УДК 519.68+004.1'414

В. Б. Шрайбман

Новосибирский государственный университет
ул. Пирогова, 2, Новосибирск, 630090, Россия
E-mail: shvb@ngs.ru

RDF КОНТЕНТ-МЕНЕДЖЕР КАК СРЕДСТВО ВЫСОКОУРОВНЕВОЙ РАБОТЫ С ДАННЫМИ И ЕГО ПРИМЕНЕНИЕ

Технологии Semantic Web

Сегодня Web представляет собой огромное хранилище информации, которая, к всеобщему сожалению, слабо структурирована и за редкими исключениями пригодна для использования только человеком. С целью преодоления этих недостатков несколько лет назад в работе W3C консорциума выделилось новое направление, названное «Semantic Web».

Semantic Web – это расширение текущего Web, в котором информации придается строго определенная семантика, что открывает новые перспективы совместной работы людей и компьютеров [Hendler et al., 2002]. Целью разработчиков этого проекта является создание и внедрение в Web технологий, способных помочь программам правильно интерпретировать данные, чтобы они смогли предоставлять более богатые возможности поиска, сбора и интеграции информации, полученной из различных источников, а также автоматизировать выполнение заданий по обработке разнообразных сведений.

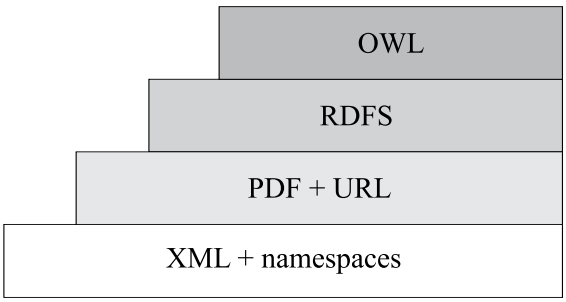

Рис. 1

Стандарты, составляющие основу Semantic Web, часто изображают в виде пирамиды (рис. 1), полный вариант и детальное описание которой можно найти, например, в [Koivunen, Miller, 2001].

На каждом уровне (если смотреть по восходящей) расположена технология, опирающаяся на предыдущие и использующая их, и каждый следующий слой – это шаг по направлению к машинной логике и «осмысленной» программной обработке информации.

XML (Extensible Markup Language) предоставляет универсальный способ записи данных, поддерживаемый и распознаваемый многими программами. Таким образом, этот язык обеспечивает синтаксическую интероперабельность приложений в Web.

RDF (Resource Description Framework[1]) позволяет изложить произвольные сведения при помощи утверждений вида «*субъект – предикат – объект*» или иначе «*субъект – свойство – значение*». Использование универсальных идентификаторов URI (Universal Resource Identifier) для обозначения элементов троек, называемых в этом случае ресурсами, позволяет субъекту одного утверждения быть объектом другого. Это превращает информацию, выраженную на языке RDF, в семантическую сеть. Только в качестве значений свойств помимо ресурсов разрешено использовать непосредственно текстовые данные, называемые литералами.

Одним из нескольких возможных способов записи RDF является RDF/XML-нотация. Таким образом, RDF добавляет к XML семантическую интероперабельность приложений [Шрайбман, 2003].

[1] Resource Description Framework (RDF) / http://www.w3.org/RDF/

Технология RDFS (RDF Schema[2]) предлагает способ задания системы типов для RDF. Она позволяет определить классы ресурсов и свойства как элементы словаря, а также специфицировать, ресурсы каких классов какими свойствами могут быть описаны.

Информация о типах выражается средствами RDF с применением дополнительных понятий, определенных в RDFS, что позволяет хранить и использовать ее точно так же, как и сами описываемые данные. Более того, разработчики спецификации сознательно не регламентировали то, как именно приложения должны интерпретировать описания словарей: одни могут воспринимать их в качестве дополнительного источника сведений о ресурсах, другие – как жесткое определение структуры классов, третьи – в роли помощника для выявления ошибок.

Язык RDFS, т. е. набор определяемых этой технологией терминов, равно как и любой другой RDF словарь, может быть легко расширен новыми понятиями. Одним из таких расширений RDFS является язык OWL (Web Ontology Language[3]), определяющий дополнительные свойства и ресурсы для более сложного и богатого концептуального моделирования различных предметных областей.

По замыслу разработчиков Semantic Web, сведения, изложенные на языке RDF при помощи понятий, задаваемых RDF схемой или онтологией, будут семантически интерпретируемыми программами, знакомыми с соответствующими словарями.

Средства работы с RDF-данными

Вопреки часто встречающемуся мнению RDF следует рассматривать шире, чем просто вспомогательный язык для указания метаданных, т. е. добавления к имеющимся ресурсам нескольких характеризующих их утверждений (наиболее известным примером чего является словарь Dublin Core[4]). RDF предлагает удобный способ связного описания данных, который может быть успешно использован для представления больших объемов информации. Одно RDF-хранилище может быть одновременно использовано как источник сведений для внешнего программного агента, работающего в среде Semantic Web, и в качестве хранилища данных некоторой информационной системы, предназначенной для человека. Для создания таких приложений требуется иметь удобные средства программного доступа к данным и работы с ними.

