Интеграция XML-коллекций данных в посреднике неоднородных коллекций электронных библиотек

Осипов М.А.
МГУ им. Ломоносова
osipov@newtech.ru

Калиниченко Л.А.
ИПИ РАН
leonidk@ipi.ac.ru

Аннотация

В статье рассматривается задача создания адаптера коллекций XML-документов в посреднике неоднородных информационных ресурсов. Так как технология XML развивается и уже существует множество программных средств, ориентированных на XML, предлагается использовать некоторые из этих средств, например XML-ориентированную СУБД Tamino, языком запросов которой является язык XQL. В рамках решаемой задачи потребовалось создание собственной алгебры на основе языка XQL.

1 ВВЕДЕНИЕ

Данная работа выполняется в рамках проектов РФФИ "Создание интегрированных библиотек на основе неоднородных распределенных электронных коллекций научной информации", грант (98-07-91061) и "Методы и инструментальные средства создания XML-базированных интегрированных информационных систем для научных исследований", грант (00-07-90086). В проекте по созданию интегрированных библиотек неоднородных ресурсов предлагается разработать инфраструктуру доступа к множественным неоднородным коллекциям информации. Основной идеей этой инфраструктуры является введение промежуточного слоя между электронными коллекциями и потребителями информации. Основными компонентами промежуточного слоя являются информационные посредники, существующие независимо от электронных коллекций. В посреднике различаются три уровня представления информации: локальный уровень, представляющий метаинформацию о разнородных коллекциях, федеративный уровень и персонализованный уровень. Подробную информацию о структуре посредника можно найти в [1].

Важным видом коллекций данных являются XML-коллекции данных, которые получают все большее распространение. XML (eXtensible Markup Language)[11] - язык разметки документов, позволяющий хранить документы в структурированном виде, является новым стандартом для представления и обмена информацией в Web. Несмотря на то, что большая часть информации в Web получена из структурированных источников данных, эта информация хранится в неструктурированном или слабоструктурированном виде, а именно, в виде текстовых документов с HTML-разметкой. В отличие от языка HTML, ориентированного только на форматную разметку документа, XML предоставляет средства структурированного представления информации. Структура XML-документов представляется посредством языка DTD (Document Type Definition). XML-элемент является основной логической единицей XML-документа. Один XML-элемент может быть вложен в другой. С каждым XML-элементом может быть связан список пар идентификатор атрибута - значение. Типы элементов и их атрибутов специфицируются DTD.

Канонической моделью данных в посреднике является модель данных языка СИНТЕЗ[2]. В качестве языка запросов для этой модели данных используется язык SOQL, который является подмножеством языка запросов OQL для объектной модели данных с изменениями, обусловленными спецификой модели данных СИНТЕЗ. Важным элементом, обеспечивающим интеграцию коллекций в посреднике, является адаптер. Адаптер - компонент архитектуры посредника, обеспечивающий доступ посредника к информационным коллекциям. Взаимодействие посредника и адаптера осуществляется средствами интерфейса JDBC[10]. Посредник посылает адаптеру запрос на языке `SOQL'`, являющийся подмножеством языка SOQL, в терминах локальной схемы и после выполнения адаптером этого запроса получает результирующее множество, используя интерфейс JDBC. Запросы на SOQL' имеют стандартную SELECT-FROM-WHERE форму. СИНТЕЗ способен обеспечить эквивалентное определение разнородных информационных моделей и поэтому обладает богатым набором средств для спецификации информационных ресурсов. Для выражения модели данных XML средствами языка СИНТЕЗ необходимо подмножество этого языка, относящееся к определению абстрактных типов данных. Правила отображения модели данных XML в СИНТЕЗ основаны на отображении спецификаций ти-

©Вторая Всероссийская научная конференция
ЭЛЕКТРОННЫЕ БИБЛИОТЕКИ:
ПЕРСПЕКТИВНЫЕ МЕТОДЫ И ТЕХНОЛОГИИ,
ЭЛЕКТРОННЫЕ КОЛЛЕКЦИИ
26-28 сентября 2000г., Протвино

пов XML-документов (DTD) в спецификацию абстрактных типов данных на языке СИНТЕЗ таким образом, что определение каждого типа XML-элементов в DTD соответствует спецификации абстрактного типа данных в СИНТЕЗе. Более подробно отображение DTD в СИНТЕЗ определено в [7].

