УДК 681.3

ПРАКТИЧЕСКИЙ ПОДХОД К РЕАЛИЗАЦИИ ПРИЛОЖЕНИЙ СЕМАНТИЧЕСКОГО ВЕБ

В.А. Дерецкий, М.М. Богданова, С.И. Горошанский

Институт программных систем НАН Украины,
03187, Киев-187, проспект Академика Глушкова, 40,
тел.: 38 044 526 4342, 38 044 526 6249, e-mail: {dva, bmm}@isofts.kiev.ua

НТЦ ГП НАЭК «Энергоатом»
01054, Киев-54, ул. Б. Хмельницкого, 63-А,
тел.: 38 044 206 9727, e-mail: s.goroshansky@ntc.atom.gov.ua

Одной из ключевых характеристик приложений семантического Веб является доступность машинно-читаемых метаданных. В данной работе описывается практический подход к семантическому Веб, который охватывает данные и метаданные на предприятии. Эти данные могут быть структурированными, слабоструктурированными или неструктурированными; хранящиеся в любых базах данных и созданные любым приложением. В работе представлена архитектура, которая новым способом интегрирует возможности семантического Веб в существующие технологии. Ключевая цель архитектуры состоит в том, чтобы позволить организациям видеть сервисы в Интернете с унифицированным семантическим описанием. Приведен также прототип практической реализации подхода.

It is well understood that the key for successful Semantic Web applications depends the availability of machine understandable metadata. In this paper, we describe a practical approach to the Semantic Web called Information Grid. Information Grid resources span all the data in the organization and all the metadata required to make it meaningful. This data maybe structured, semistructured, or unstructured; stored anywhere; and created by any application. We present an architecture that integrates Semantic Web techniques into existing technologies in a novel way. We show the design and implementation of a prototype that demonstrates the ideas presented in the paper.

Введение

Существуют два подхода к управлению данными: *виртуализация* и *конвергенция*. Виртуализация вычислительных ресурсов позволяет приложениям быть независимыми от отдельных конкретных элементов Grid. Она определяет структуру для разделения ресурсов в различных средах выполнения и приводит к снижению стоимости приложения. Среда Grid-вычислений обеспечивает не только виртуализацию информационных ресурсов, таких как хранилища данных, каналы передачи данных, вычислительные ресурсы, но обеспечивает также свободную связь между приложениями и модулями, которые больше не являются монолитными клиентами и серверами. Слабосвязанные приложения выполняются различными модулями на виртуальных Веб-узлах, могут инициировать выполнение функций удаленных Веб-сервисов, обмениваться метаданными и другой информацией и проводить оркестровку модулей процесса. Технологии XML поддерживают слабосвязанные приложения Grid-вычислений. Архитектура распределенных вычислений основана на принципах SOA (Service Oriented Architectures).

Конвергенция направлена на централизацию и стремится согласовать управление всеми данными. В настоящее время менее 10 процентов информации, полученной при сборе, хранении, индексировании, поиске, анализе данных, относится к категории структурированных данных. Конвергенция дает возможность управлять оставшейся частью данных. Существуют модели данных, которые способны представлять структурированные данные (реляционные), текстовые неструктурированные данные (документы) и слабоструктурированные данные (сообщения, шаблоны, деловые документы или метаданные). Стандартизация форматов документов на основе XML позволяет приложениям динамически настраиваться в соответствии с изменяющимися требованиями бизнеса. Создавая информационное наполнение с использованием XML, поставщики облегчают задачи информационного наполнения. XML является инструментом для совместного и повторного использования информации серверными приложениями, порталами и другими информационными сервисами.

Когда неструктурированная информация становится ресурсом, она может быть интегрирована в большое число процессов, таких как поиск и согласование. Пользователи могут искать информацию, которая ранее хранилась в хранилищах, например, файловые системы, репозитории документов, Веб-сайты и электронная почта. Согласование может быть осуществлено единообразно во всех средствах. Унификация удаленного доступа к компонентам архитектуры Grid и возможность сбора, обработки и анализа информации о состоянии компонент распределенной системы в сочетании с управлением их состоянием является краеугольным камнем концепции, а архитектура управления данными, основанная на XML, является ключевой для построения новых информационных систем.

