Интеграция электронных библиотек и систем управления документами, как задача, решаемая системой обмена информации между объектно-ориентированными информационными системами

М. В. Кулагин
ЦНТК РАН,
gl@ras.ru

А. С. Лопатенко
ЦНТК РАН
andreyl@ccas.ru

Аннотация

Создано программное обеспечение передачи данных между различными информационными системами. Данный программный продукт позволяет осуществлять операции передачи данных с их трансформацией между различными объектно-ориентированными и реляционными системами. Частично решены проблемы импорта и экспорта информации, представленной в файлах с метаданными. Класс систем достаточно широк и подробно описан ниже. Одним из наиболее значимых его приложений, является передача данных между электронными библиотеками и между электронными библиотеками и другими информационными системами, например, системами управления документами.

1 ВВЕДЕНИЕ

В Вычислительном Центре РАН создается информационная система ИСИР РАН (Интегрированная Система Информационных Ресурсов РАН). Как декларировано в [1]: “Основной целью проекта является разработка концептуальной основы и инфраструктуры для интеграции разнородных информационных и вычислительных ресурсов РАН в единое информационное пространство. Этот базис должен обеспечить объединение в единое пространство всевозможных электронных библиотек, информационных и вычислительных систем, использующих как собственные принципы организации, так и технологию открытой архитектуры системы проекта ИСИР или непосредственные ее релизы.”

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

Одной из целей электронной библиотеки Интегрированной Системы Информационных Ресурсов РАН является объединение данных ИСИР РАН с данными других информационных систем, доступ к которым представляет собой большую ценность для сотрудников РАН. При этом необходимо обмениваться информацией с другими информационными системами, хранящими информацию о научных сотрудниках, институтах, публикациях, например, MathNET. Изложение разработанных модулей интеграции ИСИР РАН представлено в главе “Внедрение», приведенной ниже. Попытки решения задачи интеграции ИСИР РАН с другими информационными системами, для каждой конкретной системы своими методами, приводит к неоправданным трудозатратам, поскольку информационные системы изменяются, модифицируются, меняются объектные модели, реляционные схемы, форматы метаданных. Решение задач по созданию и поддержке программного кода для каждого конкретного случая являются трудновыполним. Кроме того, методы работы, примененные при интеграции ИСИР РАН с одним видом систем, могут быть использованы при интеграции с другими системами..Поэтому представляется необходимым создать единый программный модуль для интеграции ИСИР РАН с другими информационными системами. Уровень общности, который заложен в этом модуль позволит использовать его в качестве самостоятельного информационного приложения для интеграции данных различных информационных систем, а не только как часть ИСИР.

Этот модуль должен быть настраиваемым на схемы данных, среды передачи данных, также необходимо, чтобы он был независимым от операционной среды. Для общего случая решения пока не найдено, но выделены типы систем, с которыми данный модуль может работать.

2 ПОСТАНОВКА ЗАДАЧИ

Необходимость интеграции электронных библиотек и других информационных систем уже неоднократно обсуждалось авторами работ [2-5]. В частности, в работе [2] , согласно мнению экспертов CIIR (National Science Foundation (NSF) Center for Intelligent Information Retrieval (CIIR)), представлен перечень 10 наиболее важных проблем систем запроса информации (Informational Retrieval Systems). Обмен данными и работа в распределенной среде- две наиболее важные проблемы в этом перечне. Первая из проблем (Integrated Solutions) является темой нашей работы.

Как показано в работе [6] проблему информационной интероперабельности возможно разделить на проблемы структурной интероперабельности, системной интероперабельности, лингвистической и семантической интероперабельности. Достаточно близкая точка зрения высказывается в [10].

С точки зрения представления данных информационные системы можно разделить на объектно-ориентированные, реляционные, сетевые, фреймовые. По нашему мнению, для большинства задач объединения данных, лежащих в области электронных библиотек, необходимо работать с объектно-ориентированными и реляционными представлениями данных. Фактически все важные системы электронных библиотек описываются объектно-ориентированной парадигмой с наследованием и полиморфизмом, либо каким-либо подмножеством этой парадигмы. Например, информационная структура многих электронных библиотек (Kahn-Wilensky framework, DIENST, FEDORA) также описывается объектно-ориентированной парадигмой. Широко употребляемые прикладные информационные системы, которые будут связываться с цифровыми библиотеками, например, Системы Управления Документами (СУД), порталы являются объектно-ориентированными.

По мнению авторов, системой обмена данных, решающей большинство практических задач будет информационная система, позволяющая: экспортировать/импортировать данные

- в широко известные форматы метаданных, используемые информационными системами. Важность экспорта и импорта метаданных видна из работ [7,8,9];
- в информационные системы, базирующиеся на РСУБД;
- в информационные системы, имеющие доступ к своим ресурсам на основе объектных API (COM, EJB, CORBA).

Нами предпринята попытка решения проблем структурной интероперабельности и небольшого числа проблем семантической интероперабельности для таких систем.

Следуя классификации Sheth [10], по типу интероперабельности данных мы разделяем приложения объединения данных информационных систем на два больших класса:

- Системы структурной интероперабельности. Это системы первого и второго поколения по Sheth. В данный момент, в основном это системы, ориентированные на РСУБД (хранилища данных, сервера репликации данных и распределенных транзакций), хотя имеется ряд систем, не ориентированных ра РСУБД ;
- Системы семантической интероперабельности. Или "системы обмена информации на основе семантических описаний информационных систем"[10-14].

