Технология разработки приложений баз данных на основе декларативных спецификаций*

Е. С. Фереферов[1], И. В. Бычков[1], А. Е. Хмельнов[1]

Статья посвящена проблемам автоматизации разработки прикладных автоматизированных информационных систем, обеспечивающих взаимодействие пользователя с базами данных (приложений баз данных). Для решения этой задачи предлагаются оригинальная технология и инструментальная система, позволяющая создавать автоматизированные информационные системы на основе декларативных спецификаций приложений баз данных. Рассматриваемые декларативные спецификации являются удобным средством представления моделей приложений баз данных.

Ключевые слова: автоматизация разработки, информационные системы, приложения баз данных, декларативные спецификации.

Введение

Автоматизированные информационные системы (АИС) представляют собой программные комплексы, функции которых состоят в поддержке сбора, хранения, выполнения специфических преобразований информации, моделирования, вычислений, агрегаций, анализа и предоставления данных пользователям. Хотя некоторые АИС могут быть основаны на применении сложных алгоритмов обработки информации и взаимодействии с внешними аппаратными средствами, основная задача значительной части таких систем сводится к организации удобного взаимодействия пользователя с реляционной СУБД. Программное обеспечение, реализующее интерфейс для взаимодействия пользователей с предметной базой данных (БД), будем называть приложением баз данных (ПБД). Основные задачи ПБД — обеспечение выполнения стандартных операций: создания, чтения, модификации и удаления записей таблиц БД (CRUD операции), а также ряда дополнительных операций: поиска, фильтрации данных, формирования отчётов и т. п. Приложения баз данных могут быть как частью (подсистемой) АИС (например, частью биллинговой системы), так и самостоятельными АИС (например, городской реестр адресов, реестр объектов недвижимости). Кроме того, ПБД могут входить в состав вычислительных систем (ВС), например, в качестве подсистемы для регистрации пользователей.

Современные технологии разработки АИС, основанные на императивном программировании и использовании развитых библиотек визуальных компонентов (например, VCL [1], MFC [2], FCL [3]), предоставляют общецелевые инструментальные средства. Среди них имеются компоненты, реализующие как части визуального интерфейса, так и части бизнес-логики для доступа к данным в БД и их модификации. При разработке

[1] *Институт динамики систем и теории управления СО РАН, Иркутск, Россия*
Контактный e-mail: fereferov@icc.ru

*Работа выполнена при финансовой поддержке РФФИ в рамках проекта № 14-07-31339.

ПБД, как правило, сначала создаётся структура (схема) БД, а затем ПБД (иногда эти процессы могут происходить одновременно). При этом для используемых в приложении таблиц БД необходимо описать в коде или средствами визуального проектирования их структуру, а также реализовать пользовательский интерфейс для просмотра и редактирования каждой таблицы. Для больших таблиц дополнительно необходимо реализовать поиск записей по некоторым атрибутам и их фильтрацию. Фрагменты кода для работы с разными таблицами различаются в основном только именами и типами используемых полей, что приводит к созданию большого объёма однотипного кода. При использовании средств визуального программирования этот объём не сокращается, но лишь несколько возрастает скорость создания программы. При изменении структуры БД, например добавлением новых таблиц или полей, необходимо вносить изменения в программный код ПБД, заново реализуя функции для доступа к новым элементам БД. Невысокий уровень автоматизации реализации однотипных функций для работы с таблицами БД вызывает большие временные и трудовые затраты при создании ПБД и существенно усложняет их модернизацию, поскольку, например, изменение типа поля может потребовать коррекции всех частей кода, где оно упоминается.

Существующие подходы в области объектно-реляционного отображения (например, Hibernate/NHibernate [4], Entity Framework [5]) позволяют ускорить разработку ПБД за счёт автоматизации построения объектной модели, обеспечивающей взаимодействие с сущностями реляционной БД и являющейся описанием структуры БД для приложения. При этом часть программного кода, в частности классы, соответствующие сущностям БД, и классы, обеспечивающие преобразование объектных запросов в SQL-запросы, генерируется автоматически. Отметим, что следствием изменения в структуре БД является необходимость перегенерации классов. Это не сложный процесс, однако здесь, чтобы не смешивать автоматически сгенерированный и созданный вручную коды, требуется аккуратное программирование при доработке данных классов. Дальнейшая реализация системы, т. е. программирование пользовательского интерфейса, операций поиска и т. д., выполняется уже вручную отдельно для каждой сущности, только теперь через свойства и методы классов, что снова приводит к созданию больших объёмов однотипного кода.

В настоящее время активно ведутся исследования в области автоматизации разработки как пользовательских интерфейсов (например, Model-Based User Interface Development [6]), так и АИС в целом (например, Model Driven Architecture [7], порождающее программирование [8]), позволяющих повысить эффективность процесса создания приложений. Основной тенденцией в этих исследованиях является разработка современных методов структурирования метаданных (данных о структуре) АИС в виде моделей системы (иногда только пользовательского интерфейса [9]) различными средствами (например, надстройки над моделями классов UML [10], применение модельных каркасов [11] или построение онтологий предметной области [12]). Формализация знаний о структуре АИС в модели позволяет единожды выделить схожие структуры данных и присущие им бизнес-процессы в отдельные компоненты, а также сгенерировать соответствующие им сценарии создания структур в СУБД, алгоритмы обработки бизнес-процессов, экранные формы и распространять их на все подобные компоненты. Сгенерированный код в данном случае является грубым каркасом будущей системы, поэтому практически всегда требуется его доработка, причём полученные изменения не отражаются в исходных абстрактных моделях системы, что существенно затрудняет её модернизацию при необходимости изменения структуры БД.

Большинство БД содержат информацию о реально существующих физических объектах. Такие объекты, как правило, имеют определённое пространственное расположение, которое может задаваться рядом атрибутов, например, почтовым адресом, координатами, кадастровым номером и т. д. В данном случае наиболее наглядным способом представления, анализа и поиска пространственно-привязанной информации является её отображение на карте. Поэтому для решения задач обработки, представления и анализа пространственных данных (ПД) современные АИС часто должны в том или ином объёме включать соответствующие функциональные возможности геоинформационных систем (ГИС). Для реализации ГИС-функциональности современные ГИС предлагают разработчикам API (например, GisToolKit, MapX). Несмотря на развитость этих API, реализация таких функций — сложная и трудоёмкая задача, требующая знаний в области геоинформационных технологий. Реализация ГИС-функциональности существующими методами часто приводит к дублированию функций целевой ГИС в разрабатываемой системе, поскольку используется доступ к данным из приложения ГИС. Для модернизации существующих АИС, направленной на интеграцию с функциями обработки ПД, как правило, необходимо наличие исходных кодов этих АИС у разработчика, а в случае их отсутствия требуется повторная разработка системы.

Таким образом, в рассматриваемой тематике актуальными являются исследования в области разработки концептуально новых технологий и инструментальных средств, автоматизирующих процесс создания приложения баз данных, позволяющих не только сократить сроки и, как следствие, затраты на создание и модернизацию ПБД, но и решать более широкий круг задач за счёт использования встроенной ГИС-функциональности.

