David Chappell
& Associates

ЗНАКОМСТВО С WINDOWS AZURE

ДЭВИД ЧЕППЕЛ (DAVID CHAPPELL)

ОКТЯБРЬ 2010 Г.

ПРИ ПОДДЕРЖКЕ КОРПОРАЦИИ MICROSOFT

СОДЕРЖАНИЕ

ОБЗОР WINDOWS AZURE

Облачные вычисления уже стали реальностью. Запуск приложений и хранение данных на компьютерах, расположенных в доступном через Интернет центре обработки данных, может принести немало преимуществ. Однако независимо от того, где выполняются приложения, они создаются на определенных платформах. Например, для локальных приложений, выполняемых в центре обработки данных организации, в понятие платформы обычно входит операционная система, тот или иной способ хранения данных и, возможно, некоторые другие аспекты. Приложениям в облачной среде требуется аналогичная база для работы.

Именно такую базу предоставляет платформа Windows Azure — базу для запуска приложений и хранения данных в облаке. На рис. 1 это наглядно показано.

Рис. 1. Приложения Windows Azure выполняются в центрах обработки данных Microsoft и доступны через Интернет.

Вместо того чтобы предоставлять программное обеспечение, которое заказчики Microsoft могли бы самостоятельно устанавливать и выполнять на своих компьютерах, мы предлагаем Windows Azure в качестве сервиса. Заказчики используют эту платформу для выполнения приложений и хранения данных на компьютерах, принадлежащих корпорации Microsoft, а доступ к этим компьютерам осуществляется через Интернет. Приложениями как сервисом могут пользоваться и предприятия, и индивидуальные потребители. Вот несколько разновидностей приложений, которые можно создать на платформе Windows Azure.

- Независимый поставщик ПО может создать предназначенное для бизнес-пользователей приложение по модели *SaaS («программное обеспечение как сервис»).* Одной из целей создания Windows Azure была поддержка собственных приложений Microsoft, разработанных по модели SaaS, так что независимые поставщики ПО могут также использовать эту платформу в качестве основы для разнообразных облачных решений.

- Независимый поставщик ПО может создать приложение SaaS для индивидуальных потребителей, а не предприятий. Поскольку в Windows Azure заложена возможность поддержки хорошо масштабируемых программных продуктов, эта платформа прекрасно подойдет фирме, планирующей разработку нового приложения для широкого потребительского рынка.

- Предприятия могут с помощью Windows Azure создавать и выполнять приложения, предназначенные для сотрудников. В такой ситуации, скорее всего, не потребуется столь высокая масштабируемость, как в приложениях для индивидуальных потребителей. Однако благодаря своей надежности и управляемости платформа Windows Azure представляет собой привлекательный вариант.

Для поддержки облачных приложений и данных в Windows Azure имеется пять компонентов, как показано на рис. 2.

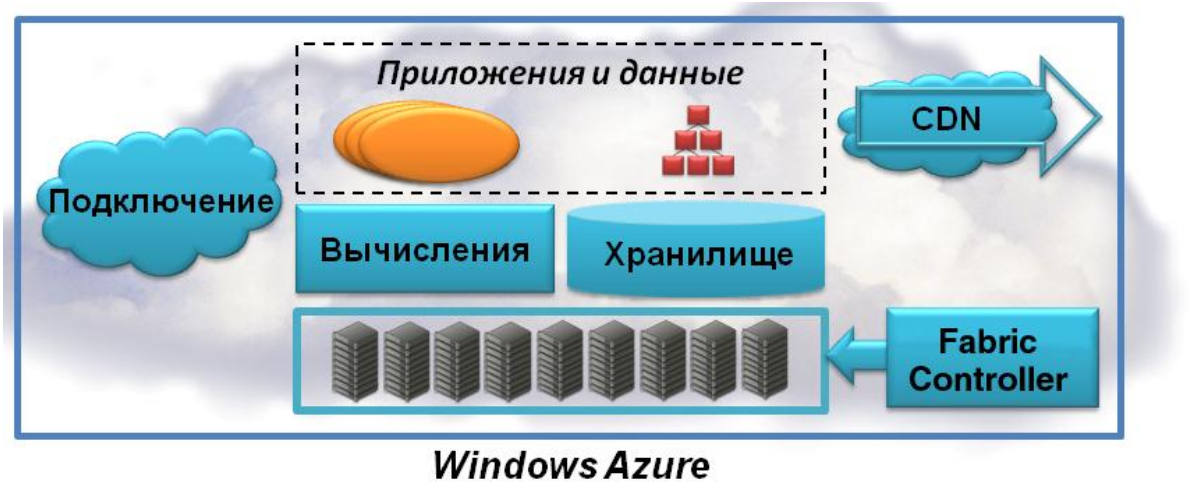

Рис. 2. В Windows Azure имеется пять основных компонентов: вычисления, хранилище, Fabric Controller, сеть доставки контента и подключения.

- Вычисления: выполнение приложений в облаке. В этом случае приложения работают в среде, во многом схожей со средой Windows Server, хотя модель программирования Windows Azure несколько отличается от модели локальных приложений Windows Server.

- Хранилище: хранение двоичных и структурированных данных в облаке.

- Fabric Controller: развертывание и мониторинг приложений, управление ими. Fabric Controller также обновляет системной ПО всей платформы.

- Сеть доставки контента (Content Delivery Network, CDN): повышает скорость глобального доступа к двоичным данным в хранилище Windows Azure за счет ведения кэшированных копий таких данных по всему миру.

- Подключения: позволяют создавать подключения IP-уровня между локальными компьютерами и приложениями Windows Azure.

Далее в этом разделе подробнее описывается каждая из этих технологий.

ВЫЧИСЛЕНИЯ

В среде вычислений Windows Azure могут выполняться приложения многих типов. Но какие бы задачи ни выполняло приложение, оно должно быть реализовано как одна или несколько *ролей*. Windows Azure выполняет несколько *экземпляров* каждой роли, используя встроенную подсистему балансировки нагрузки для равномерного распределения запросов между ними. На рис. 3 показано, как это выглядит.