Среди разработок, посвященных технологиям Semantic Web[5], можно найти немало библиотек для различных языков программирования, предоставляющих все или некоторые из следующих возможностей:

• программный интерфейс для манипулирования утверждениями, ресурсами и литералами;

• чтение RDF-данных из файлов и запись их в файл;

• хранение RDF-моделей в реляционных базах данных с универсальной структурой таблиц, которая не зависит от используемой системы типов и не заметна для внешних приложений, имеющих дело лишь с указанным интерфейсом.

Из средств для языка Java наиболее распространенным продуктом, вероятно, является Jena[6]. Ее главные достоинства – это простая и удобная система классов, реализующих программное представление RDF-моделей, а также многочисленные дополнительные функции.

Библиотеки, подобные Jena, полностью решают задачу доступа к данным, но не слишком удобны в качестве архитектурного слоя, непосредственно предшествующего слою бизнес-логики разрабатываемого приложения, поскольку предлагают работать с RDF на низком уровне, т. е. с отдельными субъектами и их свойствами. Поэтому перед автором возникла задача разработать дополнительные средства для манипулирования более сильными абстракциями, чем просто ресурсы, литералы и утверждения, а именно RDFS-классами. Иными словами, целью исследования было создание самостоятельного приложения или вспомога-

[2] RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation. 2004 / http://www.w3.org/TR/rdf-schema/

[3] Web Ontology Language (OWL) / http://www.w3.org/2004/OWL/

[4] Dublin Core Metadata Initiative / http://dublincore.org

[5] См., например, Dave Beckett's Resource Description Framework (RDF) Resource Guide / http://www.ilrt.bristol.ac.uk/discovery/rdf/resources/

[6] Jena – A Semantic Web Framework for Java / http://jena.sourceforge.net/

тельной библиотеки, позволяющей программистам работать с RDF-данными на уровне классов, используя привычные подходы объектно-ориентированного языка программирования, такого как Java.

Согласно тому же списку разработок в предложенной постановке задача пока решена не была. Высокоуровневые технологии создания информационных систем, предназначенных для конечного пользователя, сконцентрированы на использовании уже имеющихся источников данных, как правило, реляционных СУБД, чья структура таблиц соответствует конкретной предметной области. Этот подход успешно применим, но обладает двумя недостатками. Во-первых, при манипулировании информацией так или иначе учитывается ее структура и особенности хранения. А во-вторых, чтобы работать с данными как с семантической сетью, таким системам приходится отдельно реализовывать возможность отображения информации в RDF.

Наиболее яркой и масштабной из известных автору технологий создания приложений является система ИСИР [Бездушный и др., 2003], которая позволяет разрабатывать распределенные объектно-ориентированные цифровые библиотеки на основании RDFS-описаний схем данных и предоставляет большое количество сервисов по обработке, репликации информации и публикации ее в различных форматах. В качестве другого примера успешных исследований в этой области следует упомянуть проект SMART [Шрайбман, Гуськов, 2003], явившийся идейным предшественником данной работы.

RDF контент-менеджер

При решении поставленной задачи была разработана Java-библиотека, которая получает на вход некое RDFS-описание и адрес одного или нескольких соответствующих ему RDF-хранилищ и предоставляет использующим ее приложениям доступ как к программному представлению элементов схемы, так и к самим RDF-данным посредством специальной системы классов. Эта разработка была названа RDF Content Manager – RCM.

Понятие *контент-менеджер* перегружено, и его часто, но отнюдь не всегда, используют в отношении средств быстрого создания Web-сайтов и управления их страницами и данными. Здесь же такое название выбрано по причине некоторой идеологической и технологической общности библиотеки с продуктом DB2 Content Manager[7] компании IBM. В среде сообщества Java-программистов (Java Community) подобные системы часто именуют Content Repository (репозиторий).

Выбор RDFS в качестве средства описания системы типов был обусловлен тем, что схемы являются непосредственно следующим для RDF слоем в приведенной пирамиде технологий Semantic Web, и они вполне достаточны для многих приложений. В будущем по мере усложнения контент-менеджера, возможен выход и на уровень поддержки онтологий, что должно происходить достаточно легко, поскольку OWL использует и расширяет ряд понятий и концепций, определяемых еще в RDFS.

Общий взгляд на систему классов

При проектировании вышеупомянутой системы классов для работы с RDF и RDFS сначала была предпринята попытка сопоставить каждому RDFS-классу из схемы собственный Java Bean класс, а каждому RDF-ресурсу из хранилища – Java-объект соответствующего класса. Для осуществления такого сопоставления необходимо было разработать средство автоматической генерации Java Beans и так называемую «фабрику», отвечающую за создание Java-объектов и наполнение их надлежащими данными RDF-ресурсов и их свойств, а также за запись изменений этих данных обратно в хранилище.

Несмотря на то что с Java Beans приятно работать в программе, их использование оказалось неудобно в целом. Первая из причин – это необходимость генерировать классы для каждой RDF-схемы, прежде чем с ней можно начать работать. Более того, любое изменение схемы (что часто бывает на начальных этапах разработки информационных систем) требует обновления или даже полной перегенерации Java Beans, а также ручной переработки осно-

[7] IBM DB2 Content Manager / http://www-306.ibm.com/software/data/cm/cmgr/

ванного на них кода. И в-третьих, использование специализированных классов существенно затрудняло создание приложений и библиотек общего назначения, чьи функции не зависят от конкретной схемы данных RDF-хранилища, с которым ведется работа.