1. Технология спецификации приложений баз данных

Для автоматизации создания информационных систем авторами разработана технология создания ПБД на основе декларативных спецификаций, которые являются средством представления и хранения моделей ПБД. Спецификация приложения баз данных содержит минимально необходимую информацию о его структуре, которой вместе с тем достаточно для автоматической реализации приложения и, в частности, создания пользовательских интерфейсов, обеспечения выполнения CRUD-функций, построения пользовательских запросов, поддержки взаимодействия с ПД, а также организации взаимодействия с внешними подключаемыми модулями для решения специфических задач. Декларативные спецификации удобны своей компактностью, при этом они обладают предметной ориентированностью и выразительностью, а также широкой возможностью интерпретации различными трансформационными и другими процедурами. Кроме того, представление моделей в виде спецификаций позволяет поддержать модульную разработку ПБД — интегрировать готовые спецификации приложений для работы с частью БД при разработке новых, более масштабных ПБД.

Взаимодействие пользователя с данными в рамках предлагаемой технологии (рис. 1) осуществляется через настраиваемое при помощи спецификации ПБД универсальное автоматизированное рабочее место (АРМ), реализующее пользовательский интерфейс доступа к БД, а также картографический модуль для доступа к пространственным данным и подключаемые внешние модули (Plugin), расширяющие функциональные возможности универсального АРМ (например, для решения вычислительных задач).

Рассматриваемая технология может использоваться как при разработке системы с нуля, так и при создании приложения для работы с существующей БД. Иными сло-

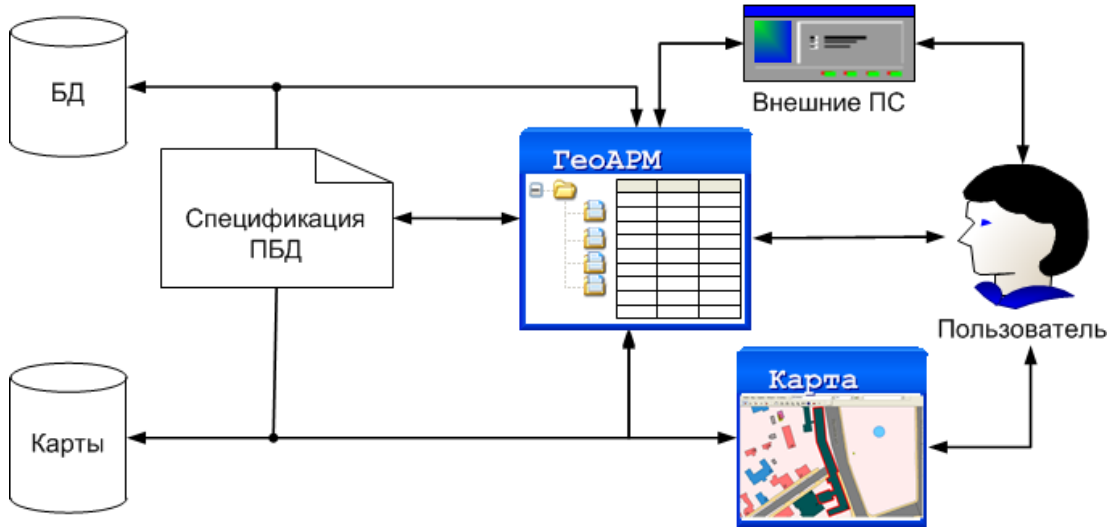

Рис. 1. Схема взаимодействия пользователя с данными в рамках технологии спецификации ПБД

вами, здесь не накладываются существенные ограничения на способ организации БД, как это происходит, например, при использовании EAV-подхода [13], поэтому во многих случаях возможна разработка ПБД в ГеоАРМ под существующую БД с сохранением совместимости с унаследованными приложениями, использующими эту БД для решения других задач.

Процесс создания АИС в рамках предлагаемой технологии выглядит следующим образом (рис. 2). В случае разработки новой АИС на первом этапе проводится структурный анализ предметной области, в результате которого выявляются сущности, формулируются требования к структуре, описываются бизнес-процессы, выявляются функциональные зависимости. Далее следует этап проектирования БД, результат которого — создание схемы БД для разрабатываемой системы, реализованной в определённой СУБД (например, СУБД MS SQL Server, Oracle). На этих этапах предложенная авторами система не используется.

На третьем этапе с применением инструментальной системы ГеоАРМ строится модель ПБД. При этом в качестве входных данных применяются метаданные о структуре БД (схема БД), хранящиеся в СУБД. Схема данных представляет собой уже структурированные знания о сущностях и связях между ними, но для создания полноценного ПБД этих знаний недостаточно. Структурную информацию БД необходимо расширить знаниями о способах представления информации пользователю и интерпретации связей между таблицами.

При построении модели данных в рассматриваемой здесь технологии имеется некоторое сходство с реализациями концепции MVC [14] (например, Oracle ADF [15]), но при этом предлагаемая модель ПБД содержит не только структурную информацию (схему БД), как в MVC, но и информацию о способе представления данных пользователю, что в концепции MVC строго выделяется в отдельный уровень. Кроме того, в модели ПБД могут содержаться связи с пространственными данными и способы взаимодействия с внешними программными системами. Отличием предлагаемой авторами технологии от MDA является отсутствие в ней процессов генерации и компиляции программного кода. Все работы по созданию АИС направлены на построение спецификаций ПБД, которые

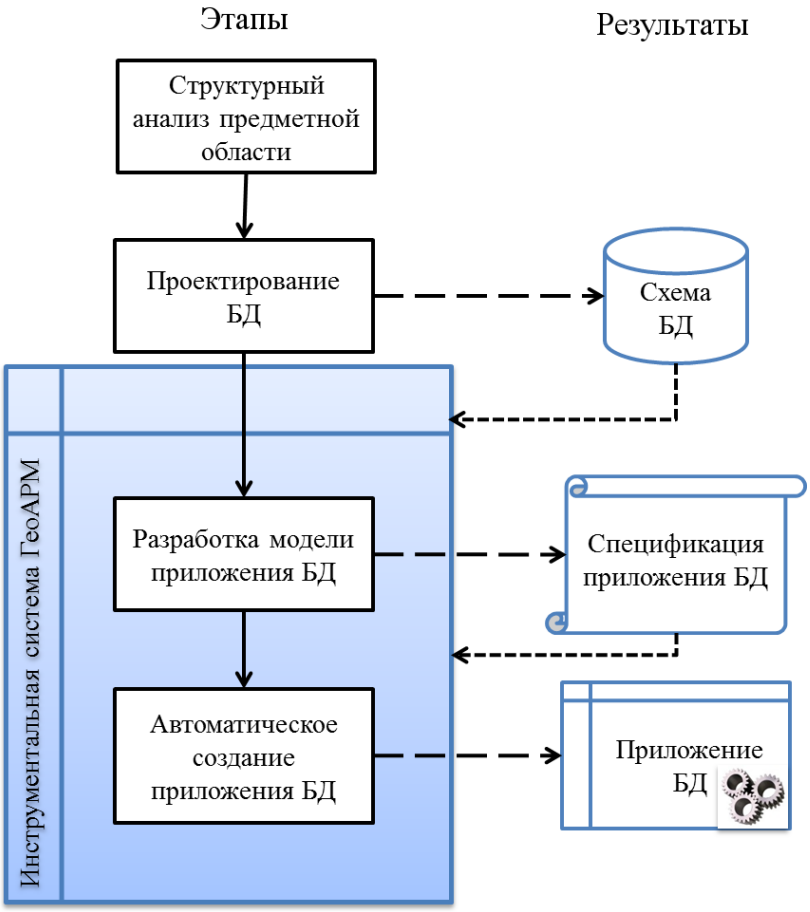

Рис. 2. Этапы технологии разработки приложения БД

интерпретируются специально разработанной универсальной инструментальной системой ГеоАРМ, обеспечивающей динамическое создание пользовательского интерфейса для доступа к БД с возможностью взаимодействия с ПД и внешними прикладными системами.