В настоящей статье рассматривается реализация адаптера, позволяющего организовать взаимодействие XML-коллекций данных с посредником. При этом мы ограничиваемся рассмотрением типизированных XML-коллекций, т.е. таких, структуру которых определяет DTD. Структура данной статьи следующая: сначала будет рассмотрена структура адаптера, правила его взаимодействия с посредником, далее определяются правила отображения запросов на языке SOQL' в запросы на языке алгебры адаптера, правила представления объектов СИНТЕЗа на языке Java.

2 СТРУКТУРА АДАПТЕРА

При создании адаптера для XML-коллекций, прежде всего предполагается интегрировать коллекций XML-документов под управлением XML-ориентированной СУБД Tamino [8] фирмы Software-AG. Tamino - одна из первых СУБД, которая полностью ориентирована на технологию XML. Все документы в Tamino хранятся в формате XML, языком запросов, используемым в Tamino, является язык XQL [9] (XML Query Language). Схема базы данных под управлением Tamino описывается на языке XML-Schema [12] на основе спецификаций типов документов (DTD). Взаимодействие с данной СУБД осуществляется посредством протокола HTTP.

Интеграция XML-коллекций данных в посреднике требует спецификации отображения модели данных XML в каноническую модель данных посредника; правил отображения запросов на языке SOQL' в запросы на языке алгебры адаптера; правил регистрации XML-коллекций в посреднике. Язык SOQL' содержит ряд конструкций, отсутствующих в OQL. Например в SOQL' используются регулярных выражений в определении искомых образцов слабоструктурированных данных. Подобные конструкции присутствуют в ряде языков запросов, в том числе в языке XQL. В связи с тем, что язык запросов XQL является простым языком запросов для слабоструктурированных данных, его возможности слабее, чем у языков для структурированных "традиционных"моделей данных; создатели этого языка сделали все, чтобы упростить XQL. Поэтому некоторые запросы на SOQL' нельзя интерпретировать исключительно средствами XQL. Эти проблемы приходится решать адаптеру XML-коллекций самому, или на основе других средств манипулирования базой данных под управлением Tamino.

Предлагается осуществлять интеграцию коллекций XML-документов в посреднике следующим образом (рис. 2):

- Каждая коллекция XML-документов из базы данных СУБД Tamino регистрируется в посреднике при помощи специальных програмных средств (например, средства загрузки схемы локальной коллекции XML-документов в базу посредника, средства согласования контекстов, антологий)

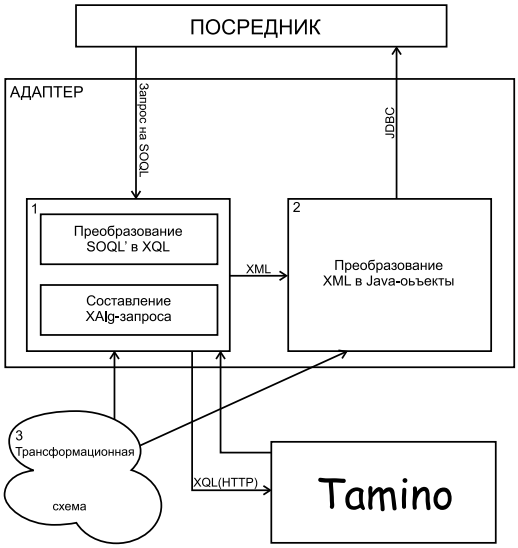

Рис. 1: Общая схема взаимодействия посредника с адаптером

- Адаптер интерпретирует запрос на SOQL', так что исходный запрос преобразуется в несколько запросов на XQL. Далее конструируется запрос на языке алгебры адаптера.

