УДК 004.4: 004.9: 528.9: 912.43

ОБЛАЧНЫЙ СЕРВИС ХРАНЕНИЯ ДАННЫХ

Суханов Владимир Иванович
д.т.н., доцент

Чагаева Ольга Леонидовна
преподаватель

Уральский федеральный университет, Екатеринбург, Россия

Описано перспективное федеративное хранилище данных в облаке

Ключевые слова: ОБЛАКО, БАЗА ДАННЫХ, ВЕБ-ПРИЛОЖЕНИЕ, СЕРВЕР

UDC 004.4: 004.9: 528.9: 912.43

CLOUD STORAGE DATA SERVICES

Sukhanov Vladimir Ivanovich
Dr.Sci.Tech., associate professor

Chagaeva Olga Leonidovna
lecturer

Ural Federal University, Yekaterinburg, Russia

The article presents a perspective cloud federation data storage

Keywords: CLOUD, DATA BASE, WEB-APPLICATION, SERVER

Введение

В настоящее время одним из ключевых направлений развития информационных технологий являются облачные вычисления. Эксперты Gartner считают, что стремление пользователей делиться своими данными и иметь к ним доступ с разных цифровых устройств приведет к тому, что к 2016 году треть всех данных будет храниться в «облаках».

В 2011 году, по подсчетам аналитиков, только 7% данных хранилось на "облачных" сервисах, а к 2016 году эта цифра должна вырасти до 36%. По прогнозам Gartner, если в 2011 пользователи хранили примерно 329 экзабайт данных, то к 2016 году эта цифра увеличится до 4,1 зеттабайт. Основную долю указанных объёмов, как правило, представляют мультимедийные файлы и аналитические копии существующих серверов поисковых машин [1].

На рынке представлено несколько типов облачных сервисов хранения пользовательских данных:

1) Сервисы, в которых пользователю предоставляется экземпляр СУБД на виртуальной машине.

2) Сервисы, в которых пользователю предоставляется СУБД без привязки к виртуальным машинам и управление виртуальными машинами, масштабируемость и отказоустойчивость хранения данных обеспечивается поставщиком сервиса.
3) Сервисы, в которых пользователю предоставляется возможность хранения данных в виде таблиц, написания простых приложений для обработки этих данных на некотором специализированном языке.
4) Сервисы хранения файлов.

В контексте создания облачного сервиса хранения данных пользователей SaaS-приложений интерес представляет вторая группа, как наиболее полно реализующая преимущества от применения облачных технологий и максимально удобная для пользователя, что связано с разработкой специального ПО для сборки разнородных источников данных по запросу клиентов. Это позволяет обобществить произвольные данные за счет программной интеграции и предотвратить неоправданную избыточность данных в сети на основе сервис-ориентированной архитектуры и соответствующих протоколов. Интеграция данных поддерживает операции с композицией и временным (виртуальным) представлением данных, хранящихся у различных владельцев. Данные остаются под контролем владельца и извлекаются по требованию клиентов для интегрированного доступа, возможно на коммерческой или свободной основе.

В тоже время клиент – потребитель интегрированной информации за счет единого для всех потребностей интерфейса избавлен от необходимости извлекать и согласовывать данные от разных источников в разном формате, что существенно экономит время и ресурсы на разработку своего оригинального ПО. При этом он может и не знать в какой системе, где и как хранятся интересующие его данные: реляционная

СУБД, файловые ресурсы поставщиков, результат работы поисковых сервисов, фиксированные провайдеры справочных данных и др. Всю работу по поиску и преобразованию данных можно автоматизировать в рамках единой федеративной системы хранения и доступа. Решение о наилучшем способе создания этого унифицированного представления часто выбирается разработчиками локальных систем с учетом доступности инструментария, опыта, квалификации и уровня информационной культуры организации. При использовании традиционных архитектур время, усилия и затраты, связанные с интеграцией, могут свести на нет ее преимущество для бизнеса. При реализации в сервис-ориентированной среде подход, основанный на использовании информационных сервисов на основе объединения в федерацию общих данных, может существенно улучшить характеристики повторного использования и снизить его стоимость.

