Анализ технологий импорта данных для систем отчетности коммерческих организаций

И.С. Берендаков (ФГУП ИТМ и ВТ им. С.А. Лебедева РАН)

Рассматривается несколько современных технологий импорта данных для систем отчетности. Проводится анализ этих решений с целью выявления их преимуществ и недостатков. Особое внимание уделяется механизму Web-служб и его использованию при построении подсистемы импорта данных. Это обусловлено тем, что применение Web-служб в качестве альтернативного подхода для загрузки информации из внешних источников еще не получило широкого распространения, и сейчас данное решение незаслуженно находится в тени других часто используемых технологий импорта данных. Подробно рассматриваются преимущества, которые можно извлечь из Web-служб, если интегрировать их с разнородными источниками информации внешних коммерческих организаций.

Импорт данных из внешних информационных источников – это важный и ресурсоемкий этап формирования отчетности. Как правило, процесс импорта данных является самым продолжительным по времени, сильно нагружает сервер отчетности и увеличивает трафик в канале связи (если данные передаются по сети). Поэтому вопрос первостепенного значения заключается в том, как сделать импорт менее затратным и более быстрым.

Системы-источники данных могут содержать десятки, сотни или даже тысячи гигабайт информации, передача которых может длиться несколько часов. Поэтому обычно полный импорт данных проводится только один раз, сразу после создания системы отчетности. Далее загружаются только те данные, которые были изменены или добавлены с момента последнего импорта. Этот тип импорта называют инкрементальным, когда происходит постепенная передача новых данных небольшими порциями. Каждая порция, например, может содержать информацию только за последний день или неделю. Исключение могут составлять лишь небольшие таблицы-справочники, на заполнение которых не требуется много ресурсов и которые можно перезаписывать полностью при каждом импорте.

Самая большая проблема при сборе информации из нескольких источников заключается в том, что они могут быть разнородными. Организации могут использовать различные СУБД (от Access до Oracle), поэтому необходимо подобрать единый формат представления данных при их передаче. На настоящий момент самым распространенным форматом для интеграции с другими системами является XML (http://ru.wikipedia.org/wiki/XML). Этот язык позволяет описывать данные любой сложности. Многие современные корпоративные системы (например, JIRA) предоставляют инструменты администрирования для выгрузки своих данных в XML-файл. В последствии этот файл может быть импортирован в систему отчетности. Но данный подход имеет ряд недостатков. Во-первых, на момент загрузки информация XML-файла может устареть и несколько отличаться от реальных данных, так как этот тип импорта не является интерактивным. Конечно, его можно проводить и ночью, когда в информационные системы не вносятся новые данные, а сами серверы не нагружены. Но часто могут возникнуть ситуации, когда требуется выполнить загрузку в течение рабочего дня. Во-вторых, как правило, не предоставляется возможность импортировать только последние изменения (инкрементальный импорт), а значит, в систему отчетности каждый раз будут загружаться все данные целиком. В-третьих, сформированный XML-дамп необходимо выкладывать на доступный из вне Web-ресурс (например, FTP), чтобы предоставить возможность переноса файла с данными на сервер отчетности. Но операцию генерации XML-файла и его размещения в Web скорее всего будет вручную выполнять администратор информационной системы, а он может по ошибке сохранить файл не в ту директорию или под другим именем. В результате этого импорт данных из внешней системы не будет проведен должным образом.

Выходом из сложившейся ситуации может стать создание ссылки (например, Linked Server в MS SQL) на сервер БД информационного источника, и использование этой ссылки в SQL запросах к внешним данным. Оформив это решение в виде хранимых процедур на сервере системы отчетности, вызываемых по определенному заданию (job), можно получить неплохое решение для импорта данных. Также будет нетрудно настроить инкрементальный импорт, добавив к SQL запросам условие выборки, в соответствии с которым будут возвращаться только данные, измененные или добавленные с момента последнего импорта. Но данный подход требует знания структуры БД внешнего источника. Зачастую эта информация является конфиденциальной и не сообщается разработчикам системы отчетности. Обойти это можно путем создания промежуточной БД на сервере внешней организации. Эта БД, например, могла бы содержать только представления (views), которые позволяли бы получать лишь необходимые для отчетности данные и скрывали бы их реальную структуру.

Соответственно все взаимодействие сервера отчетности с внешними источниками ограничилось бы запросами к этим промежуточным БД, структуры которых скрывать нет необходимости.