- Результаты запросов к Tamino преобразуются в Java-представление объектов канонической модели данных, и посредник получает окончательный результат, используя JDBC.

Трансформационная схема представляет информацию, необходимую для получения соответствия схемы локального уровня посредника и схемы конкретной коллекции. Основными функциями адаптера являются:

1. Преобразование SOQL' в XQL. Здесь происходит синтаксический анализ SOQL'-запроса и преобразование его во внутреннее представление. Далее производится сопоставление имен (типов и атрибутов), присутствующих в запросе, с именами конкретной коллекции. После этого конструируется один или несколько запросов на языке XQL, на основе которых создается запрос на языке алгебры адаптера, происходит интерпретация запросов. XQL-запросы посылаются СУБД Tamino посредством протокола HTTP.

2. Преобразование XML в Java-объекты. Здесь происходит преобразование XML в Java-представление объектов канонической модели данных и передача результирующего множества посреднику посредством JDBC-интерфейса.

3 АЛГЕБРА XML-КОЛЛЕКЦИЙ, ТИПИЗИРОВАННЫХ DTD

В силу того, что язык запросов SOQL' "сильнее"языка XQL, требуется расширенный язык, с помощью которого

можно было бы интерпретировать запросы языка SOQL'. Это обусловлено тем, что в языке XQL нет возможности осуществлять такие операции как join и projection, также в XQL нет возможности создания элементов новых типов.

3.1 Язык запросов XQL

XQL (XML Query Language) - является языком запросов для доступа и выбора элементов и текста из XML-документов. Запрос на языке XQL представляет собой путь в иерархии элементов, которую представляет XML-документ. Результатом XQL-запроса являются элементы, пути которых соответствуют спецификации запроса. В качестве примера DTD-спецификации в настоящей статье используется спецификация типов документов, содержащих информацию о статьях, опубликованных в журнале ACM SIGMOD Record (http://www.dia.uniroma3.it/Araneus/Sigmod/Record/DTD/). Соответствующая коллекция находится по адресу http://www.dia.uniroma3.it/Araneus/Sigmod/Record/SigmodRecord/SigmodRecord.xml. Примером XQL-запроса к данной коллекции является следующее выражение: 'SigmodRecord/issue/number'. Таким образом XQL-запрос это путь, элементами которого являются идентификаторы типов XML-элементов. В языке существуют средства задания фильтров. Фильтр является предикатом, который связывается с элементом запроса и означает, что необходимо выбирать только те XML-элементы, для которых условие верно. Например результатом запроса "SigmodRecord/issue/articles/article[authors/author = 'Tom Smith']/title"будут элементы типа title, для которых элемент author "родительского"элемента issue равен "Tom Smith", или иначе - "названия всех книг, автором которых является Tom Smith".

Пример 1: "SIGMOD Record DTD"

```
<!ELEMENT SigmodRecord (issue)*>
<!ELEMENT issue (volume,number,articles)>
<!ELEMENT volume (#PCDATA)>
<!ELEMENT number (#PCDATA)>
<!ELEMENT articles (article)*>
<!ELEMENT article (title,initPage,endPage,
authors)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT initPage (#PCDATA)>
<!ELEMENT endPage (#PCDATA)>
<!ELEMENT authors (author)*>
<!ELEMENT author (#PCDATA)>
<!ATTLIST author position CDATA #IMPLIED>
```

3.2 Алгебра адаптера

Алгебра адаптера включает операции join, select, projection, rename и constructor, а также запросы на любом XML-ориентированном языке запросов, порождающем множество XML-элементов, имеющих определенный тип, специфицированном DTD. Здесь таким XML-ориентированным языком является XQL. Запросы на этом языке (назовем его XAlg) определим следующим образом :

1. Запрос на языке XQL является XAlg-запросом.