Концепции облачного федеративного хранилища

Общепризнанным стандартом представления данных на транспортном уровне является XML. Если на транспортном уровне использовать XML-представление, то описание структуры хранимых данных становится синонимом файла определения синтаксиса XML-файла DTD (http://ru.wikipedia.org/wiki/DTD) или XML Schema (http://ru.wikipedia.org/wiki/XML_Schema). XML-схемы вытеснили DTD в силу их больших возможностей и универсальности. В этом случае схемы представления информации в XML файле становится некоторым аналогом таблицы в реляционной СУБД или класса объектов в объектно-ориентированных базах данных.

Таким образом, на основе подхода с использованием промежуточного представления передаваемых данных в формате документов на языке XML можно построить *федеративное хранилище*

данных, которые делегируются в него отдельными держателями этих данных как на свободной, так и на коммерческой основе. В роли языка манипулирования данных в этом случае будет выступать язык XQuery, позволяющий выполнять запросы к наборам данных, представленных как документами XML, так и традиционными таблицами реляционных СУБД. В отличие от языка SQL язык XQuery позволяет создавать запросы к гетерогенным источникам данных, что делает его хорошим инструментом для работы с распределенными хранилищами, предоставляемыми различными провайдерами, входящими в федерацию.

У такого хранилища будет единый интерфейс для манипулирования федеративными данными, в том числе и для дистанционного пополнения с последующим модерированием новых данных держателями физических копий для обеспечения их целостности и корректности. Для разработки опытного образца программного обеспечения в качестве полигона можно использовать публикуемые в сети данные о банках, страховых компаниях, классификаторы адресов и другие источники коллективных данных. Как правило такие данные изменяются редко и выборки из них могут кешироваться на локальных узлах клиентов для повторного использования. Обновление кеша выполняется администратором по запросу или регулярно по расписанию.

Если ориентироваться на создание универсальной функционально полной системы хранения структурированных данных с использованием виртуальных схем на основе общепринятой идеологии использования XML на транспортном уровне, то для такого представления нужно разработать язык и систему интерпретации для создания произвольной схемы данных и

её использования при передаче данных между абонентами корпоративного или федерального облака. Функциональная полнота системы предполагает наличие средств добавления и модификации документов авторизованными удаленными клиентами хранилища. Здесь возникают проблемы с вычисляемыми атрибутами документов, для которых в общем случае не существует алгоритма правильной декомпозиции на составляющие их значения полей первичных записей баз данных, расположенных, возможно, в разных узлах сети.

Нетривиальной представляется задача поддержки транзакций и уровней изоляции в такой распределенной системе хранения данных для формирования XML-документов при их добавлении и модификации.

Проблема администрирования хранилищ предполагает исследование стандартных задач развертывания, миграции, разграничения полномочий, резервного копирования и восстановления, балансировки загрузки, поддержания живучести на достаточном для клиентов уровне готовности системы за счет репликации копий данных, хорошо известных в облачных технологиях. Их проработка существенно зависит от арендуемой инфраструктуры облака и функциональных возможностей предоставляемых провайдерами виртуальных машин и их программного обеспечения в рамках IaaS. Для транспортировки данных между клиентами и серверами федеративного хранилища можно использовать протоколы передачи на основе SOAP и REST.

На рисунке 1 приведена укрупненная схема, иллюстрирующая прохождение запроса в федеративном хранилище.