Из систем первого типа, по нашему мнению, наиболее технологически развиты и доступны системы, ориентированны на обмен данными между РСУБД и соответственно, они могут хранить информацию о реляционной схеме источника и приемника информации. Из этих систем нам наиболее интересны хранилища данных (Data Warehouse), так как полнота схемы данных источника и приемника у них не уступает другим системам, и при этом, они обладают наиболее широкими возможностями трансформации данных. Известно, множество реализаций систем такого класса: Oracle DataMart Suite, Oracle DataWarehouse, DataMirror, Microsoft DataTransformation Services.

Анализ текущих возможностей коммерческих реализаций Data Warehouse и научных исследований в этой области показал, что для решения поставленной нами задачи интероперабельности они обладают следующими недостатками:

- Ориентированы исключительно на реляционные источники и приемники данных, либо на сравнительно слабые объектные модели
- Предлагают недостаточные мощные возможности трансформации (что связанно со структурной интероперабельностью) данных. Наиболее часто предлагаемый способ расширения возможностей трансформации – реализация ее на процедурном коде не анализируемом и не понимаемом системой хранилища данных (Обычно используется Oracle PL/SQL, ECMA Script, Perl).
- Не рассматриваются аспекты семантической интероперабельности

В России разработаны системы подобного класса, например [5] Наша разработка не противоречит этим системам. Одна из наших целей довести методы интеграции данных подобного рода систем до конкретных источников и приемников данных. Другая цель создать инструментарий для интеграции данных на основе открытых стандартов.

Возможности систем семантической и структурной интероперабельности третьего поколения достаточно велики, но:

- Отсутствуют доступные коммерческие реализации систем третьего поколения
- Большей частью, они ориентированны на частные способы представленя данных и их моделей. Инте-

грация их с распостраненными системами и поддержка существующих стандартов требует больших трудозатрат

Исходя из вышеприведенных утверждений решено, что разрабатываемое программное обеспечение должно быть

- легко интегрирующееся на синтаксическом, системном и структурном уровне с объектно-ориентированными системами, с системами, базирующимеся РСУБД, и с файловыми описанимя информации
- основанное на открытых стандартах
- позволяющее адаптировать практики решения задач структурной и семантической интероперабельности информационных систем

Существуют два стандарта для представления данных необходимых для хранилищ данных: Object Management Group Common Warehouse Model (OMG CWM) [16] и Metadata Coalition Open Informational Model (MDC OIM) [15]. Первый стандарт предназначен исключительно для решений хранилищ данных, он базируется на UML, имеет метамодели для описания моделей реляционных, XML, объектно-ориентированных источников и приемников информации, а также для описания процесса трансформации, хранения данных о процессе трансформации, о переданных данных. Второй стандарт представляет собой множество метамоделей для работы в различных областях информационных систем, Среди них имеются метамодели для моделирования РСУБД в обобщенном варианте и конкретных РСУБД (MS SQL Server, Oracle). и описания трансформации данных при их передаче из одной РСУБД в другую (Data Transformation Information Model)

Сравнение этих стандартов и некоторых технологий хранения данных о процессах и моделях данных в хранилищах данных приводится в [16-18]. При сравнении стандарта OMG (CWM) и стандарта MDC (OIM) нами был выбран стандарт OMG, который :предлагает более общую модель описания источников данных (не только реляционные, но и XML, и объектно-ориентированные), опирается на открытые стандарты, принятые многими производителями ПО (OMG MOF, OMG UML), также имеются реализации этого стандарта на многих платформах и ведутся активные работы по расширению CWM. В последней версии стандарта OIM (версия 1.1) появились модели для описания XML и record-oriented данных, но эти модели не интегрированны с Data Transformation моделью. Необходимыми для нашей системы являются модели Warehouse Process, Warehouse Operations стандарта CWM аналоги которых отсутствуют в OIM.

Одним из достоинств OIM является наличие MS Repository - бесплатного репозитория для платформы Win32. MS Repository позволяет хранить информацию представленную в OIM. Для доступа к информации представляется набор COM интерфейсов. Ряд производителей средств проектирования и моделирования (Microsoft, Rational, Platinum) поддерживает MS Repository, что существенно упрощает процесс создания средств пользователя для описания источников и приемников данных и процессов передачи данных.

В связи с этим ведутся работы по созданию конвертора CWM <-> OIM, позволяющего представлять данные CWM в виде данных OIM и обратно. Такой конвертор позволит сохранять метаданные в репозитории MS Repository.

3 РЕШЕНИЕ ЗАДАЧИ И РЕАЛИЗАЦИЯ.

Архитектура системы состоит из независимых с точки зрения реализации модулей:

1. Модуль представления данных и метаданных источника информации в виде объектной модели
2. Модуль передачи данных из источника информации в модуль 1
3. Модуль трансформации данных, представленной в виде объектной схемы источника (модуль 1).
4. Модуль представления преобразованных данных источника информации в виде объектной модели
5. Модуль загрузки данных в приемник информации.

Модуль трансформации данных (следуя модели CWM) делится по функциональности на следующие подмодули:

- подмодуль трансформации, который занимается непосредственно единичными актами извлечения данных из таблиц, объектов и их трансформацией;
- подмодуль группировки трансформаций, который группирует единичные акты трансформации в цепочки и указывает порядок их исполнения;
- подмодуль специализированных трансформаций, определяющий специализированные наиболее часто используемые единицы трансформации.