2. Операция **projection**(query, attr1, attr2, ... , attrN), первым атрибутом которой является XAlg-запрос, а остальные атрибуты - идентификаторами атрибутов XML-элементов, на которые осуществляется проекция, является XAlg-запросом. Процедура projection осуществляет проекцию XML-элементов на определенные атрибуты, идентификаторы которых указаны как параметры данной операции.

3. Операция **join**(query1, query2), с атрибутами query(i), являющимися XAlg-запросами. Операция join является XAlg-запросом. Результатом операции join является множество элементов, тип которых определяется как соединение типов операндов join [4].

4. Операция **select**(query, path) с атрибутами query (XAlg-запрос) и path (спецификация пути в элементах результирующего множества query), является XAlg-запросом. Данная процедура выполняет выборку из результата XAlg-запроса. Использовать данную процедуру имеет смысл только над сложными XAlg-запросами, т.к. над простыми запросами операцию выборки может более эффективно осуществить Tamino. Технически операция select заключается в выборе в дереве разбора узлов, соответствующих параметру path, который является XQL-запросом.

5. Операция **rename**(query, oldname1, newname1, ...) с атрибутом query (XAlg-запросом) и атрибутами oldname(i) и newname(i) - списком переименования, является XAlg-запросом. Процедура rename служит для переименования типов элементов. В результате переименования создается новый тип элементов, результатом данной операции является множество элементов данного типа.

6. Операция **constructor**(query, dtdspec, queryelem1, constructelem1 ...) - с атрибутами dtdspec (DTD-спецификация), и queryelem(i), constructelem(i) (список пар идентификаторов элементов результата запроса query и DTD, соответственно) является XAlg-запросом. Результатом операции constructor является множество элементов типа, указанного в DTD.

Таким образом, если есть XML-документы, соответствующие DTD примера 1, и необходимо получить список, содержащий заголовок и аннотацию книг, автором которых является Tom Smith, то такой результат можно получить в результате выполнения следующего XAlg-запроса: projection(SigmodRecord/issue/aricles[authors/author = 'Tom Smith'], title, initPage).

3.3 Преобразование SOQL' в XAlg

Преобразование запроса на языке SOQL' в запрос на XAlg осуществляется в несколько этапов. На первом этапе преобразования происходит отображение всех идентификаторов и путей в исходном выражении SOQL'-запроса в идентификаторы конкретной коллекции XML-документов, согласно трансформационной схеме. На втором этапе к полученным на первом этапе путям доба-

вляются фильтры, соответствующие разделу WHERE исходного запроса. Таким образом мы получаем один или несколько запросов на языке XQL, которые соответствуют выражениям путей в запросе на SOQL'. На следующем этапе преобразования конструируется запрос на языке XAlg.

3.3.1 Регистрация коллекций в посреднике и трансформационная схема

При регистрации коллекции XML-документов в посреднике все спецификации типов данной коллекции(DTD) отображаются в спецификации абстрактных типов данных на языке СИНТЕЗ. Например, DTD из примера 1 можно отобразить в спецификации абстрактных типов данных на языке СИНТЕЗ так, как это сделано в примере 2. Это отображение сделано в соответствии с правилами, определенными в [7].

Пример 2: "Спецификация АТД на языке СИНТЕЗ"

```
{SIGMODRECORD;
in : type;
issue : {sequence;
        value:ISSUE;}
};
{ISSUE;
in:type;
        volume:VOLUME;
        number:NUMBER;
        articles:ARTICLES;
};
{VOLUME;
in:type;
string:{String;}
};
{NUMBER;
in:type;
string:{String;}
};
{ARTICLES;
in:type;
article:{sequence;
        value:ARTICLE;}
};
{ARTICLE;
in:type;
        title:TITLE;
        initPage:INITPAGE;
        endPage:ENDPAGE;
        authors:AUTHORS;
};
{TITLE;
in:type;
string:{String;}
};
{INITPAGE;
in:type;
string:{String;}
};
{ENDPAGE;
in:type;
string:{String;}
};
{AUTHORS;
in:type;
author:{sequence;
        value:AUTHOR;}
};
{AUTHOR;
in:type;
string:{String;}
};
```

Трансформационная схема хранит информацию, необходимую для обратного отображения, т.е. соответствие идентификаторов типов, слотов в СИНТЕЗе типам и атрибутам XML-элементов конкретной коллекции.