1. Информационный Grid

Grid-вычисления для большинства пользователей представляются как вычислительные кластеры, которые нашли применение в Grid-приложениях. В работе рассматриваются подходы, применяемые в Grid для информационных систем.

Ресурсами в информационном Grid являются данные, а также метаданные, необходимые для описания этих данных. Данные могут быть структурированными, слабоструктурированными или неструктурированными, они могут храниться в базах данных, локальных файловых системах, почтовых серверах или создаваться любым приложением. Основой для построения информационного Grid является семантический Веб, цель которого состоит в том, чтобы дать пользователям доступ к Интернет ресурсам с использованием унифицированного семантического доступа. В Grid приложениях отдельные модули, созданные на различных платформах, могут совместно использоваться как Веб-сервисы. Каждый модуль жестко связан со своими данными (база данных, файловая система, почтовый сервер). Сведения о данных должны компилироваться в модуле приложения. В информационном Grid, наоборот, модули приложений сами могут осуществлять поиск источников, определять какими данными они обладают, существует ли жизненный цикл этих данных и как данные нужно интерпретировать. Информационный Grid строится на инфраструктуре Grid-приложений.

Например, на производстве заинтересованы в отслеживании дефектов выпускаемой продукции. Данные о дефектах поступают на производство из различных источников – электронная почта, непосредственно от заказчика, газетные материалы, телефонные звонки в службу поддержки и т.п. На прикладном уровне, организация могла бы построить параллельно анализ электронной почты, RSS-поиска или отслеживать дефекты с использованием модулей CRM, но в каждом таком модуле анализируется только один вид данных. При появлении новых видов данных о дефектах, которые являются непредвиденными (например, Интернет блоги становятся главным источником информации о дефектах), то модули, которые жестко закодированы для заданного вида данных, невозможно применить и Grid-приложение нельзя применить в данном случае. Информационный Grid, в котором могут быть описаны данные о дефекте, является более гибким подходом для понимания семантики.

2. Компоненты информационного Grid

Далее рассмотрим основные компоненты информационного Grid.

2.1. Репозиторий метаданных, управление сервисами. Недостающим звеном для эффективного совместного и многократного использования данных в Веб является недостаток машино-читаемых стандартов для контента различных областей семантического Веб. Семантический Веб часто связывается с основанными на XML стандартах метаданных для описания общей семантики, например, с использованием RDF и OWL. Каждый продукт или сервис предприятия представлен в виде XML-файла, который содержит описание данных стандартным образом для их использования другими приложениями. XML является основным стандартом для обмена финансовой отчетности (XBRL), при наполнении Веб-узла (RSS), при обмене юридической информацией (LegalXML) и т.п. В дополнение к стандартам форматов обмена появляются стандарты управления подобно языку разметки управления доступом (Access Control Markup Language – XACML) или цифровое управление правами интеллектуальной собственности (digital intellectual property rights management – XRML).

Информационный Grid использует семантическую информацию для того, чтобы сделать каждый ресурс данных доступным любому процессу в Grid приложениях, без требований априорной связи между ресурсами и Grid-процессами. На практике такая возможность возлагается на метаданные, описывающие значения и взаимоотношения между элементами данных, реализацию форматов обмена и стандартов управления.

Реляционная модель данных была одной из ранних реализаций технологии метаданных. XML является следующим шагом эволюции метаданных. Центральную часть информационного Grid представляет XML репозиторий метаданных. Этот репозиторий (физически может находиться на Веб-узлах и дисках) содержит информацию об информации, которая помогает организовать все ресурсы, участвующие в информационном Grid, в иерархические отношения, например, записи о счетах, находящиеся в базе данных A, логически принадлежат папке под названием Клиент, которая находится в файловой системе B, описание этого клиента должно быть в CRM-приложении C, взаимодействие с клиентом зарегистрировано в почтовом сервере D". Данные связи автоматически выводятся из тэгов XML.