Из-за перечисленных недостатков автор отказался от применения Java Beans в пользу универсальной системы, сопоставляющей Java-классы самим понятиям *rdfs:class*, *rdf:property* и *rdfs:resource*. Таким образом, RDFS-классы, применяемые к ним свойства и конкретные ресурсы оказываются ссылающимися друг на друга объектами универсальных классов, не ориентированных на какую-то используемую схему.

Программное представление RDF-схемы

С точки зрения программного представления системы типов устройство RCM можно изобразить в виде диаграммы (рис. 2).

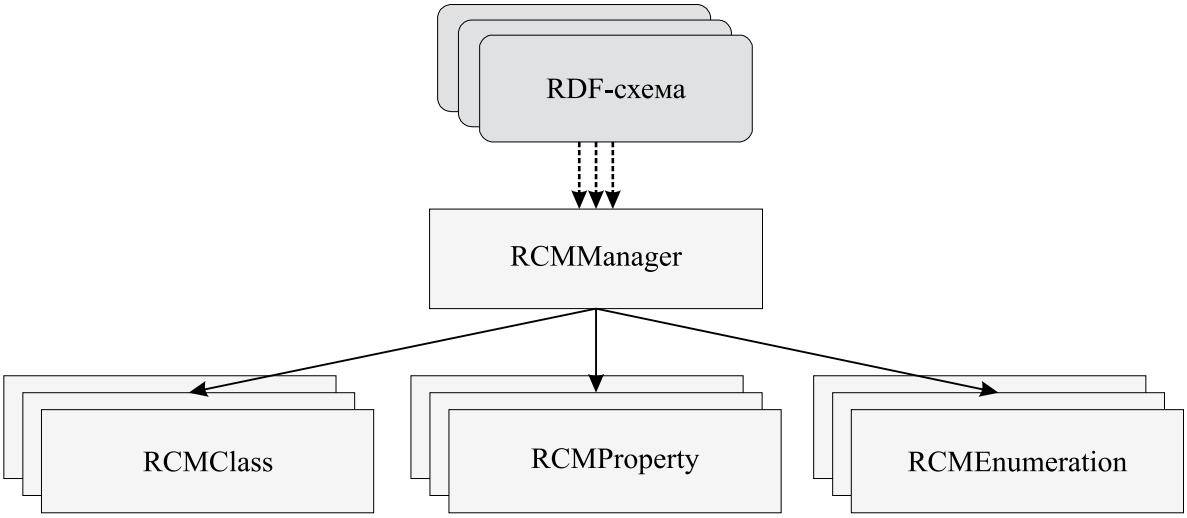

Рис. 2

RCMManager является центральным классом, а его экземпляры – отправной точкой всех работ с библиотекой RCM. В процессе инициализации объекта *RCMManager* ему передаются файлы с RDFS-описанием системы типов, которыми менеджеру предстоит манипулировать, после чего программа проверяет схемы на полноту определений и убеждается в том, что они удовлетворяют нескольким ограничениям, которые будут описаны ниже. В случае неудачной проверки объект может вернуть список ошибок и предупреждений валидации.

По окончании инициализации экземпляр *RCMManager* предоставляет доступ к коллекциям объектов классов *RCMClass*, *RCMProperty* и *RCMEnumeration*, каждый из которых несет в себе полную информацию об одном классе, свойстве или перечислимом типе, описанном в используемой RDF-схеме. Вслед за указаниями областей определения и значений свойств – *rdfs:domain* и *rdfs:range*, а также за отношением специализации классов *rdfs:subClassOf* некоторые из этих объектов ссылаются друг на друга, обеспечивая связность программного представления схемы. Заявление о полноте предоставляемой информации означает, что использующие RCM приложения посредством объектов могут получить доступ не только к утверждениям, которые соответствуют словарю RDFS (и его расширениям, поддерживаемым RCM), но также и к дополнительной информации, указанной автором схемы с применением незнакомого RCM словаря.

Перечисления, которым соответствуют объекты класса *RCMEnumeration*, являются конструкцией, добавленной к RDFS библиотекой RCM и предназначенной для определения перечислимых значений простых типов. Таким образом, если в описании схемы в качестве области значений свойства указан не класс, а перечисление, то RCM будет считать это свойство литеральным, а его допустимыми значениями будут элементы указанного перечисления. Смысл конструкции заключается в том, что она дает четкое определение простого типа данных (чего не было бы в случае использования строк или чисел в качестве значений свойства), а также

в том, что каждому элементу перечисления можно сопоставить удобную текстовую метку, предназначенную для человека и переводимую на различные языки.

Класс *RCMManager* не является статическим. Каждый его объект проходит независимую инициализацию, поэтому может работать со своей собственной системой типов, и является отдельным экземпляром RDF контент-менеджера.

Требования к схемам и данным

Как уже было отмечено, спецификация RDFS ничего не говорит о том, как схемы должны быть использованы на практике, поэтому правильно написанная схема может неполно или неоднозначно задавать систему типов или вообще не соответствовать реальным данным. С другой стороны, сам RDF оставляет некоторую свободу выбора, каким образом описывать сведения, и в похожих случаях часто применяются различные средства языка.

Ввиду такой неоднозначности RCM выдвигает несколько требований, которым должны удовлетворять схемы данных и RDF-модели. Хотя предлагаемые ограничения и несколько сужают круг поддерживаемых хранилищ, они делают данные более структурированными и в действительности не мешают создавать удобные RDF-схемы, надлежащим образом определяющие различные предметные области.