Трансформационная схема представляется в виде таблицы, состоящей из двух столбцов. Таблица определяет соответствие идентификаторов и путей в спецификации схемы коллекции на языке СИНТЕЗ (локальный уровень) и идентификаторов и путей коллекции XML-документов. Также трансформационная схема содержит информацию, необходимую для доступа к коллекции: URL базы данных, к которой принадлежит данная коллекция.

3.3.2 Обработка выражений путей

На данном этапе преобразования SOQL' запроса в запросы на XQL происходит выбор из выражения SELECT всех идентификаторов и путей и замена их соответствующими конструкциями, согласно трансформационной схеме. Например, для запроса

```
SELECT SIGMODRECORD.issue.value.articles.
article.value.title
from sigmodrecord
WHERE SIGMODRECORD.issue.value.number.
string = 1;
```

образуются пути SigmodRecord/issue/articles/ article/title и SigmodRecord/issue/number. Причем, запросом для выполнения является только первое выражение. Если в разделе SELECT выражения запроса присутствует два или более элемента, специфицирующих XML-элементы, для которых существует общий родительский элемент, то выражения путей для этих элементов отображаются в одно выражение, соответствующее ближайшему общему родительскому элементу. Далее к элементам, соответствующим данному пути, будут применены операции projection и select.

Вместо выражений путей в запросе на языке `SOQL'` могут использоваться предикаты-образцы. Для этого в язык SOQL' были введены такие средства как wild card и конструкции, обозначающие итерации. Язык XQL также использует аналогичные средства. Таким образом, предикаты-образцы отображаются в соответствующие выражения в языке XQL.

3.3.3 Добавление фильтров

На данном этапе отображения на основе раздела WHERE выражения SELECT к полученным на предыдущем этапе XQL-запросам добавляются фильтры-предикаты. Например, для выражения SELECT из пункта 3.3.2

данной статьи к XQL-запросу добавляется фильтр [number=1]. Таким образом, в результате преобразования получается следующий запрос на языке XQL : SigmodRecord/issue[number = 1]/articles/article/title , в результате выполнения которого образуется множество, содержащее заголовки статей из первого номера журнала.

3.3.4 Конструирование запроса на языке XAlg

На данном этапе происходит создание XAlg-запроса согласно исходному запросу на языке SOQL' и полученным на предыдущих этапах запросам на языке XQL. Запрос на языке SOQL' можно логически разделить на три раздела: раздел SELECT, раздел FROM, раздел WHERE. В разделе SELECT специфицируется тип объектов результирующего множества. Данная спецификация содержит пути в объектах типов, идентификаторы которых представлены в разделе FROM. Каждый элемент в разделах SELECT и FROM может быть связан с идентификатором с помощью конструкции 'as'.

Первыми применяются операции select и projection к тем XQL-запросам, для которых существует несколько элементов с общим родительским элементом в разделе SELECT исходного SOQL'-запроса. Для преобразования SOQL'-запроса, содержащего только разделы SELECT и FROM, в запрос на языке XAlg, достаточно применить операцию rename к полученным на предыдущих этапах XQL-запросам, согласно конструкции 'as', и операцию construct к полученным после применения операции rename запросам. DTD-параметр операции construct создается на основе списка выборки раздела SELECT. Операция join выполняется на основании раздела WHERE и раздела SELECT, содержащего конструкции 'as' с одинаковыми идентификаторами.