Метаданные порождаются онтологиями различных видов. Онтологии могут определять не только предметную область ("дефектом может быть фактический отказ или возможный отказ"), но и определять задачи ("как вычислить вероятность возможного отказа"). Они могут быть персонифицированными ("представление дефекта от Legal, Support, Marketing или Finance perspectives"), а также содержать аргументацию ("почему данные о дефекте были собраны, почему это было смоделировано таким способом, кто с этим соглашается и кто возражает") и т.д.

Репозиторий приложения содержит сервис управления событиями, например, реагирующий на ситуацию ("что сделать, если клиент удален"), бизнес-правила ("как вычисляются оптовые скидки"), контроль версий ("определить новую версию счета, отображающего оптовую скидку"), сервис управления доступом ("кто может видеть номер кредитной карточки клиента") и т.д.

Последние поколения базы данных Oracle включают модель данных XML, в том числе поддержку стандарта XML Schema для реализации обмена. Oracle также включает встроенную XML функциональность репозитория метаданных, поддерживающую управление событиями, бизнес-правила, управление версиями, контроль доступа, управление правами и другие.

2.2. Семантические поисковые механизмы. В Интернете поисковые механизмы обеспечиваются поисковыми агентами [1] для выявления метаданных информационных ресурсов, проводят их индексирование для дальнейшего поиска по ключевым словам. Поисковые серверы обеспечивают такую же возможность в пределах Grid. В информационном Grid семантические поисковые агенты извлекают метаданные из средств, используя существующие разметки и применяя различные эвристики для определения взаимоотношений между информационными элементами в методах поиска текстовой информации. В момент поступления сообщения на почтовый сервер семантический поисковик определяет наличие сообщения и получение ответа, указывающего на уровень удовлетворенности клиента. По запросу имени клиента поисковый сервер может нанести цветную маркировку на результаты поиска, что указывает насколько клиент удовлетворен полученными результатами.

В настоящий момент, поисковые механизмы являются плохими примерами семантической обработки, обычно два пользователя на один и тот же запрос получают одинаковый результат, даже если один искал насекомое (cricket – сверчок), а другой игру (cricket). В результате семантической неоднозначности, полученные результаты поиска являются нерелевантными запросам. В идеале поисковый запрос будет уточнен контекстом пользователя и данными контекста создателя и, в случае совпадения контекстов, будут получены однозначные результаты.

В Интранет поисковые агенты участвуют в реализации политики безопасности, информационном управлении жизненным циклом и политике конфиденциальности. Лучшие поисковые серверы объединяют безопасность, семантическую релевантность возвращаемых результатов и способность разумно представлять контекстную информацию.

Другой аспект информационного Grid состоит в интеграции. Дополнительно к поиску по ключевым словам необходимо выполнять запросы к другим источникам данных, например, выполнить расширенный запрос SQL или OLAP. Данные в информационном Grid не всегда реляционные, поэтому не всегда возможно использование механизма SQL; однако, если все информационные ресурсы будут описаны с помощью XML, то может быть использован язык запросов, основанный на XML. Появление стандарта XQuery, объединенного с технологией J2EE Connector Architecture, дает возможность использовать запросы через присоединение документов (joins), выраженных XML и реляционных данных, также выраженных XML. Мощность XQuery позволяет выражать запросы ко всем видам данных, физически сохраненных в XML форматах. Например, для определения количества людей приславших запросы, можно присоединить к системе электронной почты имя клиента, полученного из базы данных и т.п.

2.3. Информационное представление и визуализация. Люди обрабатывают информацию более эффективно если она представлена графически, а не словесно. Известно что «одно изображение стоит тысячу слов». Способ представления информации может воздействовать на принятое решение. Недостаточная или неправильная интерпретация информации может привести к ошибочным решениям и упущенным возможностям.