В процессе создания модели ПБД разработчику необходимо описать следующие объекты: “Таблицы”, “Представления”, “Правила”, “Надстройки”. “Таблицы” содержат структурную информацию, обеспечивающую взаимодействие (выполнение CRUD-операций) с соответствующими объектами (таблицами) БД, а именно информацию о полях таблиц с их типами данных и связях с другими таблицами БД. В случае грамотно спроектированной БД структурные метаданные таблиц можно получить автоматически из СУБД. Вместе с тем необходимо отметить, что встречаются БД, в которых отсутствует информация о связях (например унаследованные БД), поэтому должна быть возможность уточнения такой информации при формировании объектов модели приложения БД и реализации работы в приложении с несколькими БД. Заметим, что не всю информацию, необходимую для описания ПБД, можно извлечь из метаданных. Например, внешний ключ может выглядеть одинаково для ссылок на справочник и на запись главной таблицы из подчинённой, поэтому способ интерпретации конкретной связи в свою очередь может быть дополнительно описан в спецификации ПБД.

На четвёртом этапе при помощи спецификации инструментальная система автоматически настраивается и становится предметным приложением баз данных, обес-

печивающим поддержку всех функций работы с базой данных и (при необходимости) возможность взаимодействия с пространственной информацией. Весь пользовательский интерфейс ПБД формируется динамически при интерпретации спецификации системой ГеоАРМ. Преимущество такого подхода состоит в отсутствии потребности специально разрабатывать формы пользовательского интерфейса (создавать элементы интерфейса, связывать их с данными, компилировать, отлаживать) для каждых таблицы и представления. Механизм создания элементов пользовательского интерфейса, связывания их со структурами данных унифицирован и встроен в интерпретатор системы. Интеграция функций создания спецификаций и их интерпретации в одной системе позволяют сразу оценивать адекватность модели ПБД и существенно повысить скорость разработки в целом.

При использовании предлагаемой технологии развитие и модернизация ПБД упрощаются за счёт отсутствия необходимости изменения программного кода, его отладки и компиляции с применением специализированных средств разработки программного обеспечения. Предложенная инструментальная система обеспечивает поддержку как создания, так и модернизации спецификации ПБД за счёт встроенных визуальных инструментальных средств формирования спецификаций, что даёт возможность сократить сроки доработки ПБД на каждом этапе жизненного цикла системы.

2. Язык представления баз данных

Для формирования спецификаций ПБД разработан новый декларативный язык — ЯПБД [16], конструкции которого обеспечивают достаточно детальное и в то же время компактное описание всех элементов ПБД. Грамматика данного языка принадлежит к классу LL(1). Каждый элемент ПБД описывается отдельным предложением, кроме того существуют конфигурационные предложения для описания общих системных настроек ПБД. Предложения ЯПБД имеют следующую структуру: `<Стартовое слово> <Список обязательных выражений> [список необязательных выражений]`.

Начало предложений определяется по принадлежности первого слова к множеству стартовых слов. Каждое предложение состоит из обязательных и необязательных выражений. Все выражения разделены пробелами. Каждое выражение содержит зарезервированное (служебное) слово, описывающее имя какого-либо атрибута системы, и может включать значение этого атрибута через знак “=”. Например, `READONLY` или `SCHEMA=<Значение атрибута>`. В некоторых случаях за служебным словом может идти список выражений в скобках “(. . .)” через запятую “,”.

Конструкции ЯПБД позволяют описать способ подключения к БД, общие параметры ПБД, таблицы, представления, подключаемые модули (plugin), подключение других спецификаций, способ взаимодействия с ГИС. В способе подключения к БД содержится информация о технологии доступа к БД (поддерживается BDE и ADO) и способе аутентификации (средствами СУБД или Windows, запрашивать ли имя и пароль пользователя или хранить их в спецификации). В общих параметрах ПБД задаются настройки внешнего вида ПБД (например, отображаемое наименование приложения, способ представления списка сущностей — дерево или панель), а также общие для всех таблиц правила взаимодействия с БД (схема БД, способ формирования имён полей и таблиц для запросов). При создании спецификации ПБД в качестве входных данных используются метаданные о структуре уже созданной БД (схема БД), хранящиеся в СУБД. Схема данных является уже структурированными знаниями о сущностях и связях между ни-

ми, которые необходимо расширить знаниями о способах применения этих сущностей и связей в ПБД. Описания таблиц в спецификации (рис. 3) содержат информацию, обеспечивающую автоматическое создание пользовательского интерфейса и механизмов взаимодействия с соответствующими таблицами БД. Так, для полей таблиц указывается их вид, который, кроме типа данных, известного СУБД, определяет способ использования данного поля в приложении, влияющий, например, на выбор элементов управления и доступных в построителе запросов операций сравнения. Например, строковому полю (varchar) могут быть сопоставлены следующие виды: “строковый”, “содержащий имя файла”, “именованный”, “списочный”. “Строковый” вид указывает, что в таком поле может содержаться произвольный набор символов, а при формировании запроса пользователь будет иметь возможность задать условия на значения строки или её подстрок. Вид поля, “содержащий имя файла”, применяется для указания путей к файлам большого размера, хранение которых непосредственно в БД влияет на скорость доступа к данным. “Именованное” — это строковое поле, уникально идентифицирующее запись данной таблицы. При формировании условий запроса пользователь будет иметь возможность выбрать интересующие его значения такого поля из списка. Кроме того, если это поле используется как поле-подстановка (lookup) в представлении, то при выборе записи будет задействован механизм контекстной фильтрации значений поля (функция autocomplete). “Списочный” вид указывает на то, что строковое поле может принимать ограниченное число значений (например, “да”, “нет”). Необходимость использования этих полей возникает при описании БД, в которых разработчиком не был организован справочник для хранения списка возможных значений (при разработке ПБД на основе унаследованных БД). Вместе с тем при формировании условий запроса пользователь будет иметь возможность выбрать интересующие его значения такого поля из списка.