Первое из требований состоит в том, что для каждого RDF-свойства должны быть обязательно указаны область определения и область принимаемых значений, т. е. должны быть прописаны *rdfs:domain* и *rdfs:range*. Это позволяет рассматривать свойства как поля соответствующего RDFS-класса и, таким образом, следовать принципам объектно-ориентированного программирования, которые себя прекрасно зарекомендовали в деле представления сущностей реального мира, и не видно достаточных оснований от них отказываться даже в среде трудно структурируемой информации Web.

Помимо этого, RCM определяет для свойств дополнительный предикат *rcm:type*, который также является обязательным и сообщает программе, к какой из следующих категорий относятся значения этого свойства:

• текст, который в случае многоязыковой системы должен быть продублирован для всех поддерживаемых языков;

• простое значение, не требующее повторения при локализации и обычно применяемое для хранения чисел, дат и другой подобной информации;

• ссылка на ресурс класса, указанного в качестве области значения свойства;

• значение перечислимого типа (*RCMEnumeration*).

Вторая группа ограничений налагается непосредственно на RDF-модели и состоит в том, что RCM не поддерживает использование нетипизированных ресурсов, неименованных узлов (*blank node*) и RDF-коллекций, включая списки. По мнению автора, применение таких конструкций привносит в семантические сети элементы, отвечающие за структуру описания, а не за отражение каких-то новых сведений, и тем самым «ухудшает» качество RDF-данных.

В качестве типичного примера таких «сомнительных» конструкций можно привести многоуровневые описания ресурсов, когда значение свойства отделено от субъекта, к которому оно фактически применяется, промежуточными вершинами в RDF-графе (рис. 3).

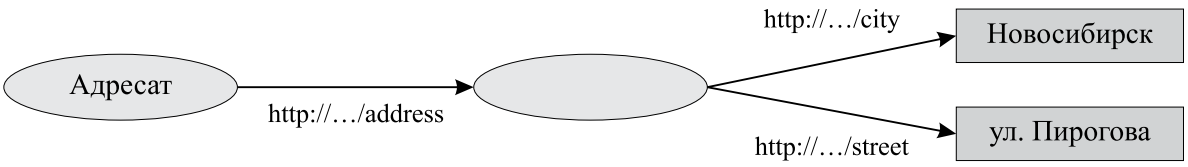

Рис. 3

Здесь в центре описания находится ресурс, про который ничего конкретного сказать нельзя – имеются только косвенные сведения. С точки зрения целостности данных было бы правильнее либо придать промежуточному узлу самостоятельную смысловую нагрузку (т. е. сделать его экземпляром класса, соответствующего подобным ресурсам), а также как-то идентифицировать его, либо обойтись без него вовсе.

В свою очередь, вместо коллекций предлагается использовать повторяющие свойства (*repeated properties* в терминологии RDF), правда, RCM позволяет предикату иметь несколько значений, только если он является ссылкой на ресурс. Для литеральных свойств и для свойств, чьи значения являются элементами перечислимого типа, повторяемость запрещена.

В дополнение к уже сказанному текущая версия RCM не поддерживает наследование (специализацию) свойств и высказывания об утверждениях (*reification*) – у автора пока нет должного понимания того, как эти возможности RDF и RDFS могут быть по-настоящему использованы в решаемых им задачах.

Опциональные расширения языка схем

Помимо описанных требований, RCM предлагает ряд необязательных, но часто полезных на практике расширений языка RDFS, об одном из которых – перечислениях – было рассказано выше.

Так, RCM предоставляет сразу несколько дополнительных предикатов, позволяющих указать, должно ли свойство непременно иметь значение, может ли ссылочное свойство иметь более одного значения, и как поддерживать целостность данных в случае удаления ресурса, на который ссылается другой ресурс, или наоборот.

Далее, RCM позволяет упорядочивать свойства классов, а также указывать предикаты, чьи значения формируют осмысленное с точки зрения человека имя объекта, которое может быть использовано сторонними приложениями самыми различными способами. Например, именем статьи можно считать значение одного ее свойства «тема», а именем факультета – сочетание значений его свойства «название» и имени вуза, на который ссылается соответствующий факультету RDF-ресурс.

И последним, одним из наиболее интересных расширений языка схем являются стереотипы, предлагающие универсальный способ добавления некоторых меток к определениям различных элементов моделирования. Их идея была позаимствована автором в языке UML (Unified Modeling Language[8]). Приложения, использующие RCM, могут поинтересоваться у программного представления схемы, присутствует ли некая метка в описании класса, свойства или перечисления, и варьировать свое поведение в зависимости от полученного ответа. Универсализм стереотипов заключается в том, что при их применении нет необходимости расширять базовый словарь RDFS всякий раз, когда требуется предоставить какую-то дополнительную информацию о типах данных.

Работа с RDF-данными

Один экземпляр контент-менеджера позволяет использующим его приложениям работать сразу с несколькими RDF-хранилищами, чьи данные удовлетворяют схеме, использованной при инициализации объекта *RCMManager*. Источниками данных могут выступать RDF-файлы и отдельные модели, хранимые библиотекой Jena в универсальных реляционных СУБД.