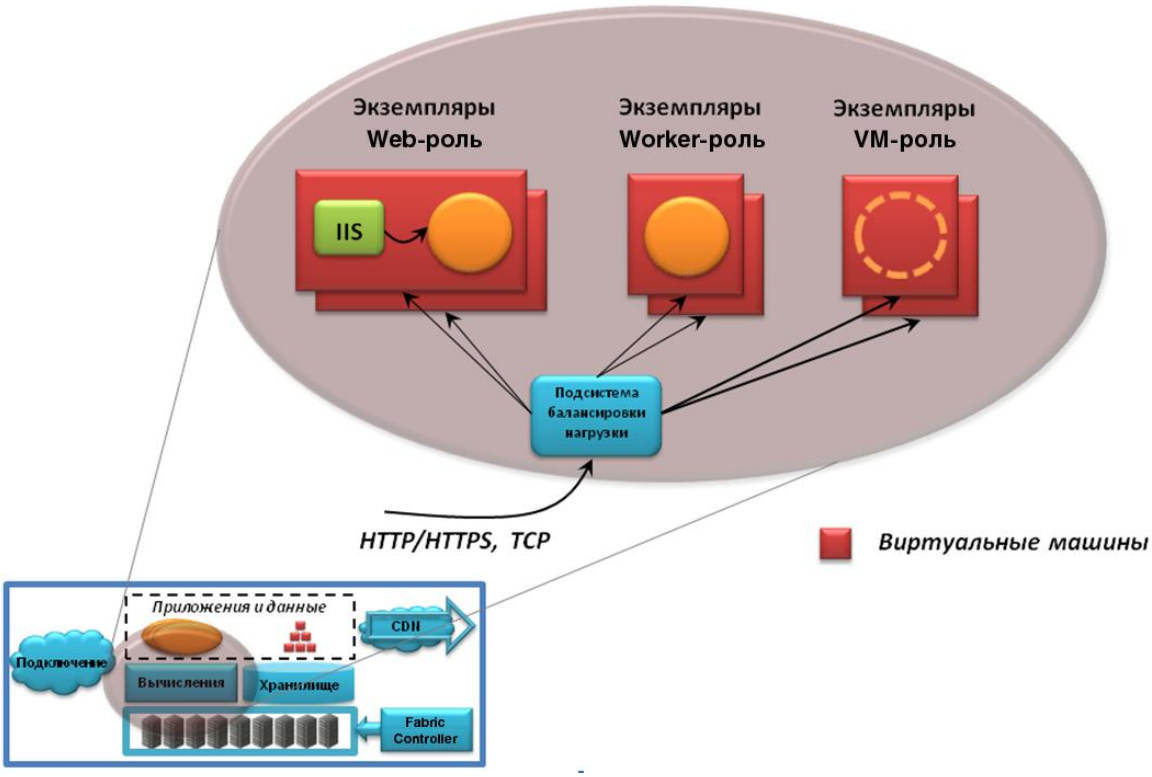

Рис. 3. Приложение, выполняемое в Windows Azure, может состоять из любого сочетания экземпляров Web-ролей, Worker-ролей и VM-ролей.

В текущей версии Windows Azure разработчики могут использовать на выбор роли следующих трех типов.

- Web-роли. Предназначены прежде всего для облегчения создания веб-приложений. В каждом экземпляре Web-роли имеются предварительно настроенные сервисы Internet Information Services (IIS) 7, что упрощает создание приложений с помощью ASP.NET, Windows Communication Foundation (WCF) и других веб-технологий. Разработчики могут также создавать приложения в машинном коде — использование .NET Framework не обязательно. Другими словами, они могут устанавливать и выполнять технологии, производимые не корпорацией Microsoft, в том числе PHP и Java.

- Worker-роли. Предназначены для выполнения всевозможного кода на платформе Windows. Самое большое различие между Web-ролью и Worker-ролью заключается в том, что в последних нет настроенных сервисов IIS, поэтому выполняемый в этих ролях код не размещается в IIS. В Worker-роли может выполняться моделирование, обработка видео или практически любая другая задача. Распространена такая схема: приложение взаимодействует с пользователями через Web-роль, а затем передает задачи для обработки Worker-роли. И в этом случае разработчик также может по своему усмотрению использовать .NET Framework или другое ПО, выполняемое в среде Windows, в том числе технологии, производимые не Microsoft.

- VM-роли. В каждой из них выполняется указанный пользователем образ Windows Server 2008 R2. Помимо прочего, VM-роль иногда может пригодиться для перемещения локального приложения Windows Server на платформу Windows Azure.

Для отправки приложения в Windows Azure разработчик может воспользоваться порталом Windows Azure. Вместе с приложением передается информация о настройках, где платформе указывается, сколько экземпляров каждой роли должно выполняться. Далее Fabric Controller Windows Azure для каждого экземпляра создает виртуальную машину и запускает код для соответствующей роли в этой виртуальной машине.

Как показано на рис. 3, запросы от пользователей приложения могут передаваться по таким протоколам, как HTTP, HTTPS и TCP. Независимо от способа передачи, для запросов выполняется балансировка нагрузки между всеми экземплярами роли. Поскольку подсистема балансировки нагрузки не позволяет привязывать сеансы к конкретному экземпляру роли, то никак нельзя гарантировать, что различные запросы от одного и того же пользователя будут направлены в один и тот же экземпляр роли. Из этого следует, что экземпляры роли в Windows Azure не должны сами сохранять свое состояние в промежутке между запросами. Вместо этого любое состояние, связанное с клиентом, должно записываться в хранилище Windows Azure, сохраняться в SQL Azure (еще один компонент платформы Windows Azure) или каким-то иным способом сохраняться вовне.

Для создания приложения Windows Azure разработчик может использовать любую комбинацию экземпляров Web-ролей, Worker-ролей и VM-ролей. Если нагрузка на приложение увеличится, с помощью портала Windows Azure можно увеличить количество экземпляров каждой роли в приложении. Если нагрузка снизится, количество запущенных экземпляров можно сократить. Windows Azure также предоставляет программный интерфейс для выполнения всех этих операций, так что для изменения числа запущенных экземпляров не обязательно личное вмешательство. Однако платформа не масштабирует приложения автоматически в зависимости от нагрузки на них.

Приложения Windows Azure создаются с помощью таких же языков и средств, как и любые приложения Windows. Разработчик может реализовать Web-роль с помощью ASP.NET и Visual Basic или, например, использовать WCF и C#. Таким же образом разработчик может создать Worker-роль на одном из языков .NET, работать непосредственно в C++ без .NET Framework или использовать Java. Хотя для Windows Azure предоставляются подключаемые модули Visual Studio, использование этой среды разработки не является обязательным. Например, разработчик, установивший PHP, при написании приложения может по своему выбору использовать другое средство.

В целях мониторинга и отладки приложений Windows Azure каждый экземпляр может вызывать программный интерфейс ведения журнала, записывающий сведения в общий журнал приложения. Кроме того, разработчик может настроить систему для ведения счетчиков производительности приложения, замерять загрузку ЦП приложением, сохранять аварийные дампы в случае сбоев и т. п. Эта информация содержится в хранилище Windows Azure, и разработчик может написать код для анализа этих сведений. Например, если в течение одного часа три раза происходит сбой экземпляра Worker-роли, специально написанный код может отправлять администратору приложения письмо по электронной почте.

Возможность выполнения кода является основополагающей, но не достаточной частью облачной платформы. Приложениям также требуется постоянное хранилище. Для выполнения этой задачи предназначен сервис хранилища Windows Azure, рассматриваемый далее.

ХРАНИЛИЩЕ

Приложения работают с данными самыми разными способами. В связи с этим сервис хранилища Windows Azure предлагает несколько вариантов, которые продемонстрированы на рис. 4.

