ПРОГРАММНО–
ТЕХНИЧЕСКИЕ
КОМПЛЕКСЫ

В.Г. ТУЛЬЧИНСКИЙ, А.К. ЮЩЕНКО, Р.А. ЮЩЕНКО

УДК 681.3

ГРАФОВЫЕ ЗАПРОСЫ ДЛЯ ИНТЕГРАЦИИ ДАННЫХ ПОСРЕДСТВОМ XML

Ключевые слова: *язык XML, базы данных (БД), представления БД, язык QBE, XML-представления, семантическая сеть.*

ВВЕДЕНИЕ

XML [1] — популярный язык разметки документов, позволяющий структурировать информацию, используя произвольный набор инструкций. XML-документ имеет древовидную структуру. Среди основных применений XML — Web-сервисы, Интернет-приложения, информационные хранилища, системы документооборота. Преимуществом XML является текстовое представление, не требующее специальных программ для создания, просмотра и редактирования. В то же время XML, как правило, имеет более громоздкий формат по сравнению с аналогичными по содержанию бинарными форматами.

XML, в отличие от HTML, не является языком для кодирования определенной информации. Он служит для представления произвольных иерархий данных, выраженных в текстовом виде. Каждое конкретное применение XML порождает формально определенный язык или диалект. Существует множество стандартизированных диалектов XML, например MathML, RSS, GpaphML, XHTML и т.п. Такими диалектами также являются спецификации схемы XML-документа: DSD, DTD, XSD, XDR [2]. Синтаксический анализатор XML выдает ошибку при несоответствии содержимого XML-документа объявленной структуре. Спецификация схемы может быть встроена в XML-документ (в таком случае он сам себя описывает) или храниться в виде отдельного файла (это удобно, если одинаковую структуру должны иметь разные документы). На практике часто используется промежуточное решение: спецификация схемы, встроенная в XML-документ, ссылается на термины, определенные в других файлах. Консорциум всемирной сети (World Wide Web Consortium, W3C) предлагает средства обработки XML-данных, обладающие большими возможностями по интеграции данных разных XML-документов, поиску конкретной информации и трансформации XML-деревьев.

В настоящее время большая часть данных хранится в реляционных БД. Для облегчения разработки Интернет-приложений, межпрограммного взаимодействия и интеграции данных актуальна публикация (экспорт) данных БД в XML. Производители СУБД предлагают специальные средства для решения этой задачи. Расширение СУБД DB2 (IBM) SilkRoute [3] эффективно вычисляет моментальные представления, описанные на языке RXL. Другая разработка той же группы, XPERANTO [4], предназначена для объектно-реляционных баз. Microsoft SQL Server, начиная с версии 2000, позволяет определять как моментальные, так и виртуальные (динамические) XML-представления [5]. Для этого разработано расширение SQL, названное FOR XML и впоследствии положенное в основу стандарта SQL/XML [6]. В то же время усилия предпринимались в направлении манипулиро-

вания реляционными данными из языков запросов XML — XML-QL [7] и более гибкого XQuery [8], разработанного W3C. Для отображения таблиц базы данных в XPath-запросах в виде виртуальных XML-документов применяется язык XDR, уступающий по полноте и выразительности RXL, но позволяющий достаточно просто описать отображение схемы одной таблицы в схему виртуального XML-документа. Однако перечисленные стандартные средства обладают рядом ограничений:

- в XPath/XDR таблицы БД представляются отдельными виртуальными XML-документами, поэтому недостающую семантику приходится вписывать непосредственно в запрос преобразования, что его сильно усложняет;
- все данные, экспортируемые в XML-документ посредством SQL/XML, должны быть представлены одним соединением, что снижает эффективность реализации сложных XML-представлений и повышает затраты ресурсов;
- традиционные средства автоматизации разработки запросов не поддерживают XML-представления, что затрудняет их программирование и повышает вероятность ошибки.

В [9, 10] предлагается восстанавливать расширенную модель сущность-связь из реляционной схемы по шаблонам и на ее основе описывать XML-дерево в виде подсхемы или на языке вложенных отношений. Недостаток отображения XML на языке вложенных отношений — необходимость в специальном нереляционном расширении SQL. В [11] предлагается простое эвристическое решение: описание дерева XML деревом SQL-запросов. Графовые запросы [12, 13] объединяют преимущества предложенных решений, обеспечивая визуальную технологию разработки дерева запросов на основе концептуальной модели.

1. ГРАФОВЫЕ ЗАПРОСЫ

Графовый запрос (ГЗ) — средство выделения и визуализации связанных фрагментов данных, имеющих сетевую структуру. Подобная структура данных широко применяется в информационных системах на уровне концептуальной модели (например, диаграммы сущность-связь, объект-связь). Основой выделения и визуализации фрагментов сети (подграфов) в ГЗ являются деревья обхода. Главные особенности ГЗ — использование семантики концептуальной модели и результат в виде дерева — непосредственно связывают их с задачей формирования XML-представлений.

1.1. Структурированная семантическая сеть (ССС) — специальная модель данных, позволяющая отвлечься от конкретных особенностей сетевых концептуальных моделей. В основу ССС положены семантические сети — известная модель представления знаний. В отличие от моделей данных, применяемых в деловой сфере, семантические сети предназначены для систематизации знаний общего характера. Вершины могут быть двух типов: экземпляры и классы. ССС отличаются строгой типизацией вершин и дуг сети. Подграф семантической сети, состоящий из вершин-классов и соединяющих эти вершины дуг, выделяется в случае ССС в отдельную сеть, которая выступает в роли схемы; это приближает ССС к классическим моделям данных. Схема образует остов ССС.

Структурно ССС представляет собой раскрашенный орграф, вершины которого — сущности модели предметной области (МПрО), дуги — связи между сущностями. Его дополнение образует схема — раскрашенный орграф классов. Дуга, соединяющая вершины-классы, называется бинарным отношением, а дуга, соединяющая вершины-экземпляры, — утверждением. Модели сущность–связь и объект-связь элементарно сводятся к ССС, что позволяет непосредственно распространить дальнейшие результаты на данные, описанные ER-диаграммами, диаграммами классов или последовательностей UML. Схемы XML также задают ССС. (Во всех перечисленных случаях вершинам ССС целесообразно сопоставлять сущности/объекты/элементы, атрибуты XML — атрибутам ССС.)

Вопросы восстановления отсутствующей концептуальной модели из схемы реляционной или объектно-ориентированной базы данных рассмотрены в [10, 14, 15].