Доступ к каждому репозиторию осуществляется через экземпляр класса *RCMDataProvider*, получаемый приложениями у контент-менеджера и олицетворяющий для них сессию работы с хранилищем. Все действия с RDF-ресурсами происходят посредством объектов *RCMObject*, за получение и жизненный цикл которых и отвечает конкретный поставщик *RCMDataProvider*. Более точно его функциями являются:

• поиск ресурсов в хранилище по их URI, классу, значению свойства и нескольким более сложным критериям;

• получение объектов *RCMObject*, соответствующих ресурсам, и наполнение их надлежащими данными;

• создание новых объектов требуемых классов и сохранение соответствующих ресурсов в репозитории;

• изменение данных существующих ресурсов и сохранение их в хранилище;

• проверка наличия у ресурсов (объектов) свойств, являющихся обязательными согласно описанию типов;

[8] Unified Modeling Language (UML) 1.5 / http://www.omg.org/technology/documents/formal/uml.htm

• удаление ресурсов из хранилища;

• обеспечение целостности связей: принятое RCM расширение RDF-схемы поддерживает запрет, простое и каскадное удаления связных ресурсов (объектов).

Таким образом, работу отдельно взятого экземпляра *RCMDataProvider* можно представить так:

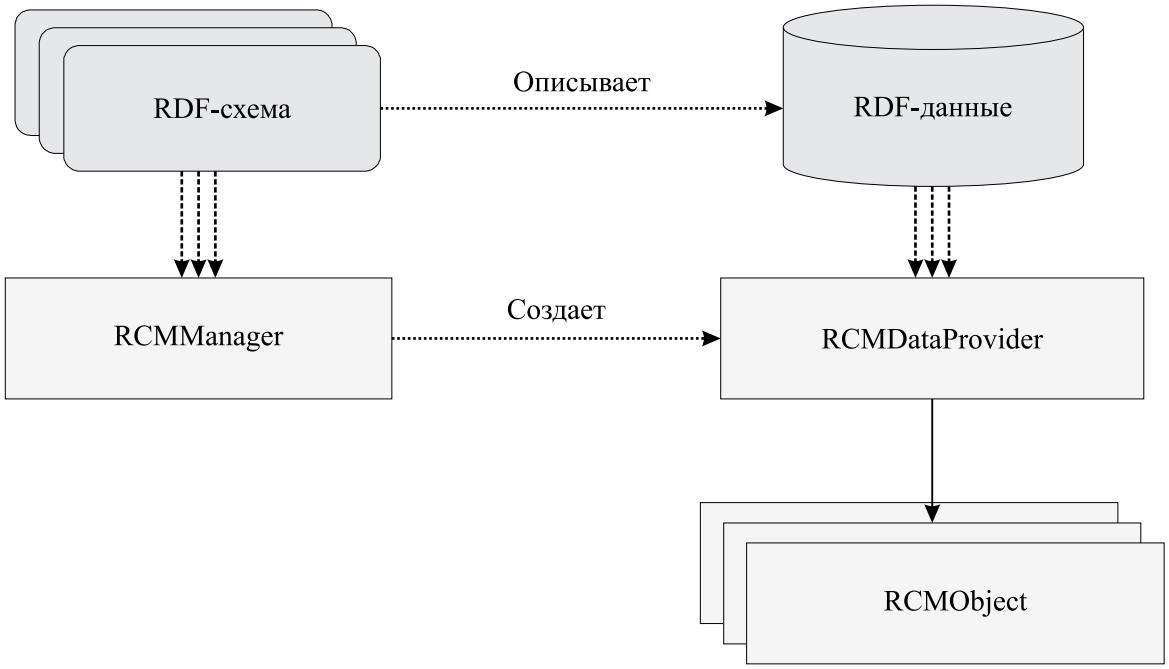

Рис. 4

В текущей версии RCM каждый экземпляр *RCMObject* связан в точности с одним из созданных в программе объектов *RCMDataProvider*, а через него с конкретным репозиторием. В будущем же, возможно, будет сделана попытка подобной реализации работы с ресурсами, чьи данные распределены между несколькими хранилищами.

Все действия с ресурсами, включая чтение, производятся в рамках либо явных транзакций, начало и конец которых указывает сам разработчик, либо неявных, автоматически завершаемых системой после выполнения каждой отдельной операции. Транзакции гарантируют целостность и непротиворечивость данных в любой момент времени. Например, если один пользователь записывает изменения свойств какого-то ресурса, а другой одновременно читает тот же самый объект, то второй никогда не получит интересующие его сведения в промежуточном состоянии, когда часть значений уже сохранена в репозитории, а часть – нет. Точно так же RCM гарантирует, что при одновременной записи одного ресурса изменения будут приняты последовательно и разные данные не будут смешаны.

Многоязыковая поддержка

Важной особенностью RCM является полноценная многоязыковая поддержка, предусмотренная на уровне API (Application Programming Interface) и в отличие от многих систем, работающих, например, только с русским и английским, не ограниченная ни списком конкретных локализаций, ни их допустимым количеством.

Как уже упоминалось, литеральные свойства в RDF-схеме могут зависеть от языка, и в этом случае соответствующие значения повторяются в хранилище в различных переводах.

Кроме того, был также предусмотрен механизм локализации чисел, дат и других подобных данных. Для этого RCM предлагает Java-интерфейс *RCMValuesFormatter*, реализации которого подключаются к контент-менеджеру на стадии инициализации и осуществляют двусторонние преобразования значений определенных простых типов из стандартных, используемых репозиторием, форматов в специфические, учитывающие региональные предпочтения пользователей, и наоборот. Иначе говоря, RCM может как возвращать, так и принимать лока-

лизованные данные и, если это необходимо, автоматически их попутно перекодировать. В состав контент-менеджера входят и используются по умолчанию реализации *RCMValuesFormatter*, форматирующие значения основных типов данных, встроенных в XML Schema[9].