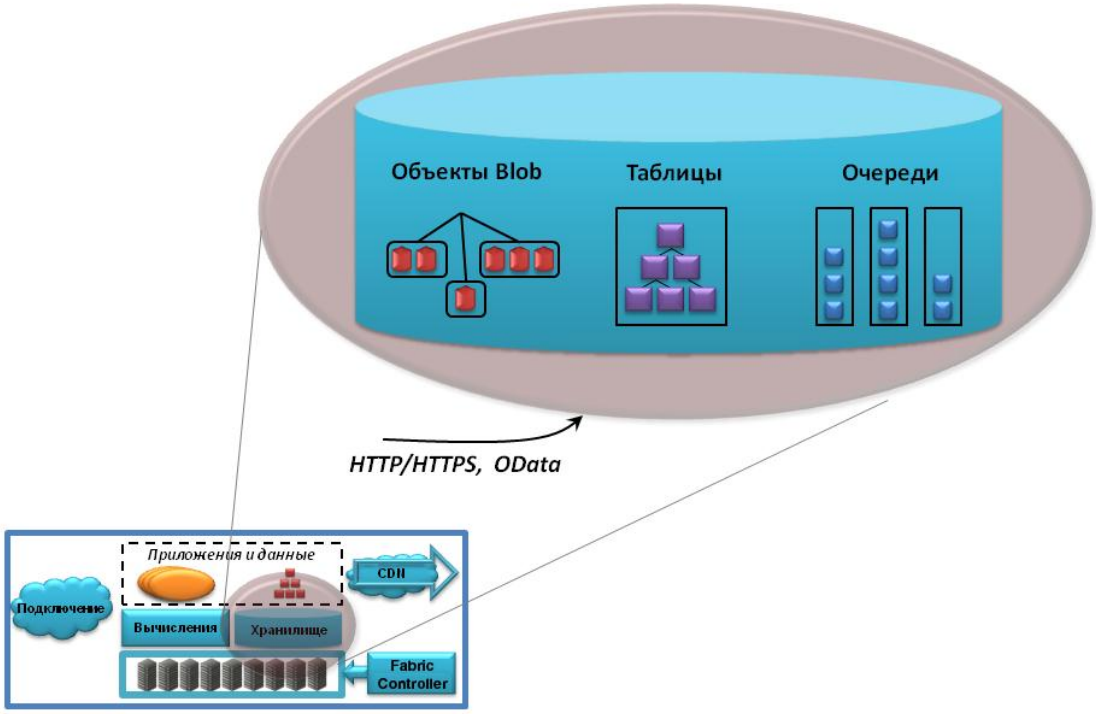

Рис. 4. Хранилище Windows Azure предоставляет объекты BLOB, таблицы и очереди.

Самый простой способ хранения данных в хранилище Windows Azure — это использование больших двоичных объектов (BLOB). В большом двоичном объекте содержатся двоичные данные, и, как показано на рис. 4, иерархия весьма проста. В каждом *контейнере* может содержаться один или несколько больших двоичных объектов. Большие двоичные объекты могут быть очень большими — вплоть до терабайта. С ними могут быть соотнесены метаданные, такие как информация о том, где была сделана фотография в формате JPEG или кто является исполнителем музыки, записанной в MP3-файле. Большие двоичные объекты могут также предоставлять базовое хранилище для *дисков Windows Azure* — механизма, позволяющего экземпляру роли Windows Azure взаимодействовать с постоянным хранилищем, как будто это локальная файловая система NTFS.

Большие двоичные объекты прекрасно подходят для ряда ситуаций, но в других случаях они оказываются слишком неструктурированными. Для того чтобы позволить приложениям более детально работать с данными, хранилище Windows Azure предоставляет таблицы. И пусть их название не вводит вас в заблуждение. Это не реляционные таблицы. Фактически данные, содержащиеся в каждой таблице, хранятся в группе *сущностей*, обладающих *свойствами*. Для извлечения данных из таблицы в приложениях используются не запросы SQL, а соглашения, заданные протоколом OData. Такой подход позволяет горизонтально масштабировать хранилище, при необходимости равномерно распределяя данные по нескольким компьютерам более эффективно, чем в стандартной реляционной базе данных. В действительности в одной таблице Windows Azure могут находиться миллиарды сущностей и терабайты данных.

Большие двоичные объекты и таблицы ориентированы на хранение данных и доступ к ним. Третий способ хранения данных в хранилище Windows Azure, которым являются очереди, имеет совершенно другое предназначение. Основная задача очередей — обеспечивать асинхронный способ взаимодействия экземпляров Web-роли с экземплярами Worker-роли. Например, пользователь может отправить запрос на выполнение какой-то требующей интенсивной обработки задачи через веб-интерфейс, реализованный Web-ролью Windows Azure. Экземпляр Web-роли, который получил этот запрос, может написать в очередь сообщение с описанием работы, которая должна быть выполнена. Экземпляр Worker-роли, который ожидает в этой очереди, может затем прочитать сообщение и выполнить задачу, указанную в нем. Полученные результаты можно вернуть с помощью другой очереди или обработать каким-то другим способом.

Вне зависимости от способа хранения данных — в больших двоичных объектах, таблицах или очередях — все данные, имеющиеся в хранилище Windows Azure, реплицируются трижды. Это повышает отказоустойчивость, поскольку потеря копии не так страшна. При этом система обеспечивает высокую степень согласованности данных, так что приложение, незамедлительно считывающее данные, которые оно только что записало, гарантированно получит именно то, что было записано. Windows Azure также хранит резервную копию всех данных в другом центре обработки в той же части света. Если центр обработки данных, где содержится основная копия, недоступен или уничтожен, то остается доступной резервная копия.

Доступ к хранилищу Windows Azure можно осуществлять с помощью приложения Windows Azure, локального или размещаемого приложения, а также из другой облачной платформы. Во всех этих случаях все три способа хранения Windows Azure используют соглашения REST для выявления и предоставления данных, как показано на рис. 4. Для именования больших двоичных объектов, таблиц и очередей используются универсальные коды ресурсов (URI), а доступ к ним осуществляется с помощью стандартных операций HTTP. Для этой цели клиент .NET может использовать библиотеку, предоставляемую Windows Azure, но это не обязательно — приложение может также использовать простые HTTP-вызовы.

Создание приложений Windows Azure, использующих большие двоичные объекты, таблицы и очереди, определенно может обеспечить важные преимущества. Приложения, полагавшиеся на реляционное хранилище, могут вместо него использовать SQL Azure — еще один компонент платформы Windows Azure. Благодаря этой технологии приложения, выполняемые в Windows Azure (или в других местах), могут осуществлять доступ к облачному хранилищу, пользуясь привычными методами доступа на основе SQL, как при работе с реляционными хранилищами.

FABRIC CONTROLLER

Все приложения Windows Azure и все данные в хранилище Windows Azure располагаются в каком-либо центре обработки данных Microsoft. В его рамках Fabric Controller управляет комплексом из компьютеров, выделенных для Windows Azure, и выполняемого на них программного обеспечения. На рис. 5 это наглядно показано.