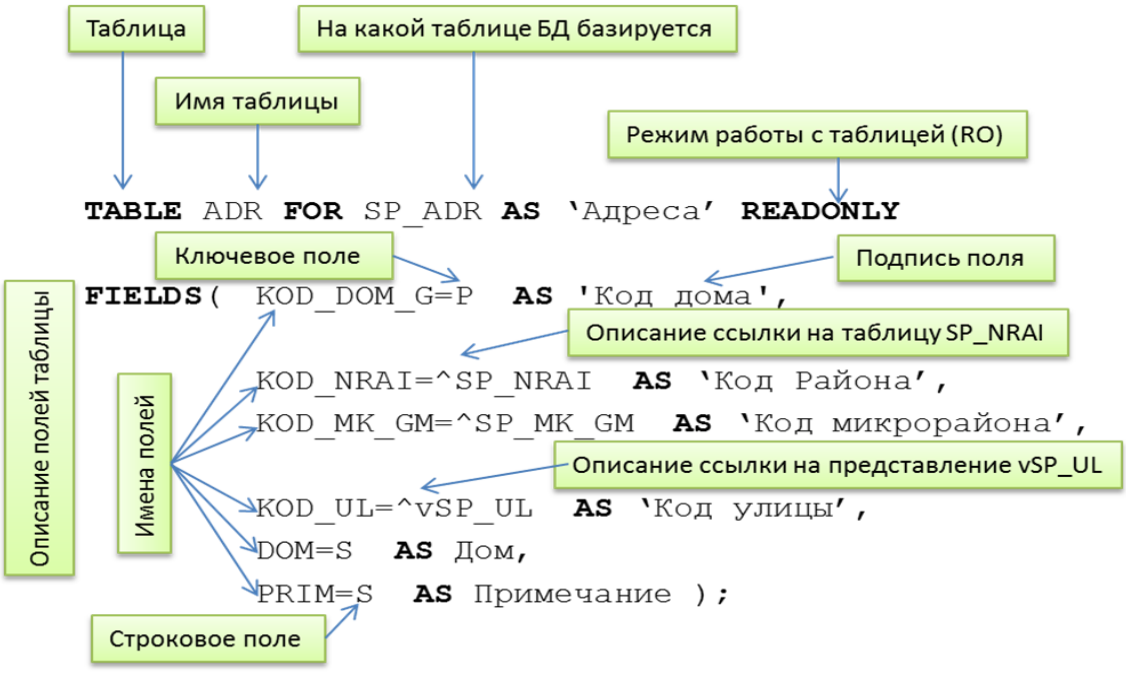

Рис. 3. Пример описания таблицы

Существуют также два вида описания blob-полей: “графический” и “документный”. Если вид поля таблицы определён как “графический”, то при просмотре записи в виде формы пользователю будет доступно изображение из файла, хранящегося в данном поле. Для “документных” полей в целях реализации загрузки и просмотра содержимого поля соответствующими приложениями дополнительно требуется указание расширения файла.

Кроме вида, в описание полей может быть включено указание на их роль во взаимодействии с пространственными объектами (объектами цифровых карт). Так, группу полей таблицы (представления) можно описать как адресную (например, поля, содержащие значения “Улица” и “Дом”). Тогда пользователю при работе с записями такой таблицы будет доступна возможность поиска и отображения пространственных объектов с соответствующими значениями семантик. Для полей таблиц, содержащих координаты пространственных объектов, в спецификации можно определить способ создания соответствующих объектов на цифровой топооснове, что позволяет реализовать автоматическое создание цифровых карт на основе записей таблиц БД.

Все современные СУБД предоставляют информацию о наличии связей между таблицами, но при создании ПБД для понимания того, как должна быть реализована работа пользователя со связанными таблицами, её недостаточно. В спецификации ПБД при описании таблиц, кроме наличия внешних ключей (ссылок на справочники), могут быть указаны связи типа “мастер-детали”, что позволяет автоматически создать формы, обеспечивающие при просмотре записей справочников доступ к записям-деталям (записям из таблиц, содержащих ссылку на данную запись справочника). При этом поддерживается описание связей типа “мастер-детали” через несколько уровней ссылок. Для примера рассмотрим структуру БД, реализующую хранение информации об объектах городского хозяйства, где объект характеризуется почтовым адресом (запись в таблице SP_ADDR). Объект может состоять из нескольких строений (записи таблицы BUILDINGS, являющиеся деталями по отношению к записям SP_ADDR). В каждом здании может существовать множество помещений (записи в таблице FLATS, являющиеся деталями для записей BUILDINGS). При просмотре записей объектов городского хозяйства пользователю на форме необходимо видеть список помещений всех зданий объекта. Для реализации такого представления данных в спецификации в описании таблицы SP_ADDR необходимо указать связь следующего вида:

$$\text{REFS}(=< \text{vFLATS.Buildings\_ID.SP\_ADDR\_ID}),$$

где vFLATS — представление на основе таблицы FLATS, Buildings_ID — поле таблицы FLATS, содержащее ссылку на таблицу BUILDINGS, SP_ADDR_ID — поле таблицы BUILDINGS, содержащее ссылку на таблицу SP_ADDR.

В правильно разработанной (нормализованной) БД используется большое количество таблиц-справочников, а основные таблицы содержат внешние ключи — ссылки на эти справочники. При отображении информации пользователю вместо искусственных внешних ключей необходимо показать значения одного или нескольких полей из справочника, например, вместо кода улицы — её наименование и тип. В то же время не все поля, доступные по ссылкам из таблицы, следует сразу показывать на формах этой таблицы: например, для улицы могут быть известны дата переименования, старое название и ещё ряд атрибутов, которыми нет необходимости загромождать форму с информацией о здании. Для выбора отображаемых полей из доступных в некоторой таблице и таблицах, связанных с ней по ссылкам, в спецификации требуется описать

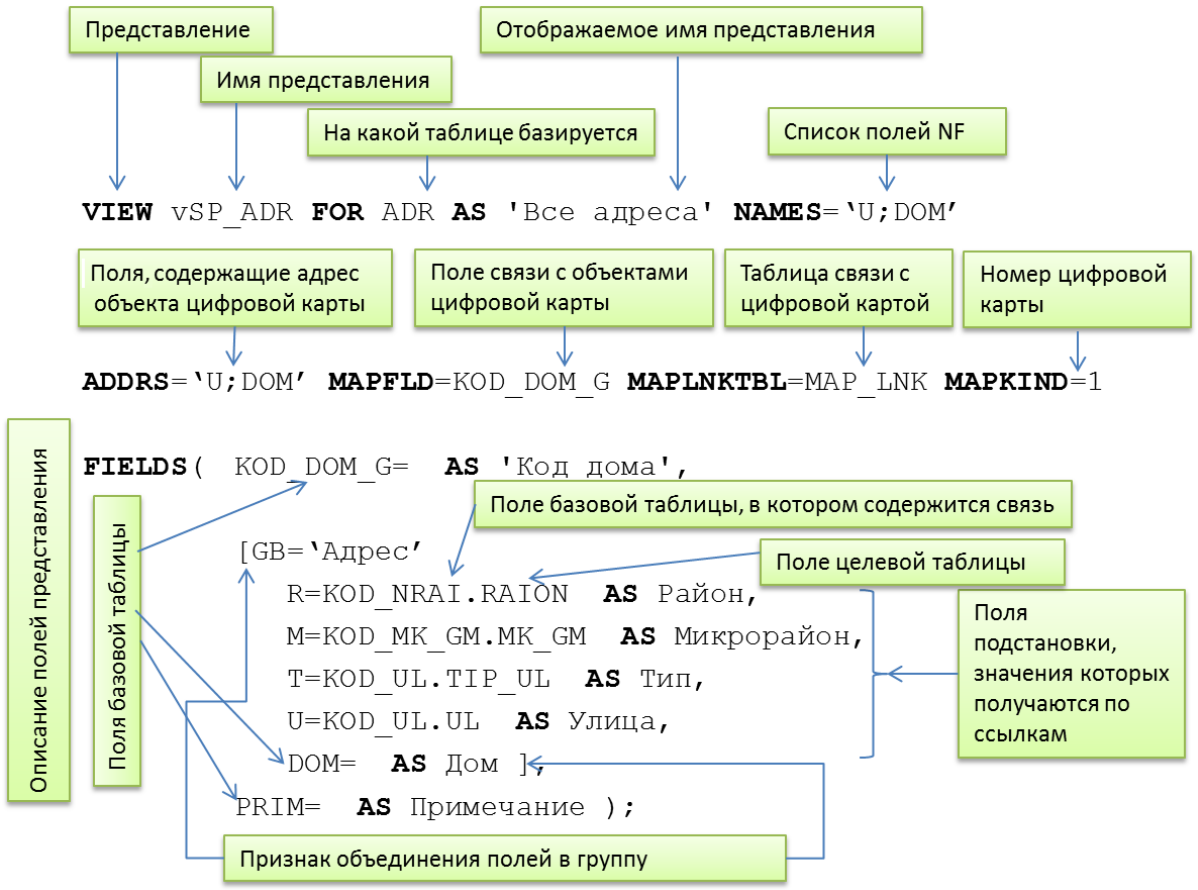

Рис. 4. Пример описания представления

“Представление”. В отличие от представлений (View) реляционных СУБД, которые являются сохранёнными произвольными SQL-запросами, представления в предлагаемой технологии являются наборами данных (рис. 4), построенных на одной базовой таблице и содержащих информацию о декодировании связей для получения данных из связанных таблиц для полей-подстановок. Для полей представлений задаётся путь получения значений из полей базовой таблицы или из таблиц, на которые имеются ссылки из базовой таблицы. Причём значения полей-подстановки в представлении могут быть получены через несколько уровней ссылок как из таблиц, так и из других представлений.

