{f}_p^{\varphi}\left(r_{\varphi}\right)$ в случае необходимости реализована возможность вызова из процедур $\tilde{f}_p^{\varphi}\left(r_{\varphi}\right)$, то выражение (1) можно представить в виде $f_p^{\varphi}\left(r_{\varphi}\right)=\tilde{f}_p^{\varphi}\left(r_{\varphi}\right),\hat{f}_p^{\varphi}\left(r_{\varphi}\right)$. В этом случае, уровень бизнес-логики для АС в целом, можно представить выражением

$$F_p=\left\{\tilde{f}_p^{\varphi}\left(r_{\varphi}\right),\hat{f}_p^{\varphi}\left(r_{\varphi}\right)\right\} \text{ для всех } \varphi=1,\ldots,\Phi.$$

Уровень представления. На уровне представления в области формализации имеются отдельные результаты. В частности, использование XSL-процессора на уровне пользовательского интерфейса привело к унификации фрагментов ФУ, осуществляющих отображение ИР в виде аутентичных документов [5]. Однако XSL-процессоры не охватывают аспекты функционирования клиентской составляющей в части ввода информации ДЛ при выполнении операций доступа.

Поэтому для формализации процедур в части ввода информации было предложено использовать xForms-процессор, который для визуализации элементов управления (ЭУ) экранных форм (ЭФ) использует данные xHTML-файлов, реализованных по заданному стандарту. Но этот стандарт оказался довольно сложным при реализации и пока еще не получил широкого распространения в виде простых и удобных xForms-процессоров.

Таким образом, на клиентском уровне мы имеем ситуацию, в которой:

- с одной стороны, XSL-процессоры, хотя и просты в применении, но решают лишь незначительную часть проблем;
- с другой стороны, xForms-процессоры, с помощью которых можно решить практически все задачи клиентского уровня, слишком сложны и выгода от их применения пока еще не очевидна.

По перечисленным причинам формальные модели, унифицирующие клиентскую составляющую, сложно применить в реальных разработках и в современных АС они практически не используются. В результате, данный уровень архитектуры АС по-прежнему реализовывается в виде привычных ЭФ, обеспечивающих выполнение отдельных видов операций по доступу к данным ИР отдельных типов.

С учетом изложенного клиентскую составляющую ФУ $f_{\varphi}\left(r_{\varphi}\right)$ будем описывать без учета данных технологий, т. е. рассматриваем как неформализованную и обозначаем выражением $F_c=\left\{\tilde{f}_c^{\varphi}\left(r_{\varphi}\right)\right\}$.

2 Анализ уровня данных

Проведенный анализ структуры ФУ АС показал, что из трех его уровней только уровень данных полностью формализован, что позволило унифицировать процедуры обработки данных на этом уровне. В данном разделе приведем описание формальной модели уровня данных и определим механизм унификации процедур обработки данных, основанный на этой модели, с целью постановки задачи его применения для унификации процедур обработки данных на уровнях представления и приложений ФУ АС.

В дореляционный период данные хранились в файлах и сведения об их структуре нигде не фиксировались. Поэтому объективно существующие связи между элемен-

тами данных приходилось учитывать в программном коде. Стандартов представления структур данных в то время еще не существовало и программисты описывали и обрабатывали эти связи по своему усмотрению.

С определением понятия структур данных разработчики реляционной модели данных обнаружили, что любая структура порождает класс эквивалентных реализаций данных, обработка которых, т. е. реализация допустимых преобразований на множестве эквивалентных реализаций, может осуществляться по общим процедурам. Это открытие привело к разработке унифицированных процедур обработки данных, описываемых схемой данных, хранимой в БМД.

2.1 Схема БД

Схема БД представляет собой описание перечня таблиц, каждая из которых описывается набором атрибутов. Между частью таблиц БД устанавливаются связи, определяющие зависимости некоторых полей соответствующих таблиц.