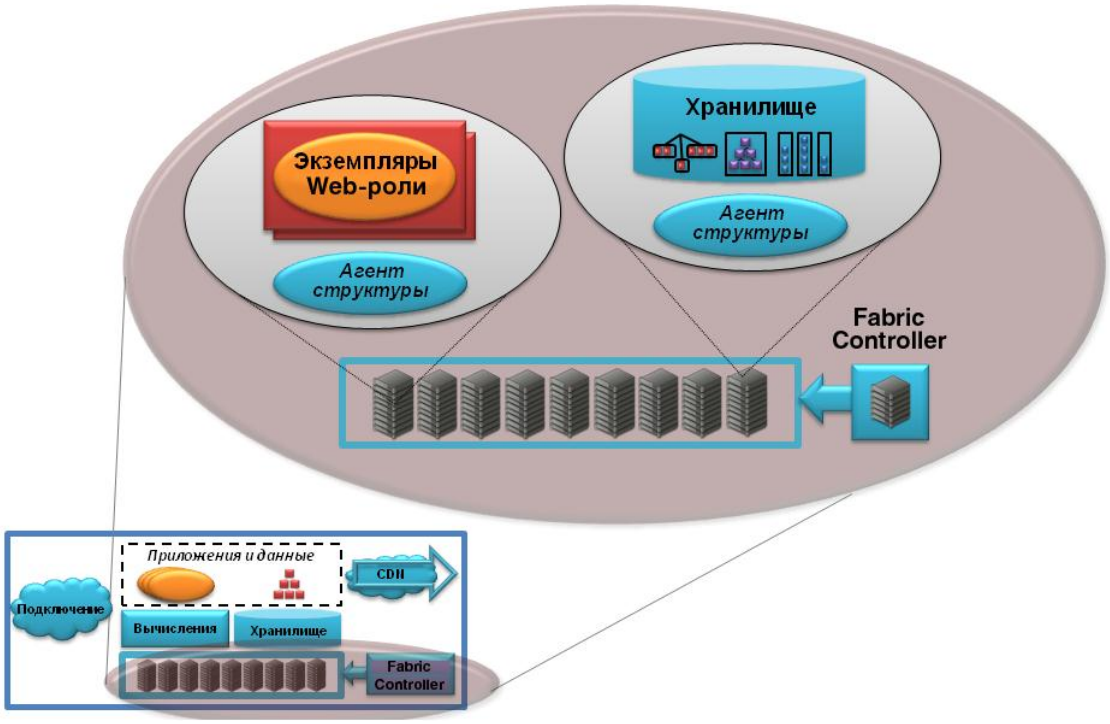

Рис. 5. Fabric Controller взаимодействует с приложениями Windows Azure через агент структуры.

В свою очередь, Fabric Controller является распределенным приложением, реплицированным в группе компьютеров. Ему принадлежат все ресурсы среды: компьютеры, коммутаторы, подсистемы балансировки нагрузки и т. д. Поскольку он взаимодействует с *агентом структуры* на каждом компьютере, ему также известно о каждом приложении Windows Azure в этой структуре. (Интересно, что с точки зрения Fabric Controller хранилище Windows Azure является просто еще одним приложением, так что подробности управления данными и репликации данных ему не видны.)

Широкая информированность позволяет Fabric Controller выполнять ряд важных задач. Он осуществляет мониторинг всех запущенных приложений — например, следит за актуальным общим состоянием всего происходящего. Он также решает, где должны запускаться новые приложения, выбирая для этого физические серверы, чтобы оптимизировать использование оборудования. Для этой цели Fabric Controller использует сведения о конфигурации, загружаемые с каждым приложением Windows Azure. В таком файле в формате XML описывается, что необходимо приложению: сколько экземпляров Web-ролей, сколько экземпляров Worker-ролей и т. п. Когда Fabric Controller развертывает новое приложение, он с помощью такого файла конфигурации определяет, сколько виртуальных машин нужно создать.

После того как виртуальные машины созданы, Fabric Controller осуществляет мониторинг каждой из них. Например, если приложению требуется пять экземпляров Web-ролей и один из них перестанет работать, то Fabric Controller автоматически запустит новый экземпляр. Таким же образом, если перестанет исправно функционировать компьютер, на котором была запущена виртуальная машина, Fabric Controller запустит новый экземпляр нужной роли на другом компьютере, соответствующим образом изменив параметры подсистемы балансировки нагрузки с учетом новой виртуальной машины.

В настоящее время Windows Azure предлагает разработчикам на выбор пять размеров виртуальных машин, а именно:

- минимальный размер — с одноядерным процессором с частотой 1,0 ГГц, ОЗУ 768 МБ и 20 ГБ места в хранилище для экземпляра;
- малый размер — с одноядерным процессором с частотой 1,6 ГГц, ОЗУ 1,75 ГБ и 225 ГБ места в хранилище для экземпляра;
- средний размер — с двухъядерным процессором с частотой 1,6 ГГц, ОЗУ 3,5 ГБ и 490 ГБ места в хранилище для экземпляра;
- большой размер — с четырехъядерным процессором с частотой 1,6 ГГц, ОЗУ 7 ГБ и 1000 ГБ места в хранилище для экземпляра;
- максимальный размер — с восьмиядерным процессором с частотой 1,6 ГГц, ОЗУ 14 ГБ и 2040 ГБ места в хранилище для экземпляра.

Экземпляр минимального размера использует ядро процессора совместно с другими экземплярами минимального размера. А для всех остальных размеров каждому экземпляру предоставляется одно или несколько выделенных ядер процессора. Это означает, что производительность приложения будет прогнозируемой, а продолжительность выполнения экземпляра никак не будет ограничиваться. Например, экземпляр Web-роли может обрабатывать запрос от пользователя столько времени, сколько потребуется, а экземпляр Worker-роли может вычислять значение числа π (пи) с точностью вплоть до миллионов знаков после запятой.

Для Web-ролей и Worker-ролей (но не VM-ролей) Fabric Controller также управляет операционной системой в каждом экземпляре, в том числе применяет пакеты исправлений ОС и обновляет другое системное ПО. Благодаря этому разработчики могут полностью сосредоточиться на написании приложений и не заботиться об управлении платформой. Тем не менее важно понимать, что Fabric Controller всегда предполагает, что выполняется не менее двух экземпляров каждой роли. Поэтому он может остановить один из них для обновления ПО, не прекращая работу приложения в целом. В связи с этим обстоятельством и по ряду других причин обычно не рекомендуется, чтобы выполнялся только один экземпляр какой-либо роли Windows Azure.

СЕТЬ ДОСТАВКИ КОНТЕНТА

Часто большие двоичные объекты используются для хранения информации, доступ к которой может осуществляться из различных мест. Например, представьте приложение, показывающее видеоматериалы в клиентах Flash, Silverlight или HTML 5 по всему миру. Чтобы повысить производительность в подобных ситуациях, в Windows Azure имеется сеть доставки контента. В этой сети копии большого двоичного объекта хранятся как можно ближе к клиентам, которые его используют. На рис. 6 это наглядно показано.