И наконец, RCM предоставляет доступ к различным переводам текстовых меток *rdfs:label* классов, свойств и перечислений, указанным в RDF-схемах, а также к именам объектов, о которых было рассказано выше.

Примеры использования RCM

Благодаря своему устройству RCM может быть успешно использован в качестве основы не только специализированных информационных систем, применяющих RDF и RDFS, но и универсальных программ и компонент, чьи возможности не зависят от конкретной схемы и RDF-хранилища, с которым ведется работа.

Универсальный редактор RDF-данных

Первым примером такой надстройки над RCM служит Web-приложение, получившее название RDF Content Viewer – RCV.

RCV умеет отображать структуру классов, определяемую схемой данных, которая передается на вход RCM, поддерживает двустороннюю навигацию по отношениям наследования и ссылочным свойствам, осуществляет поиск классов по их именам, заданным предикатами *rdfs:label*.

Далее это приложение позволяет просматривать коллекции ресурсов, хранящихся в указанном в его настройках RDF-репозитории и являющихся экземплярами классов, определяемых RDF-схемой. Объекты могут показываться единым списком с разделением на страницы либо, если их сравнительно немного, RCV автоматически разбивает всю коллекцию на части, каждая из которых характеризуется собственной первой буквой в именах объектов, и отображает ссылки для быстрого переключения между частями. Также приложение предоставляет доступ ко всем средствам поиска данных, заложенным в RCM, что позволяет сократить общее количество показываемых ресурсов (рис. 5).

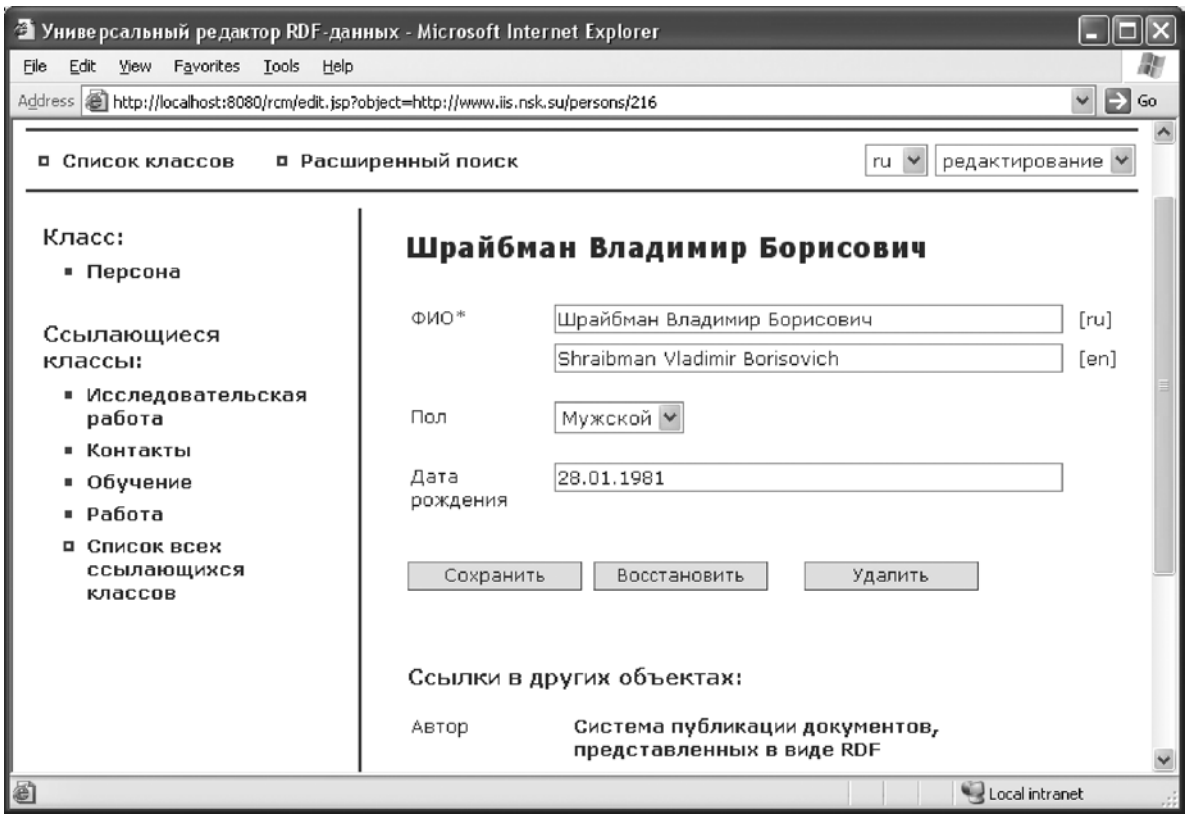

Рис. 5

[9] XML Schema Part 2: Datatypes Second Edition / http://www.w3.org/TR/xmlschema-2/

И последняя, наиболее важная функция RCV – это предоставление универсального интерфейса для просмотра и редактирования самих данных, т. е. значений свойств ресурсов. На приведенном ниже рисунке изображена страница для изменения первичных сведений об авторе статьи, хранящихся в экспериментальном RDF-репозитории кафедры программирования Новосибирского государственного университета (рис. 6).