Преимущества такой архитектуры заключаются в том, что новые типы информационных систем источников и приемников информации добавляются расширением некоторых модулей (2, 5), а не изменением всей системы; Модуль трансформации независим от других модулей; возможно использование различных стандартов на трансформацию данных (Data Warehouse модели, XSLT модель трансформации). Например, наша система изначально была настроена на импорт данных из DMA источников. При появлении тех же данных в виде XML файлов потребовалось только дописать модуль импорта информации из XML (описан ниже), в то время как модули трансформации данных и загрузки остались без изменений.

Для решения задач структурной интероперабельности необходимо разрешить следующие проблемы.

3.1 Проблемы актуальности информации и идентификации информапционных ресурсов.

В случае если данные источника информации преобразуется в данные приемника, возникает вопрос, имеются ли в приемнике эти данные или нет, требуется ли соз-

давать в приемнике новые объекты, атрибуты, отношения или обновлять уже имеющиеся.

Предлагается решения для варианта трансформации объект -> объект. Решения для случаев, когда объекты источника описывают отношения или атрибуты приемника находится в разработке.

Возможно несколько типов ситуаций, каждая из которых имеет самостоятельный вариант решения:

- Ситуация: объекты системы-приемника идентифицированы по единой схеме с системой-источником информации. Решение: если их идентификаторы, совпадают с идентификаторами приемника информации, то при загрузке данных, загрузчик пытается обновить информацию об этих объектах, либо выдает диагностическое сообщение. Совместное использование идентификаторов возможно в случаях совместного использования общей идентификационной системы (HANDLE System. DOI, сервисы имен CORBA), наличия в приемнике данных, ранее полученых из источника с сохраненными идентификаторами, совместного использования данных.
- Ситуация: объекты системы-приемника информации были ранее импортированы из системы-источника, но приемник и источник информации не имеют общей схемы идентификации объектов. Решение: в системе приемнике-информации используется самостоятельная схема идентификации ресурсов. Для хранения информации о соответствии объектов источника и приемника информации для каждого акта передачи данных об объекте хранится объект класса StepExecutionObj, являющегося расширением класса CWM StepExecution. Описания классов даны на языке Java.

```
1. class ISIR.Infoexchange.ObjToObj {
2.   Object SourceOID; // OID объекта в системе-источнике
     данных
3.   Object Target OID; // OID объекта в системе-примнике
     данных
4. }
5. class ISIR.Infoexchange.StepExecutionObj extends
   CWM.StepExecution {
6.     Vector ObjYoObjs; // список актов трансформации
7. }
```

- Ситуация: помимо OID каждый объект источника информации имеет первичный ключ – совокупность атрибутов уникально идентицифирующую объект, и этот ключ сохраняется в системе-преимнике. Предполагается, что все атрибуты ключа объекта источника передаются в объект приемника без изменения их значений. Решение: зная значения атрибутов первичного ключа данного объекта возможно установить имеется ли его копия в системе приемнике данных и уникально идентифицировать эту копию. Объект класса CWM.UniqieKey системы-приемника указывает какие атрибуты идентифицируют объект. Используя объекты класса AttributeMap, описывающие трансформацию объектов данного класса находятся атрибуты объекта класса-приемника, принадлежащие перивчному ключу. Используя соответствие атрибутов и их значение идентифицируется объект приемника.

3.2 Проблемы интеграция с представлением информации в файлах.

Одним из наиболее распространенных методов доступа к информации является ее представление в виде файловых описаний. Такой обмен данными часто используется в случаях 1) если нет возможности функционального контакта между системами в силу несовместимости их моделей данных, API или по административным причинам; 2) данные одной системы будут использоваться многими другими системами и нежелательно перегружать систему-источник. У этого метода есть недостатки, среди них 1) нет гарантии актуальности данных, которые описаны в файле; 2) синтаксис и семантика описаний данных многих типов еще на стандартизованы и по сути для обмена данными с каждой новой системой приходится писать отдельный модуль. Наше решение частично разрешает вторую проблему. Имея анализатор такого синтаксиса данных, представляющий данные файла в виде объектной модели, возможно не писать приложение заново для каждой новой модели данных, а только настраивать на имеющуюся модель.

Одним из часто употребляемых языков описаний данных сейчас служит XML. Семейство этого языка включает язык RDF, которые позволяет описывать сложные схемы информационных ресурсов (направленные помеченные графы ресурсов), а также XLink язык для описания связей между ресурсами и XPointer для указания множества ресурсов. Стандартом для API анализаторов XML документов является стандарт W3 DOM (Document Object Model)[19].

Неоднократно рассматривалась совместимость схемы данных XML/SGML описаний и совместимость ее с объектными описаниями [5,20,21]. Создан стандарт представления информационной модели XML документов (W3.ORG XML Information Set) и ведутся разработки производителями ПО (Microsoft XML Object Model [22])

Существует ряд технологий (компаний IBM, ObjectSpace и др.) для сериализации Java щбъектов в виде XML. Несмотря на то, что эти технологии обладают большими выразительными возможностями при рабоет с Java объектами, на текущий момент невозможно используя их специфицировать формат XML/RDF представления и указать как должна производиться генерация.

Для того, чтобы посредством нашей системы генерировать XML/RDF представление информации необходимо 1) создать схему RDF или XML используя объекты ниже представленных классов, 2) для блока трансформации указать, как объекты информационной системы-

источника должны быть представлены в виде объектов RDF/XML.