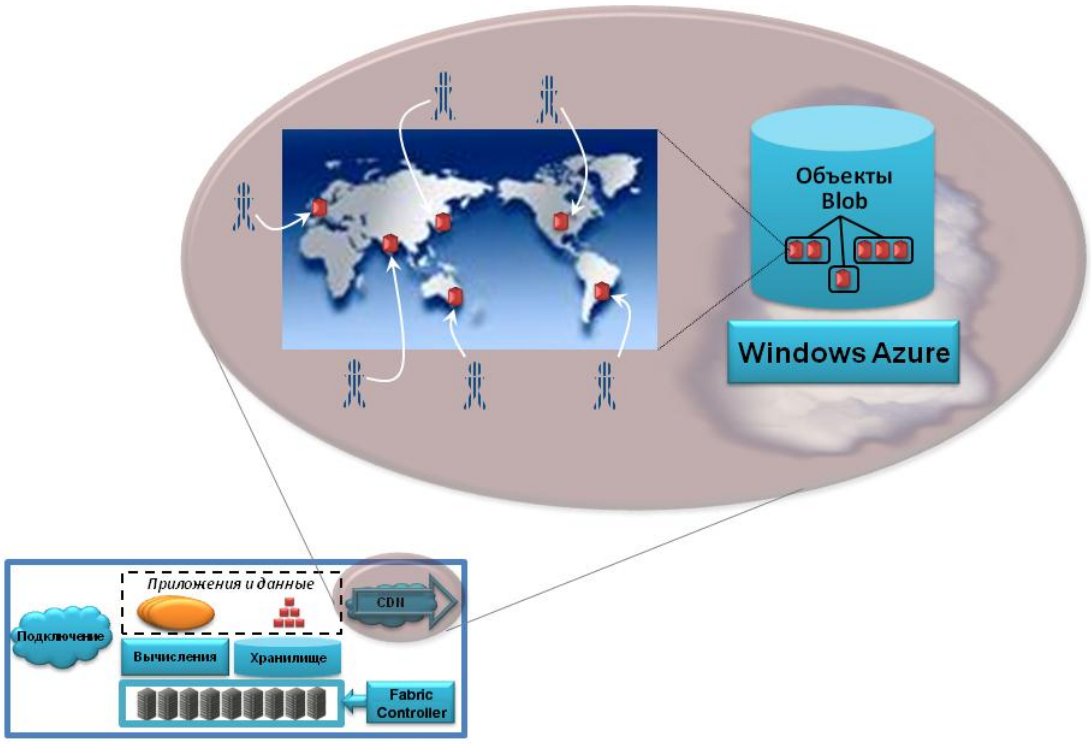

Рис. 6. Сеть доставки контента Windows Azure кэширует копии объектов BLOB по всему миру, благодаря чему пользователи могут быстрее осуществлять доступ к такой информации.

Эту иллюстрацию не следует воспринимать буквально — в сети доставки контента Windows Azure на самом деле намного больше мест кэширования по всему миру, чем показано на рисунке, — но общий принцип здесь продемонстрирован верно. Когда пользователь впервые запрашивает определенный двоичный объект, в сети доставки контента копия этого объекта сохраняется географически как можно ближе к местонахождению пользователя. В следующий раз при доступе к большому двоичному объекту его контент будет доставлен из кэша, а не из далеко находящегося исходного хранилища.

Для примера представим, что с помощью Windows Azure предоставляются видеоматериалы ежедневных спортивных событий для обширной аудитории. Первый пользователь, открывающий определенный видеоматериал, не воспользуется преимуществом сети доставки контента, поскольку большой двоичный объект еще не кэширован ближе к нему. Зато для всех остальных пользователей в том же регионе производительность будет выше, поскольку с использованием кэшированной копии видеоматериал будет загружаться быстрее.

ПОДКЛЮЧЕНИЕ

Выполнение приложений в облаке Microsoft обеспечивает ряд преимуществ. Но поскольку приложения и данные, хранимые внутри организаций, перестанут использоваться еще не скоро, важно, чтобы локальные среды можно было эффективно подключать к Windows Azure.

Для этой цели предназначен Windows Azure Connect. Этот компонент облегчает взаимодействие между приложением Windows Azure и компьютерами, работающими вне облака Microsoft, предоставляя возможность подключения IP-уровня. На рис. 7 это наглядно показано.

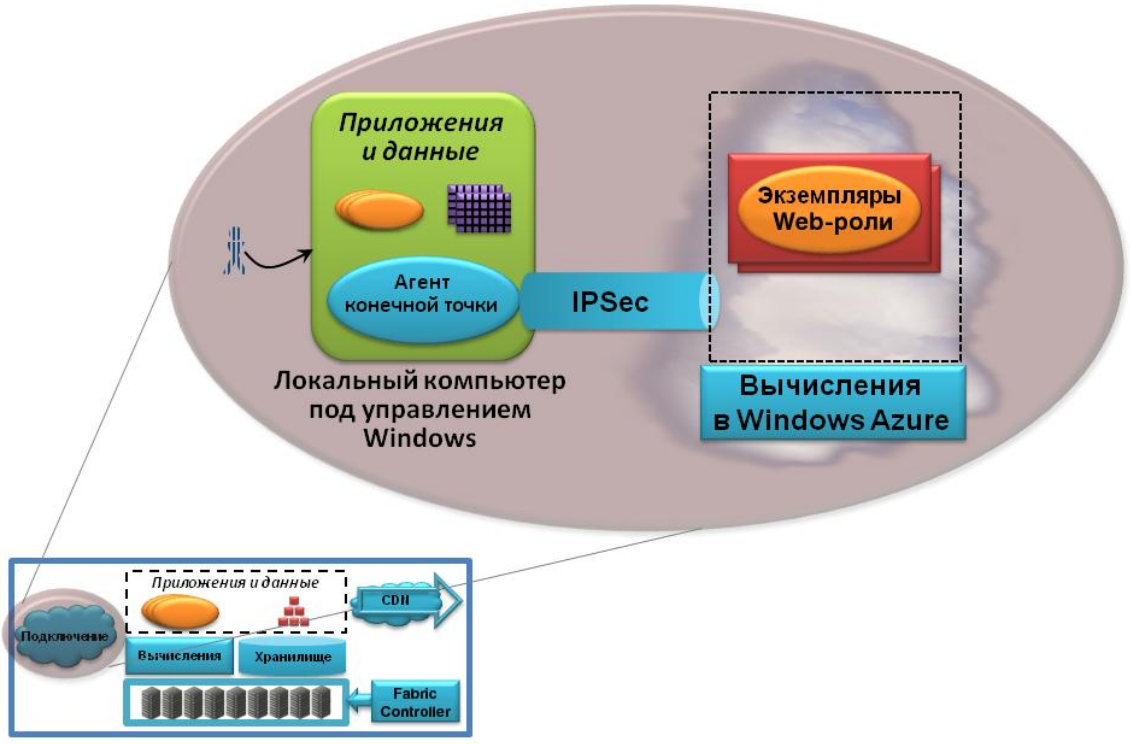

Рис. 7. Windows Azure Connect предоставляет возможность подключения IP-уровня между локальными компьютерами под управлением Windows и приложением Windows Azure.

Как показано на рисунке, для использования Windows Azure Connect требуется установить *агент конечной точки* на каждом локальном компьютере, подключающемся к приложению Windows Azure. (Поскольку эта технология базируется на протоколе IP v6, агент конечной точки в настоящее время доступен только для операционных систем Windows Server 2008, Windows Server 2008 R2, Windows Vista и Windows 7.) Приложение Windows Azure также необходимо настроить для работы с Windows Azure Connect. После этого агент может использовать IPsec для взаимодействия с конкретной ролью в приложении.

Следует отметить, что это подключение не является полноценной виртуальной частной сетью. Корпорация Microsoft объявила о планах предоставить такой функционал в дальнейшем, однако в настоящее время Windows Azure Connect представлен в виде более простого решения. Например, для установки такого подключения не требуется обращаться к администратору сети вашей организации. Все, что требуется, — это возможность установить агент конечной точки на локальном компьютере. Такой подход также позволяет избежать потенциально сложной настройки IPsec — ее за вас выполнит Windows Azure Connect.

После установки этой технологии роли в приложении Windows Azure как бы находятся в той же IP-сети, что и локальный компьютер. В частности, это открывает следующие возможности.

- Приложение Windows Azure может напрямую осуществлять доступ к локальной базе данных. Для примера представим, что организация перемещает существующее приложение Windows Server, созданное с использованием ASP.NET, в Web-роль Windows Azure. Если используемая этим приложением база данных должна остаться локальной, подключение Windows Azure Connect дает возможность этому приложению, теперь выполняемому в Windows Azure, осуществлять доступ к локальной базе данных, как и прежде. Не придется даже менять строку подключения.

- Приложение Windows Azure может быть присоединено к домену локальной среды. Это позволит использовать единый вход для подключения локальных пользователей к облачному приложению. Кроме того, для управления доступом приложение сможет использовать существующие учетные записи и группы Active Directory.