Для большого количества информации в информационном Grid методы визуализации могут помочь пользователям обрабатывать большие объемы данных, создавая краткие аннотации и механизмы интерпретации определенных контекстов. В различных приложениях пользователи должны иметь возможность увидеть взаимосвязь между информацией и контекстом.

3. Обзор технологии Oracle

Далее рассмотрим основные особенности базы данных Oracle в контексте информационного Grid.

3.1. **Репозиторий XDB.** Oracle XDB обеспечивает независимое хранение, независимый контент и независимую инфраструктуру языка программирования для хранения и управления данными XML. Он предоставляет новые методы навигации в запросах для XML и использует репозиторий XML для управления иерархиями XML документов [2].

Репозиторий XDB позволяет авторам работать напрямую с XML контентом, хранящимся в базе данных. Большинство инструментов, используемых для написания XML, могут использовать протоколы доступа типа HTTP, FTP и WebDAV. XDB предоставляет средства поддержки этих протоколов, обеспечивая прямой доступ к контенту, управляемому хранилищем XDB.

3.2. API поиска. Oracle обеспечивает API полнотекстового поиска, который может использоваться при построении приложений. Являясь расширением структуры Oracle, API поиска обеспечивает специализированные текстовые индексы для поисковых приложений типа механизмов поиска в Интранет, текстовых хранилищах и каталогов приложений. Инфраструктура обеспечивает развитый язык для спецификации запросов. Существует также набор пакетов PL/SQL, которые содержат средства доступа к текстам подобно классификаторам и кластерам.

3.3. Поддержка RDF. Современная поддержка RDF основана пространственной сетевой модели данных (Spatial Network Data Model – NDM), которая является решением для управления графами в пределах базы данных Oracle. Модель данных RDF поддерживает три типа объектов базы данных: модель (RDF, состоящая из набора троек), rulebase – набора правил и индекса правил (граф RDF) [3, 4].

4. Прототип приложения

Далее рассмотрим прототип приложения, который демонстрирует вышеописанные концепции. Для построения прототипа используется коллекция DBLP, которая обеспечивает копию набора данных в RDF и простую онтологию OWL [5].

Архитектура, используемая в прототипе, состоит из нескольких серверных прикладных процессов, которые собирают данные, извлекают метаданные и создают индексы наряду с другими процессами, которые управляют набором данных RDF. Другим аспектом архитектуры является сервлет (servlet), который обеспечивает поиск, просмотр, кластеризацию и возможности визуализации.

Прототип написан на Java и усиливает Oracle10gR2 возможностями XML.

Есть несколько способов включения новых источников данных в приложении. В случае DBLP копия набора данных RDF доступна в Веб. Используя утилиты загрузчика SAX анализируем и заполняем внутреннюю схему. Другие подходы основаны на непосредственной загрузке информации в базу данных. Существуют два основных индекса для поиска и выборки. Каталогизация, кластеризация и пакетный вывод обеспечивают постобработку данных RDF. На рис. 1 показаны основные серверные компоненты приложения.

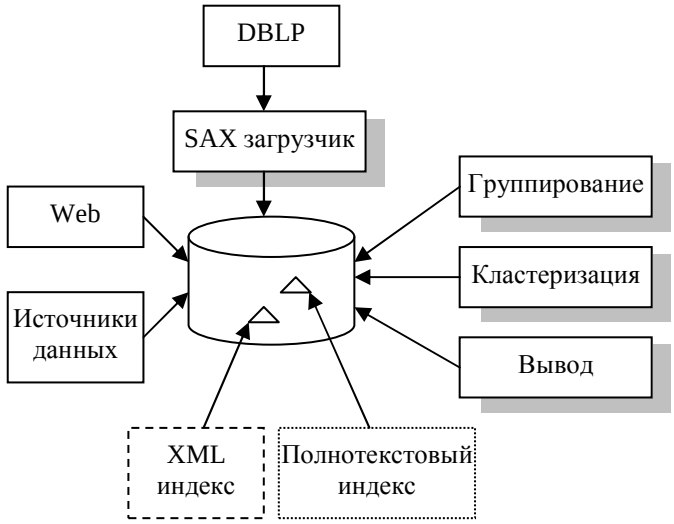

Рис. 1. Основные серверные компоненты приложения

Сервлет обеспечивает простой и расширенный поиск, просмотр онтологии и кластеризацию. Существует также уровень визуализации, обеспечивающий промежуточное представление в виде карты иерархии для кластера или граф социальной сети. На рис. 2 показаны основные компоненты сервлета Веб-приложения.