Нами предлагается основанная на UML спецификация генерации XML, RDF описаний информационных ресурсов, совместимое с CWM. Для генерации RDF описаний предлагается следующее множество классов.

Класс, описывающий генерацию RDF описания информационного ресурса

```
8.  class RDFClass extends UML.Core.Class {
9.    String RdfName;  // наименование типа rdf описания, например, rdf:description или ISIR:person
10.   String IdTAG;    // используемый таг идентификатора (ID | about)
11.   UML.Core.Attribute IdProperty;          // атрибут, содержащий значение id
12.   String RDFType;     // RDF тип
13. }
```

Класс, описывающий генерацию описания свойства ресурса (не контейнера)

```
14. class RDFProperty extends UML.Core.Attribute {
15.   String RdfName;  // наименование rdf свойства, например, dc:Creator
16.   String ParseType; // тип свойства (Resource | Literal | Simple | Reference)
17.   String Type;    // rdf тип
18.   UML.Core.Attribute IDProperty; // значение какого атрибута является id данного свойства
19.   String eq; // entity qualifier
20.   String vq;  // value qualifier
21. }
```

Класс, описывающию генерацию свойства-контейнера

```
22. class RDFContainerProperty extends RDFProperty {
23.   String ContanerType;          // тип контейнера (Bag | Seq | Alt)
24.   String ReferencedorInline;   // ссылочное свойство или inline
25. }
```

При импорте данных RDF необохимо привести их к базовой модели, для последующих операций. Для этого предлагаются следующие описания (основанные на [20])

```
26. class RDFClassImport extends RDFClass {
27.   UML.Core.Class Class; // какой класс должен быть создан для представления данного информационного ресурса
28.   UML.Core.Attribute ParentAttr ;  // в случае принадлежности элемента, описывающего данный информационный ресурс, другому элементу, следует ли сохранять отношение принадлежности. И если следует, то какой атрибут объекта родителя должен хранить данный элемент
29.   UML.Core.Attribute ContentAttr; // В какой атрибут помещать content данного элемента
30.   UML.Core.Attribute attrName;  // атрибут, устанавливаемый по умолчанию
31.   Object attrValue;   // значение, устанавливаемое по умолчанию
32. }
33.
34. class RDFPropertyImport extend RDFProperty {
35.   UML.Core.Attribute contentAttr; // в какой атрибут импортировать значение данного свойства
36.   String pcdataClass;
37.   UML.Core.Class objectType; // объект какого класса необходимо создавать для представления данного свойства (для свойств типа Resource)
38.   UML.Core.Relationship referenceType;  // ссылку какого типа необходимо создавать для представления ссылки между описания ресурсов в RDF
39. }
```

3.3 Интеграция с системами РСУБД

Нами не ставится задача предложить методы по представлению объектно-ориентированных схем в виде реляционных схем. Данная тема достаточно подробно разработана теоретически и есть много подробных методологий, и инструментов по представлению объектно-ориентированных схем в виде СУБД(и обратно)[23-28]. В этих источниках освещено решение достаточно многих проблем объектного представления данных РСУБД и хранения данных РСУБД. В этой статье мы описываем решение проблем, не расмотренных ранее, спецификации на хранение информации о преобразовании реляционных данных в объектные. Отметим, что с объектно-ориентированным подходом сильно связана модель сущность – связь (ER) [29]. Наша цель описать архитектуру объектно-ориентированного слоя к реляционным базам данным, а именно для схем РСУБД и объектно-ориентированной схемы необходимо создать формат описания на основании которого программное обеспечение сможет предоставлять заданный объектно-ориентированный интерфейс к РСУБД.

Наша схема является расширением метамодели OMG CWM. Объектную схему источника информации можно разбить на две части, на описания самих объектов и на описание связей между ними. Поэтому задача переноса данных в РСУБД разбита на две: перенос информации об объекте и перенос связей между ними. Это разделение является достаточно условным, так как, например, атрибут объекта может быть сложной структурой данных и не отличаться от вложенного объекта. Кроме того, возможны ситуации в которых при загрузке объектов в БД необходимо учитывать связи между объектами. Например, в случае, если таблицы в БД, хранящие объекты, связаны идентифицирующим ключом, то невозможно загрузить зависимые объекты до идентифицирующих.

Задача интеграции с система РСУБД состоит из двух частей: хранение данных в РСУБД и извлечение данных из РСУБД. Задача извлечения данных более проста, чем задача хранения данных. Поэтому ниже описывается решение для задачи хранения.

3.4.1 Проблема хранения 'ядра' объекта

Под ядром объекта в даннйо статье понимается совокупность атрибутов объекта, ответственных за его идентификацию (OID) и совокупность атрибутов объекта с multiplicity=1.

Обычно разработчики баз данных используют один из трех подходов для представления иерархии классов в схеме данных РСУБД[27]:

- отображение всей иерархии классов в одну таблицу БД (плюс дополнительные таблицы, хранящие дополнительные атрибуты конкретных классов);

- отображение каждого конкретного класса в отдельную таблицу БД;
- отображение каждого класса в свою собственную таблицу.

В любом из этих случаев для каждого класса объектов существует таблица (набор таблиц), хранящая набор его базовых атрибутов. Первичный ключ этой таблицы используется для связывания с ней всех других таблиц, ответственных за хранение атрибутов объекта и его связей.