Важно обеспечить хорошее взаимодействие между облаком и существующей локальной средой. Предоставляя возможность прямого подключения IP-уровня, Windows Azure Connect облегчает эту задачу для приложений Windows Azure.

СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ WINDOWS AZURE

Значение компонентов Windows Azure понимать важно, но недостаточно. Чтобы понять, какую пользу эта платформа может принести вашей организации, лучше всего рассмотреть примеры ее возможного применения. Итак, в этом разделе будет рассмотрено несколько сценариев применения Windows Azure: создание масштабируемого веб-приложения, создание приложения с параллельной обработкой, создание веб-приложения с фоновой обработкой, создание веб-приложения с реляционными данными, перенос локального веб-приложения с реляционными данными и использование облачного хранилища из локального или размещаемого приложения.

СОЗДАНИЕ МАСШТАБИРУЕМОГО ВЕБ-ПРИЛОЖЕНИЯ

Представим, что организация хочет создать веб-приложение, доступное через Интернет. Обычным вариантом в настоящее время было бы выполнять такое приложение в центре обработки данных внутри организации или у поставщика услуг размещения. Однако во многих случаях облачная платформа, такая как Windows Azure, будет более удачным вариантом. Например, если приложение должно справляться с большим количеством одновременных пользователей, есть смысл создавать его на платформе, специально для этого предназначенной. Встроенная поддержка масштабируемых приложений и данных в Windows Azure позволяет успешно выдерживать существенно более высокие нагрузки, чем традиционные веб-технологии.

Или, допустим, нагрузка приложения будет значительно варьироваться — посреди долгих периодов низкой нагрузки возможны пики активности. Вот несколько примеров: сайт для заказа билетов; новостной сайт с видеоматериалами, где иногда бывают сенсационные репортажи; приложение, которое в основном используют в определенное время суток; и т. п. Выполнение таких приложений в традиционном центре обработки данных потребовало бы постоянно использовать столько компьютеров, сколько необходимо для обработки пиковых нагрузок, несмотря на то что они простаивали бы большую часть времени. Если вместо этого создать приложение в Windows Azure, организация может увеличивать число используемых экземпляров, только когда это необходимо, а затем снова сокращать до небольшого количества. Поскольку плата за Windows Azure зависит от использования (почасовая за каждый экземпляр), такой вариант, скорее всего, будет более выгодным, чем постоянно иметь множество компьютеров, которые большую часть времени не нужны.

Для создания приложения с высокой степенью масштабируемости на платформе Windows Azure разработчик может использовать Web-роли и таблицы. На рис. 8 упрощенно показано, как это выглядит.

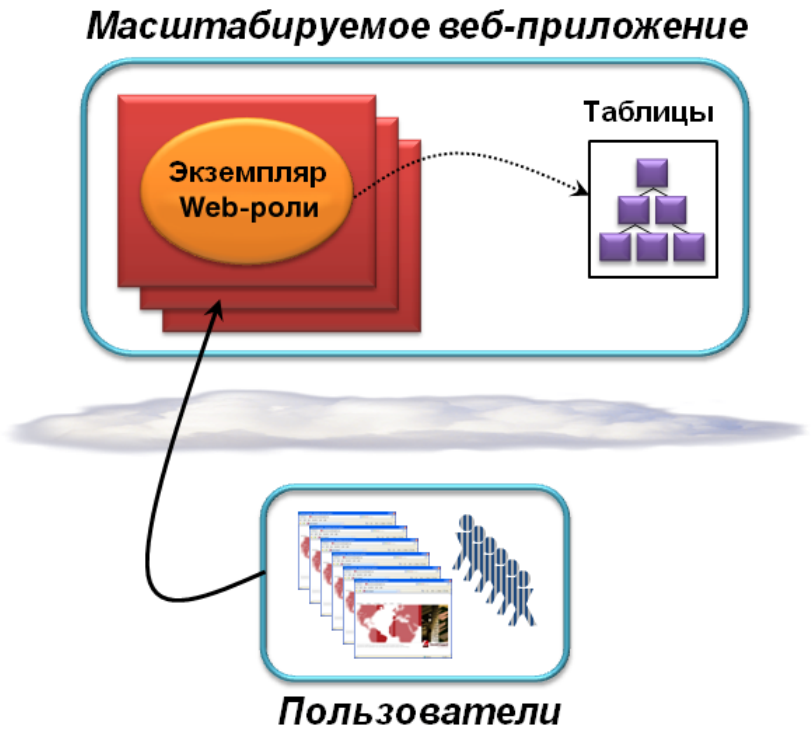

Рис. 8. Масштабируемое веб-приложение может использовать экземпляры Web-ролей и таблицы.

В показанном здесь примере клиентами являются браузеры, так что логика приложения может быть реализована с помощью ASP.NET или другой веб-технологии. Также можно создать масштабируемое веб-приложение, предоставляющее для доступа веб-сервисы на основе RESTful и (или) SOAP с использованием WCF, и эти сервисы будут вызываться, например, из клиента Silverlight. В любом из этих вариантов разработчик указывает, сколько экземпляров Web-роли должно выполняться, а Fabric Controller платформы Windows Azure создает столько виртуальных машин. Как описано ранее, Fabric Controller также осуществляет мониторинг этих экземпляров, обеспечивая постоянную доступность требуемого их количества. Для хранения данных приложение использует таблицы хранилища Windows Azure, предоставляющие горизонтально масштабируемое хранилище, которое способно обрабатывать очень большие объемы данных.

СОЗДАНИЕ ПРИЛОЖЕНИЯ С ПАРАЛЛЕЛЬНОЙ ОБРАБОТКОЙ

Масштабируемые веб-приложения дают преимущества, но это далеко не единственная ситуация, в которой целесообразно использовать Windows Azure. Представим организацию, которой периодически требуются очень большие вычислительные мощности для приложения с параллельной обработкой. Этому есть множество примеров: финансовое моделирование в банке, рендеринг спецэффектов в киностудии, разработка нового лекарства в фармацевтической компании и т. п. Можно постоянно содержать большой кластер компьютеров для таких эпизодических задач, однако это обходится дорого. Вместо этого Windows Azure может обеспечивать такие ресурсы по мере необходимости, предоставляя вычислительный кластер по запросу.

Для создания такого приложения разработчик может использовать Worker-роли. Хотя это не единственный вариант для решения такой задачи, параллельные приложения, как правило, используют большие наборы данных, которые можно хранить в больших двоичных объектах Windows Azure. На рис. 9 наглядно показано приложение такого типа.