Формально схему БД можно описать с помощью следующего выражения:

$$S = \left| \begin{matrix} \left\{ T_\theta \left(a^\theta_{1(\theta)} \dots a^\theta_{\alpha(\theta)} \dots a^\theta_{\mathrm{A}(\theta)} \right) \right\}, \\ \left\{ b_1, \dots, b_\beta, \dots, b_\mathrm{B} \right\} \end{matrix} \right|, \quad (2)$$

где каждая таблица T_θ $(\theta = 1, \dots, \Theta)$ в БД представлена атрибутами $a^\theta_{\alpha(\theta)}$ $(\alpha(\theta) = 1(\theta), \dots, \mathrm{A}(\theta))$;

между таблицами T_θ БД установлены связи $b_1, \dots, b_\beta, \dots, b_\mathrm{B}$, выраженные соответствием значений первичного ключа a^ϱ_{pk} родительской таблицы T_ϱ и вторичного ключа a^δ_{fk} дочерней таблицы T_δ, т. е. $b_\beta : a^\varrho_{pk} = a^\delta_{fk}$, для $\varrho, \delta \in 1, \dots, \Theta$.

В обозначениях атрибутов в выражении (2) используются верхние и нижние индексы для указания номера таблицы $\theta = 1, \dots, \Theta$ и номера атрибутов $\alpha = 1, \dots, \mathrm{A}$, соответственно. При этом номера атрибутов в одних и тех же позициях в зависимости от таблиц принимают различные значения. Поэтому мы вынуждены указывать обозначения таблиц и в верхнем, и нижнем индексах.

Для снижения избыточности в обозначениях, когда значения одних индексов зависят от значений других индексов, предлагается использовать нотации вида $\alpha(.)$, где точка в данном конкретном случае заменяет идентификатор таблицы $\theta = 1, \dots, \Theta$, используемый в верхнем индексе. Описанный прием будем использовать не только для обозначения номеров атрибутов, но и в обозначениях связей и записей таблиц. При этом нужно иметь в виду, что данные сокращения действуют только в заданном контексте. Например, $\alpha(.)$ в выражении $a^1_{\alpha(.)}$ означает $\alpha(1)$, в выражении $a^\theta_{\alpha(.)} - \alpha(\theta)$, а в выражении $a^\Theta_{\alpha(.)}$ – $\alpha(\Theta)$.

Еще один прием упрощения записей индексов связан с удалением скобок, используемых для обозначения иерархических зависимостей. Например, в выражениях вида $a^{\tau(\rho)}$ индексы предлагается писать без скобок, т. е. индекс $\tau(\rho)$, означающий «таблица τ ИР типа ρ», пишется как $\tau\rho$, а выражение $\epsilon(\tau(\rho))$, означающее «экземпляр ϵ таблицы τ ИР типа ρ», – как $\epsilon\tau\rho$.

В случаях, когда использование предлагаемой нотации может вызвать неоднозначное толкование, как, например, в выражении ${}^{\tau\varphi}_{\alpha(.)}d^{q\varphi}_{\xi(.)}$, необходимо привести уточняющие определения вида $\alpha(.) = \alpha\tau\varphi = 1\tau\varphi, \dots, \mathrm{A}\tau\varphi$, $\xi(.) = \xi q\varphi = 1q\varphi, \dots, \Xi q\varphi$.

2.2 База данных

С помощью схем S определяются конкретные данные, удовлетворяющие схеме. При этом множество различных реализаций схемы определяет БД D или, другими словами, инвариантная структура S порождает класс реализаций, называемый БД. БД D, описываемая схемой S, состоит из записей таблиц $\{T_\theta\}$ для всех $\theta = 1, \dots, \Theta$, и представляется матрицами

$$T_\theta = \left\| {}^{w(.)}v^\theta_{1(.)} \dots {}^{w(.)}v^\theta_{\alpha(.)} \dots {}^{w(.)}v^\theta_{\mathrm{A}(.)} \right\|,$$

для всех $w(.) = w(\theta) \in 1(\theta), \dots, W(\theta)$.

В дальнейшем для краткости, записи ${}^{w(.)}v^\theta_{1(.)} \dots {}^{w(.)}v^\theta_{\alpha(.)} \dots {}^{w(.)}v^\theta_{\mathrm{A}(.)}$ таблиц T_θ $(\theta = 1, \dots, \Theta)$ с номером $w(.)$ будем обозначать символом $z^\theta_{w(.)}$.