4.1. Представление RDF и OWL в базе данных. Данные в формате RDF представлены как единый файл с корневым узлом RDF [6]. Этот узел содержит большую коллекцию элементов описания. В данном примере информационное наполнение было представлено в виде нескольких описаний для обеспечения доступа к контексту каждого элемента.

Технология обработки SAX позволяет эффективно обрабатывать контент RDF. Каждый элемент описания помещается в таблицу XML (rdf_document_table).

Для управления системой метафор и доступа к элементам описания была сформирована иерархия папок онтологий (OWL), связанных с данными RDF. Каждый элемент описания был “упакован” согласно классификации, с использованием пакета dbms_xdb.createResource.

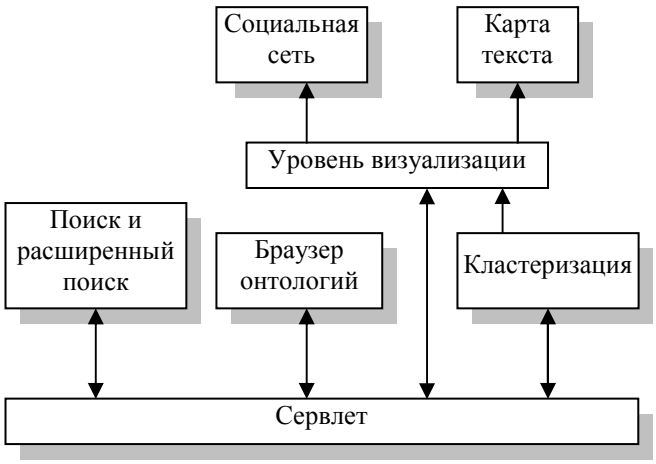

Рис. 2

БД Oracle XML включает репозиторий, который позволяет организовать контент как набор файлов и папок. Содержимое в репозитории доступно через URL таким же образом, как в реляционной таблице содержимое может быть доступно через первичный ключ. Папки и файлы могут быть получены из SQL через стандартные протоколы, такие как HTTP, FTP и WebDAV.

Инструментальные средства редактирования XML и XML-схем стали стандартом для пользователей, в том числе для пользователей, которые работают и определяют онтологии для организаций. В качестве редактора OWL используются средства SemanticWorks, которые обеспечивают возможность редактирования кода RDF и работы с данными на уровне UML [7].

4.2. Запросы к базе данных. Для извлечения данных можно использовать несколько механизмов запросов, доступных через SQL. Для примера представим запросы «Выборка авторов» и «Выборка публикаций в журналах», в которых используются различные операторы.

Выборка авторов:

```
select value(auth).getClobVal()
from rdf_document_table, table(xmlsequence(
extract(
object_value,'/rdf:Description/author','xmlns:rdf=
"http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns="http://example.org/"'));
```

Выборка публикаций в журналах, содержащих контент Software:

```
select extractValue (object_value,
'/rdf:Description/title',
        'xmlns:rdf=http://www.w3.org/1999/02/22-rdfsyntax-ns# xmlns="http://example.org/"')
from rdf_document_table where
contains(object_value,'Software
inpath(/rdf:Description/journal)')>0;
```

Язык запросов достаточно гибок, что позволяет пользователям выполнять традиционные SQL запросы, XPath, полнотекстовый поиск или любую их комбинацию.