Описанное решение является довольно распространенным, но сам такой подход отличается своей низкоуровневой направленностью. Интерфейс взаимодействия системы отчетности и информационного источника реализован здесь набором таблиц или представлений на удаленном сервере внешней организации. Все команды получения данных будут запрограммированы с явным указанием этих таблиц (или представлений). Конечно это не самое удобное средство интеграции систем, хотя бы с точки зрения возможностей его описания. Далеко не всегда разработчики системы отчетности могут понять, какие вообще таблицы существуют и для хранения чего они предназначены. Документы, описывающие состав и предназначение таблиц, могут либо отсутствовать, либо быть получены не сразу, что приведет к задержке выполнения работ. Другой, более серьезной причиной, по которой этот подход не всегда применим, является то, что удаленный доступ непосредственно к серверу БД внешнего источника может быть запрещен. Это продиктовано желанием организаций обеспечить большую безопасность своих информационных ресурсов и ограничить число точек доступа к своим системам.

Также следует учесть, что источников данных может быть множество, и в них могут использоваться СУБД разных типов. Поэтому на сервере системы отчетности должны быть установлены специальные программные средства – провайдеры данных (data providers), позволяющие оперировать с данными, содержащимися во внешних хранилищах различного формата. Для каждого типа удаленной СУБД должен использоваться подходящий провайдер. В некоторых случаях в качестве источников данных применяются, например, файлы Excel или даже текстовые файлы. Как правило, это имеет место в небольших коммерческих организациях, которые еще не перешли на использование СУБД для хранения своей информации. Но подобная ситуация встречается все реже и реже, и здесь подход с использованием присоединенного сервера полностью неприменим.

Решение описанных проблем можно найти, если обратиться к современным информационным технологиям. Преобладающая часть этих инновационных достижений непосредственно связана с развитием сети Internet. Сейчас все большую роль начинает играть механизм Web-служб (Web Services) – это приложения, которые расширяют концепцию распределенной обработки, предоставляя API (Application Programming Interfaces), методы которых могут быть вызваны через Internet. Эти приложения могут быть написаны на любом развитом языке программирования, а связь с ними осуществляется с помощью открытых протоколов, не зависящих от платформы[1]. Следовательно, просто необходимо рассмотреть возможность применения Web-служб в качестве альтернативного способа организации импорта данных из внешних источников. И действительно, ничто не мешает использовать удаленные методы, предоставляемые Web-службами, для организации программного интерфейса взаимодействия с БД коммерческих организаций.

Для начала рассмотрим структуру современных информационных систем, применяемых для управления корпоративными бизнес-процессами. В большинстве случаев организации, предоставляющие свои данные для построения отчетности, имеют специализированное ПО, состоящее из приложения и БД. Приложение предоставляет графический интерфейс пользователю-оператору системы для ввода, редактирования и просмотра данных, а БД служит для хранения всей введенной информации. Обычно применяют Web-приложения, так как для их использования оператору потребуется только браузер. А это значит, что нет необходимости устанавливать дополнительное ПО на ПК всех пользователей системы. Существует только одно приложение, которое работает под управлением Web-сервера (сервер приложений). Следовательно, это предоставляет возможность централизованной доработки, модификации или настройки системы и избавляет администраторов от необходимости распространения и установки каждой новой версии программы на ПК пользователей.

Использование Web-приложения в качестве интерфейса взаимодействия с информационной системой существенно упрощает процесс интеграции этой системы с Web-службой. Это объясняется тем, что Web-служба также является приложением глобальной сети Internet, поэтому не имеет смысла создавать ее изолированно от Web-приложения. Обычно функционал Web-службы гармонично встраивается в систему путем добавления соответствующего плагина – дополнительного программного модуля, расширяющего возможности системы (например, добавление jar-архива для Web-приложения, написанного на языке Java). Плагин предоставляет возможность гибкого включения или исключения функционала Web-службы из системы. Единственное ограничение этого подхода заключается в том, что Web-служба и само приложение должны быть написаны на одном и том же языке программирования. В случае использования изолированной Web-службы возможно применение двух различных языков высокого уровня, но в такой ситуации необходимо серьезно подумать, стоит ли вообще так делать. Ведь если, например, Web-приложение системы разработано с помощью технологий Java, а Web-служба написана на C#.NET, то на сервере приложений должны быть установлены сразу две различные платформы: виртуальная машина Java и .NET Framework. Это приведет к возникновению целого ряда серьезных проблем, таких как организация совместной работы двух программных платформ и Web-сервера, нехватка аппаратных ресурсов ПК и т. д. Поэтому выбирать подобное решение следует лишь в случае

[1] *Либерти Д.* Программирование на C# /Пер. с англ. СПб: Символ-Плюс. 2003.