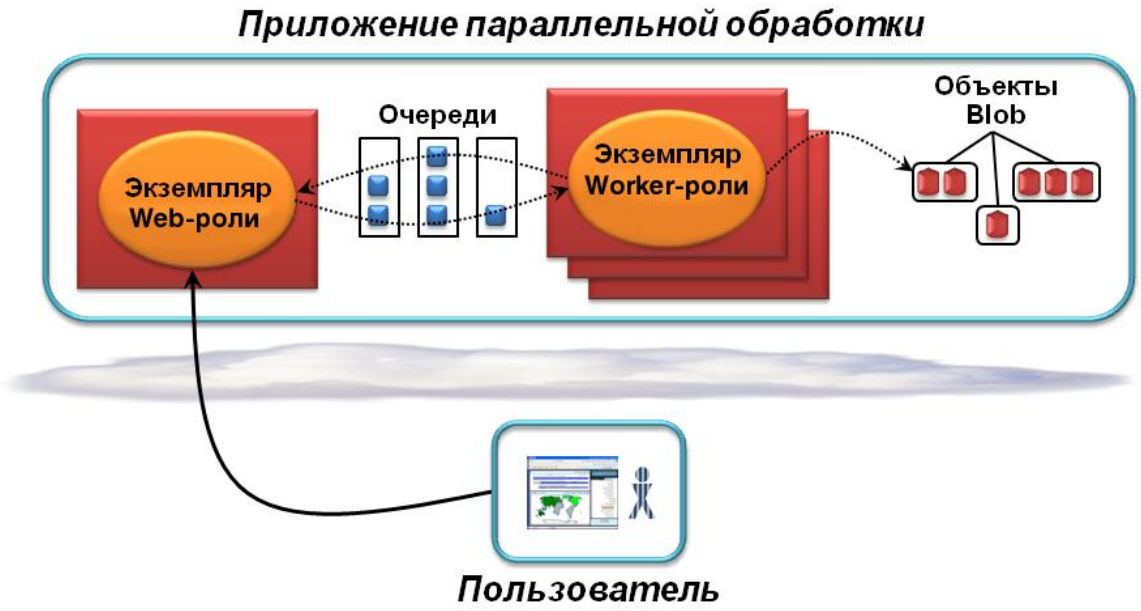

Рис. 9. Приложение с параллельной обработкой может использовать экземпляр Web-роли, несколько экземпляров Worker-ролей, очереди и объекты BLOB.

В показанном здесь сценарии параллельная работа выполняется несколькими экземплярами Worker-роли, которые выполняются одновременно и каждая из которых использует данные больших двоичных объектов. Поскольку Windows Azure не ограничивает продолжительность выполнения экземпляра, каждый экземпляр может выполнять произвольный объем работы. Взаимодействие пользователя с приложением осуществляется через один экземпляр Web-роли. С помощью этого интерфейса пользователь может задавать количество выполняемых экземпляров Worker-ролей, запускать и останавливать экземпляры, получать результаты и выполнять ряд других действий. Связь между экземпляром Web-роли и экземплярами Worker-ролей осуществляется через очереди хранилища Windows Azure.

Учитывая огромный объем вычислительных мощностей, доступных в облаке, такой передовой подход становится поворотной точкой в развитии высокопроизводительных вычислительных систем. Например, Microsoft Windows HPC Server уже позволяет создавать вычислительный кластер с использованием экземпляров Worker-ролей

Windows Azure вместе с локальными физическими серверами или вместо них. Как бы это ни было реализовано, использование этого нового источника вычислительных мощностей целесообразно во многих случаях.

СОЗДАНИЕ МАСШТАБИРУЕМОГО ВЕБ-ПРИЛОЖЕНИЯ С ФОНОВОЙ ОБРАБОТКОЙ

Пожалуй, не будет преувеличением сказать, что большинство создаваемых в настоящее время приложений предоставляют интерфейс на основе браузера. И хотя приложения, которые только принимают запросы из браузера и отвечают на них, дают некоторые преимущества, они также связаны с определенными ограничениями. Во многих ситуациях программному продукту с веб-интерфейсом требуется инициировать работу, которая должна выполняться в фоновом режиме независимо от запросов и ответов пользовательского интерфейса приложения.

Для примера возьмем веб-приложение для публикации видеоматериалов. Оно должно принимать запросы из браузера — возможно, от большого числа одновременных пользователей. В том числе будут запросы на загрузку новых видеоматериалов, каждый из которых нужно будет обработать и сохранить для последующего доступа. Нет смысла заставлять пользователя ждать, пока завершится такая обработка. Вместо этого та часть приложения, которая принимает запросы из браузера, должна иметь возможность запустить фоновую задачу, выполняющую такую работу.

Для реализации этого сценария можно использовать сочетание Web-ролей и Worker-ролей. На рис. 10 показано, как может выглядеть подобное приложение.

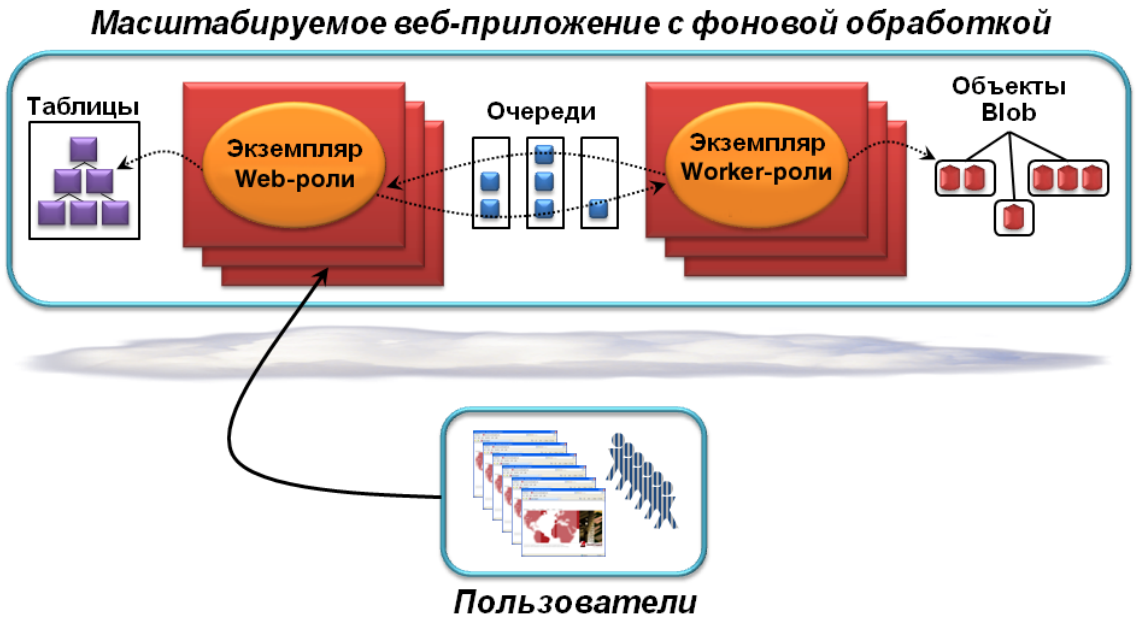

Рис. 10. Масштабируемое веб-приложение с фоновой обработкой может использовать различные возможности Windows Azure.

Как и в ранее показанном масштабируемом веб-приложении, в этом примере используется некоторое количество экземпляров Web-ролей для обработки запросов пользователей. Чтобы поддерживать большое число одновременных пользователей, сведения их профилей хранятся в таблицах. Фоновая обработка осуществляется через экземпляры Worker-ролей, которым через очереди передаются задания. В этом примере экземпляры Worker-ролей работают с данными в больших двоичных объектах, но возможны и другие подходы.

Данный пример показывает, как в приложении могут сочетаться несколько основных возможностей, предлагаемых Windows Azure: экземпляры Web-ролей, экземпляры Worker-ролей, большие двоичные объекты, таблицы и очереди. И хотя на рисунке это не показано, приложение для публикации видеоматериалов может для ускорения доступа использовать сеть доставки контента Windows Azure. Даже если все эти возможности понадобятся не в каждом приложении, их наличие важно для поддержки более сложных сценариев, таких как описанный выше.