4.3. Интерфейс пользователя. Пользовательский интерфейс в современных средствах поиска в Интернете состоит из единого окна поиска, для основного, поиска и нескольких добавочных полей – для расширенного поиска. Вообще пользовательский интерфейс привязан к серверу и ограничивается возможностями настройки. Другими словами, средства поиска позволяют пользователю проводить поиск по определенным полям. Поисковый двигатель нуждается в достаточном пространстве для размещения служебной информации.

Интерфейс пользователя приложения состоит из трех представлений для доступа к контенту. Первое представление должно обеспечить просмотр статей, используя классы онтологий. Второе представление использует метки или тэги, основанные на кластерах, как альтернатива схеме классификации. Третье представление – предложено окно поиска, которое обеспечивает прямой доступ к любому элементу.

Одно из преимуществ наличия классов в OWL является то, что мы можем построить простой механизм для просмотра контента. Следующий запрос SQL возвращает иерархию верхнего уровня, представленную как путь:

```
select path from path_view where
under_path(RES,'/home/DBLP/DocumentHierarchy')=1;
```

На следующим примере представлена процедура добавления уровней иерархии и представления результатов для Веб интерфейса:

```
String statementText = "select distinct extractValue"
+" object_value, "
+" ′ /rdf:Description/title′, 'xmlns:rdf=\"http://www.w3.org/1999/02/22-rdf-syntax-ns#\ " "
+" xmlns:rdf=\"http://www.w3.org/2000/01/rdf-schema#\" xmlns:dc=\"http://pur1.org/dc/elements/1.1/ \" "
+" xmlns=\"http://exemple.org/ \ " ')title, "
+" from RDF_DOCUMENT_TABLE "
+" where existsNode ( "
+" object_value, ′/rdf:Description/rdf:type[@rdf:resource=\http://exemple.org/#"+displayPath+"\"]',"
        +" 'xmlns:rdf=\http://www.w3.org/1999/02/22-rdf-syntax-ns#\
  xmlns:rdf=\"http://www.w3.org/2000/01/rdf-schema#\"
+" xmlns:dc=\"http://pur1.org/dc/elements/1.1/\" xmlns=\"http://exemple.org/ \ " ') = 1 ";

OracleDriver ora = new OracleDriver ( );
connection = ora.defautConnection( );
```

```
OracleCallableStatement  statement = (OracleCallableStatement) connection.prepareCall(statementText);
statement.execute(statementText);
ResultSet rs = statement.executeQuery(statementText);
out.printIn("<table width=\"100%\" cellpadding=2 cellspacing=0 border=0><tbody>");
while(rs.next( ) )
  {
```

Для улучшения результатов поиска необходимо использовать кластеризацию [8], которая используется также как альтернативный подход группирования информации [9]. Пользователи используют кластеризацию при исследовании данных или создании краткого обзора по теме [10].

В прототипе приложения объединяются эти два подхода. Большинство популярных кластеров основаны на заголовках публикаций, которые используют традиционные подходы, подобно k-методам [11]. Размер шрифта в метке прямо указывает на качество кластера. Второй подход состоит в кластеризации результатов поиска. Используется список совпадений кластеризации, основанный на суффиксных деревьях, который более точно соответствует контекстным результатам [12].

Существует также возможность представления кластеров в виде карты деревьев.

В любом современном поисковом механизме есть возможность осуществить поиск по умолчанию с опцией переключения в расширенный режим. В отличие от простых поисковых механизмов Интернет, расширенный режим обеспечивает интерфейс, который позволяет пользователям искать поля в зависимости от доступности различных источников.

Как вышеуказано, в ядре системы есть мощный механизм для обнаружения и формирования метаданных XML. Использование AJAX (Асинхронный JavaScript и XML) (как основную методику) предоставляет возможность строить гибкий механизм для динамического поиска источника и атрибутов. Компонент загрузки метаданных формирует запрос к базе данных. Извлекаются все метаданные источника данных и индексы, которые возвращаются в виде XML (для каждого источника данных).