1.2. Грамматическое представление ССС. Зададим непустое множество корневых вершин ССС $T_0 \subseteq E$. Введем множество нетерминальных символов N_E и определим взаимно однозначное отображение $n: E \leftrightarrow N_E$.

Рассмотрим грамматику $\Gamma = (N = N_E \cup \{A\},\ T = E \cup L, A,\ P = P_E \cup P_E^* \cup P_L \cup \cup P_L^*)$, в которой каждому экземпляру сопоставлены терминальный и нетерминальный символы, а каждой корневой вершине ССС и каждому утверждению ССС — продукции четырех типов: $P_E = \{p_s^1 : A \to sS \mid s \in T_0,\ S = n(s)\}$, $P_E^* = \{p_s^2 : A \to s \mid s \in T_0\}$, $P_L^* = \{p_\alpha^2 : S \to r \mid r = (\alpha, s, t) \in L, S = n(s)\}$, $P_L = \{p_\alpha^2 : S \to rT \mid r = (\alpha, s, t) \in L, S = n(s), T = n(t)\}$. Назовем такую праволинейную грамматику грамматикой ССС.

Множество строк языка грамматики ССС Γ тождественно множеству путей обхода ССС, начинающихся с корневых вершин ССС.

Грамматика Γ_2 непосредственно сцеплена с грамматикой Γ_1, если продукции грамматики Γ_2 помечены метками из числа терминальных/нетерминальных символов грамматики Γ_1, причем хотя бы для некоторых цепочек, выводимых в Γ_1, все символы каждой цепочки являются метками продукций Γ_2 и при этом однозначно определяют последовательность продукций, выводящих цепочку в Γ_2. Если каждой меткой помечено не более одной продукции Γ_2, то Γ_2 детерминировано сцеплена с Γ_1. Для удобства дальнейшего изложения несколько идентичных продукций с разными метками будем представлять как одну продукцию с несколькими метками.

Пусть $\Gamma_1 = (N_1, T_1, A_1, P_1)$ и $\Gamma_2 = (N_2, T_2, A_2, P_2 = P_{E2} \cup P_{E2}^* \cup P_{L2} \cup P_{L2}^*)$ — грамматики ССС с общей схемой.

Обозначим множество экземпляров ССС Γ_1 как E_1, множество утверждений — L_1, множество корневых вершин — $T_{01} \subseteq E_1$. Пусть каждая продукция Γ_2 помечена меткой из числа терминальных символов Γ_1, при этом выполняются следующие условия:

1) $\forall\ p_s^1 \in P_{E2}\ \ \text{label}(p_s^1) \in T_{01} \wedge f(s) = f(\text{label}(p_s^1))$ — корневая вершина соответствующего типа;

2) $\forall\ p_\alpha^1 \in P_{L2}\ \ \text{label}(p_\alpha^1) \in L_1 \wedge r = (\alpha, s, t) \in L_2 \Rightarrow f(r) = f(\text{label}(p_\alpha^1))$ — дуга соответствующего типа;

3) метки парных продукций совпадают — $\text{label}(p_x^1) = \text{label}(p_x^2)$;

4) $\forall\ t, s \in T_{01}\ (f(t) = f(s) \Rightarrow t = s)$ — тип каждой корневой вершины ССС Γ_1 уникален;

5) $\forall\ r = (\alpha, t, s),\ q = (\alpha, o, p) \in L_1\ (t = o \Rightarrow r = q)$ — тип и входной экземпляр каждого утверждения ССС Γ_1 однозначно определяют выходной экземпляр данного утверждения.

Назовем Γ_2 подсхемой Γ_1. Грамматика Γ_2 непосредственно сцеплена с грамматикой Γ_1. Назовем подсхему Γ_2 грамматики ССС Γ_1 полной, если каждой терминальной цепочке, выводимой в грамматике Γ_1, сопоставляется ровно одна терминальная цепочка, выводимая в грамматике Γ_2. Назовем сокращением подмножество $L^*(\Gamma) \subset L(\Gamma)$ языка Γ, состоящее из цепочек, не включающих повторяющиеся символы. Справедливо следующее утверждение.

Теорема 1. Сокращение языка произвольной грамматики ССС обладает полной подсхемой с конечным языком.

Конечному числу цепочек подсхемы соответствует то же число конечных соединений реляционных отношений или объектов, на которых можно стандартным образом выполнять отбор (селекцию), что позволяет непосредственно использовать язык запросов СУБД для выделения заданного фрагмента ССС. Комплексное описание фрагмента ССС (подсхема + условия отбора на языке запросов вмещающей СУБД) называется графовым запросом.

Теорема 2. Фрагмент ССС, состоящий из конечного множества цепочек $L'(\Gamma_1)$, которое можно описать конечной дизъюнкцией ограничений на языке вмещающей СУБД, представляется некоторым ГЗ.

Таким образом, механизм ГЗ обладает полнотой с точки зрения представления сетевой концептуальной модели посредством иерархического представления. Ограничив набор атрибутов, можем получить иерархическое представление XML-доку-

ментом, который является ССС по определению. XML-документ для ГЗ — то же самое, что результирующая таблица для запроса реляционной БД. В силу полноты, ГЗ является универсальным средством экспорта фрагментов БД в XML.

2. РАЗРАБОТКА ГЗ ПО ОБРАЗЦУ

Для визуального формирования ГЗ разработан QBE-подобный язык запросов ER-QBE и алгоритм трансляции запросов ER-QBE в SQL-подобный язык запросов вмещающей СУБД. В отличие от QBE для реляционной модели, ER-QBE для ГЗ — не одна форма/таблица, а дерево форм. В отличие от XDR, ER-QBE фокусируется не на синтаксических деталях формата XML (выборе представления атрибутов элементами или атрибутами XML, преобразовании имен), а на информационном содержании XML-документа.

2.1. ER-QBE формирует ГЗ с учетом следующих ограничений:

1) запрос не создает новых объектов или типов;

2) если какие-нибудь атрибуты объекта/записи участвуют в формировании обобщенного отношения, идентификатор данного объекта или первичный ключ типа IDENTITY этой записи записывается в XML-документ;

3) ничего, кроме атрибутов, в запросе не выбирается.

Ограничения 2, 3 обеспечивают реализацию обновления БД на основе модификаций XML-представления. Если задача состоит в получении статического (моментального) XML-представления, условия 2 и 3 могут быть ослаблены: достаточно потребовать наличия ключа (возможно, составного) среди выбранных атрибутов и разрешить простые вычисления без агрегации.

