Особенности построения цифровых библиотек со связанным контентом

© А.Г. Марчук, П.А. Марчук

Институт систем информатики им. А.П. Ершова СО РАН, г. Новосибирск
mag@iis.nsk.su, peter@iis.nsk.su

Аннотация

В работе обсуждается подход к построению цифровых архивов или библиотек, существенной особенностью которых является предоставление доступа не только к записям базы данных, но и к электронным образам хранимых документов. Файлы, дополнительные к базе данных, будем называть контентом библиотеки. Работа с контентом включает хранение, транспортировку и обработку файлов, публикацию внешнего образа контента, изменение содержимого файлов и др. В статье обсуждаются также структура хранения данных, использование хранилищ, интеграция данных, расположенных в разных хранилищах, миграция данных по мере изменения технологий и стандартов. Работа выполнена при поддержке гранта РГНФ 10-03-12116в и программы РАН р 2/12.

1 Постановка задачи

Во многих случаях можно сказать, что цифровая библиотека представляет собой множество метаинформационных записей (карточек) о предметах (например, публикациях, экспонатах), попавших в фонд данной библиотеки. Но сами предметы также могут являться частью библиотеки в виде их электронных образов (оцифрованные фотоснимки и видеозаписи, файлы документов и т. д.). Поскольку на такие образы (файлы) есть ссылки в базе данных, то можно говорить, что файлы связаны с цифровой библиотекой. Такие файлы являются содержимым или, часто говорят, контентом библиотеки.

Стандартным решением проблемы хранения файлов документов коллекции и доступа к их содержимому являются прямая публикация документов в интернет-пространстве и использование URL публикации для предоставление доступа к ней. Если URL отражает структуру хранения данных на сервере, то он же может использоваться и для доступа к файлу со стороны серверного приложения. Такой способ публикации и предоставления доступа обладает рядом недостатков. Во-первых, фиксируется адрес (URL) публикуемого файла. Это не так страшно для библиотечной системы, поскольку при ее переносе на другой сервер можно изменить также и адреса связанных документов. Но если речь идет о других информационных системах, сопряженных с данной, или о поисковых системах интернета, то при перемещении или реструктуризации публикационной зоны возникают труднопреодолимые проблемы. Например, в других информационных системах могут появиться ссылки на документы нашей электронной библиотеки. Эти ссылки не всегда можно изменять, а долго поддерживать переадресацию затруднительно.

Еще одна проблема этого способа публикации контента связана с тем, что может возникнуть потребность в предоставлении рабочего доступа не к исходному, оригинальному файлу, а к некоторому переработанному варианту, более подходящему для конкретной ситуации его использования. Например, оригинал фотодокумента, хранящегося в библиотеке, может быть слишком большим для оперативного использования или он может быть защищен правом интеллектуальной собственности, в этом случае следует предоставлять доступ к уменьшенным вариантам фотографии.

В последнее время все более популярными становятся такие способы работы с цифровой библиотекой, когда база данных и контент формируются силами самих пользователей. Соответственно, возникает проблема перемещения данных от пользователя к серверу, на котором они хранятся. Протокол HTTP становится малопригодным для этих целей по мере роста числа передаваемых файлов и увеличения их объема. Интеграция же, в рамках одного веб-интерфейса, частей, работающих по разным протоколам, например, HTTP и FTP, – это довольно сложная задача.

При увеличении числа информационных единиц в электронной библиотеке усложняется доступ к каждой из единиц. В серверном приложении приходится переходить на использование базы данных для быстрого нахождения запрашиваемого документа. Это, во-первых, излишне централизует хранилище, во-вторых, «привязывает» информационную систему к конкретной СУБД. Судьба многих малых и не очень малых информационных проектов часто плачевна: авторы закончили работу над проектом и перестали его поддерживать, а ценные дан-

Труды 12[й] Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» – RCDL'2010, Казань, Россия, 2010

ные «застревали» в недрах какой-то базы данных и потом уничтожались при очередной реконфигурации серверного хозяйства. Кроме того, при использовании централизованной базы данных трудно предоставлять доступ к контенту, расположенному в хранилищах, внешних по отношению к ИС.