3.3.5 Пример

Предположим, что места работы авторов определены другой коллекцией документов, спецификация типов элементов которой имеет вид:

```
<!ELEMENT JobPlaces
        (JobPlace)*>
<!ELEMENT JobPlace
(author, orgname,
address)>
<!ELEMENT author
        #PCDATA>
<!ELEMENT orgname
        #PCDATA>
<!ELEMENT address
        #PCDATA>
```

```
{JOBPLACES;
in : type;
JobPlace:{sequence;
    value:JOBPLACE;
          }
};
{JOBPLACE;
in : type;
author : AUTHOR;
orgname : ORGNAME;
address : ADDRESS;
};
{AUTHOR;
in : type;
string : string;
};
{ORGNAME;
in : type;
string : string;
};
{ADDRESS;
in : type;
string : string;
};
```

Чтобы узнать названия организаций, к которым принадлежат авторы статьи "Research in Knowledge Base Management Systems", можно задать запрос на языке SOQL':

```
SELECT JOBPLACES.JobPlace.value.
orgname.string as org
FROM jobplaces, sigmodrecord
WHERE JOBPLACES.JobPlace.value.
author.string =
SIGMODRECORD.issue.value.articles.value.
article.authors.author.value.string
AND
SIGMODRECORD.issue.value.articles.article.
value.title.string =
"Research in Knowledge Base
Management Systems."
```

На первом этапе преобразования происходит сопоставление путей в запросе и конкретной коллекции. В данном случае:

```
/JobPlaces/JobPlace/
SigmodRecord/issue/articles/article
```

На втором этапе к XQL-запросам добавляются фильтры. Таким образом получаются запросы:

```
/JobPlaces/JobPlace
/SigmodRecord/issue/articles/
article[title =
'Research in Knowledge Base
Management Systems.']
```

Далее из данных запросов конструируется сложный запрос на языке XAlg:

```
project(
join(
 rename(projection(/JobPlaces/JobPlace/,
        orgname, author), orgname, org),
 select(
   /SigmodRecord/issue/articles/
   article[title = 'Research in Knowledge
             Base Management Systems.']),
  authors/author)),
orgname)
```

4 ИНТЕРПРЕТАЦИЯ ЗАПРОСОВ НА ЯЗЫКЕ XALG В АДАПТЕРЕ

Запрос на языке XAlg является либо одной из операций select, projection, join, rename или constructor, либо XQL-запросом. Назовем XQL-запросы в выражении XAlg-запроса простыми запросами. Адаптер выполняет XAlg-запросы по следующему плану:

1. Выполняются все простые запросы.
2. Над результирующими множествами, полученными в результате выполнения простых запросов, выполняются операции select, join, projection, rename и constructor. Их применение может быть рекурсивным.

Первый этап планирования состоит в том, что адаптер посылает запросы XML-ориентированной СУБД Tamino и сохраняет результаты. Второй этап адаптер осуществляет сам.

4.1 Выполнение простых запросов

Взаимодействие адаптера и СУБД Tamino осуществляется посредством протокола HTTP. Адаптер получает из трансформационной схемы URL всех коллекций XML-документов, участвующих в запросе, и создает простые запросы. Пусть <Collection-URL> - URL определенной коллекции, а <XQL-Query> - простой запрос к данной коллекции, тогда адаптер получает результирующее множество, соответствующее запросу <XQL-Query>, по адресу "<Collection-URL>?_xql=<XQL-Query>". Полученный результат обрабатывается и сохраняется для дальнейшего использования.

4.2 Выполнение сложных запросов

Сложный запрос является одной из процедур языка XAlg - select, projection, join, rename или constructor. Данные процедуры выполняет непосредственно адаптер. Множества, полученные в результате выполнения простых XQL-запросов, поступают на вход лексическому анализатору [13]; далее полученное дерево разбора XML-документов используется при выполнении сложных XAlg-запросов. Результатом выполнения сложного запроса также является иерархия элементов, над которой может быть выполнен сложный XAlg-запрос.