Структуру ГЗ задает подсхема — ССС, вершины которой соответствуют классам схемы (таблицам БД), а дуги — бинарным отношениям (ссылкам, внешним ключам). В ER-QBE подсхема строится последовательно.

Сначала добавляются корневые вершины. Остальные вершины подсхемы присоединяются к ранее добавленным вершинам по дугам. Пользователь выбирает, к какой вершине добавить новую и по какому бинарному отношению. Для выбора предлагаются все пары «первичный ключ класса выбранной вершины — соответствующий внешний ключ какого-то класса» и «первичный ключ какого-то класса — соответствующий внешний ключ класса выбранной вершины». Таким образом, связи между классами вершин могут идти по ссылке в любом направлении. Вместо новой вершины к выбранной можно присоединить ранее добавленную вершину соответствующего класса. Одному классу может соответствовать несколько вершин подсхемы, одному бинарному отношению — несколько дуг. Разрешены циклические связи, в том числе дуга может соединять вершину подсхемы с ней самой. Однако соединение пары вершин двумя одинаково направленными дугами одного бинарного отношения не допускается. Свойством дуги является степень зависимости: разрешен ли экспорт экземпляра первой вершины дуги, если с ним не связан ни один экземпляр второй вершины дуги. В DTD этому соответствует выбор между модификаторами вложенного элемента * и +. Каждой вершине подсхемы в ER-QBE соответствует форма, напоминающая транспонированные формы QBE [16], — таблица, в первой колонке которой перечислены атрибуты соответствующего класса схемы БД, а остальные колонки служат для задания условий отбора экземпляров данной вершины. Имена атрибутов, выбранных для экспорта в XML, помечаются. Подсхема с наложенными условиями отбора и выбранными атрибутами описывает ГЗ.

ER-QBE реализован в программе «Селектор» СУБД «МикроПоиск» [17]. В «Селекторе» имена атрибутов выводятся в колонку «A», условия — в колонки «B», «C» и т.д. (рис.1).

Имя вершины и имена атрибутов в каждой форме можно изменять в соответствии с требуемой схемой XML. В «Селекторе» имена вершин и атрибутов выводятся, как в XML-документе. Однако в другой реализации может быть дополнительно предусмотрена визуализация исходных имен из концептуальной и/или логической схемы БД.

Принципы формирования условий на скалярные атрибуты в рамках одной формы аналогичны QBE. Условия в одной колонке объединяются конъюнкцией, группы условий разных колонок — дизъюнкцией. В условиях разрешено использовать скалярные значения (строки, числа), другие атрибуты той же вершины, атрибуты вершин, предшествующих этой вершине в дереве обхода подсхемы в формате <вершина>.<атрибут> или <вершина>[<индекс>].<атрибут>. Отсутствие индекса интерпретируется как указание на первый экземпляр вершины <вершина> среди предков текущей вершины в дереве обхода. Положительный индекс обозначает порядковый номер экземпляра вершины <вершина> среди предков текущей вершины в дереве обхода, считая от корня, модуль отрицательного индекса — порядковый номер, считая от текущей вершины. Если условие содержит указание на вершину, которой среди предков текущей вершины в дереве обхода нет, оно игнорируется. Условия по связи между вершинами поддерживаются автоматически и задавать их в форме не нужно.

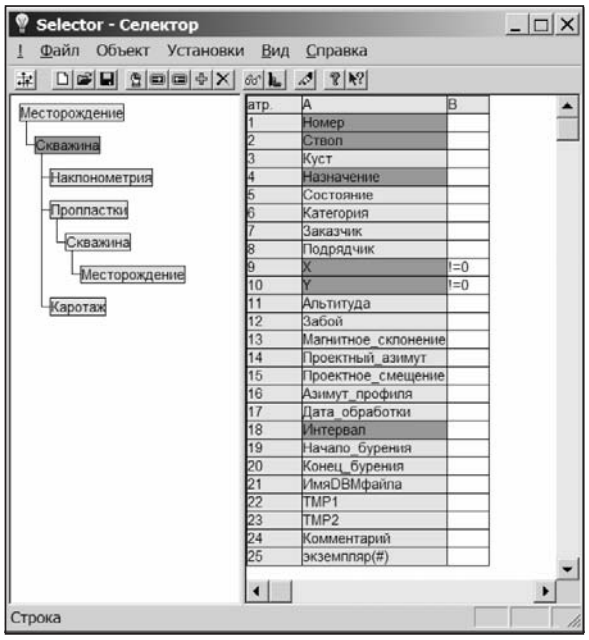

Рис. 1. ER-QBE в программе «Селектор»

В дополнительные строки можно добавлять вычисляемые атрибуты в виде выражений над теми же элементами: скалярными значениями, атрибутами текущей вершины и предшествующих ей вершин. Графовые запросы с дополнительными строками в вершинах не поддерживают обновлений XML-представления.

ССС, сформированная в результате выполнения ГЗ, может быть по-разному представлена в XML. Концепции ССС наиболее отвечает схема XML-документа, в которой объекты ССС представлены элементами XML, а атрибуты — атрибутами. Однако с точки зрения совместимости с SQL/XML целесообразна поддержка альтернативного представления атрибутов ССС в виде элементов XML с возможностью кодировать их дополнительные свойства. Например, признак отсутствия значения (NULL) кодируется значением атрибута xsi:nil="true". Выбор варианта представления атрибутов — параметр настройки ER-QBE. В «Селекторе» — это общая настройка на весь XML-документ, как FOR XML AUTO в SQL/XML. В другой реализации может быть предусмотрена индивидуальная настройка представления каждого атрибута, как в XDR. Однако гибкости FOR XML IMPLICIT ER-QBE не достигает: полнота ГЗ относится к информации в XML-документе, а не к ее форматированию. (Например, в реляционных базах SQL и QBE не обеспечивают гибкости генератора отчетов.)

В целом пользовательский интерфейс ER-QBE очень близок к интерфейсу традиционных QBE-подобных генераторов запросов. Его особенности:

- вертикальное, а не горизонтальное расположение атрибутов в форме (оно соответствует вертикальному представлению элементов одного уровня при визуализации XML-документа стандартными программами, например Internet Explorer);
- атрибуты разных классов не собираются в общую форму, а рассматриваются отдельно.