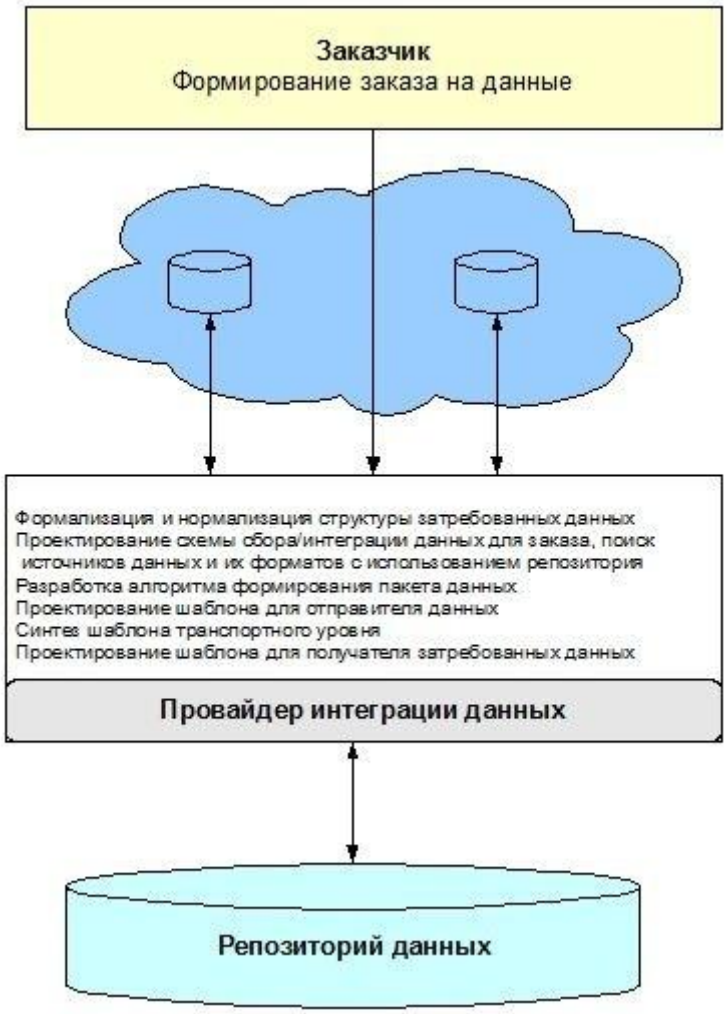

Рисунок 1 - Укрупнённая схема функционирования хранилища

Формирование заказа на передачу данных получателем предполагает работу пользователя с редактором, позволяющим сформулировать требования к запрашиваемым данным со стороны пользователя в виде шаблонов, содержащих перечень данных и их формат. Требования формируются на основе метаданных о всей номенклатуре данных облачных сервисов, хранимых в репозитории системы.

Пользователю должны быть предложены шаблоны запросов из репозитория, ранее созданные другим пользователями, с возможностью их адаптации под конкретные требования.

Формализация и нормализация структуры затребованных данных предполагает построение формально-логической схемы запрошенных

данных, контроль их доступности, форматов хранения и форматов представления в результирующих наборах.

Проектирование схемы сбора/интеграции данных для заказа, поиск источников данных и их форматов с использованием репозитория предполагает формирование гетерогенных запросов к владельцам данных, получение и перекодирование данных в соответствии с требованием результирующей схемы.

Разработка алгоритма формирования пакета данных предполагает автоматизированное создание программной реализации исполнения спроектированных ранее запросов, получение и формирование данных в соответствии с построенными схемами, формирование выходных наборов данных для передачи пользователю.

Проектирование шаблона для отправителя данных предполагает создание XML Schema для отправки данных по сети заказчику.

Синтез шаблона транспортного уровня предполагает запаковку данных с использованием существующих доступных протоколов передачи данных.

Проектирование шаблона для получателя затребованных данных предполагает распаковку данных протокола транспортного уровня и передачи их заказчику.

Репозиторий системы является базой данных, в которой хранятся сведения о всей доступной информации в системе. По каждой единице хранения (ресурсу) в репозитории должна быть информация о смысле (семантике), представленном метаданными, локализации источника, типе хранилища, структуре таблицы или документа, типе, формате хранения.

Наиболее общими возможностями располагают средства языков программирования, например, Java для обработки XML-документов [2]. Java-разработчику для создания XML-приложений доступны не менее шести расширений Java-платформы:

а) Java API для обработки XML (JAXP - Java API for XML Processing);

б) Java API для связи XML с Java (JAXB - Java API for XML/Java Binding):

в) Долгосрочная персистенция JavaBean-компонентов;

г) Java API для обмена сообщениями XML (JAXM - (JAXM - Java API for XML Messaging);

д) Java API для удаленного вызова процедур XML (JAX RPC - Java API for XML RPC);

е) Java API для реестра XML (JAXR - Java API for XML Registry).