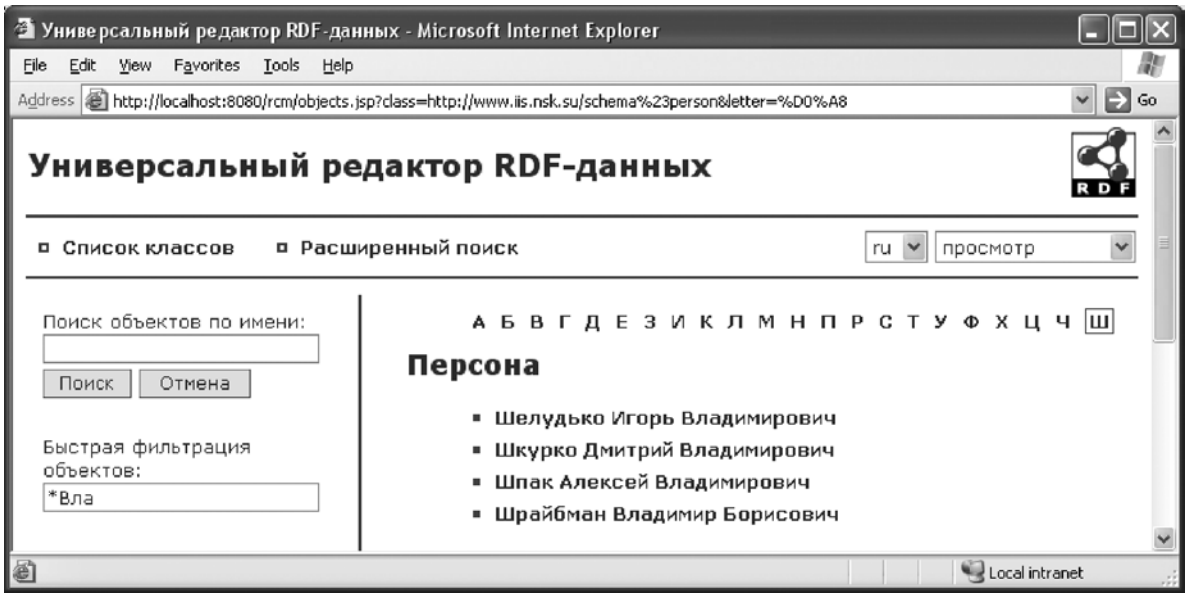

Рис. 6

Одним из немногочисленных конфигурационных параметров RCV является список языков, поддерживаемых RDF-хранилищем. Как показано на примере, для требующего локализации свойства «ФИО» следует указывать все подходящие переводы. Остальные значения редактируются в соответствии с текущим выбранным языком. В частности, «дата рождения» указывается в российском варианте написания, а для выбора значения перечислимого типа «пол» программа предлагает понятные человеку текстовые метки на правильном языке. Интерфейс RCV также легко локализуем, и если соответствующий перевод поставляется вместе с Web-приложением, то язык, используемый на страницах, автоматически синхронизуется с тем, который выбран для работы.

Основной способ применения RCV – это использование его для быстрого просмотра и редактирования информации, хранящейся в некотором RDF-репозитории. Также приложение может быть с успехом задействовано для отображения системы типов данных, как проектировщиками новых схем, так и просто людьми, желающими разобраться в уже имеющихся. Помимо этого программа может применяться и в образовательных целях для наглядной демонстрации фактов и понятий, описанных при помощи RDF и RDFS.

Другое применение

Хотя RCV может быть использован и в качестве средства почти мгновенного построения готовых информационных систем, это не слишком целесообразно, поскольку отображение классов и ресурсов тяготеет к технической точке зрения и происходит в точном соответствии со схемой данных, а не так, как было бы максимально удобно конечному пользователю.

Вместо этого идея универсальной работы с RDF-данными может быть расширена до отдельного имеющего самостоятельное значение приложения, которое будет обеспечивать доступ уже не к отдельным ресурсам, а к целым множествам контекстно связанных информационных объектов. Получив на вход схему классов, а также дополнительное, специально подготовленное описание логики отображения их отношений, такая программа позволяла бы просматривать и редактировать сразу несколько RDF-ресурсов. Например, одна страница этой системы могла бы показывать не только первичные данные о человеке, но и хранимую отдельно контактную информацию, сведения о месте работы, семейном положении и роли, исполняемой им в рамках конкретной предметной области.

Если в описанное приложение добавить возможность разнообразного форматирования выходных данных при помощи XSLT или каких-то других стилевых шаблонов, то полностью будет покрыт и даже превышен класс задач, решаемых технологией SMART.

Помимо готовых программ могут быть созданы вспомогательные модули и библиотеки, упрощающие разработку специализированных информационных систем. Сюда можно отнести JSP-теги, инкапсулирующие использование RCM на страницах сайтов, средства генерации Web-сервисов, какие-то крупные, логически законченные блоки будущих приложений и многое другое.

Платформа для разработки приложений

Текущая версия RCM имеет две недоработки. Первая из них заключается в том, что контент-менеджер, хотя и умеет взаимодействовать с несколькими RDF-хранилищами одновременно, но не предоставляет средств поиска и манипулирования распределенной информацией, которые осуществляли бы свою работу автоматически, а не под руководством программы, использующей RCM. Второй пробел – это отсутствие механизмов для ограничения доступа к классам ресурсов и даже к конкретным их свойствам.