2.2. Реализация XML-представлений основана на трансляции ГЗ, описанного ER-QBE, в набор обычных запросов к БД. Будем придерживаться синтаксиса языка SQL, который обеспечивает универсальную синтаксическую основу подавляющего большинства современных языков запросов реляционных, объектно-реляционных и объектно-ориентированных БД.

Пусть $\Lambda_S^{\Gamma_2}$ — язык вмещающей СУБД для БД со схемой S, дополненной синонимами из подсхемы Γ_2. Обозначим ГЗ $G_2 = (\Gamma_2, \Sigma)$, где Γ_2 — подсхема, а $\Sigma \subseteq \Lambda_S^{\Gamma_2} \times T_2$ — множество ограничений, сопоставленных терминалам ГЗ.

Определим логическое ограничение, наложенное на правильную цепочку γ подсхемы ССС Γ_2, как конъюнкцию ограничений, сопоставленных вершинам и дугам γ_k. Сопоставим этой цепочке последовательность δ имен вершин подсхемы, пройденных при обходе подсхемы согласно цепочке γ, а именно $\delta_1 = n(\gamma_1)$ в первой позиции и $\delta_k = Z$, если $(p_\gamma^1 : X \to \gamma Z) \in P_{L_2}^1 \wedge \gamma = \gamma_k$ для k-й позиции.

Обозначим λ_k проиндексированные номером вхождения символы последовательности δ имен вершин подсхемы.

Сформируем запрос на чтение данных из БД по следующим правилам.

• В части SELECT запроса перечисляются выбранные атрибуты с префиксом λ_k, уточняющим, к какому символу цепочки относится атрибут. Для статических XML-представлений в формулы вычисляемых атрибутов префиксы λ_k атрибутов-аргументов подставляются аналогичным образом. Экземпляры, идентификаторы которых не специфицированы в части SELECT какого-нибудь запроса, в XML-документе не отображаются (пропускаются), хотя и участвуют в его формировании. Для динамических XML-представлений отсутствующие атрибуты-идентификаторы экземпляров подставляются автоматически согласно ограничению 2 (п. 2.1).

• В части FROM перечисляются классы $f(\delta_k)$ схемы ССС с указанием в качестве синонимов соответствующих символов λ_k.

• В части WHERE для спецификации условий используется соответствующее исчисление. Для классической реляционной модели — это исчисление предикатов первого порядка. Условия соединения по связи вставляются в конъюнкцию автоматически с учетом степени зависимости дуги: в коммерческих СУБД помимо простого соединения (inner join) — $\lambda_k\ f(\gamma_k) = \lambda_{k+1} f(\gamma_k)$ можно применять и внешнее соединение (left outer join) — $\lambda_k f(\gamma_k)^* = \lambda_{k+1} f(\gamma_k)$. В случае внешнего соединения λ_k должны выводиться в XML независимо от наличия связанных с ними λ_{k+1}. К условиям соединения посредством конъюнкций добавляются ограничения, соответствующие символам цепочки: $\{\sigma \,|\, (\sigma, \gamma_k) \in \Sigma\}$.

• Для упорядочения результатов запроса добавляется ORDER BY с перечислением атрибутов точно таким, как в части SELECT.

Описанные правила обеспечивают динамическую (в процессе формирования XML-представления) трансляцию ГЗ в совокупность запросов СУБД.

Соединение, полученное в результате выполнения сформированного описанным образом запроса по одной правильной цепочке подсхемы, соответствует фрагменту XML-документа — лесу деревьев обхода ССС. Восстановить иерархическую структуру достаточно просто, так как при последовательном просмотре записей о переходе к следующему экземпляру на каждом уровне иерархии сигнализирует изменение соответствующего ключевого атрибута.

2.3. Динамические (виртуальные) XML-представления реализуются современными СУБД в форме URL-запросов на основе SQL/XML или запросов XPath на основе XDR. Обе технологии хорошо согласуются с ГЗ. Если к запросу, сгенерированному для одной правильной цепочки подсхемы, добавить часть FOR XML AUTO или FOR XML AUTO ELEMENTS, СУБД, поддерживающие SQL/XML, такие как Microsoft SQL Server, восстановят иерархическую структуру XML-представления по правилам, аналогичным правилам ГЗ. Разным исходным таблицам сопоставляются разные уровни иерархии, атрибуты представляются атрибутами или элементами, в зависимости от отсутствия или наличия уточнения ELEMENTS. Например, для запроса

```
SELECT Customers.CustomerID, Customers.ContactName, Orders.OrderID
FROM Customers, Orders
WHERE Customers.CustomerID = Orders.CustomerID
FOR XML AUTO
```

результат может иметь вид

```
<Customers CustomerID="ALFKI" ContactName="Maria Anders">
    <Orders OrderID="10643"/>
    <Orders OrderID="10692"/>
    <Orders OrderID="10702"/>
    <Orders OrderID="10835"/>
    <Orders OrderID="10952"/>
    <Orders OrderID="11011"/>
</Customers>
```

а для запроса

```
SELECT Customers.CustomerID, Customers.ContactName, Orders.OrderID
FROM Customers, Orders
WHERE Customers.CustomerID = Orders.CustomerID
FOR XML AUTO, ELEMENTS
```

иметь вид

```
<Customers>
  <CustomerID>ALFKI</CustomerID>
  <ContactName>Maria Anders</ContactName>
      <Orders><OrderID>10643</OrderID></Orders>
      <Orders><OrderID>10692</OrderID></Orders>
      <Orders><OrderID>10702</OrderID></Orders>
      <Orders><OrderID>10835</OrderID></Orders>
      <Orders><OrderID>10952</OrderID></Orders>
      <Orders><OrderID>11011</OrderID></Orders>
</Customers>
```

Для получения полного динамического XML-представления ГЗ достаточно перечислить URL-запросы, соответствующие всем правильным цепочкам подсхемы в тексте запроса XQuery, например:

```
<result>
{
    doc("http://IISServer/db1?sql=SELECT+*+FROM+Providers+FOR+XML+AUTO
    &root=Providers")
    doc("http://IISServer/db1?sql=SELECT+CustomerID+ContactName+FROM
    +Customers+Orders+FOR+XML+AUTO&root=Customers")
}
</result>
```

Применение запросов XPath аналогично применению URL-запросов, за исключением того, что по каждому подзапросу типа SELECT нужно создавать реляционное представление (CREATE VIEW...) и генерировать XDR-схему — описание соответствия между схемой реляционного представления и схемой соответствующего виртуального XML-представления. Механизм ГЗ обеспечивает автоматическое формирование XDR-описаний с выбором формы представления каждого атрибута БД — более гибких, чем FOR XML AUTO, но не FOR XML IMPLICIT. С точки зрения простоты реализации использование URL-запросов на основе SQL/XML для ГЗ предпочтительнее.