Популярные ныне системы управления контентом (CMS) и системы управления документами (DMS) [2, 6], не полностью решают указанные проблемы и, кроме того, вовлекают в решение реляционные СУБД, что для малых электронных библиотек имеет свои недостатки. Также активно развивается направление хранения документов в специализированных или универсальных сервисных порталах [1, 5, 7, 8]. Использование такого способа имеет ограничения в виде необходимости быть подключенным к интернету, работы с файлами очень большого размера и сложность в увязывании интернет-хранилища с базой данных библиотечной информационной системы и неопубликованной частью контента. Однако перспективность подобных интернет-сервисов не вызывает сомнений.

2 Механизм кассет в качестве малого хранилища данных

Предлагается в качестве хранилища данных использовать специальную файловую структуру – кассеты, т. е. группу файлов организовать таким образом, чтобы их структура была удобной для обеспечения базовых возможностей по включению/исключению файлов, предоставлению доступа к оригиналам и специальным копиям, прямой и косвенной публикации в интернете. Такой формат файловой сборки был разработан и ныне используется в разработках ИСИ СО РАН. Структура кассеты является простой системой файлов и директорий, начинающейся от одного корня, ее можно переносить или делать резервные копии с помощью обычных системных средств. Кассета содержит RDF-базу данных, структурирующую ее содержимое и предоставляющую эффективный доступ к отдельным элементам кассеты. Еще одно важное свойство кассетного механизма – возможность эволюции структуры хранилища по мере изменений в технологии.

Реализуется односторонняя связь кассеты с базой данных цифровой библиотеки, т. е. база данных может ссылаться на объекты кассеты, используя URI помещенных туда файлов. Рассмотрим некоторые особенности предлагаемого решения.

Предположим, что в базе данных описан некоторый документ. Это может быть как текстовый, так и фото-видео-аудио документ или файл иной природы. База данных позволяет увязать документ с другими сущностями путем установления интуитивно понятных отношений типа «авторство», «отражение» и др. Кроме этого документ, как правило, является также носителем информации, представленной в виде байтов, конкретной информации, внешней относительно базы данных. Типичным оформлением этой информации является файл, поток байтов которого и есть содержимое документа, например, для фотодокумента имеется файл графического формата (TIFF, JPEG), представляющий собственно фотографию.

Особое внимание обратим на документы, контент которых вовлекается в оперативную работу информационной системы, например, фотографии, имеющиеся в базе данных. Для каждого такого документа, прописанного в базе данных, надо иметь структуру и набор полей, а также связи с другими документами, т. е. «маленькую» базу данных, обеспечивающую доступ к нему и к дополнительной информации. Такая структура может оказаться излишне громоздкой, если ее реализовывать на уровне основной базы данных. В качестве решения предлагается информацию группировать, порождать описание для групп элементов и включить ее в кассету, о которой идет речь в данном разделе. Соответственно, доступ к элементам кассеты будет регламентирован такой локальной базой данных.

Файловые сборки давно используются для разных целей, обычно сочетаясь со сжатием информации. Новизной предлагаемого подхода является использование сборок, т. е. кассет для объединенного хранения базы данных и набора документов электронной библиотеки и обеспечения их функционирования. В качестве модели кассеты можно рассматривать подобие DVD-диска, который обладает такими свойствами, как перемещаемость, наличие внутренней структуры, множественность хранимых файлов, наличие дополнительной информации.

3 Общая структура кассет

Вернемся к связи записи документа с файлом документа. Предлагается использовать для этого URI (Universal Resource Identifier). Можно сказать, что информация, синтаксически оформленная в едином URI, представляет собой четверку {протокол, имя кассеты, код документа, класс документа}. Протокол указывает, какой вариант структуризации используется, а также способ интерпретации имени кассеты и кода документа. Имя кассеты идентифицирует кассету, а код документа идентифицирует документ. Класс документа определяет категорию документа, что нужно для многих целей, в частности, для корректной интерпретации содержимого файла документа.

В качестве модельного примера рассмотрим простой кассетный протокол simple. В этом протоколе файлы документов упорядочиваем в дерево директорий (папок) в соответствии с обычным строением файловой системы. Тогда именем кассеты будет имя папки, а кодом документа – относительный путь (path) к файлу от папки, корневой для данной кассеты.