В случае отображения каждого класса в свою собственную таблицу для хранения ядра объекта необходимо обращаться ко множеству таблиц. Множество таблиц и порядок обращение кодируется использованием объекта класса TransformationActivity, который хранит множество объектов класса TransformationTask, каждый из которых ответственнен за сохранение в опредленную таблицу.

После того как сохранено 'ядро' объекта в БД, необходимо выполнить дальнейшие операции по сохранению его атрибутов.

3.4.2. Проблемы хранения атрибутов объектов.

Простые атрибуты.

В случае необходимости отображения нескольких атрибутов в один кортеж используется объект типа ISIR.InfoExchange.AttributeMapExt, который в атрибуте source хранит список атрибутов источника, а в атрибуте target список атрибутов приемника. Тем самым указывается, что данное отображение осуществляется одним актом в одну строку. Класс AttributeMap CWM не позволяет конкретизировать то, что мы указали, нами расширена модель CWM и введен класс ISIR.InfoExchange.AttributeMapExt, который является подклассом CWM.AttributeMap и содержит в себе коллекцию объектов AttributeMap, определяющих конкретное отображение атрибутов.

В случае отображение каждого атрибута собственный кортеж используется отдельный объект класса AttributeMap для каждого атрибута.

В случае отображение атрибутов в индивидуальные кортежи зачастую необходимо классифицировать эти кортежи. Это необходимо, например, в случае, если значения классифицируемы справочником. Для этого введен подкласс ISIR.InfoExchange.AttributeMapIdentified класса AttributeMap. В этом классе введена функция GetValue(sourceAttr, targetAttr) , возвращающая значения атрибутов вставляемых в таблицу БД в частности ключей связи со справочниками. Большинство реальных задач решается ведением массива структур {sourceAttrMap, KeyValue }.

Сложных атрибуты.

Для хранения таких атрибутов-списков вводится класс ISIR.InfoExchange.AttributeMapList подкласс AttributeMapIdentified с атрибутами isToOneRow, IsOrdered, FirstNumber и функцией GetOneString. Булевский атрибут isToOneRow указывает не следует ли отображать данный список в одно значение и если следует, то необходимо использовать public String GetOneString (Attribute source). Если не следует, тои каждое значение из списка заносится в отдельную строку таблицы. Следует ли упорядочивать эти строки указывает атрибут IsOrdered, атрибут FirstNumber указывает первое значение счетчика, используемого для нумерации строк. Все остальное как и в предыдущем случае. Методы отображения более сложных атрибутов в базу данных пока не разработаны. Для такого случая введен объект AttributeMapExt с методом Save.

Для случая, когда атрибут есть список одинаковых кортежей, то вводится класс ISIR.InfoExchange.AttrubuteMapListCorteges, который является подклассом AttributeMapList.Этот класс позволяет идентифицировано сохранять упорядоченные и неупорядоченные элементы. Кроме того, в нем храниться массив структур {номер атрибута, столбец, функция}. Причем номер атрибута указывает, какой атрибут отображается в какой столбец с использованием какой функции преобразования.

3.4.3. Проблемы хранения отношений между объектами. Проблемы разрешения зависимостей объектов.

Во многих приложениях ссылочна целостность базы данных требует упорядочивания ввода объектов (соотвествующих строк в таблицу). Правильный объект должен быть введен ранее зависящих от него слабых объектов. Целевой кортеж должен быть введен ранее целевого кортежа.

При условии различной структуры представления данных возможно возникновение проблем ввода информации.

В качестве примера рассмотрим следующую ситуацию. База данных описывает персоны и проекты. В каждом проекте может участвовать неограниченное число ичполнителей. Каждый проект введет один и только один ответственный исполнитель. Предположим, в данной базе данных персоны являются правильными объектами, а проекты слабыми.

Описания персон хранятся в таблице Persons. Описания проектов в таблице Projects. Таблица Projects (ссылающееся отношения) связана с таблицей Persons (целевое отношение) идентифицирующим внешним ключом. Отношение исполнитель (многие ко многим) хранится в таблице ProjectParticipation.

Также предположим, что данные поступают в виде XML файлов следующего формата

```
40.  <projects>
41.  <project id=47384 url="http://www.projects.org/soft/47384">
42.    <project-name lang="ru">
43.    <registered>Электронные библиотеки</registered>
44.     <internal>…</internal>
```

```
45.     </project-name>
46.     <chief>
47.       <person id=27384>
48.       ….
49.       </person>
50.     </chief>
51.     <participants>
52.     …
53.     </participants>
54.   </project>
55.   ….
56.   </projects>
```

'Поступенчатый' ввод данных в том порядке в котором разбирает их анализатор (вообще, в том порядке в каком они могут быть извлечены из источника) не обязательно будет тем же в каком их требуется вводитьв базу данных. В случае хранилищ данных эта задача решается просто: определяется порядок выборки данных из источника информации, ввод их в приемник осуществляется в том же порядке. Но решить проблему этим способом не всегда возможно.

Предлагается следующее частичное решение проблемы. Данное решение реализуемо в случае, если возможна выборка элементов из источника данных в соотвествие с критериями выбора (для XML источников предлагается использовать XPath/XPointer запросы, для объектных репозиториев соответствующие объектные языки запросов). Решение корректно, если зависимости между элементами выражены только во внешних отношених, и нет других зависимостей (которые могут быть выражены в тригерах и. т. д.) Данное решение не требует спецификация порядка ввода элементов в базу данных, необходимо иметь описание внешних ключей между отношениями. При расширение объектной схемы и схемы базы данных адопатция к новым условиям будет происходить автоматически. Недостатком данного решения является неэффективность – число запросов к источнику информации равно числу классов в системе приемнике (при более эффективной реализации числу зависимых классов от классов в источнике информации).