Базовые инструменты для работы с документами JAXP предоставляет разработчику Java и технологии для обработки XML документов, которые зависят от JAXP. Среди них:

а) SAX – простой API для XML;

б) DOM – программный интерфейс API объектной модели документа (Document Object Model) от консорциума W3C;

в) XSLT – язык преобразований каскадной таблицы стилей XML от консорциума W3C;

г) XPath – язык XML Path от консорциума W3C;

д) JDOM – API оптимизированной объектной модели документа от jdom.org.

Веб-сервисы есть приложения сервера и приложения клиента, которые связываются через протокол World Wide Web (WWW) HyperText Transfer Protocol (HTTP). На концептуальном уровне, сервис является программным компонентом, доступным через конечную точку сети (endpoint). Потребитель и поставщик сервиса используют сообщения, чтобы обмениваться вызовами запросов и получения ответной информацией в форме самодостаточных документов, которые

предполагают лишь некоторые допущения о технологических возможностях получателя [3].

На техническом уровне, веб-сервисы могут быть реализованы разными способами: **JAX-WS** веб-сервисы "Big" и **JAX-RS** веб-сервисы "RESTful".

JAX-WS технология Java EE обеспечивает функциональность для сервисов, использующие сообщения Extensible Markup Language (XML), которые используются стандартом Simple Object Access Protocol (SOAP), определяющим архитектуру и форматы сообщений на языке XML. В таких системах часто машиночитаемое описание сервисных операций предлагается записывать на языке Web Services Description Language (WSDL) – варианте языка XML для синтаксического определения интерфейсов.

JAX-RS веб-сервисы на базе REST (REpresentational State Transfer - "RESTful") являются коллекциями сервисов, идентифицируемых своими URI. Каждый документ или процесс моделируется веб-ресурсом со своим уникальным URI. Эти веб-ресурсы управляются действиями, которые могут быть специфицированы в заголовке HTTP. Не используются ни стандарт SOAP, ни WSDL, ни стандарты WS. Вместо них обмен сообщениями может производиться в разных форматах: XML, JSON, HTML и др. Часто клиентом является веб-браузер. Технология в силу возможностей используемых транспортных протоколов не обеспечивает защиту транспортируемой информации от несанкционированного доступа, что является важной особенностью RESTful. RESTfull хорошо подходит для основных и оперативных (на лету) сценариев интеграции. Веб-

сервисы RESTful часто лучше интегрированы в HTTP, чем SOAP-ориентированные сервисы.

Средства JAX-RS в Java EE обеспечивают функциональное назначение для сервиса RESTful. Они не требуют XML сообщений или определений WSDL сервиса. Проект JAX-RS - промышленная готовая реализация для JSR 311: JAX-RS: Java API для RESTful Web Services. JAX-RS осуществляет поддержку аннотаций, определенных в JSR-311, облегчая разработчикам формирование веб-сервисов RESTful для Java и Java Virtual Machine (JVM).

Для работы с XML документами консорциумом W3C разработан язык запросов XQuery [4], [5], специально ориентированный на извлечение данных из сложных структур документов.

На рынке информационных технологий существует широкий набор свободных и коммерческих процессоров для обработки XQuery: Zorba (http://sourceforge.net/projects/zorba/), Xqilla (http: //xqilla. sourceforge.net/ HomePage) и другие. Полный список продуктов этой серии можно посмотреть на http://www.w3.org/XML/Query/#implementations. Примером свободного ПО для обработки запросов является Xqilla – процессор XQuery написанный на языке C++ и вызываемый в интерпретаторе командной строки: xqilla [options] <XQuery file>...

Для выполнения требований разработки федеративного хранилища облачный сервис хранения данных пользователей SaaS-приложений должен включать балансировщик нагрузки и несколько одинаковых узлов центрального репозитория с сервером обработки запросов пользователей хранилища (рисунок 2).