2.4. Автоматическое обновление БД на основе изменений XML-представления в общем случае — трудная задача (в связи с возможностью проявления побочных эффектов). Основные подходы в этой области:

1) сведение задачи к обновлению реляционных представлений;

2) выделение классов иерархических представлений, для которых операции изменения значений атрибутов, вставки и удаления могут быть оттранслированы в безопасную с точки зрения побочных эффектов последовательность операций.

В рамках второго подхода на основе модели вложенных отношений выделен класс хорошо вложенных NPSJ-представлений (от Nest-last Project-Select-Join — вложенность в форме проекции результатов простого отбора и объединения), обладающий указанным свойством [18]. Признак данного класса — наличие единствен-

ной записи-источника (экземпляра) любого элемента иерархии. Хорошая вложенность не гарантирует корректности любой модификации XML-представления. Она гарантирует только внутреннюю непротиворечивость модифицированного XML-документа: изменение каждого атрибута и элемента выполняется независимо от других (отсутствие избыточности). Однако изменение значения может привести к нарушению ограничений по значениям, заложенных в ГЗ, или условий целостности БД.

Для XML-представления на основе ER-QBE принадлежность к классу хорошо вложенных NPSJ-представлений обеспечивается алгоритмом трансляции ГЗ в язык вмещающей СУБД, изложенным в п. 2.2 при выполнении следующих ограничений:

1) отображение n_2 вершин подсхемы на классы схемы — взаимно однозначное: $q \in N_2 \wedge p \in N_2 \wedge q \neq p \Rightarrow n_2(q) \neq n_2(p)$;

2) естественный иерархический порядок: $p_\alpha^2 \in P_{L2} \cup P_{L2}^* \wedge (f(\alpha), s, t) \in \in L_1 \Rightarrow \forall t \exists ! s$, иначе говоря, $n_2(\alpha)$ — связь типа $(1{:}N)$ или $(0{:}N)$.

Поскольку обновляемые XML-представления на основе ER-QBE всегда содержат ID экземпляра, стандартные параметрические запросы СУБД на изменение значений атрибутов, вставку и удаление экземпляра для каждой вершины ГЗ генерируются элементарно. Однако в реальных приложениях их, как правило, необходимо дорабатывать. В СУБД «МикроПоиск» эти запросы хранятся в виде исходных текстов параметрических запросов в вершинах ГЗ. Отсутствие текста запроса означает запрет соответствующей модификации через изменение XML-представления. В отличие от классического QBE, ER-QBE не предоставляет средств визуального формирования сложных запросов на модификацию данных в БД. Выбор сделан в пользу простоты, так как в рамках одного ГЗ таких запросов понадобилось бы в три раза больше числа вершин и в каждом из них могли бы потребоваться данные из таблиц, не используемых при формировании XML-представления.

Более актуальным для обновления XML-представления на основе ГЗ является вопрос генерирования последовательности необходимых изменений БД в связи с поступлением измененного XML-документа. Для хорошо вложенных XML-представлений порядок модификации совпадает с порядком следования символов вершин λ_k в строках грамматики подсхемы. Пусть URL исходного XML-документа — π_1, а URL модифицированного XML-документа — π_2. Для формирования в виде XML-документа задания на модификацию экземпляров, соответствующих элементам λ_k, воспользуемся языком XQuery, который в начале 2007 года рекомендован консорциумом W3C для преобразования XML-данных в Интернет-приложениях [8, 19]. Будем использовать полуформальные выражения XPath, имея в виду, что текст XQuery автоматически генерируется для каждого символа каждой правильной цепочки подсхемы:

<result>{
  let $a := doc("$\pi_1$")/$\lambda_1$ / λ_2 /…/ λ_k
  let $b := doc("$\pi_2$")/$\lambda_1$ / λ_2 /…/ λ_k
  <delete>{
    for $c in $a
    where not exists($b[@ID=$c/@ID])
    return(<entity dbname="$n_2(\lambda_k)$" key="$n_2(\lambda_k$.ID)" ID="{$c/@ID/text()}">)
  }</delete>
  <insert>{
    for $c in $b
    where not exists($a[@ID=$c/@ID])
    return(<entity dbname="$n_2(\lambda_2)$" refname="$n_2(\lambda_{k-1})$" refkey="$n_2(\lambda_{k-1}$.ID)"
    refID="{$c/../../@ID/text()}" ID="{$c/@ID/text()}">
    for $d in $c/@*
    return(<attribute dbname="$n_2(\lambda_k$.[$d / nmtoken()])">{$d/text()}</attribute>)
    for $d in $c/*
    where(not(exists($d/*)))
    return(<attribute dbname="$n_2(\lambda_k$.[$d / nmtoken()])">{$d/text()}</attribute>)
    </entity>)
  }</insert>

```
    <update>{
        for $c in $b, $d in $a
        where $c/@ID=$d/@ID and exists(return(
        for $e in $c/@*, $f in $d/@*
        where $e/nmtoken()=$f/nmtoken() and $e/text()!=$f/text()
        return (<dummy/>)
        for $e in $c/*, $f in $d/* where (not exists($e/*))
        and $e/nmtoken()=$f/nmtoken() and $e/text()!=$f/text()
        return (<dummy/>)
        ))
        return(<entity dbname="n2(λk)" key="n2(λk .ID)" ID="{$c/@ID/text()}">
        for $e in $c/@*, $f in $d/@*
        where $e/nmtoken()=$f/nmtoken() and $e/text()!=$f/text()
        return(<attribute dbname="n2(λk .[$e / nmtoken()])">{$e/text()}</attribute>)
        for $e in $c/*, $f in $d/* where (not exists($e/*))
        and $e/nmtoken()=$f/nmtoken() and $e/text()!=$f/text()
        return(<attribute dbname="n2(λk .[$e / nmtoken()])">{$e/text()}</attribute>)
        </entity>)
    }</update>
}</result>
```

В результате выполнения описанного запроса для текущего символа λ_k правильной цепочки подсхемы γ формируется XML-документ с тремя разделами: <delete> (удалить), <insert> (вставить), <update> (изменить), в которых перечислены экземпляры БД и атрибуты, подлежащие модификации на данном шаге. Для вставки сохраняется также информация об экземпляре на предыдущем уровне иерархии. Собственный идентификатор записи выводится в этом случае только для обеспечения ссылочной целостности на следующих уровнях иерархии (для следующего символа цепочки γ, если он есть). Запрос работает для XML-представлений с атрибутами БД как в форме элементов, так и в форме атрибутов XML.