Связи b_β в БД представлены множеством пар записей $\left\{ z^\varrho_{w(.)} \right\}$ родительской таблицы T_ϱ и записей $\left\{ z^\delta_{w(.)} \right\}$ дочерней таблицы T_δ. Пары записей связи определяются соответствием значений первичного ключа $a^\varrho_{pk(.)}$ таблицы T_ϱ и вторичного ключа $a^\delta_{fk(.)}$ таблицы T_δ, где $\varrho, \delta \in 1, \dots, \Theta$. Если допустить, что в качестве первичных ключей выбираются первые атрибуты, а в качестве вторичных ключей – последние атрибуты таблиц, то полное определение связи между записями $z^\varrho_{w(.)}$ и $z^\delta_{w(.)}$ таблиц T_ϱ и T_δ можно представить выражением

$$b_\beta \left(\varrho, \delta, a^\varrho_{1(.)}, a^\delta_{A(.)} \right) : \left({}^{w(.)}v^\varrho_{1(.)} = {}^{w(.)}v^\delta_{\mathrm{A}(.)} \right).$$

В современных реляционных СУБД поддерживаются связи типов:

- «$1 : 1$», когда каждая запись родительской таблицы соединяется только с одной записью дочерней таблицы;
- «$1 : m$», когда каждая запись родительской таблицы может соединяться с m записями дочерней таблицы;
- «$n : m$», когда каждая запись родительской таблицы может соединяться с m записями дочерней таблицы, а каждая запись дочерней таблицы может соединяться с n записями родительской таблицы.

Использование связей типа «$1 : 1$» встречается редко, и, как правило, такие связи заменяются объединениями таблиц. А те редкие случаи, в которых использование связей типа «$1 : 1$» целесообразно, можно рассматривать как частный случай связи типа «$1 : m$», где $m = 1$. Связи типа

«$n:m$» используются только на концептуальном уровне, а на физическом – заменяются двумя связями «$1:n$» и «$1:m$». Поэтому в реляционных БД в основном используются связи типа «$1:m$», которые можно описать с помощью выражения

$$b_\beta: z^{\varrho}_{w(.)} \to z^{\delta}_{w1(.)}, \ldots, z^{\delta}_{wm(.)}.$$

При этом дочерние записи $z^{\delta}_{w1(.)},\ldots,z^{\delta}_{wm(.)}$ могут иметь свои дочерние записи, описываемые другой связью из заданного множества. В результате описания всех связей $b_1,\ldots,b_\beta,\ldots,b_{\mathrm{B}}$ на уровне БД данные отдельных таблиц T_θ, $\theta=1,\ldots,\Theta$ превращаются в БД.

Следует отметить, что БД D, определяемая схемой S, «зримо» представлена только записями $z^{\theta}_{w(.)}$ таблиц T_θ и описывается выражением

$$D=\left\{T_\theta \left| {}^{w(.)}v^{\theta}_{1(.)}\ldots {}^{w(.)}v^{\theta}_{\alpha(.)}\ldots {}^{w(.)}v^{\theta}_{\mathrm{A}(.)}\right|\right\}$$

для всех T_θ $(\theta=1,\ldots,\Theta)$ и их записей $w(.)=w(\theta)\in 1(\theta),\ldots,W(\theta)$. Связи $\{b_\beta\}$ $(\beta=1,\ldots,\mathrm{B})$ между записями таблиц БД выявляются путем сопоставления значений атрибутов, описанных в схеме S.

2.3 База метаданных

Выше было отмечено, что в современных СУБД обращение к данным осуществляется не непосредственно, а через описание схемы БД. Для этого сведения о структуре данных, описанных в виде схемы БД, должны быть определенным образом зафиксированы в физической памяти машины. В современных СУБД для этой цели используется специальный раздел БД, называемый БМД или СК, в котором определены следующие фиксированные структуры:

- таблица T_μ, записи $\left(t^{\theta}_1,\ldots,t^{\theta}_\psi,\ldots,t^{\theta}_\Psi\right)$ которой содержат сведения о наименованиях, идентификаторах и других свойствах таблиц множества $\{T_\theta\}$ для всех $\theta=1,\ldots,\Theta$;
- таблица A_μ, записи $\left(a^{\alpha\theta}_1,\ldots,a^{\alpha\theta}_\varepsilon,\ldots,a^{\alpha\theta}_{\mathrm{E}}\right)$ которой содержат сведения об атрибутах $a^{\theta}_{\alpha(.)}$ таблиц из множества $\{T_\theta\}$ для всех $\alpha(.)=1(.),\ldots,\mathrm{A}(.)$, $\theta=1,\ldots,\Theta$;
- таблица B_μ, записи $\left(b^{\beta}_1,\ldots,b^{\beta}_\eta,\ldots,b^{\beta}_{\mathrm{H}}\right)$ которой содержат сведения о связях из множества $\{b_\beta\}$ для всех $\beta=1,\ldots,\mathrm{B}$. В записях этой таблицы содержатся данные о родительской T_ϱ и дочерней T_δ таблицах связи и их первичном a^{ϱ}_{pk} и вторичном a^{δ}_{fk} ключах, соответственно.