Поскольку имя кассеты рассматривается в общем пространстве интернета, нам понадобится еще пространство имен, в котором определено имя кассеты. В итоге общая структура URI для идентифи-

кации файла документа строится в соответствии с рекомендациями WWW-консорциума:
protocol://cassette-name@domain/document-code. extension.

Здесь в качестве пространства имен используется домен, зарегистрированный пользователем.

В случае простого кассетного протокола URI документа может выглядеть следующим образом:
simple://demo-cassette@iis.nsk.su/dir1/sub-dir11/ photo1.jpg.

Вышеуказанная интерпретация простого протокола предполагает следующий способ нахождения координаты файла:
координатаКассеты(demo-cassette)/dir1/sub-dir11/ photo1.jpg.

Заметим, что координата одной и той же кассеты вычисляется динамически и может быть разной для разных агентов, поскольку пути от разных агентов к файлу могут отличаться.

Простой кассетный протокол в ряде случаев может быть полезен, но его структура слишком примитивна и не содержит мета- и предвычисленной информации. Рассмотрим более общую сложную структуру кассеты, в настоящее время применяемую в проектах ИСИ СОРАН.

Сохраним простую структуру хранения файлов в виде директорий, но добавим две дополнительные – meta и preview. В директорию meta будем помещать метаинформацию, соответствующую хранимому массиву файлов, а в preview – дополнительную информацию, предназначенную, как правило, для быстрой визуализации при просмотре. Мы постулируем, что раздел preview может быть вычислен по протоколу и массиву файлов. В разделе meta собирается информация, являющаяся локальной частью базы данных, предназначенной для загрузки в базу данных при формировании модели и предоставлению к документным файлам доступа.

Структура информационных блоков, помещаемых в meta и preview, зависит от системы соглашений по каждому конкретному протоколу. Тем не менее, следует сохранять общие свойства такой информации. Рассмотрим один из возможных подходов к размещению мета- и дополнительной информации в кассете.

В раздел meta желательно поместить следующую информацию: имя протокола, имя кассеты, комплектация кассеты, имя корневой коллекции, идентификатор корневой коллекции. Эта информация является интерфейсной и не предназначена для прямого включения в базу данных модели. Другой частью meta-раздела является RDF-документ базы данных, которая обычно включает описатели документов кассеты и формирует иерархию вложения документов в коллекции, соответствующие иерархии директорий хранения документов. Третья часть meta-раздела определяет контекст документов. Вообще говоря, контекст можно связывать с директориями, по которым распределены документы. Документ помещается в директорию, исходя из логики пользователя, определяющей группирование документов по определенным критериям. Если эта логика предполагает наследование контекста от директорий к поддиректориям, то это наследование можно было бы напрямую использовать при формировании базы данных. Однако структуру рабочих (не meta и не preview) директорий желательно не искажать дополнительной информацией, хотя и не исключено, что следует допустить обе возможности.

Раздел preview также может быть устроен специфическим образом, но общая конструкция этого раздела – хранилище файлов, производных от документных оригиналов, и базы данных для того, чтобы ускорить порождение специальных (просмотровых или транспортируемых) вариантов некоторых видов документов, хранящихся в кассете.

Установление связи между зарегистрированным документом кассеты и просмотровыми вариантами возможно либо через (простую) вспомогательную базу данных, либо через систему соглашений, позволяющих вычислять координаты просмотрового варианта, например, по URI исходного документа.

Последний вариант рассмотрим на примере. Пусть мы имеем дело с фотодокументами и нам требуется предварительно вычислить и поместить в preview «маленькие» копии этих фотографий. Можно поступить достаточно просто: по URI документа вычислить идентификатор с помощью некоторой взаимно однозначной функции и в соответствии с ним помещать различные по размерам имиджи в раздел preview. Имена таких вспомогательных файлов вычисляются, например, по формуле:

ФункцияПреобразования(URI-документа) + “_” + спецификатор + “.jpg”, где спецификатор – какой-нибудь код, соответствующий размеру имиджа (“s”, “m” и т. д.).

Соответственно, когда потребуется найти координату вспомогательной копии, то по URI документа можно вычислить имя копии, и файл будет доступен при фиксации базы для preview кассеты.