3. ПРИМЕНЕНИЕ XML-ПРЕДСТАВЛЕНИЙ ДЛЯ ИНТЕГРАЦИИ ДАННЫХ В ПРОМЫСЛОВОЙ ГЕОФИЗИКЕ

Обработка и интерпретация геофизических данных — сложный процесс, включающий несколько уровней детализации. На каждом уровне применяются отдельные методы исследования, технологии обработки и интерпретации, программные решения, а для обмена данными между ними используются как внутренние, так и открытые форматы данных. Задачи интерпретации данных геофизических исследований в большинстве случаев относятся к классу недоопределенных многопараметрических задач с недостоверными входными данными, поэтому геолого-геофизические исследования представляют собой итеративный процесс. В связи с поступлением дополнительной информации регулярно проводится переинтерпретация данных, переподсчет запасов и т.п. Геофизическая аппаратура, методы интерпретации и методы обработки данных постоянно совершенствуются, поэтому ограничиться внедрением единственной программной системы с единой БД нереально. В добывающих и геофизических компаниях принято хранить все исходные данные и варианты их интерпретации. К настоящему времени накоплены огромные объемы такой информации в виде файлов и БД различных программ, часто устаревших. Актуальной задачей является создание и поддержка информационных хранилищ геофизической информации [20]. Информационное хранилище должно обеспечивать единую, полноценную, систематизированную и непротиворечивую совокупность данных, извлеченных из различных источников и ставших доступными конечным пользователям для понимания и применения.

Для синхронизации геофизических данных между различными БД, в частности с целью поддержки информационного хранилища, XML обладает рядом важных преимуществ по сравнению с таблицами БД:

• иерархическая структура XML-документов соответствует общей иерархической структуре геофизических данных (регион-месторождение-куст-скважина-ствол-исследование) и облегчает представление многообразия вложенных иерархий на одном уровне (с месторождением связаны пласты, разломы, скважины, сейсмические исследования и т.п., в свою очередь структурированные); реляционная модель «переворачивает» эту иерархию, так как связи $1{:}N$ реализуются внешними ключами и ограничениями целостности, и разбивает на фрагменты, поскольку представление всей базы данных одним «суперотношением» неэффективно;

• конструкция «//» XQuery позволяет искать данные на любом уровне иерархии без детальной спецификации пути; в SQL-запросах необходимо учитывать все промежуточные уровни иерархии, т.е. детально знать схему каждого источника данных;

• XQuery позволяет переносить вложенные иерархии целиком; в реляционных базах для реализации одной такой операции нужно написать процедуру, состоящую из специальной последовательности запросов;

• традиционные клиент-серверные интерфейсы СУБД требуют наличия специального программного обеспечения для доступа к БД (драйверов, библиотек). Для интеграции данных двух разных БД необходимо организовать совместную работу с двумя СУБД. XML-представления построены по принципу «тонкого клиента» — не требуют администрирования и специального программного обеспечения на стороне клиента;

• реализовать экспорт в XML данных любого известного формата, как и экспорт XML из программ, поддерживающих генерацию текстовых отчетов, гораздо легче, чем обеспечить доступ к таким данным в SQL (например, путем разработки ODBC- или OLE-DB-провайдера); в геофизических файловых архивах такие специальные данные преобладают, предварительное сохранение всех данных файла в один XML-документ уменьшает риски рассогласования данных и потери части информации;

• XML — основа межпрограммного взаимодействия согласно международным стандартам ebXML (ISO/TS 15000:2004) и OpenXML (ECMA 376).

В рамках описанной задачи интеграции данных целесообразно пренебречь производительностью (скоростью выполнения автоматических операций) ради снижения объема ручного труда. Часто архивные системы БД успевают морально устареть до того, как в них удается организовать информационный поиск по десяткам или сотням тысяч скважин геофизического архива.

Рассмотрим вариант использования XQuery для сведения к унифицированной форме данных XML-представлений различных БД на примере интеграции данных пакетов «ГеоПоиск» [21] (СУБД «МикроПоиск»), и «Tesseral Pro» (СУБД Microsoft SQL Server 2003). Экспорт XML-документов, фрагменты которых показаны на рис. 2, выполнен с помощью программы «Селектор». Известны несколько реализаций XQuery: Saxon [22]), Galax [23], Quizx [24], SEDNA [25]. Для выполнения XQuery-запросов использован Saxon. На рис. 3 приведен XQuery-запрос, написанный для их интеграции.

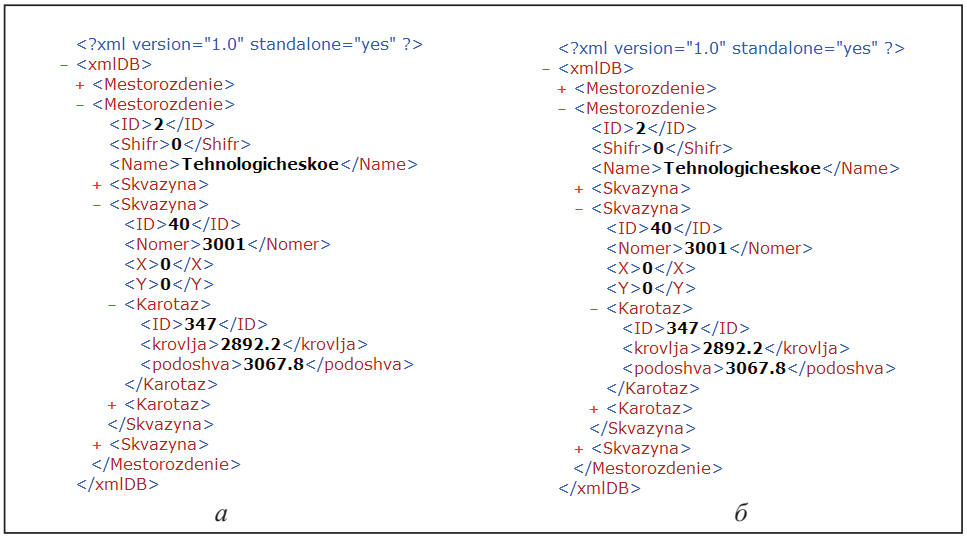

Рис. 2. XML-документы для БД: *а* — месторождение-скважина-каротаж; *б* — месторождение–скважина–пропластки