Таким образом, СК или БМД в части описания схемы БД можно представить выражением $d_\mu=(T_\mu,A_\mu,B_\mu)$, в котором структуры для представления сведений о произвольных таблицах БД, их атрибутов и связей четко определены. Это позволяет написать унифицированные процедуры считывания данных из этих структур, а фрагменты программного кода с конкретными данными заменить выражениями вызова этих процедур.

В результате программный код, содержащий вместо конкретных данных унифицированные процедуры вызова необходимых данных, определяемых атрибутами этих процедур, также становится унифицированным и пригодным для обработки широкого диапазона данных, описываемых схемой БД, представленной в БМД d_μ.

Следует отметить, что на сегодняшний день структуры для хранения метаданных БД не стандартизованы и разработчики СУБД определяют их по своему усмотрению. Поэтому приведенное в разделе описание структуры СК отражает лишь идею, а не конкретную ее реализацию.

Разработчики реляционной модели данных сделали очень важное открытие в технологии разработки программ: они вывели из процедур доступа к данным сведения о структуре данных и представили их в виде декларативных данных с фиксированной структурой.

В соответствии с данным подходом в реляционных СУБД информация о структуре данных ПрО, оформленная в виде схемы БД S, хранится в БМД $d_\mu=(T_\mu,A_\mu,B_\mu)$ и используется при реализации операций доступа к данным D.

3 Постановка задачи

Наша задача в данной статье: воспользоваться описанным подходом для формализации двух оставшихся уровней ФУ. Этот подход предусматривает выведение из кода процедур доступа к данным элементов, описывающих структурные особенности обрабатываемых данных, и реализацию их в виде метаданных, хранимых в фиксированных структурах памяти.

С этой целью предлагается:

- проанализировать данные и процедуры их обработки и представления, используемые в ФУ;
- выявить источники изменчивости приложений при выполнении практически одинаковых по смыслу алгоритмов доступа, различающихся лишь составом и структурой обрабатываемых данных;
- предложить конструктивные подходы, направленные на унификацию процедур доступа к данным на уровне ФУ АС.

Общая структура ФУ АС, описанная в разделе 1 на трех уровнях выражением

$$F=F_c,F_p,F_d,$$

где $F_c=\left\{\tilde{f}^{\varphi}_c\left(r_\varphi\right)\right\}$, $F_p\left\{\tilde{f}^{\varphi}_p\left(r_\varphi\right),\hat{f}^{\varphi}_p\left(r_\varphi\right)\right\}$, $F_d=f_d\left(q\left(r_\varphi,d_\mu\right)\right)$, для всех $\varphi=1,\ldots,\Phi$, приведена на рисунке 2.

На ***уровне данных*** для управления данными реляционной модели были созданы СУБД, предоставляющие в распоряжение пользователей высокоуровневый язык SQL, с помощью которого создается БД и формулируются запросы для выполнения операций доступа к данным базы в физической памяти.

СУБД, описываемая выражением $f_d\left(q\left(r^j_\varphi,d_\mu\right)\right)$, реализована в виде совокупности унифицированных процедур, выполняющих операции доступа к произвольным данным базы. При этом в качестве входных аргументов $f_d()$

используются SQL-выражения $q\left(r_{\varphi}^{j}, d_{\mu}\right)$, формируемые либо пользователями, либо программными средствами.

Уровень приложений в предыдущем разделе был разделен на две составляющие с целью исключения из рассмотрения заведомо не формализуемой составляющей $\tilde{f}_{p}^{\varphi}\left(r_{\varphi}\right)$ и выделения из него функций $\hat{f}_{p}^{\varphi}\left(r_{\varphi}\right)$, реализующих алгоритмы формирования выражений SQL-запросов операций доступа для выполнения в СУБД. При этом необходимо отметить следующее: в выражении запроса $q\left(r_{\varphi}^{j}, d_{\mu}\right)$, формируемом ПБД, определяется операция доступа к фрагменту БД r_{φ}^{j}, тогда как БМД d_{μ} описывает всю БД D в целом.