3. Инструментальная система создания приложений баз данных

Для автоматизации создания и модернизации спецификаций ПБД авторами была разработана инструментальная система ГеоАРМ [17]. Эта система позволяет интерактивно создавать спецификации ПБД, настраивать взаимодействие с БД, внешними подсистемами, цифровыми картами. Реализация предметного ПБД осуществляется динамически в результате интерпретации спецификации в той же инструментальной системе.

ГеоАРМ состоит из подсистем: “Ядро”, “Управление спецификациями”, “Редактор БД”, “Построитель пользовательских запросов”, модуль “Программный интерфейс” (рис. 5). Общая схема разработки и исполнения ПБД в инструментальной системе ГеоАРМ выглядит следующим образом: при помощи подсистемы “Управления специфи-

кациями" создаётся спецификация ПБД, которая загружается в "Ядро" системы, в результате чего инструментальная система становится предметным ПБД и обеспечивает взаимодействие с БД через подсистемы "Редактор БД", "Построитель пользовательских запросов", а также позволяет решать специфические задачи при помощи библиотек-надстроек, подключаемых через встроенный программный интерфейс, и осуществлять

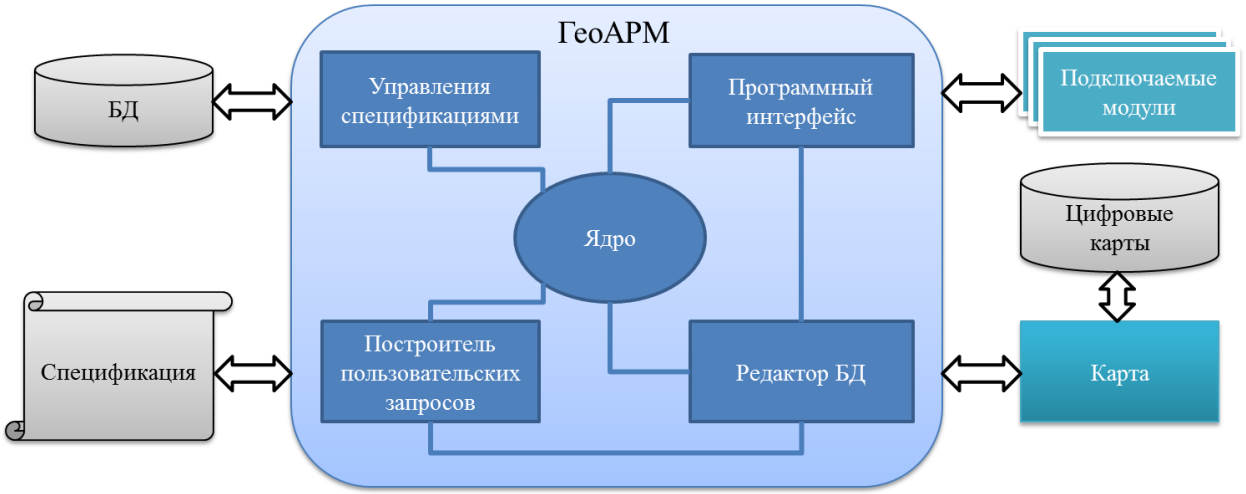

Рис. 5. Архитектура инструментальной системы создания ПБД

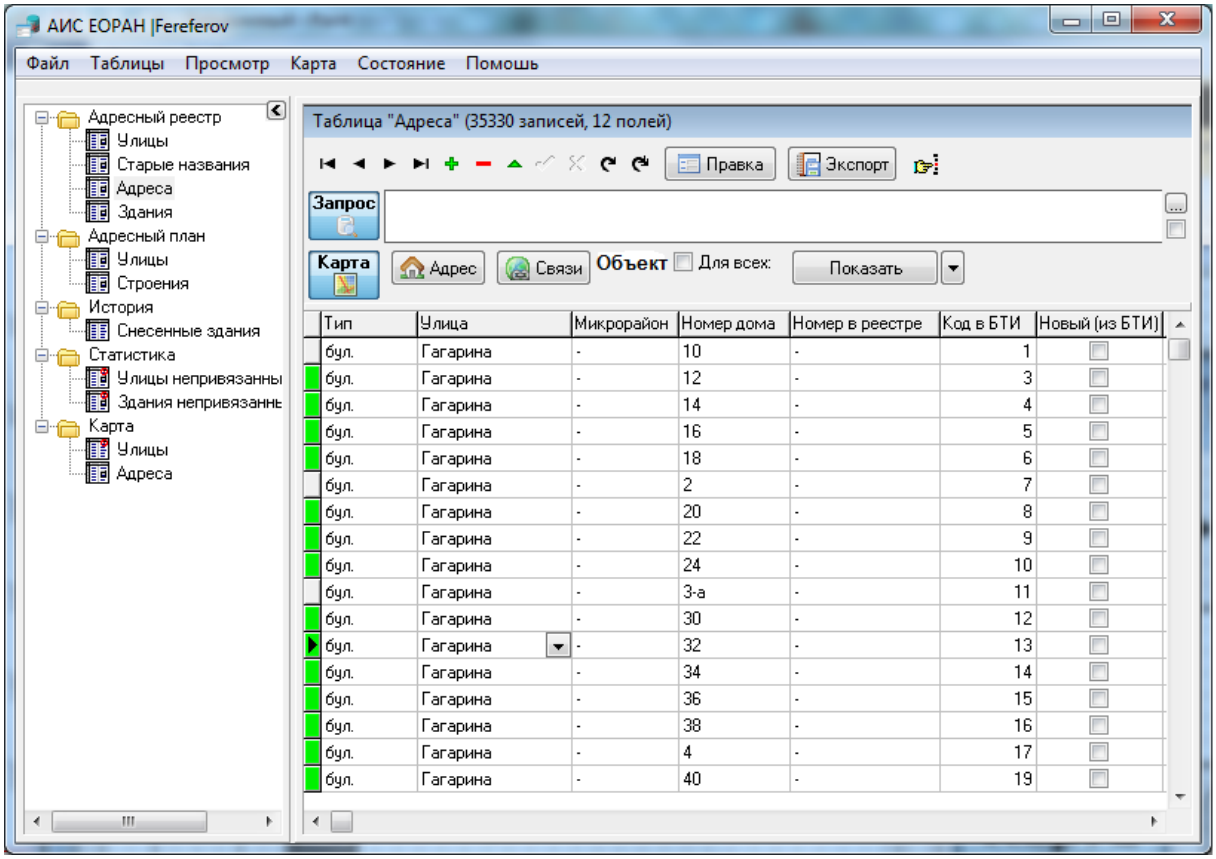

Рис. 6. Подсистема "Редактор БД" в табличном режиме

взаимодействие с пространственными данными через подсистему “Карта”. “Ядро” системы отвечает за взаимодействие всех подсистем приложения с СУБД, обеспечивая интерпретацию спецификаций и преобразование команд подсистем ПБД в команды интерфейсов для взаимодействия с СУБД.