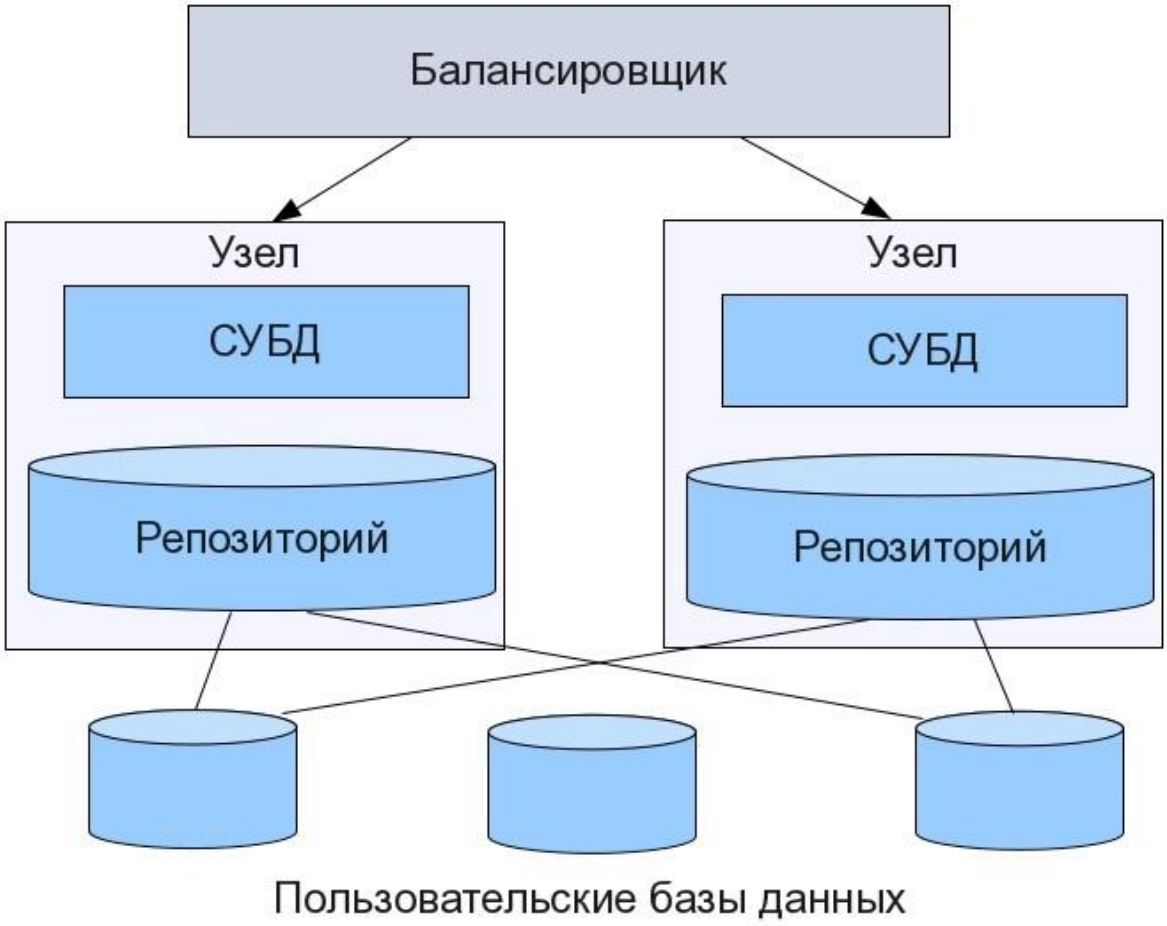

Рисунок 2 - Архитектура сервиса хранения

Репозиторий содержит описание всех доступных данных хранилища, содержащее следующие сведения:

а) внутренний идентификатор записи репозитория;

б) внешнее имя блока данных для идентификации пользователями;

в) метаданные – комментарий содержимого блока данных, полей, формата, правил доступа и др.;