Отсутствие описания структур фрагментов БД, соответствующих ИР, приводит к тому, что данную работу приходится выполнять на этапе разработки ПБД ФУ. Процедура фрагментации БД D в виде совокупности ИР $r_{\varphi}(\varphi=1,\ldots,\Phi)$ на сегодняшний день не формализована и даже не регламентирована. Программист, реализующий этот процесс, не располагает никакими руководящими документами, кроме описания схемы БД в целом.

Кроме этого, на характер выражения SQL-запроса существенное влияние оказывает и вид выполняемой операции доступа $q=\{ins,upd,del,sel\}$, где ins обозначает операцию вставки данных ($insert$), upd – операцию обновления ($update$), del – операцию удаления ($delete$), sel – операцию выборки ($select$).

Поэтому на современном этапе развития технологии разработки АС для каждой функции $\hat{f}_{p}^{\varphi}\left(r_{\varphi}\right)$ уровня приложений приходится реализовывать набор процедур для всех выполняемых операций доступа q, что для общего случая можно записать выражением

$$\hat{f}_{p}^{\varphi}\left(r_{\varphi}\right)=\left\{{}^{q}\hat{f}_{p}^{\varphi}\left(r_{\varphi}\right)\right\}.$$

Наша задача: представить данную совокупность ПБД в виде унифицированной функции

$$q\left(r_{\varphi}^{j}, d_{\mu}\right)=f_{p}\left(r_{\varphi}^{j}, d_{q}^{\varphi}, d_{s}^{\varphi}\right),$$

формирующей выражения SQL-запросов $q\left(r_{\varphi}^{j}, d_{\mu}\right)$, для реализации заданных операций доступа, и использующей данные:

- r_{φ}^{j} об экземпляре ИР r_{φ};
- d_{q}^{φ} о синтаксисе выражений SQL-запросов выполняемых операций доступа, размещенные в БМД;
- d_{s}^{φ} о структуре фрагмента ИР r_{φ}, размещенные в БМД.

Назначение ***уровня представления*** заключается в обеспечении ввода и вывода данных ИР r_{φ} в чувственно воспринимаемом виде v_{φ} и их взаимно обратного преобразования $v_{\varphi}^{j} \leftrightarrow r_{\varphi}^{j}$ в логическую форму r_{φ}^{j}, пригодную для использования в ПБД $\hat{f}_{p}^{\varphi}\left(r_{\varphi}\right)$. С учетом этого обстоятельства перепишем выражение клиентской составляющей ФУ в виде:

$$r_{\varphi}=\tilde{f}_{c}^{\varphi}\left(v_{\varphi}\right). \quad (3)$$

С целью преобразования визуальных данных v_{φ} в логические данные r_{φ} и обратно используются ЭФ, представляющие собой выделенные фрагменты экрана с:

- пустыми полями, оставленными для заполнения пользователем при вводе данных, т. е. при преобразовании v_{φ} в r_{φ};
- полями, заполненными данными, извлеченными из базы, при просмотре данных, т. е. при преобразовании r_{φ} в v_{φ}.

ЭФ могут допускать различный тип входной информации, соответствующей данным ИР r_{φ}, содержать командные кнопки, переключатели, выпадающие меню или списки для выборки используемых данных. Современные технологии компонентного программирования предоставляют в распоряжение разработчиков так называемые ЭУ, представляющие собой «полуфабрикаты» для реализации на ЭФ полей ввода и отображения данных, кнопок, переключателей, выпадающих списков и других элементов визуализации со встроенной функциональностью. Данная технология на сегодняшний день является доминирующей и обеспечивает эффективную разработку программного кода для реализации ЭФ.

В основе проблем, связанных с тиражированием практически одинаковых по смыслу алгоритмов для реализации кода ЭФ, лежит все то же отсутствие описаний d_{s}^{φ} структур ИР r_{φ} $(\varphi=1,\ldots,\Phi)$. К этой проблеме добавляется отсутствие предварительно определенных сведений d_{v}^{φ} для визуализации соответствующих элементов