Подсистема “Управление спецификациями” реализует функции пользовательского интерфейса разработчика спецификаций ПБД. При помощи данной подсистемы разработчик выполняет операции по созданию и модернизации спецификаций ПБД: настройку соединения с БД, выбор и загрузку метаинформации из СУБД, управление описанием таблиц, создание описаний представлений, настройку общих параметров ПБД и параметров взаимодействия с цифровыми картами и внешними программными системами.

Подсистема “Редактор БД” реализует пользовательский интерфейс для взаимодействия с предметной БД. Формы для доступа и модификации таблиц и представлений создаются динамически на основе описаний соответствующих структур в спецификации ПБД. “Редактор БД” (рис. 6) содержит главное меню, дерево сущностей, область задач и область данных. Главное меню содержит все функции АИС для быстрого доступа. В дереве сущностей (см. слева на рис. 6) отображаются описанные в спецификации имена таблиц и представлений АИС, объединённые в смысловые группы исходя из их назначения или функций.

В области данных (см. справа на рис. 6) отображаются элементы управления для работы с таблицами и представлениями. Работа с данными осуществляется в двух режимах: в режиме таблицы для просмотра набора записей или в режиме формы для работы с конкретной записью. Состав элементов управления и стиль работы с полями таблиц или представлений определяются автоматически на основе информации из спецификации. Например, в табличном режиме для полей-подстановок автоматически формируются списки значений, причём при осуществлении работы с несколькими полями таблицы-источника может быть определена последовательность их выбора. В режиме формы (рис. 7) все элементы пользовательского интерфейса создаются автоматически. Для редактирования значений полей базовой таблицы в зависимости от указанного вида поля в спецификации создаются элементы визуального интерфейса типа “Поле для ввода”, “Флаг”, “Календарь”. Поля, полученные по одной ссылке, если не определены иные группировки полей в спецификации, автоматически объединяются в группы. Взаимодействие с полями-подстановками реализуется через элемент “Выпадающий список”, значения в который загружаются из запроса, построенного на основе информации из спецификации. При наличии описаний в спецификации связей типа “мастер-детали” на формах отображаются записи из таблиц-деталей. За оптимальную с точки зрения удобства работы пользователя расстановку в области задач элементов управления отвечает менеджер компоновки [18].

Построитель пользовательских запросов является универсальной формой, настраивающийся на работу с конкретной таблицей или представлением при помощи спецификаций. Построитель может работать как в упрощённом, так и в расширенном режиме. В первом случае пользователь видит таблицу с именами полей, в которой он может задать ограничения на значения некоторых из них, при этом конъюнкция условий образует условие запроса. При необходимости задать более сложное условие можно перейти в расширенный режим, где есть возможность редактировать выражения в виде горизонтально ориентированного дерева (рис. 8). В расширенном режиме поддерживается формирование условий на записи подчинённых таблиц. При формировании таких