Когда оба указанных недостатка будут преодолены, а вспомогательные библиотеки, описанные в предыдущем разделе, созданы, можно будет говорить о создании вокруг RCM целой технологической платформы для разработки разнообразных приложений, использующих RDF. Ее общее устройство будет выглядеть следующим образом (рис. 7).

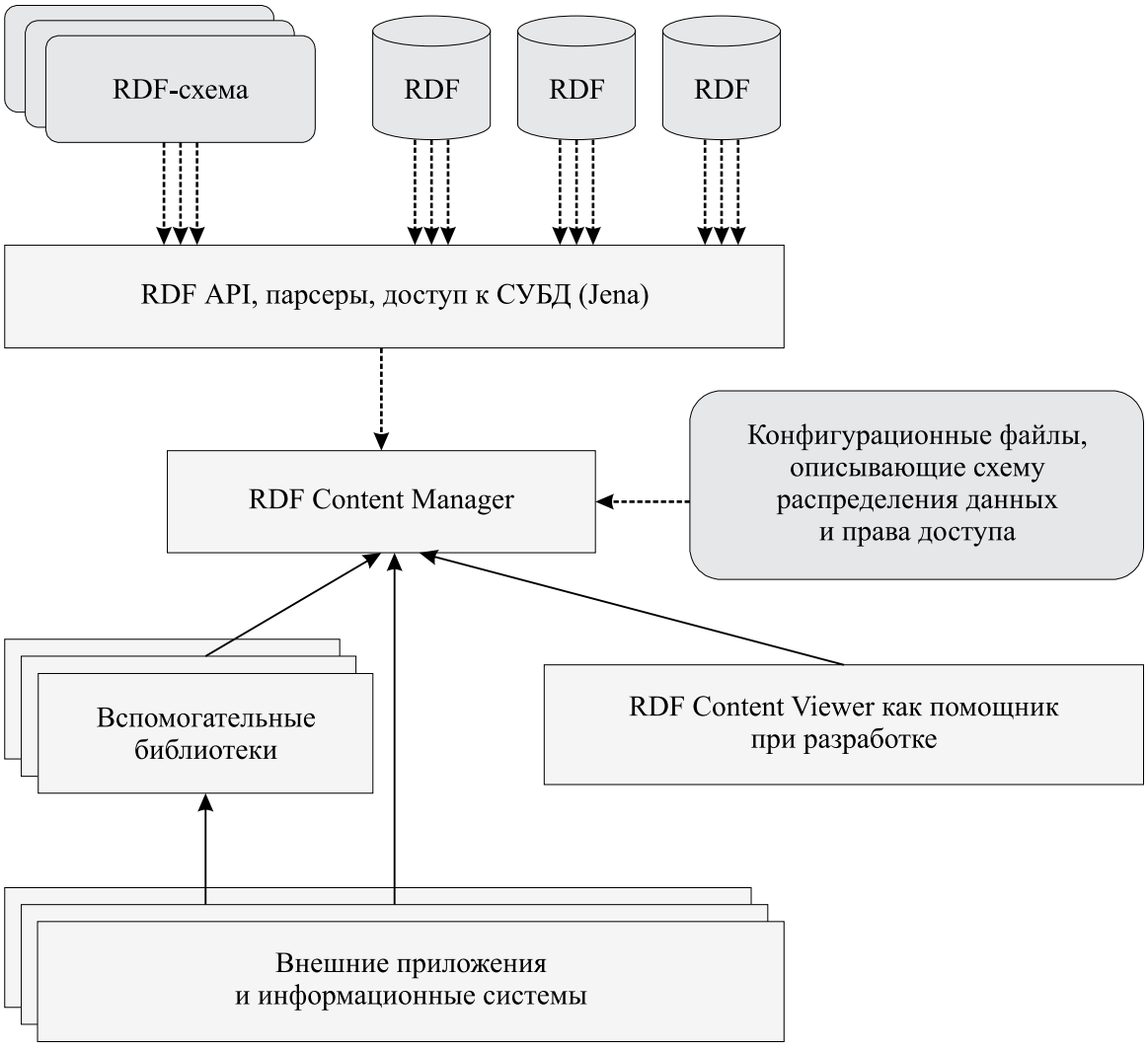

Рис. 7

Тем не менее, как было показано на примере RCV, уже сегодня библиотека RCM может быть успешно использована в различных проектах, рассматривающих RDF как основной способ представления и хранения информации.

Список литературы

Бездушный А. А., Бездушный А. Н., Нестеренко А. К. и др. Архитектура RDFS-системы. Практика использования открытых стандартов и технологий Semantic Web в системе ИСИР // Тр. пятой Всерос. науч. конф. «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» (RCDL2003). СПб., 2003. С. 45–60.

Шрайбман В. Б. Выражение семантики данных. RDF против XML. 2003 // http://www.citforum.ru/internet/xml/rdf_xml.

Шрайбман В. Б., Гуськов А. Е. Разработка информационных систем на основе RDF технологии // Тр. XLI Междунар. науч. студ. конф. «Студент и научно-технический прогресс». Новосибирск, 2003. Ч. 1. С. 143–150.

Hendler J., Berners-Lee T., Miller E. Integrating Applications on the Semantic Web. 2002 // http://www.w3.org/2002/07/swint.

Koivunen M., Miller E. W3C Semantic Web Activity. 2001 // http://www.w3.org/2001/12/semweb-fin/w3csw.

Материал поступил в редколлегию 19.09.2006