Использование базы данных для установления соответствия между документом и дополнительной информацией обеспечивает большую гибкость. Во-первых, база данных сама может являться хранилищем дополнительной информации. Во-вторых, с помощью базы данных можно регулировать наличие или отсутствие такой информации, значит, необходимость ее вычисления в динамике доступа. В этом случае мы можем организовать дополнительную информацию по принципу кэша и экономить на вычислениях, выполняя их по необходимости (on demand). В-третьих, база данных может содержать сведения о текущем варианте дополнительной информации к документу. Например, если в качестве предварительно вычисляемой информации для (больших) фотодокументов используются их уменьшенные копии, то размеры этих копий могут быть разными, а требуемый в конкретный момент времени размер может быть динамически вычислен не по большому оригиналу, а по близкой по размеру копии. Использование базы данных для хранения дополнительной информации полезно, например, в

случае, когда часть оригиналов изменяется и требуется где-то фиксировать эту ситуацию и синхронизировать дополнительную информацию с оригиналом.

4 Доступ к данным и документам

Важную роль в фактографических системах играет способ доступа к данным, т. е. к базе данных и контенту документов. Обычное решение этой задачи заключается в том, что базовое прикладное программное обеспечение, например, веб-приложение, имеет внутренние механизмы такого доступа и осуществляет его по мере формирования отдельных визуальных образов и обработки данных, поступивших при «нажатии» кнопок и по мере заполнения форм. Организация такого доступа в виде одного или нескольких сервисов, выполняющих запросы как от простых пользовательских интерфейсов, так и от сложно устроенных машинных агентов, выступающих в качестве клиентов к таким сервисам, представляется более интересной.

О таком интерфейсе часто говорят как об источнике данных. Но источник данных можно «нагрузить» дополнительными функциями поиска информации и выполнения специальных выборок. Источник данных может стать и «приемником» данных в случае реализации функций редактирования или других функций по изменению содержимого цифровой библиотеки.

Интерфейс к данным и документам, точнее, к содержимому документов может быть реализован в разных технологиях, в данном случае мы будем предполагать, что используется протокол однократного запроса в стиле объектно-ориентированного программирования. Таким примером и основой для практической реализации является протокол HTTP. Подход фирмы Google, реализующей доступы к своим сервисам [8] средствами API [9], основу которого составляет довольно простое использование HTTP, наиболее близок к рассматриваемому здесь. Но в этом и сила подхода, поскольку вне зависимости от платформы, на которой реализована его информационная система, пользователь может простыми средствами использовать предоставляемые сервисы.

Мы также рассматриваем HTTP как базовый протокол взаимодействия с данными, но в дальнейшем планируем расширить номенклатуру используемых протоколов взаимодействия клиента с сервисом данных. Предполагается, что это будут хорошо зарекомендовавшие себя протоколы, реализующиеся «поверх» HTTP, такие, как SOAP, WCF и другие. Запрос представляет собой набор параметров, например, идентификатор объекта, операция и др. Кроме того, HTTP позволяет в режиме POST передавать произвольные последовательности байтов. Ответ от сервера также может иметь параметры, а результирующий объект может быть бинарным, текстовым или форматированным (текстовым). Соответственно, все «правильные» решения, типа Content-type для HTTP здесь уместны.

Рассмотрим различные виды запросов к источнику данных. Первая группа – запрос контента файла документа. В эту группу входит запрос оригинала и запрос специальных копий (preview). Основным параметром запроса является идентификатор или URI документа. Дополнительным параметром является квалификатор вида копии, например, квалификатор размера изображения. В этой группе есть свои особенности. В частности, в ряде случаев URL запроса должен выглядеть как URL прямо опубликованного файла, что связано с особенностями кэширования в прокси-серверах и особенностями использования данного контента, например, потокового видео. Такая постановка требует специального кодирования запроса и его специальной обработки интерфейсом сервиса. Для этой группы существенно также использование компрессии и декомпрессии данных. Например, сжатие данных при запросе документов формата XML может многократно уменьшить объем передаваемой информации.

Вторая группа запросов к источнику данных – это запросы к базе данных. В базовом варианте это следующие запросы: получение единичной записи по ее идентификатору, получение множества «обратных» записей, т. е. таких, которые имеют ссылку на запись с задаваемым идентификатором, получение множества записей, соответствующих некоторому поисковому запросу. Эксперименты показали, что запросы, группирующие более простые, и запросы на специальное оформление получаемого результата также являются существенными.