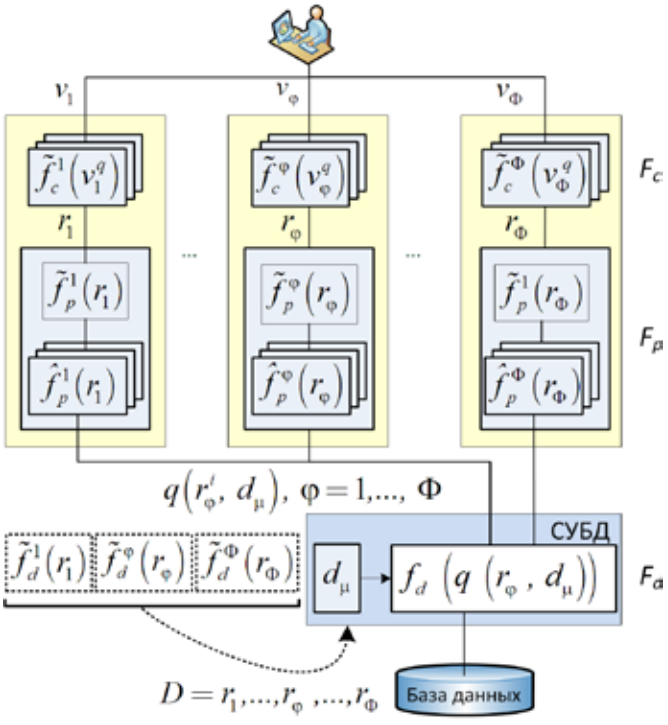

Рис. 2. Трехуровневое представление ФУ АС

ИР r_φ. Это приводит к тому, что данные для визуализации полей ввода и их наименования определяются в ходе разработки программного кода ЭФ и «встраиваются» непосредственно в программный код ЭФ.

На этом уровне наряду с проблемой зависимости алгоритмов отображения ЭФ от обрабатываемых данных имеет место еще и проблема многократного дублирования одних и тех же параметров отображения, используемых для выполнения разных операций доступа с одними и теми же данными. Таким образом, на уровне представления мы имеем дело с еще одним фактором изменчивости, и выражение (3), описывающее уровень представления ФУ, необходимо переписать с учетом этого обстоятельства и представить как:

$$\tilde{f}_c^{\varphi}\left(v_\varphi\right) \Rightarrow \left\{{}^{q}\tilde{f}_c^{\varphi}\left(v_\varphi\right)\right\}.$$

С учетом изложенного наша задача на уровне представления заключается в преобразовании этой совокупности функций, реализующих ЭФ для каждого типа ИР $\varphi = 1, \ldots, \Phi$ и вида выполняемой над ним операции доступа q, в унифицированную функцию для:

а) ввода и изменения данных

$$r_\varphi^j = f_c^{inp}\left(v_\varphi^i, d_v^\varphi, d_q^\varphi, d_s^\varphi\right),$$

формирующей логические данные об экземпляре r_φ^j ИР r_φ для ПБД $f_p\left(r_\varphi^j, d_q^\varphi, d_s^\varphi\right)$, по визуальным данным v_φ^j, введенным пользователем с ЭФ;

б) вывода данных

$$v_\varphi^j = f_c^{out}\left(r_\varphi^j, d_v^\varphi, d_q^\varphi, d_s^\varphi\right),$$

формирующей визуальные данные v_φ^j на ЭФ, по логическим данным об экземпляре r_φ^j ИР r_φ, поступившим от ПБД $f_p\left(r_\varphi^j, d_q^\varphi, d_s^\varphi\right)$.

Перечисленные функции используют данные:

- v_φ^j об экземпляре ИР r_φ, введенные пользователем с ЭФ (при вводе данных);
- r_φ^j об экземпляре ИР r_φ, поступившие от ПБД (при выводе данных);
- d_v^φ для визуализации ЭФ в целом и каждого ее ЭУ, размещенные в БМД;
- d_q^φ об особенностях выполняемых операций доступа, размещенные в БМД;
- d_s^φ о структуре фрагмента ИР r_φ, размещенные в БМД, и данные d_μ^φ о структуре фрагмента ИР r_φ, размещенные в БМД.

Таким образом, после выполнения намеченной формализации компоненты ФУ АС должны описываться выражениями:

- компонент F_c – унифицированными функциями $f_c^{inp}\left(v_\varphi^j, d_v^\varphi, d_q^\varphi, d_s^\varphi\right)$ – ввода данных в визуальной форме и $f_c^{out}\left(r_\varphi^j, d_v^\varphi, d_q^\varphi, d_s^\varphi\right)$ – вывода данных в визуальной форме;
- компонент F_p – унифицированной функцией $f_p\left(r_\varphi^i, d_q^\varphi, d_s^\varphi\right)$ – формирования выражений SQL-запросов и набором неформализованных функций $\left\{\tilde{f}_p^{\varphi}\left(r_\varphi\right)\right\}$ – реализации алгоритмов управления;
- компонент F_d – унифицированной функцией $f_d\left(q\left(r_\varphi^j, d_\mu\right)\right)$ – обработки данных в базе.