Генератор интерфейса пользователя использует объект *xmlHttpRequest* для заполнения определенных элементов интерфейса пользователя наряду с JavaScript и CSS. Результатом генерации является интерфейс, который дает пользователю возможность осуществить расширенный поиск. На рис. 3 показаны основные компоненты интерфейса пользователя с применением AJAX.

Пользователь выбирает источник данных, а интерфейс автоматически отображает доступные атрибуты для поиска.

4.4. Информационная визуализация. Существует несколько способов визуализации больших наборов данных. В приложении используются карты деревьев для кластерной визуализации [13].

В прототипе визуализации карта деревьев кластера читает конфигурацию в форме XML, а также данные, полученные с использованием SQL. Применяется встроенная в Oracle технология XML, как основной подход к интеграции.

4.5. Социальные сети. Одна из наиболее перспективных особенностей семантического Веб состоит в представлении предметной области приложений в соответствии с понятиями и выводами.

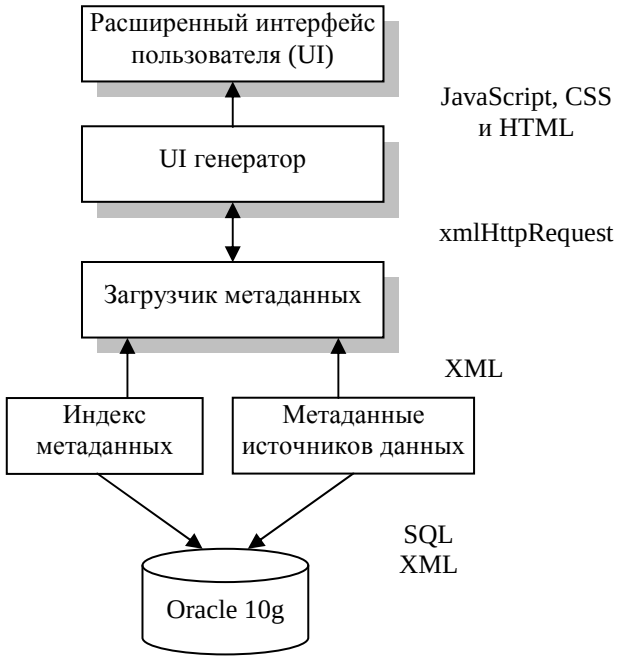

Рис. 3

Рассмотрено использование функциональности Oracle для управления RDF и OWL. Информационный Grid содержит более одного источника данных. Возможность подключения различных источников и работы с содержанием является ключевой частью процесса. Например, мы хотели бы добавить к нашему приложению подробную информацию об исследователях. Используя имена исследователей и эвристики в поисковом агенте, можно получить документы XML, которые содержат желаемую информацию.

Когда новые источники данных становятся доступными и документы первый раз попадают в репозиторий, индексируется содержимое документов и их метаданные становятся доступными в расширенном интерфейсе пользователя. Полученную информацию можно использовать для вывода (computing inference).

Например, мы хотим знать, действительно ли публично отображается персональная информация об исследователе и его социальной сети. Как только документ XML, который представляет исследователя, доступен в системе можно получить вывод о некоторых взаимоотношениях авторов, которые представлены в социальной сети. Идея состоит в том, чтобы осуществлять вывод, используя: a) описание RDF в базе данных или b) механизм логического вывода.

Социальные сети могут служить мощными инструментами для интеллектуального сбора информации и разработки приложений.

5. Другие приложения семантического Веб

Рассмотрим некоторые приложения, которые поддерживают концепции семантического Веб.

Один из подходов к разработке программных приложений с использованием семантических Веб-сервисов рассмотрен в [14].

Haystack – это система для слабоструктурированного управления данными с ориентацией на конечных пользователей [15]. Данный подход обеспечивает просмотр информации, определяя важность данных.