Информация о внешних ключах в приемнике информации хранится в виде набора объектов CWM.ForeignKey. Информация о том является ли данный внешний ключ идентифицирующим выводится из условия nullable для колонок внешнего ключа ссылающегойся таблицы. Отношения зависимости вводят частичную упорядоченность на множестве отношений (предполагается отсуствие циклов).

Организуется цикл ввода данных. В каждой новой итерации цикла выбирается класс – параметр данного цикла таким образом, что в будущих итерациях не будет классов от которых зависит данный. Из источника информации выбираются объекта класс соотвествующего параметру цикла. Все эти объекта вводятся в базу данных, если необходимо вводятся отноешния между ними и уже введенными объектами. Для ввода отношений между ними и уже введенными объектами используется первая схема пункта 3.1.

3.5 Распределенная среда

В распределенных средах возникают дополнительные требования к безопасности данных и администрированию системы. Для обеспечения безопасности данных и легкости администрирования системы необходимо расширить функциональность системы от функций, которые она выполняет сейчас (Data Management, Data Presentation, Communication по классификации основных задач интероперабельности электронных библиотек [30]), до функций типов Operations и Protection той же классификации. Наиболее важные с нашей точки зрения направления такого расширения указаны ниже.

- Расширяемость схемы данных. Необходимо, чтобы расширение информационной модели узла не влияло на процессы обмена информации между узлами. Новые типы объектов в системе-источнике семантически эквивалентные имеющимся с точки зрения системы-приемника информации должны автоматически, если возможно, передаваться в приемник во время процесса передачи данных.
- Гибкость. Правила передачи данных могут быть общими, но не обязательными в каких-то конкретных условиях, позволяющими работать в распределенной системе состоящей из похожих, но не обязательно одинаковых узлов. Если в системе-источнике или системе-приемнике данных информационные ресурсы представлены не точно таким способом, как это декларировано в правиле передачи данных, то должно быть возможна передача данных на том уровне насколько представления информационных ресурсов соответствуют требованиям правила.
- Безопасность. В распределенной системе возможно наличие общих для всех узлов правил безопасности данных. Необходимо иметь механизмы декларации правил безопасности данных соблюдаемые на всех узлах. При выполнении актов передачи данных, система должна проверять не нарушает ли данная передача правила безопасности данных.

4 ВНЕДРЕНИЕ

В рамках проекта по созданию Информационной Системы Информационных Ресурсов РАН создаются АРМ ученых секретарей и АРМ руководителей научных проектов, коллективов. Главная цель создания этих АРМ - повысить качество и эффективность информационной работы, связанной с документационной деятельностью, соответствующих работников. Данные АРМ позволяют хранить, искать, использовать, связывать электронные копии документов, свойственных этим рабочим местам, организовывать ход работ. АРМ ученых секретарей и АРМ руководителей научных групп созданы на базе одной из промышленных Систем Управления Документами

Важной особенностью этих рабочих мест является наличие в документах информации, имеющей значение для научной общественности. Механизмы интеграции данных соответствующих АРМ с цифровой библиотекой Академии Наук позволят публиковать информацию в интернет для публичного доступа.

Кроме того, при эксплуатации цифровой библиотеки ИСИР РАН необходимо разрешить проблему актуальности данных. Электронная библиотека хранит данные о персоналиях и организациях РАН, публикациях научных сотрудников РАН и проектах, ведомых в рамках РАН. ИСИР РАН предусматривает возможности ввода информации посредством пользовательских интерфейсов. В данный момент функции ввода актуальной информации возложены на ученых секретарей Отделений и институтов РАН. Как показывает практика, в незначительном числе случаев обеспечивается актуальность и полнота информации. Причина этого – отсутствие непосредственного эффекта от ввода информации для соответствующих лиц. Но в то же время сотрудники пользуются информационными системами, связанными с их непосредственной деятельностью и разрешающей их проблемы. Если данные необходимые для цифровой библиотеки будут храниться в АРМ сотрудников, то интеграция АРМ и ЦБ позволит разрешить проблемы актуальности данных цифровой библиотеки.

Наиболее важными по представленным в них данных являются документы АРМ ученых секретарей “Протоколы ученых советов”, “Годовые отчеты”. В годовых отчетах содержится информация о научной деятельности сотрудников институтов, о проведенных конференциях, структурных изменениях. В протоколах отделения Президиума РАН содержится информация об структурных и кадровых изменениях в Отделении, о назначении директоров институтов и редакторов журналов.

Все документы АРМ ученых секретарей являются подтипом общего типа “Документ”. Документооборот Академии Наук не соответствует всем правилам ГОСТ Р 6.30-97 “Требования к оформлению документов“, но можно выделить ряд формальных требований к документам ученых секретарей. Формальная структура данных представлена ниже.