крайней необходимости, когда нет другого выхода.

Иногда информационные системы внешних коммерческих организаций уже содержат Web-службы, которые позволяют программно получить информацию прямо из хранилищ данных этих систем. Нужно лишь вызвать удаленные методы, которые предоставляют имеющиеся Web-службы, и получить запрашиваемые данные. Но и здесь есть серьезная проблема. Она заключается в том, что корпоративная информационная система редко когда содержит готовую и продуманную Web-службу, особенно если система является самодельной, то есть разработана силами самого предприятия с привлечением собственных ресурсов. Если же система является коммерческим продуктом организации, специализирующейся на разработке ПО, то наличие Web-службы возможно.

Анализ существующих Web-служб некоторых информационных систем показал, что в большинстве случаев требуется серьезная доработка их методов. Дело в том, что организации, занимающиеся созданием корпоративных систем и порталов, разрабатывают Web-службы для общих нужд потребителей. Поэтому при создании системы отчетности часто возникают проблемы, связанные с невозможностью получения некоторых данных из внешних источников. Существующих методов Web-служб оказывается просто недостаточно для удовлетворения информационных потребностей конкретной системы отчетности. Это приводит к созданию новых или модификации старых функций, входящих в состав API используемых Web-служб. Отсюда можно сделать вывод о том, что применение механизма Web-служб для организации импорта данных требует индивидуального подхода, то есть при разработке системы отчетности необходимо создавать новые или изменять уже имеющиеся службы в соответствии с тем набором данных, который требуется для формирования содержимого самих отчетов.

За использование Web-служб говорит тот факт, что данный механизм имеет развитую инфраструктуру. Например, компания Microsoft предоставляет набор утилит, таких как Discovery и wsdl, которые существенно облегчают работу с Web-службами. Discovery позволяет приложениям обнаруживать и запрашивать описания Web-служб, что является предварительным этапом перед обращением к этим службам. Именно на этом этапе будущие пользователи Web-службы узнают о ее существовании, возможностях и о способах взаимодействия с ней. Прежде чем создавать клиентское приложение, взаимодействующее с Web-службой, необходимо создать класс-посредник. Инструментальное средство под названием wsdl создает исходный текст этого класса на основании информации из WSDL-файла. WSDL является XML-схемой, применяемой для описания методов, предоставляемых Web-службой (то есть ее интерфейса). Чтобы просмотреть WSDL-документ, следует добавить фразу "?WSDL" к URL-адресу Web-службы.

Очевидным преимуществом использования Web-служб для организации импорта данных является возможность применения всех средств высокоуровневых языков программирования, на которых написаны эти службы. Такие языки, как Java или C#.NET обладают обширными библиотеками классов. Следовательно, при разработке подсистемы импорта данных на основе Web-служб в руках у программистов оказываются мощные инструменты, предоставляемые этими библиотеками. Каким бы ни был внешний информационный источник – БД Oracle, БД MS SQL, файл Excel или XML-дамп – встроенные классы и средства развитого языка программирования дают возможность получения необходимых данных для системы отчетности. Таким образом, подход с применением механизма Web-служб для организации импорта данных полностью скрывает структуру хранилища внешнего источника, тип используемой СУБД (или формат файлов данных) и предоставляет разработчику все возможности библиотеки классов высокоуровневого языка программирования.

Итак, механизм Web-служб является хорошей альтернативой при выборе подхода к реализации импорта данных для системы отчетности. Выделим преимущества и недостатки данного решения.

Преимуществами этого решения являются: использование средств высокоуровневых языков программирования при построении подсистемы импорта данных; использование протоколов, не зависящих от платформы; развитая инфраструктура Web-служб; удобный и понятный программный интерфейс взаимодействия с источником данных; полное скрытие структуры источника данных; возможность взаимодействия с различными типами внешних хранилищ информации. Но, используя данный подход, необходимо учесть, что потребуется индивидуальная разработка Web-служб.

Берендаков Илья Сергеевич *– ведущий разработчик ФГУП ИТМ и ВТ им. С.А. Лебедева РАН.*
Контактный телефон: 8-910-486-72-25. E-mail: ber83@mail.ru