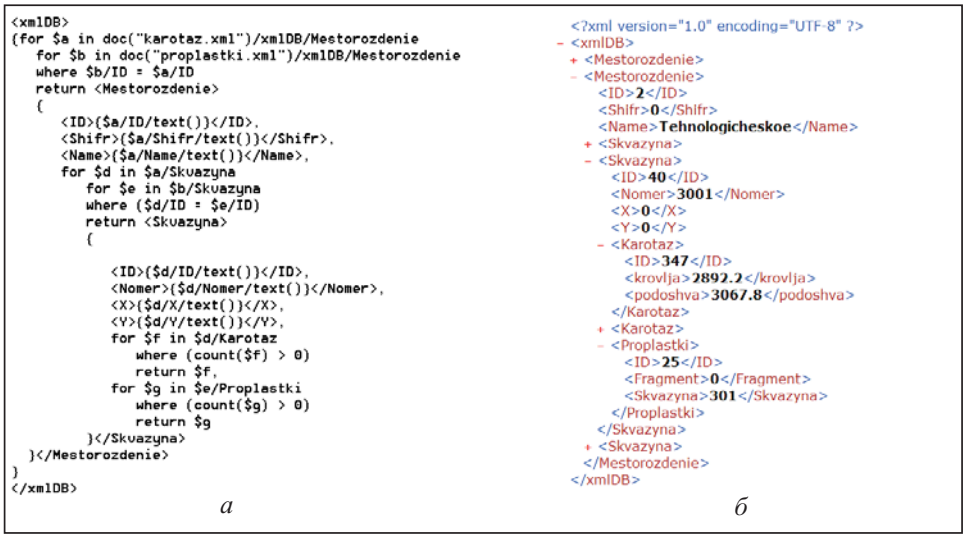

Рис. 3. Интеграция данных систем «ГеоПоиск» и «Tesseral Pro»: *а* — XQuery-запрос; *б* — результат интеграции XML-документов

ЗАКЛЮЧЕНИЕ

Язык ER-QBE на основе графовых запросов обеспечивает интуитивный визуальный способ спецификации XML-представлений реляционных, объектно-реляционных и объектно-ориентированных БД. Графовые запросы обладают полнотой с точки зрения выделения иерархически-упорядоченных фрагментов БД на основе ER-подобной концептуальной модели. ER-QBE просто и эффективно реализуются в любой СУБД с SQL-подобным языком запросов. Реализация ER-QBE автоматически обеспечивает возможность обновления БД на основе изменения XML-представлений, в сочетании с SQL/XML поддерживает динамические XML-представления. Совместное использование графовых запросов и XQuery позволяет существенно упростить интеграцию данных разных БД, поддержку согласования данных в информационных хранилищах. Реализация ER-QBE в составе СУБД «МикроПоиск» опробована в задачах интеграции геофизических данных. В ходе дальнейшего исследования представляет интерес рассмотрение возможности применения графовых запросов для спецификации преобразования XML-документов средствами XQuery.

ПРИЛОЖЕНИЕ

Доказательство теоремы 1. Рассмотрим произвольную грамматику ССС Г. Построим по $L^*(\Gamma)$ граф переходов автомата в виде дерева с ветвями, соответствующими правильным цепочкам. По построению он является ССС. Язык грамматики Γ_1 по построению — сокращение языка грамматики $\Gamma: L^*(\Gamma) = L(\Gamma_1)$.

Покажем, что любая грамматика ССС с конечным языком имеет полную подсхему с конечным языком. Для этого достаточно привести алгоритм построения такой подсхемы. Обозначим k-й символ терминальной цепочки $\gamma \in L(\Gamma_1)$ как γ_k. Рассмотрим пары вида $o = [f(\gamma_k), k]$, $n(o) = [n(f(\gamma_k)), k]$ в качестве символов алфавита с индексом k. Пусть

$$\Gamma_2 = (N_2 = \{n(o) \mid o \in O \times \tilde{N}\} \cup \{A_2\},$$

$$T_2 = (O \cup B) \times \tilde{N},\ A_2,\ P_2 = P_{E2} \cup P_{E2}^* \cup P_{L2} \cup P_{L2}^*),$$

где $\tilde{N}$ — множество целых чисел,

$$P_{E2} = \{p_x^2 : A_2 \to [f(x), 1]\, n([f(x), 2]) \mid \gamma \in L(\Gamma_1) \wedge x = \gamma_1\},$$

$$P_{E2}^* = \{p_x^2 : A_2 \to [f(x), 1] \mid \gamma \in L(\Gamma_1) \wedge x = \gamma_1\},$$

$$P_{L2} = \{p_y^2 : n([f(n^{-1}(X)), k]) \to [f(y), 1]\, n([f(n^{-1}(Z)), k+1]) \mid \gamma \in L(\Gamma_1) \wedge$$
$$\wedge\, y = \gamma_k \wedge (X \to yZ) \in P_{L1}\},$$
$$P_{L2}^* = \{p_y^2 : n([f(n^{-1}(X)), k]) \to [f(y), 1] \mid \gamma \in L(\Gamma_1) \wedge y = \gamma_k \wedge (X \to y) \in P_{L1}^*\}.$$

Пометим каждую продукцию ее нижним индексом — терминальным символом соответствующей продукции из Γ_1.

Условия 1–5 подсхемы обеспечиваются для Γ_2 по построению.

Если цепочка $\gamma \in L(\Gamma_1)$ и $\delta \le \gamma$, по построению $\delta \in L(\Gamma_1)$. Так как язык $L(\Gamma_1)$ конечный, длина цепочек ограничена сверху. Таким образом, индексы нетерминалов Γ_2 ограничены. Если продукция Γ_2 содержит в правой части нетерминал, его индекс на единицу превосходит индекс нетерминала в ее левой части. Отсюда следует, что длина цепочек, выводимых в Γ_2, ограничена. Следовательно, язык $L(\Gamma_2)$ — конечный.