Класс, моделирующий все документы ученых секретарей. Наименования атрибутов данных в соответствии с ГОСТ Р 6.30-97

```
57. Class Document {
58.     Nomenclature Type;  // тип документа по номенклатуре ()
59. Organization Org;      // организация-автрор документа(реквизит 06)
60. Date          dtData;  // дата документа. Семантика зависит от типа документа(реквизит 09)
61. String        Index;   // регистрационный номер документа (реквизит 10)
62. Query         Addressat; // адресат документа(реквизит 14).
63. Vector        SignedBy;  // подпись (реквизит 21). Список объектов класса  Person.
64. Person        PreparedBy;  // лперсона, подготовившая документ
65. Person        Executive;       //  исполнитель(реквизит 26)
66. String        Resolution; // резолюция (реквизит 16)
67. String        Header; //заголовок к тексту (реквизит 17)
68. XMLDescription  Text;    // структурированное описание содержимого документа
69. }
```

Класс, моделирующий годовые отчеты

```
70. class YearReport extends Document{
71.     Integer         Year; //отчетный год
72.     Organization  aboutOrg; // организация, о которой составлен отяет
73.     Organization  toOrg;         // организация, в которую направляется данный отчет
74. }
```

Протокол ученого совета

```
75. class Protocol  extends Document {
76.     Vector          Persons; // список лиц, учавствующих в работе ученого совета
77.     Person          Chief; // председатель
78.     Person          Secretary; // секретарь
79. }
```

Для представления метаданных, описывающих контекст документа (точнее c-context, следуя терминологии принятой в [31]),решено использовать XML представления. Причина – невозможность описывать сложные структуры данных используя СУД. Кроме того, на этом примере будет продемонстрирована возможность объединять различные типы источников информации в одной операции переноса данных.

Пример RDF описания содержимого документа-протокола ученого совета

```
80. <ISIR.EDMS:protocol index=”07.05.2000 /2132”>
81.
82. <item type=”elections”>
83. Протокол выборов академиков и членн-кореспондентов РАН
84. <elected><person>7283</person></elected>
85. <electedto ><occupation>8394</occupation></elected >
86. </item>
87.
88. </ISIR.EDMS:protocol>
```

При передаче данных в цифровую библиотеку этой информации у персоны с id 7283 должна появится связь типа “занимает” со званием 8394 (член-корреспондент РАН по специальности Информационные системы в математике). Конечно, возникает еще ряд необходимых действий. Предыдущая персона должна потерять связь с этой должностью и т. д. В будущем будет предусмотрена возможность определения такой информации и в нашей системе, но сейчас такие действия выполняются триггерами электронной библиотеки.

Для решения этой задачи составлено правило трансформации, что

для каждого элемент item типа elections следует выполнить следующее действие: связать все ресурсы, которые указаны в элементе elected с ресурсом указанным в элементе electedto. Типы ресурсов определяются по типам элементов. Тип связи между ресурсами задается типами элементов. Установлено правило, что для элемента elections создается связь между person и occupation типа “занимает должность”.

При решении этой задачи возможны проблемы в случае отсутствия соответствующих ресурсов в ЦБ. Для решения таких проблем предусмотрено следующее: персона и должность могут описываться в виде RDF описаний импортируемых системой, например

```
89. <elected><person id="x-
    doc.person:453"><FirstName> Алексей
    </FirstName><MiddleName> Борисович</MiddleName><Last
    Name>Жижщенко</LastName></person></elected>
90. <electedto ><occupation id="x-
    doc:odccupations:43"><name> Член-
    корреспондент</name><organization>3849</organization>
    </occupation></elected >
```

системе дан порядок ввода элементов. Сначала вводятся все элементы типа персона, затем все элементы типа должность, а затем все результаты выборов. Выборка элементов нужного типа происходит в результате применения XPath запросов над документом. В результате чего во время ввода связей между ресурсами, все ресурсы введены в систему (в случае их корректного определения).

АРМ ученого секретаря использует СУД с объектным интерфейсом и интерфейсом на основе SQL. Модуль 2 (см. выше) для создания объектного представления документов имел доступ к данным посредством SQL интерфейса. В настройке модуля есть возможность для отдельных классов, атрибутов подключать другие модули и указывать, как должны соединяться данные. Для атрибута XMLDescription указывается, что необходимо извлечь строку.

То есть, трансформация объекта класса YearReport описывается объектом ClassMapExt transfYearReport. В списке объектов класс listAttributeMapExt этого класса имеется объект класса AttrMap.

```
91. AttrMap mapXMLDescription
92.     Source  = XMLDescription
93.     Target  = {Person, Occupation, Relation}
94.     Id =
95.     Related = {{'/descendant::election', 'takenFrom'}}
```

Первая версия продукта извлекает данные из баз данных, поддерживающих OLE DB интерфейсом, а также из Document Management Alliance совместимых источников. Модули экспорта данных разработаны для РСУБД (OLE DB, JDBC), XML, RDF описаний.

Для разработки используются расширения UML для описания реляционных схем [32,33], которые позволяют моделировать реляционные схемы данных.

Проводится тестирование MS Repository как средства хранения моделей. В качестве средства редактирования UML диаграмм источника и приемника информации, правил трансформации данных предлагается MS Visual Modeler и Rational Rose.

Список литературы

1 С. В. Агошков, А. Н. Бездушный, М. П. Галочкин, М. В. Кулагин, А. М. Меденников, В. А. Серебряков "Интегрированная Система Информационных Ресурсов (ИСИР) РАН – подход к созданию интегрированных электронных библиотек".