PiggyBank наследует некоторые идеи, исследованные в Haystack, и делает их доступными на Веб-браузере [16]. В нем используется комбинация поиска и категорий или кластеров. Подход основан на открытой технологии (open source) и зависит от единой платформы управления данными. Обеспечивается потребность клиентов быть более “семантически осведомленными”.

Swoogle [17] предлагает новую перспективу в поиске и индексировании семантического контента. Он заменяет или расширяет нынешнюю методику ранжирования.

Magnet [18] и BANKS [19] представляют собой примеры систем управления и просмотра больших наборов данных. Прототип приложения обеспечивает просмотр как основную особенность системы. Представленный в виде RDF контент [20] является важным аспектом любых семантических приложений.

WEESA – хороший пример автоматизации процесса генерации метаинформации [21]. Реализация осуществлена в рамках Apache. Подобные результаты могут быть использованы в инструментальных средствах Oracle XML.

Заключение

Представлен информационный Grid как новая перспектива управления семантической информацией. В работе рассмотрены новые подходы и архитектура, которая может быть использована при построении Grid-приложений.

Продемонстрировано практическое применение идей семантического Веб, их реализация на прототипе приложения с использованием набора данных DBLP и существующих технологий Oracle 10gR2.

Предлагаемый подход основан на использовании существующих инструментальных средств XML для расширенного использования семантического контента, представленного в OWL / RDF. Следующий шаг должен обеспечить сервисы, которые могут использовать в своих интересах семантику как поиск, кластеризацию, вывод и социальные сети. Метаданные XML позволят легко разрабатывать сложные методологии визуализации и новые подходы к разработке интерфейсов пользователя.

1. *Андон П. І., Дерецкий В О.* Проблеми побудови сервіс-орієнтованих прикладних інформаційних систем в Semantic web середовищі на основі агентного підходу // Проблеми програмування. − 2006. − № 2–3. − С. 493–502.
2. *Oracle* Database 10g Release 2 XML DB Application developers guide. Oracle Corp. – 2005.
3. *Oracle* Spatial Resource Description Framework (RDF) 10g Release 2 Reference. Oracle Corp. – 2005.
4. *E. Chong et al*. An Efficient SQL-based RDF Querying Scheme. VLDB – 2005.
5. http://dblp.uni-trier.de
6. http://www.semanticweb.org/
7. http://www.altova.com

8. http://www.vivisimo.com
9. http://www.flickr.com/
10. *Aula A., Jhaveri N. and M. Kak*i Information Search and Re-access Strategies of Experienced Web Users WWW. – 2005.
11. *Jain A., Murty M., Flynn P.* Data clustering: a review, ACM Computing Surveys. – 1999.
12. *Zamir O. and Etzioni O.*Web document clustering: a feasibility demonstration. SIGIR. – 1998.
13. *Card S., Mackinl J. and Shneiderman B.* Readings in Information Visualization. Morgan Kaufman, San Francisco, 1999.
14. *Дерецкий В., Богданова М., Горошанский С.* Подход к разработке программных приложений с использованием семантических Веб-сервисов // Проблеми програмування. − 2009. − № 4. − С. 59–70.
15. *Karger D. et al. Haystack A Customizable.* General-Purpose Information Management Tool for End Users of Semistructured Data. CIDR – 2003.
16. *Huynh D., Mazzocchi S. and D. Karger PiggyBank.* Experience the Semantic Web Inside Your Web Browser // In4th International Semantic Web Conf. ISWC. – 2005.
17. *Ding L. et al*. Swoogle A Search and Metadata Engine for the Semantic Web. CIKM. – 2004.
18. *Sinha V. and Karger D.*Magnet: Supporting Navigation in Semistructured Data Environment, SIGMOD. – 2005.
19. *Bhalotia G. et al*. Keyword Searching and Browsing in Databases using BANKS. ICDE. – 2002.
20. *Rutledge L., J. van Ossenbrunggen and L. Hardman* Making RDF Presentable. WWW. – 2005.
21. *Reif G., Gall H. and Jazayeri M.* WEESA – Web Engineering for Semantic Web Applications WWW. – 2005.