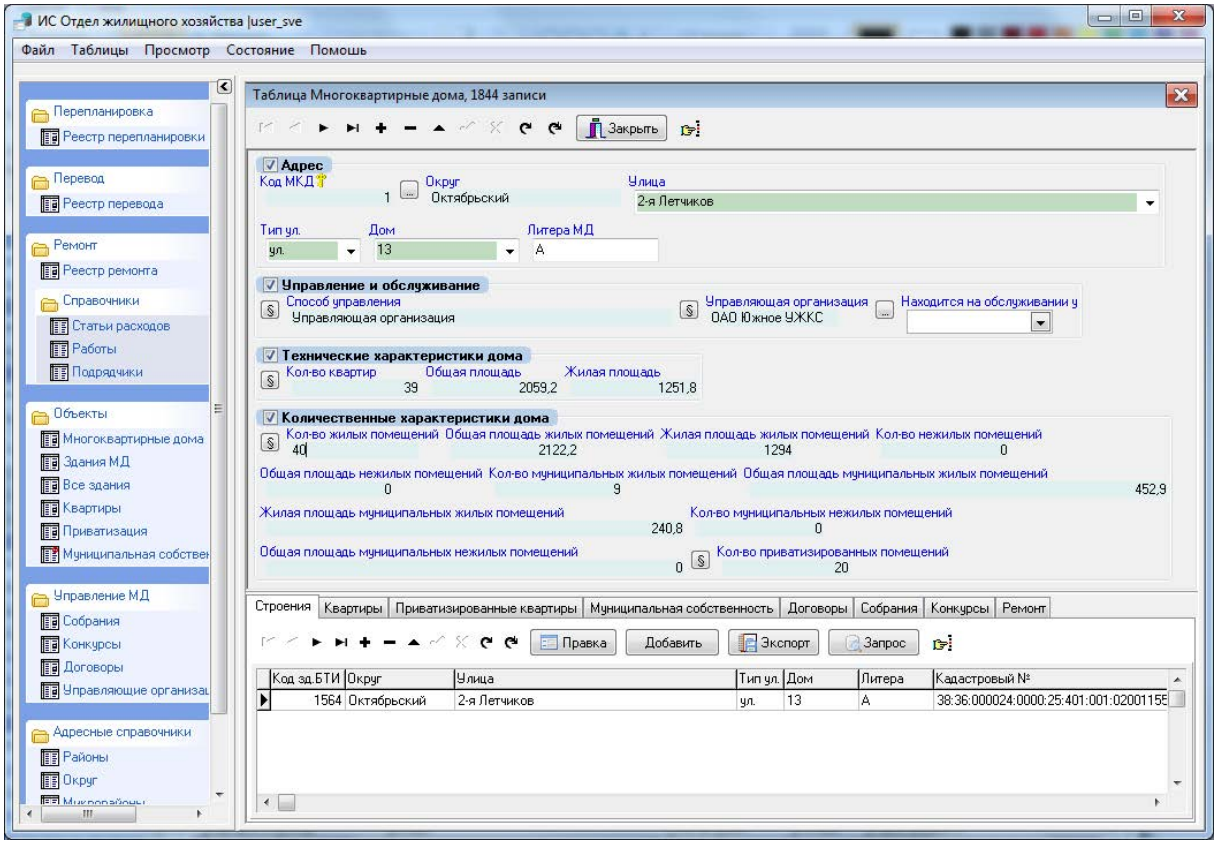

Рис. 7. Подсистема “Редактор БД” в режиме формы

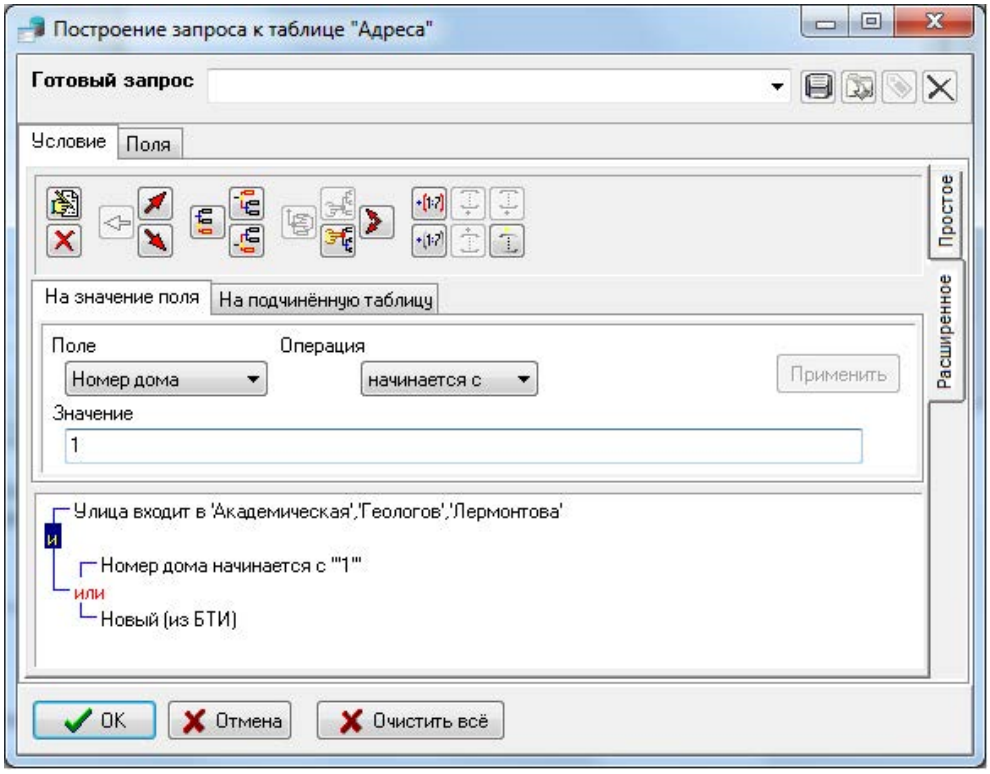

Рис. 8. Построитель пользовательских запросов

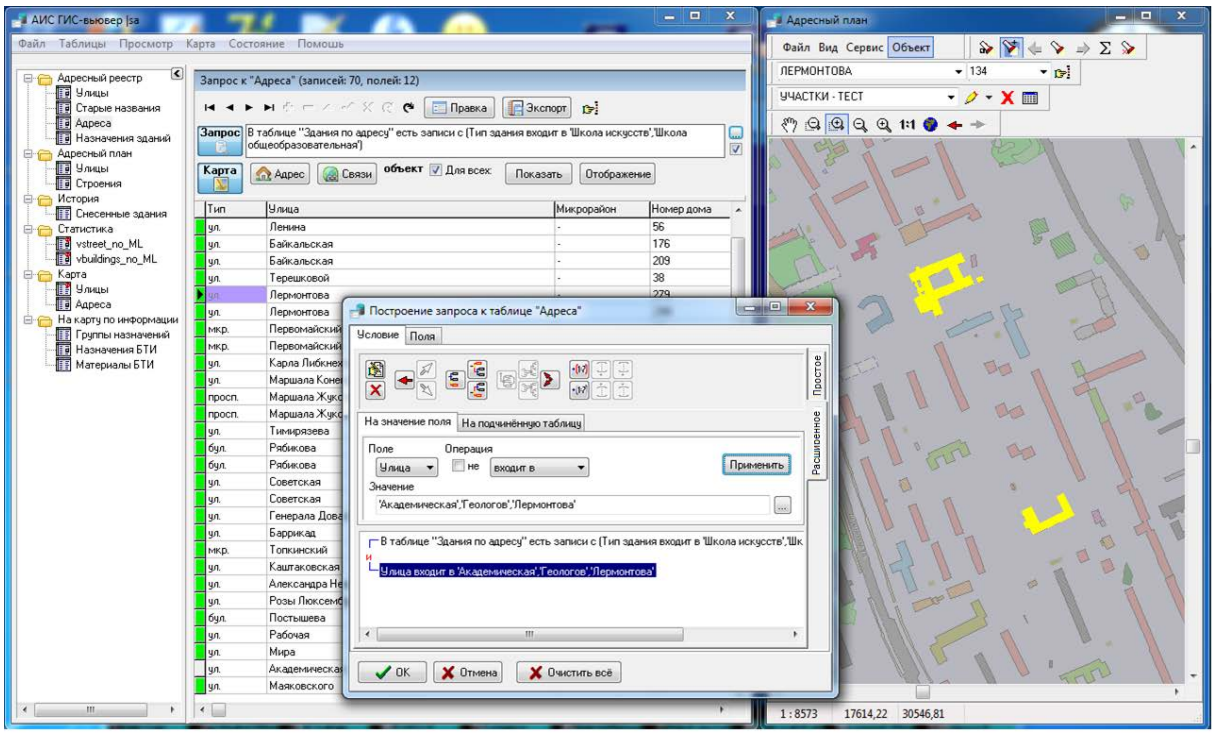

Рис. 9. Отображение результатов запроса на карте

условий построитель запросов вызывается рекурсивно. В обоих режимах способ редактирования условий на значения поля определяется информацией о виде поля таблицы или представления из спецификации ПБД.

Программный интерфейс предназначен для взаимодействия с надстройками (внешними подсистемами), позволяющими решать специфические, не заложенные в архитектуру ГеоАРМ, задачи. Надстройки реализуются в виде динамически подключаемых библиотек и позволяют расширить возможности системы.

Подсистема “Карта” (рис. 9) обеспечивает возможность работы с цифровой топоосновой. Этот модуль реализован при помощи пакета GIS ToolKit из ГИС Панорама в виде DCOM-сервера. Данная подсистема позволяет просматривать картографическую информацию, находить на карте объекты, информация о которых содержится в БД, и, наоборот, находить в БД информацию, связанную с объектом карты. В ней также реализованы стандартные механизмы просмотра и редактирования цифровой карты, такие как загрузка карты, масштабирование, перетаскивание карты, создание, удаление объекта, получение информации об объекте. Привязка БД к карте может быть осуществлена двумя способами: через геокодирование, т. е. по полям, содержащим адреса домов, или привязкой к произвольным объектам карты через таблицу связей.

4. Технология применения системы ГеоАРМ для разработки АИС

Итак, для создания ПБД с использованием системы ГеоАРМ необходимо разработать или иметь готовую БД, после чего структура этой БД описывается в спецификации ПБД. При этом автоматически генерируются формы для просмотра содержимого таблиц БД в табличном виде (см. рис. 6), а также формы редактирования отдельных записей (см. рис. 7). Для отображения информации пользователю в основном исполь-

зуются не непосредственно таблицы, а основанные на них представления, в которых вместо кодов в ссылочных полях отображаются осмысленные для пользователя поля из таблиц-справочников (Lookup-поля). На формах редактирования записей применяется механизм размещения полей [18], позволяющий эффективно использовать свободное место формы для отображения информации и перераспределять его между полями редактирования данных при изменении размеров формы.

В случае, если разработчика не устраивает автоматически определённый порядок размещения полей на форме, описание автоматически сгенерированной формы редактирования записи можно включить в спецификацию, после чего оно может быть скорректировано.

В результате указанных операций создаётся приложение, обеспечивающее реализацию основных операций просмотра, поиска и редактирования информации в описанных таблицах БД. При этом для реализации более сложной бизнес-логики и обеспечения целостности данных необходимо использовать программные механизмы СУБД (триггеры, ограничения). Существует также возможность вызова хранимых процедур и функций СУБД непосредственно пользователем в виде вызова запускающей их надстройки (plugin). В системе не используются дополнительные механизмы безопасности (разграничения доступа), поскольку предоставляемых СУБД механизмов оказывается достаточно, но при этом есть возможность устанавливать в спецификации запрет на модификацию определённых таблиц или полей на уровне пользовательского интерфейса и отображать только тот фрагмент БД, который имеет отношение к деятельности пользователя. Реализованный механизм модульности спецификаций позволяет создавать АРМы с различной функциональностью и регулировать доступ к данным через отдельные модули (например, адресные справочники описаны в отдельной спецификации, которая подключена в режиме Только чтение).