2 W. B. Croft "What do people want from informational Retrival?", D-Lib Magazine, November 1995 (http://www.dlib.org/dlib/november95/11croft.html)

3 Ch. Nikolaon, M. Marazakis, "System Infrastructure for Digital Libraries: A Survey and Outlook"

4 G Alonso., D. Agrawal G, A. El. Abbadi, , C. Mohan, "Functionality and Limitations of Current Workflow Management Systems." IEEE Expert, 1(9), 1997. Special Issue on Cooperative Informational Systems

5 L. A. Kalinichenko, "Integration of Heterogeneous Semistructured Data Models in the Canonical one", First Russian National Conference on Digital Libraries: Advanced Methods and Technologies, Digital Collections

6 H. Chen, "Semantic Research for Digital Libraries", D-Lib Magazine, Oct. 1999, (http://www.dlib.org/dlib/october99/chen/10chen.html)

7 М. Р. Когаловский, "Научные коллекции информационных ресурсов в электронных библиотеках". Труды Всероссийской научной конференции Электронные библиотеки: перспективные методы и технологии. Электронные коллекции - 1999

8 М. А. Осипов, О. Л. Мачульский, Л. А. Калиниченко, "Отображение модели данных XML в объектную модель языка СИНТЕЗ", Первая Всероссийская научная конференция ЭЛЕКТРОННЫЕ БИБЛИОТЕКИ: ПЕРСПЕКТИВНЫЕ МЕТОДЫ И ТЕХНОЛОГИИ, ЭЛЕКТРОННЫЕ КОЛЛЕКЦИИ

9 L. Dempsey, R. Heery, others, "Specification for resource description methods. Part 1. A review of metadata: a survey of current resource description formats". (http://www.ucoln.ac.uk)

10 Amit Sheth , "Changing Focus on Interoperability in Information Systems: From System, Syntax, Structure to Semantics", (http://lsdis.cs.uga.edu/~amit)

11 Y. Arens, C. Chee, C. Hsu, C. Knoblock. "Retrieving and Integrating Data from Multiple Informational Sources.", International Journal of Intelligent and Cooperative Information Systems, 2(2), June 1993

12 V. Kashyap, A. Sheth. "Semantic-Based Information Brokering". In Proceeding of the Third International Conference on Information and Knowledge Management (CIKM), November 1994

13 G. Wiederhold, "Interoperation, Mediation and Ontologies". In FGCS Workshop on Heterogeneous Cooperative Knowldege-Bases, December 1994

14 S. Heiler, "Semantic interoperability", Computing Survey, 27(2):271-273, June 1995

15 Metadata Coalition Open Informational Model (http://www.mdcinfo.com)

16 OMG CWM (Object Management Group Common Warehouse Model)

17 M. Staudt, A. Vaduva, T. Vetterli, “The Role of Metadata for Data Warehousing”, Institur fur Informatic der Universitat Zurich, TR-1999, (http://www.ifi.unizh.ch/techreports/TR_1999.html)

18 H. H. Do, E. Rahm “On Metadata Interoperability in Data Warehouses”, (http://dol.uni-leipzig.de/pub/1999-22/en)

19 W3 Document Object Model (http://www.w3.org/DOM/)

20 G. F. Simons, “Using architectural forms to map TEI data into an object-oriented database”, Tenth Anniversary User Conference on Text Encoding Initiative, (http://www.stg.brown.edu/conferences/tei10/tei10.papers/Simonspaper.html)

21 G. Simons, ”Importing SGML data into CELLAR by means of architectural forms” (http://www.sil.org/cellar/import)

22 D. Washa, The Revised XML Object Model for Internet Explorer 5.0, (http://msdn.microsoft.com/msdn-online/voices/news/xml_objm.asp)

23 К. Дж. Дейт, “Введение в системы баз данных“, Киев, Москва, Диалектика, 1998

24 M. Andersson. “Extracting an entity relationship schema from a relational database trough reverce engineering”. In. 13th Int. Conf. On ER Approach, 1994

25 A. Behm, A. Geppert, K. R. Dittrich “On the migration of relational schemas and data to object-oriented database systems,” In Proc. Of the 5th International Conference on Re-Technologies in Information Systems, Klagenfurt, Austria, Dec. 1997

26 S. W. Ambler “Mapping objects to relational databases”, (http://www-4.ibm.com/software/developer/library/mapping-to-rdb/)

27 M. L. Fussell, “Foundations of Object Relational Mapping”, (http://www.chimu.com/publications/objectRelational/)

28 Wolfgang Keller, “Object/Relational Access Layers, Object/Relational Access Layers”, Proceedings of the 3rd European Conference on Pattern Languages of Programming and Computing, 1998

29 J. Biskup, R. Menzel, T. Polle. “Transforming an entity-relationship schema into object-oriented database schemas”, Technical Report 17, Hildersheimer Informatic—Berichte, June 1994.

30 A. Paepske, C.-C. K. Chang, H. Garcia-Molina, T. Winograd, “Interoperability for Digital Libraries: Problems and Directions”, CACM 41(4),1998

31 V. Kashyap, A. Sheth, “Semantic Heterogeneity in Global Information System: The Role of Metadata, Context and Ontologies, Cooperative Information systems: Current Rends and Directions”

32 Rational. White paper.”The UML and Data Modeling”. (http://www.rational.com/products/rs/prodinfo/whitepapers/dynamic.jtmpl?doc_key=101516)

33 G. Booch, M. Christerson, M. Fuchs, J. Koistinen UML for XML Schema Mapping Specification, (http://www.rational.com/uml/resources/documentation/media/uml_xmlschema33.pdf)