После реализации поставленной цели трехуровневое представление ФУ АС приобретает вид, показанный на рисунке 3.

Использование методов формализации данных с помощью метаданных в свое время привело к разработке реляционной модели данных, реализованной в виде такого программного продукта, как СУБД. Как было описано во введении, при этом фрагменты кода, обеспечивающие доступ к данным в памяти машины были заменены на выражения запросов на языке SQL.

ПБД ФУ также обеспечивают доступ к данным ИР, и реализация намеченной унификации в виде функции $q\left(r_\varphi^i, d_\mu\right) = f_p\left(r_\varphi^i, d_q^\varphi, d_s^\varphi\right)$ могла бы содействовать разработке модели данных, представляющей ИР ФУ в виде совокупности объектов, описываемых в терминах ПрО.

При наличии такой модели каждую ФУ $f_\varphi(r_\varphi)$ АС можно реализовать в виде компонента $\tilde{f}_p^{\varphi}\left(r_\varphi\right)$, в котором ПБД $\hat{f}_p^{\varphi}\left(r_\varphi\right)$ заменяются высокоуровневыми (в терминах объектов ПрО) запросами, для формулировки которых используются компоненты клиентского уровня $f_c^{inp}(\ )$ и $f_c^{out}(\ )$, обеспечивающие взаимодействие с пользователем при вводе и выводе необходимых данных.

На основе выполнения описанной формализации трехуровневого представления ФУ АС можно разработать высокоуровневую модель данных [6], обеспечивающую формирование запросов на выполнение операций доступа к данным в терминах ПрО.

Вопрос разработки новой модели данных, повышающей уровень представления данных и приближающей его к уровню представления данных в человеческой памяти, возникает всякий раз, когда расширение применения информационных технологий в какой-либо отрасли обостряет проблему несоответствия двух способов представления данных до критического уровня. В этом случае вопрос разработки новой модели данных становится актуальным, а текущая модель данных рассматривается как исходное низкоуровневое представление [7].

В нашем случае исходной моделью данных является реляционная, в соответствии с которой объекты ПрО представляются подсхемами БД и требуют разработки приложений для выполнения манипуляций с ними с целью поддержания актуальности их представления в машинной памяти в соответствии с изменениями, происходящими в реальном мире.

Требуется такая модель данных, которая рассматривала бы объекты ПрО как единицы обработки, для манипуляции с которыми достаточно сформулировать запрос в терминах объектов ПрО, а не разрабатывать ЭФ для ввода

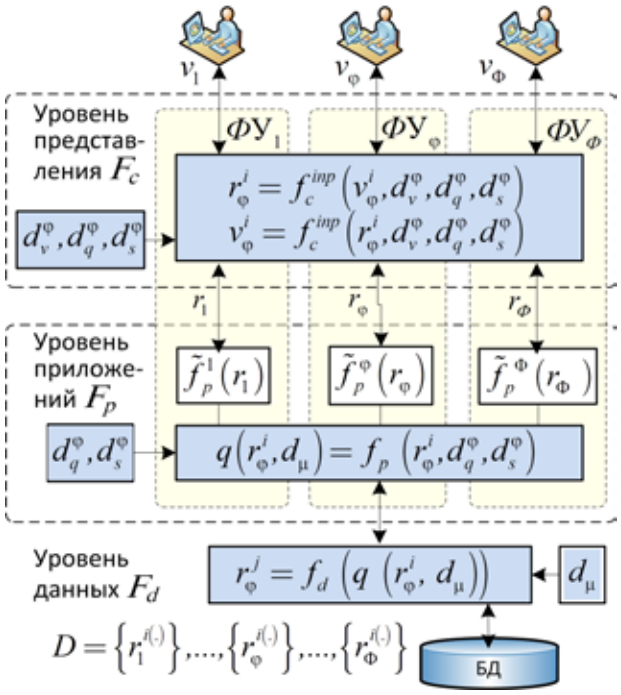

Рис. 3. Формализованное трехуровневое представление ФУ АС

и вывода данных и приложения БД для каждого типа ИР, как в настоящее время.

Итак, для решения поставленных задач необходимо:

1. Реализовать формализацию и унификацию приложений БД, выполняющих формирование SQL-выражений доступа к реляционной БД.