В некоторых приложениях могут использоваться бизнес-операции, затрагивающие несколько записей из, возможно, нескольких таблиц. Трудно представить ситуацию, в которой факт проведения такой операции не требуется отразить в БД. Для этого необходимо добавить запись о данной операции в специальную таблицу выполненных действий. Таким образом, чтобы реализовать бизнес-операцию в системе ГеоАРМ, необходимо вызвать её из триггера на вставку соответствующей таблицы действий, а само выполнение операции в данном случае сводится к добавлению новой записи в таблицу действий (при этом удаление и редактирование таких записей может быть запрещено механизмами безопасности СУБД).

Для решения более сложных задач расширения возможностей системы ГеоАРМ предусмотрен механизм подключения модулей (plugin). Кнопки вызова операций из подключаемых модулей могут быть включены на панели управления форм соответствующих таблиц, а также отображаться рядом с конкретными полями на формах редактирования записей. При выполнении операции подключаемый модуль получает доступ к информации редактируемой записи, подчинённых таблиц и т. д. Например, при помощи подключаемого модуля реализованы механизмы генерации отчётов по содержащим теги шаблонам.

Заключение

Предложенная технология позволяет автоматизировать создание приложений БД, являющихся неотъемлемой частью большинства современных АИС, значительно сокра-

тить сроки создания и трудозатраты. Например, приложение баз данных для БД Pubs (тестовая БД, входящая в состав MS SQL Server, содержит 11 взаимосвязанных таблиц) было реализовано за один час. При этом объём спецификации ПБД составил 160 строк. Кроме того, технология спецификации ПБД была эффективно применена при создании ряда АИС для органов местного самоуправления (АИС МИСОГД, МГИС г. Иркутска, АИС Управление многоквартирными домами) и позволила реализовать межсистемное взаимодействие и интеграцию данных информационных систем подразделений администрации г. Иркутска. Разработанная инструментальная система ГеоАРМ обеспечивает поддержку создания спецификаций ПБД и может быть настроена без перекомпиляции на работу с любыми БД, а также позволяет организовывать работу с цифровой картой для баз данных, в которых изначально эта возможность не была предусмотрена.

Список литературы

[1] VCL Overview // Embarcadero. URL: http://docs.embarcadero.com/products/rad_studio/delphiAndcpp2009/HelpUpdate2/EN/html/devwin32/vclov_xml.html

[2] Приложения MFC для рабочего стола // Microsoft Developer Network. URL: http://msdn.microsoft.com/ru-ru/library/d06h2x6e.aspx

[3] .NET Framework Class Library Overview // Microsoft Developer Network. URL: http://msdn.microsoft.com/en-us/library/hfa3fa08(v=vs.110).aspx

[4] Hibernate ORM documentation // Hibernate ORM. URL: http://hibernate.org/orm/documentation/

[5] Entity Framework (EF) Documentation // Microsoft. Data Developer Center. URL: http://msdn.microsoft.com/en-us/data/ee712907

[6] Paulo Pinheiro da Silva, Tony Griffiths and Norman W. Paton. International Working Conference on Advance Visual Interfaces2000 (AVI2000) // Generating User Interface Code in a Model Based User Interface Development Environment. Palermo, Italy. 2000. p. 155–160.

[7] Грибачёв К. Delphi и Model Driven Architecture. Разработка приложений баз данных. СПб.: Питер, 2004.

[8] Чарнецки К., Айзенекер У. Порождающее программирование. Методы, инструменты, применение. Для профессионалов. СПб.: Питер, 2005. 736 с.

[9] Грибова В.В., Кисленок Р.С. Автоматизация разработки визуального представления пользовательского интерфейса по модели предметной области // Искусств. интеллект. 2006. № 4. С. 148–152.

[10] Черкашин Е.А., Фёдоров Р.К., Бычков И.В., Парамонов В.В. Автоматизация синтеза ядра информационной системы с использованием UML-описания // Вычисл. технологии. 2005. Т. 10. Спецвыпуск: Труды IX рабочего совещания по электронным публикациям (El-Pub2004). С. 114–121.

[11] Черткова Е.А. Применение модельных каркасов для разработки графического пользовательского интерфейса // Вестник Астраханского гос. техн. ун-та. 2007. № 1(36). С. 150–153.

[12] Грибова В.В., Клещев А.С. Концепция разработки пользовательского интерфейса на основе онтологий // Вестник ДВО РАН. 2005. № 6. С. 123–128.

[13] Introduction to EAV Model in Magento // Magento Planet. URL: http://magentoplanet.wordpress.com/2014/03/19/introduction-to-eav-model-in-magento/

[14] Martin Fowler. Patterns of Enterprise Application Architecture. Addison-Wesley Professional, 2003. 533 p.

[15] Oracle Application Development Framework — Oracle ADF. URL: http://www.oracle.com/technetwork/developer-tools/adf/overview/index.html

[16] Фереферов Е.С., Хмельнов А.Е. Язык представления баз данных // Материалы III международной конф. "Инфокоммуникационные и вычислительные технологии и системы". Улан-Удэ: БГУ, 2010. С. 269–272.

[17] Бычков И.В., Гаченко А.С., Ружников Г.М. и др. Интеграция информационно-аналитических ресурсов и обработка пространственных данных в задачах управления территориальным развитием. Новосибирск: Изд-во СО РАН, 2012. 369 с.

[18] Фереферов Е.С., Хмельнов А.Е. Реализация менеджера размещения визуальных компонентов в Delphi // Совместный выпуск по материалам Международной конференции "Вычислительные и информационные технологии в науке, технике и образовании". Вычисл. технологии. Т. 13. Вестник КазНУ № 4(59). Ч. III. Алматы — Новосибирск, 2008. С. 283–287.

Поступила в редакцию 7 августа 2014 г.,
с доработки — 25 сентября 2014 г.

Technology for database applications based on declarative specifications

Fereferov Evgenii S.[1], Bychkov Igor V.[1], Hmelnov Aleksei E.[1]

Purpose. The article deals with the problems of automation development for automated information systems that provide user interaction with the database (database applications).

Design/methodology/approach. To solve this problem, we propose an original technology, allowing AIS creation based on declarative specification of database applications. The proposed declarative specifications provide a convenient method representation of models for database applications. A declarative language to form database application specifications was developed. It provides sufficiently detailed and at the same time compact description for all elements of database applications, as well as methods for interaction with external plugable software modules which include GIS. To automate design and modification processes for specifications of database applications, a tool system was developed. This tool system provides a universal workbench customizable for specifications of a specific database.

Findings. The proposed approach was successfully applied in the development of a number of AIS for local governments and will significantly reduce development time and modernization.

Research limitations/implications. Specific functions that can't be implemented within this approach ought to be implemented in the plug-ins. **Originality/value.** The author's technology allows creating database application without writing code on common programming languages. Unlike systems that use EAV-model, the proposed system does not impose restrictions on the database schema, and allows creating database application for any formerly constructed databases.

Keywords: automating development, information systems, database applications, declarative specification.

Received 7 August 2014
Received in revised form 25 September 2014

[1] *Institute for System Dynamics and Control Theory of SB RAS, Irkutsk, Russia*
Corresponding author: Fereferov Evgenii S., e-mail: fereferov@icc.ru
Aknowlegements: This work was supported by RFBR project number 14-07-31339.