Третья группа запросов – это запросы на изменение базы данных и клиент-серверная синхронизация по данным. Логика этих запросов диктуется общей логикой синхронизации моделей в распределенной фактографической системе [3, 4]. В эту группу входят: внесение новых записей в базу данных или модификация имеющихся, операторы уничтожения записей и замены идентификаторов, запросы на обновление клиентской модели в соответствии с изменениями серверной.

К четвертой группе относятся запросы на добавление и изменение документов или набора документов данной информационной системы. Внесение новых документов порождает создание архивной копии и, возможно, некоторых специальных (preview) вариантов данного контента, а также приводит к созданию необходимых записей в базе данных, регистрирующих документ и связывающих документ с другими элементами базы данных.

5 Использование кассет в информационных системах

Кассеты могут быть использованы в самых разных случаях построений информационных систем. Их основная функция – группировать файлы хранимых документов и предоставлять к ним доступ. Поскольку в наших проектах все существенные ре-

сурсы отображаются на файлы, кассета может быть единственным носителем всего конкретного проекта. В кассете легко разместить и RDF-базу данных, и OWL-спецификации, различные конфигураторы, XSLT, CSS, JS файлы для оформления визуального интерфейса пользователя и обеспечения его функциональности. Такой подход проверялся на проектах малого и среднего размера, когда количество документов не превышало нескольких десятков тысяч, а количество фактов в базе данных – одного миллиона.

Оформление данных и документов в кассеты позволяет гибко управлять содержимым различных информационных систем, использующих общее информационное поле. Это означает, что под конкретную информационную систему собирается набор кассет, содержащих нужные данные, эти кассеты загружаются в оперативную зону системы и обеспечивают данными ее функционирование. Например, информационная система может использовать несколько общих кассет, содержащих публичную информацию о людях, организациях, событиях, и приватные кассеты с непубличными данными пользователей.

6 Заключение

Предлагаемый подход был реализован и использован в нескольких исследовательских и прикладных проектах, которые проводятся в ИСИ СО РАН. В частности, это Фотоархив Сибирского отделения, Электронная энциклопедия ММФ НГУ, База данных летних школ юных программистов, информационная система кафедры программирования ММФ НГУ.

Литература

[1] Бесплатное защищенное паролем интернет-хранилище. – http://office.live.com.

[2] Коржов В. Использование сетевой модели данных для управления информационным наполнением// Computerworld Россия. – 2000. – № 21.

[3] Марчук А.Г. Распределенные электронные архивы, библиотеки и базы данных // Препринт 122, Институт систем информатики им. А.П. Ершова СО РАН, Новосибирск, 2004. – 25 с.

[4] Марчук А.Г., Марчук П.А. Платформа интеграции электронных архивов // Электронные библиотеки: перспективные методы и технологии, электронные коллекции. Тр. девятой Всерос. конф., Переславль, 2007. – С. 89-94.

[5] Парамонов В. Google открывает «облачное» хранилище данных. – http://www.citforum.ru/news/23833/, 2010.

[6] Пьянзин К. Универсальные системы управления документами // Lan/Журнал сетевых решений. – Ноябрь 1998. – Т. 4, № 11.

[7] Bradley T. Google unveils new cloud data storage for developers. – http://www.pcworld.com/ businesscenter/article/196539/google_unveils_new_cloud_data_storage_for_developers.html, 2010.

[8] Create and share your online documents, presentations, and spreadsheets. – http://docs.google.com.

[9] Google Data Protocol. – http://code.google.com/intl/ru/apis/gdata/docs/developers-guide.html.

Special features of the construction of digital libraries with document content

A.G. Marchuk, P.A. Marchuk

In the work is discussed the approach to the construction of digital archives or libraries, essential feature of which is the granting of access not only to the records of the database, but also to the digital content of documents. Files, additional to the data base, we will call content of library. Work with content includes storage, transportation, publication in the Internet, a change of files and others. In the article the following questions are discussed: the structure of storage of data, the use of depositories, the integration of data, located in the different depositories, the migration of data in proportion to a change in the technologies and standards.