г) тип хранимых данных: SQL, ключ-значение, XML-документ, файл и т. д.;

д) адрес хранения;

е) протокол доступа;

ж) порт доступа;

з) Login/Password/уровень доступа пользователей (чтение, запись);

и) текст запроса по умолчанию и его параметры;

к) XML-схема представления результата запроса;

л) место обработки запроса: локальный узел пользовательской БД или центральный сервер;

м) конвертор результата в XML-документ;

н) владелец ресурса;

о) цена единицы ресурса;

п) дата обновления ресурса;

р) объем ресурса;

с) обработчик запроса GET;

т) обработчик запроса POST;

у) прочие сведения.

Обработка запроса GET.

В запросе пользователя на получение ресурса указываются:

а) идентификатор ресурса в репозитории хранилища;

б) характер запроса: по умолчанию или пользовательский;

в) параметры для запроса по умолчанию;

г) текст запроса пользователя, включающего форму результата.

Результат выполнения запроса по умолчанию всегда в виде XML-документа для обеспечения универсальности (прозрачности) представления. Пользовательские запросы форматируются самостоятельно.

В зависимости от типа ресурса и СУБД обработка запроса может быть передана на локальный сервер СУБД хранения ресурса или выполнена на центральном сервере репозитория для запросов на языке XQuery к XML-документам локальных хранилищ. Для этого ресурс запрашивается на центральный сервер и может быть кеширован, например, в отдельном поле записи о ресурсе для экономии трафика в будущих запросах.

Если содержимое ресурса контролируется только локальной SQL СУБД, то контроль даты обновления затруднен/невозможен. Поэтому кешировать такой ресурс не имеет смысла.

Обработка запроса POST.

В запросе пользователя на изменение ресурса указываются:

а) идентификатор ресурса в репозитории хранилища;

г) текст запроса пользователя на изменение записей, в зависимости от типа СУБД.

После обработки запроса корректируется поле даты изменения и, возможно, копия в кеше.

Заключение

Работы в области создания федеративных хранилищ в мировой практике находятся в начальном состоянии, что позволяет надеяться на получение в этом направлении результатов мирового уровня, получить приоритет в продвижении этой технологии как в научном плане, так и в плане коммерциализации результатов интеллектуальной деятельности. Этому способствует готовность рынка свободных инструментальных средств разработки веб-приложений и веб-сервисов на языках программирования Java, Python, Ruby on Rails, широко используемых в облачных технологиях, и имеющийся широкий опыт использования этих инструментов для разработки веб-технологий для облаков.

Список использованных источников

1 Облачные вычисления (мировой рынок) [Электронный ресурс] // TAdviser. - М., 2012. - Режим доступа: http: //www.tadviser.ru/index.php /Статья: Облачные_вычисления_ %28мировой_рынок%29 (дата обращения: 15.01.2013).

2 Технология Java и XML. Часть 1: Введение в API для обработки XML. [Электронный ресурс]. Режим доступа: http:// javagu.ru/portal/dt?last=false&provider=javaguru&ArticleId=GURU_ARTICLE_81146&SecID=GURU_SECTION_80704 (дата обращения: 08.02.2013).

3 The Java EE 6 Tutorial, Volume II Advanced Topics . [Electronic resource] Sun Microsystems, Inc. 4150 Network Circle Santa Clara, CA 95054 U.S.A. Part No: 820–7628–10 December 2009 (accessed date: 21.12.2012).

4 Д.Чемберлин. XQuery: язык запросов XML. Журнал Открытые системы #01/2003. [Электронный ресурс]. Режим доступа: http://citforum.ru/internet/articles/xqlzxml.shtml (дата обращения: 10.01.2013).

5 С. М. Саракко, Д. Чамберлин, Р. Ахуджа. DB2 и XML: Запрос XML-данных при помощи Xquery. [Электронный ресурс]. Режим доступа: http://www.ibm.com/developerworks/ru/library/db2xml-5/index.html (дата обращения: 30.01.2013).

Работа поддерживается Министерством образования и науки Российской федерации, ГК №14.514.11.4014