Докажем, что любой терминальной цепочке γ грамматики Γ_1 соответствует хотя бы одна последовательность продукций Γ_2, выводящая терминальную цепочку. Для цепочек из одного символа это утверждение очевидно. Пусть оно очевидно для цепочек длиной k. Для $|\gamma| = k+1$ заменим последнюю продукцию в последовательности, выводящей цепочку, соответствующую $\gamma_1\gamma_2 \ldots \gamma_k$, на парную к ней с нетерминалом в правой части. По построению Γ_2 содержит продукцию, приводящую последний нетерминал к терминальному символу. По индукции Γ_2 является полной подсхемой Γ_1.

Теорема доказана.

Доказательство теоремы 2. Достаточно привести алгоритм построения ГЗ, удовлетворяющего условию данной теоремы. Разобьем исходное множество цепочек $L'(\Gamma_1)$ на однотипные группы. Для этого каждой цепочке экземпляров γ сопоставим такую цепочку классов η, что $\eta_k = f(\gamma_k)$, и отнесем к одной группе s_η все цепочки $\gamma \in L'(\Gamma_1)$, сопоставленные η. Рассмотрим граф переходов автомата в виде дерева с ветвями, соответствующими избранным из каждой группы представителям. По построению он является ССС с конечным языком. Обозначим соответствующую грамматику Γ_2; ее можно выбрать в качестве полной подсхемы Γ_1 по определению. По построению $L(\Gamma_2)$ содержит ровно одну цепочку для каждой терминальной вершины. Нагрузим эту цепочку η ограничением на языке вмещающей СУБД так, чтобы запрос, построенный по описанным в п. 2.2 правилам, описывал подмножество $s_\eta \in L'(\Gamma_1)$. Это возможно по условию. Сопоставим полученное ограничение терминальной вершине Γ_2, соответствующей последнему символу цепочки η. Полученный ГЗ назовем $G_2 = (\Gamma_2, \Sigma)$. Обозначим множество цепочек G_2 как $M(G_2)$. По построению $(M(G_2) \subseteq L'(\Gamma_1)$. Допустим, $\exists\, \gamma \in L'(\Gamma_1) \wedge \gamma \notin M(G_2)$. В силу того, что Γ_2 — полная подсхема, терминальной цепочке γ соответствует ровно одна терминальная цепочка $\eta \in L(\Gamma_2)$. Следовательно, γ не удовлетворяет единственному условию, описывающему s_η. Получено противоречие.

Теорема доказана.

СПИСОК ЛИТЕРАТУРЫ

1. Bray T., Paoli J. Extensible Markup Language (XML) 1.0 (2nd Edition) / Sperberg-McQueen (Eds). — W3C Recommendation. (http://www.w3.org/TR/2000/REC-xml-20001006)
2. Lee D., Chu W. Comparative analysis of six XML schema languages // ACM SIGMOD Record. — 2000. — N 29. — P. 76–87.
3. Shanmugasundaram J., Shekita E., Barr R. et al. Efficiently publishing relational data as XML documents // Proc. of VLDB. — Cairo, Egipt: ACM SIGMOD, 2000. — P. 65–76.
4. Carey M., Florescu D., Ives Z. et al. XPERANTO: Publishing object-relational data as XML // Informal Proc. of WebDB 2000. — Dallas, USA: ICM SIGMOD, 2000. — P. 105–110.
5. Rys M. Bringing the internet to your database: using SQLServer 2000 and XML to build loosely-coupled systems // Proc. of ICDE'01. — Heidelberg, Germany: IEEE Comp. Soc., 2001. — P. 465–472.
6. www.sqlxml.org.

7. Deutsch A., Fernandez M., Florescu D. et al. A query language for XML // Proc. of WWW8. — Toronto, Canada: CNRC, 1999. — P. 77–91.
8. Chamberlin D., Florescu D., Robie J. et al. XQuery: a query language for XML. — 2001. (http://www.w3.org/TR/query)
9. Rodriguez-Gianolli P., Mylopoulos J. A semantic approach to XML-based data integration // Proc. of ER'01. — Yokohama, Japan: Springer-Verlag, 2001. — P. 117–132.
10. Lee D., Mani M., Chiu F. et al. NeT and CoT: Translating relational schemas to XML schemas using semantic constraints // Proc. of ACM CIKM. — McLean, Virginia, USA: ACM SIGMOD, 2002. — P. 206–217.
11. Самохвалов Н.А. Поддержка обновлений XML-представлений над реляционными базами // Пробл. программирования — 2006. — № 2–3. — С. 455–457.
12. Гречко В.О., Тульчинский В.Г., Тульчинский П.Г. От QBE к графовым запросам // Тр. VI Междунар. конф. «Знание–Диалог–Решение». — Киев: Ассоциация создателей и пользователей интеллектуальных машин, 1997. — Т. 1. — С. 216–224.
13. Тульчинский В.Г., Тульчинский П.Г. Графовый прототип приложения // Пробл. программирования. — 2002. — № 1-2. — С. 489–498.
14. Agrawal S., Chaudhuri S., Narasayya V. Automated selection of materialized views and indexes in SQL databases // Proc. of VLDB, 2000. — Cairo, Egypt: VLDB, 2000. — P. 496–505.
15. Miller R.J., Haas L., Hernandez M.A. Schema mapping as query discovery // Proc. of VLDB, 2000. — Cairo, Egypt: VLDB, 2000. — P. 227–236.
16. Zloof M. M. Query by Example // Proc. of AFIPS, 1975. — Anaheim, CA, USA: AFIPS, 1975. — N 44. — P. 431–438.
17. Тульчинский В.Г., Тульчинский П.Г. Графовые запросы для интеграции данных посредством XML // Тр. III Междунар. конф. «Теоретичні та прикладні аспекти побудови програмних систем». — Киев: Київ. нац. ун-т ім. Т.Г. Шевченка. — 2006. — Т. 1. — С. 92–95.
18. Braganholo V. On Updatability of XML Views over Relational Databases // Proc. of WebDB 2003. — 2003. (http://www.cse.ogi.edu/webdb03/presentations/06.pdf)
19. Deutsch A., Fernandez M., Florescu D. et al. A query language for XML // Proc. of WWW8. — Toronto, Canada: CNRC, 1999. — P. 77–91.
20. Гречко В.О., Тульчинский В.Г., Тульчинский П.Г., Харченко А.В. Информационные хранилища в промысловой геофизике // Пробл. программирования. — 2000. — № 1–2. — С. 81–90.
21. www.geopoisk.com.
22. saxon.sourceforge.net.
23. www.galaxquery.org.
24. www.axyana.com/xquest.
25. modis.ispras.ru/sedna/index.htm.

Поступила 16.05.2007