4.3 Динамическая база метаданных запроса

Выполнение операций rename, projection, join и constructor приводит к созданию новых типов элементов. Все созданные типы элементов сохраняются в определенной структуре, которую назовем базой метаданных запроса. Также в базе метаданных сохраняются соответствия полученных динамически типов и типов локального уровня посредника. База метаданных необходима для преобразования элементов результирующего множества запроса в Java-представление объектов языка СИНТЕЗ.

5 JAVA-ПРЕДСТАВЛЕНИЕ ОБЪЕКТОВ НА ЯЗЫКЕ СИНТЕЗ

В результате выполнения XAlg-запроса, соответствующего запросу на языке SOQL, получается множество, состоящее из элементов строкового типа, абстрактного типа данных, либо типа структуры (structure). Необходимо определить Java-представление объектов данных типов, а также типов union и sequence языка СИНТЕЗ. Для строкового типа данных, его аналогом в языке Java является тип String.

6 ЗАКЛЮЧЕНИЕ

В данной статье предложена архитектура адаптера коллекций XML-документов, алгебра XML-коллекций, типизированных DTD. Определены правила отображения запросов на языке SOQL' модели данных языка СИНТЕЗ в алгебру XAlg, представление объектов языка СИНТЕЗ в виде Java-объектов. Предложена структура адаптера для коллекций XML-документов, организованных средствами СУБД Tamino. Адаптер реализуется на языке Java в среде Windows NT.

Список литературы

[1] Kalinichenko L.A., Briukhov D.O., Skvortsov N.A., Zakharov V.N., Infrastructure of the subject mediating environment aiming at semantic interoperability of heterogeneous digital library collections. Institute of Problems of Informatics RAS, Proceedings of the Second All-Russian Conference Digital Libraries 2000.

[2] Л.А. Калиниченко, СИНТЕЗ - язык определения, проектироания и программирования интероперабельных сред неоднородных информационных ресурсов, ИПИ РАН, Москва, 1993.

[3] Roy Goldman, Jason McHugh, Jennifer WidomFrom Semistructured Data to XML: Migrating the Lore Data Model and Query Language, ACM SIGMOD Workshop on WebDB'99, June 3-4, 1999

[4] Kalinichenko L.A. Compositional Specification Calculus for Information Systems Development. Proceedings of the East-West Symposium on Advances in Databases and Information Systems (ADBIS'99), Maribor, Slovenia, September 1999, Springer Verlag, LNCS, 1999

[5] ACM SIGMOD Record: XML Version, http://www.acm.org/sigmod/record/xml/

[6] Kalinichenko L.A.,Integration of Heterogeneous Semistructured Data Models in the Canonitial One, Proceedings of the First All-Russian Conference Digital Libraries, October 19-22, 1999

[7] Осипов М.А., Мачульский О.Л., Калиниченко Л.А., Отображение модели данных XML в объектную модель языка СИНТЕЗ, Труды Первой Всероссийской конференции "Электронные библиотеки", Санкт-Петербург, 19-22 октября 1999 г.

[8] Tamino - The Information Server for Electronic Business, http://www.softwareag.com/tamino/

[9] XQL Tutorial (XML Query Language), http://metalab.unc.edu/xql/xql-tutorial.html

[10] JDBC (TM) Technology, http://www.javasoft.com/products/ jdbc/index.html

[11] eXtensible MarkupLanguage (XML) 1.0,http://www.w3.org/TR/REC-xml

[12] XML-Shema definition language, W3C Technikal Report 1999, http://www.w3.org/TR/XMLschema-1

[13] "XML Parser for Java", http://www.alphaworks.ibm.com/formula/xml

[14] ObjectSpace Products: JGL , http://www.objectspace.com/jgl/prodJGL.asp