2. Реализовать формализацию и унификацию клиентских приложений, генерирующих ЭФ, обеспечивающие ввод данных для формирования SQL-выражений выбранной операции доступа к требуемому объекту ПрО.

3. Разработать модель данных, описывающую ИР ФУ в виде совокупности объектов ПрО, высокоуровневых операций, обеспечивающих формирование SQL-выражений доступа к ним в БД, и интерфейсной составляющей, обеспечивающей ввод пользователем необходимых данных для формирования SQL-выражений доступа.

Заключение

В статье рассмотрена проблема формализации компонентов ФУ современных АС для устранения тиражирования практически одинаковых программных модулей, обеспечивающих доступ к данным.

С этой целью в статье проанализирована ФУ, рассматриваемая как деятельность ДЛ по формированию ИР. При этом ФУ разделена на три уровня и отмечено, что только один из этих уровней, а именно уровень данных, формализован и реализован в виде унифицированной функции. Остальные два уровня формализованы только частично.

Для выявления механизма формализации данных и унификации процедур их обработки приводится формальное описание уровня данных, в ходе которого структура ИР представляется в виде схемы БД, реализованной в БМД в качестве декларативной составляющей уровня данных. Процедурная составляющая при этом становится унифицированной, так как все структурные особенности, различающие ИР разных типов, учитываются в БМД.

В статье предлагается использовать выявленный механизм формализации остальных двух уровней ФУ. Для этого на основе анализа структур данных и процедур уровней представления и приложений ФУ сформулированы постановки задач по их формализации. Конечная цель предложенной формализации заключается в разработке высокоуровневой модели данных, описывающей ИР в терминах ПрО.

Реализация сформулированных целей предполагается в последующих статьях данной серии.

СПИСОК ЛИТЕРАТУРЫ

1. Единое информационно-функциональное пространство ВМФ: от идеи до реализации / под общ. ред. В.И. Кидалова. – СПб. : Ника, 2003 – 490 с.

2. Шкрыль А.А. Разработка клиент-серверных приложений в Delphi. – СПб. : БХВ-Петербург, 2006. – 480 с.

3. Энциклопедия SQL : 3-е изд. / Дж. Грофф, П. Вайнберг. – СПб. : Питер, 2004. – 896 с.

4. Брукшир Дж. Информатика и вычислительная техника : 7-е изд. – СПб. : Питер, 2004. – 620 с.

5. Валиков А.Н. Технология XSLT. – СПб. : БХВ-Петербург, 2002. – 544 с.

6. Цикритзис Д., Лоховски Ф. Модели данных : пер. с англ. – М. : Финансы и статистика, 1985. – 344 с.

7. Токмаков Г.П. Интеграция информационных ресурсов: эволюция, механизмы и формальная модель // Автоматизация процессов управления. – 2009. – № 1 (15). – С. 3–14.

REFERENCES

1. Kidalov V.I. and Others. *Yedinoye informatsionno-funktsionalnoye prostranstvo VMF: ot idei do realizatsii* [Common Navy Information Space: from an Idea to the Implementation]. Sankt-Peterburg, Nika Publ., 2003. 490 p.

2. Shkryl A.A. *Razrabotka kliyent-servernykh prilozheniy v Delphi* [Development of client-server applications in Delphi]. Sankt-Peterburg, BHV-Peterburg Publ., 2006. 480 p.

3. Groff J.R., Weinberg P.N. *Entsiklopediya SQL : 3-e izd.* [The Complete Reference SQL. Third Edition]. Sankt-Peterburg, Piter Publ., 2004. 896 p.

4. Brookshear J.G. *Informatika i vychislitelnaya tekhnika : 7-e izd.* [Computer Science. 7th Edition]. Sankt-Peterburg, Piter Publ., 2004. 620 p.

5. Valikov A.N. *Tekhnologiya XSLT* [XSLT-Technology]. Sankt-Peterburg, BHV-Peterburg Publ., 2002. 544 p.

6. Tsichritzis D., Lochovski F. *Modeli dannykh* [Data Models]. Moscow, Finansy i statistika Publ., 1985. 344 p.

7. Tokmakov G.P. Integratsiya informatsionnykh resursov: evolyutsiya, mekhanizmy i formalnaya model [Integration of Information Resources: Evolution, Mechanisms, and Formal Model]. *Avtomatizatsiya protsessov upravleniya* [Automation of Control Processes], 2009, no. 1(15), pp. 3–14.