СОЗДАНИЕ ВЕБ-ПРИЛОЖЕНИЯ С РЕЛЯЦИОННЫМИ ДАННЫМИ

Большие двоичные объекты и таблицы в Windows Azure прекрасно подходят для ряда случаев, однако в других ситуациях удобнее использовать реляционные данные. Представим, что предприятие хочет создать для своих сотрудников приложение на платформе Windows Azure. Есть вероятность, что время его существования будет небольшим или непредсказуемым, поэтому выделение сервера в корпоративном центре обработки данных не очень оправдано. Или, возможно, требуется как можно быстрее запустить приложение в эксплуатацию, поэтому неприемлемо ждать, пока ИТ-отдел выделит и настроит сервер. Или, например, в организации пришли к выводу, что эксплуатировать приложение на платформе Windows Azure будет дешевле и проще.

Каковы бы ни были причины, вероятно, такому приложению не потребуются огромные масштабы, доступные при использовании таблиц Windows Azure. Вместо этого разработчики могут выбрать уже знакомый им реляционный подход и привычные средства создания отчетов. В такой ситуации в приложении можно использовать Windows Azure вместе с SQL Azure, как показано на рис. 11.

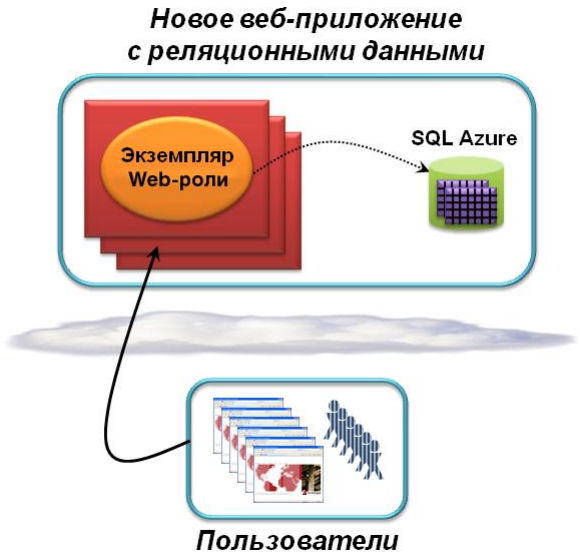

Рис. 11. Приложение Windows Azure может использовать SQL Azure для работы с реляционными данными.

SQL Azure, являясь управляемым облачным сервисом, предоставляет многие функции SQL Server, в том числе для создания отчетов. Приложения могут создавать базы данных, выполнять запросы SQL и выполнять другие действия, но при этом не придется администрировать базы данных или оборудование, на котором выполняется такая система, поскольку эту задачу берет на себя Microsoft. Доступ к базе данных SQL Azure можно осуществлять с помощью протокола потока табличных данных (Tabular Data Stream, TDS), как и в случае с локальной версией SQL Server. Благодаря этому приложение Windows Azure может осуществлять доступ к реляционным данным с помощью привычных механизмов, таких как Entity Framework и ADO.NET. А поскольку SQL Azure является облачным сервисом, плата зависит от использования.

Так как Windows Azure и SQL Azure предоставляют облачные варианты подобных локальных компонентов, это упрощает перемещение кода и данных для таких приложений как в одну, так и в другую сторону. Есть некоторые различия — например, приложение Windows Azure должно быть способно выполнять несколько экземпляров, — однако во многом облачная среда весьма напоминает локальную. Такая переносимость полезна, когда целесообразно создавать приложение, код и данные которого могут потенциально использоваться либо в локальной среде, либо в облаке.

ПЕРЕНОС ЛОКАЛЬНОГО ВЕБ-ПРИЛОЖЕНИЯ С РЕЛЯЦИОННЫМИ ДАННЫМИ

Представим, что вместо создания нового веб-приложения для Windows Azure организация хочет переместить на эту облачную платформу существующее приложение Windows Server. Один из способов это сделать — использовать VM-роль Windows Azure. Общая схема во многом похожа на предыдущую, как показано на рис. 12.

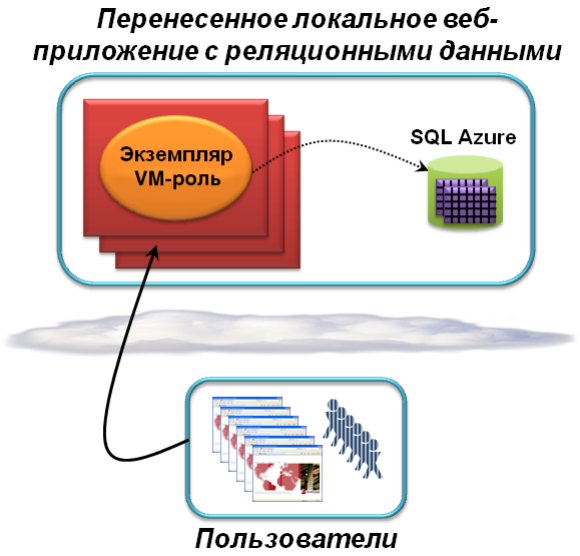

Рис. 12. Некоторые локальные приложения можно перенести в Windows Azure с помощью VM-роли и SQL Azure.

Чтобы использовать VM-роль, организация создает из локального компьютера под управлением Windows Server 2008 R2 виртуальный жесткий диск. Затем этот образ можно загрузить в Windows Azure и выполнять в VM-роли. Как показано на рисунке, приложение может осуществлять доступ к реляционным данным в SQL Azure. Другой вариант — оставить данные локально и осуществлять к ним доступ напрямую с помощью Windows Azure Connect, как описывалось ранее.

Применение VM-роли может быть весьма эффективным, однако важно понимать, что для переноса приложения Windows Server в Windows Azure могут потребоваться дополнительные действия помимо упаковки в виртуальный жесткий диск и выполнения в VM-роли. Во-первых, вспомним, что Fabric Controller платформы Windows Azure предполагает, что всегда выполняется не менее двух экземпляров каждой роли. (Фактически, этого требует соглашение об уровне обслуживания Windows Azure.) Также следует учитывать, что Windows Azure балансирует нагрузку всех пользовательских запросов, распределяя их по экземплярам роли. Если переносимое приложение разработано с учетом этого (например, оно уже выполняется в веб-ферме с балансировкой нагрузки), то оно может с успехом выполняться в Windows Azure без существенных изменений. А если приложение разрабатывалось для выполнения только в одном экземпляре, возможно, его придется переработать, чтобы оно с успехом выполнялось в Windows Azure.

ИСПОЛЬЗОВАНИЕ ОБЛАЧНОГО ХРАНИЛИЩА ИЗ ЛОКАЛЬНОГО ИЛИ РАЗМЕЩАЕМОГО ПРИЛОЖЕНИЯ

В то время как Windows Azure предоставляет широкий спектр возможностей, приложению нередко требуется только одна из них. Представим локальное или размещаемое приложение, которому нужно хранить большие объемы данных. Например, предприятие хочет архивировать старую электронную почту, чтобы сэкономить деньги на хранилище, но при этом обеспечить доступность архива со старой почтой. Новостному веб-сайту, размещенному на стороннем хостинге, может понадобиться масштабируемое и доступное из любой точки мира хранилище для больших объемов текстов, графики, видеоматериалов и сведений о профилях пользователей. Создатели сайта для публикации фотографий могут захотеть делегировать сложные задачи хранения информации надежному стороннему поставщику услуг. Во всех этих случаях можно применить хранилище Windows Azure, как показано на рис. 13.

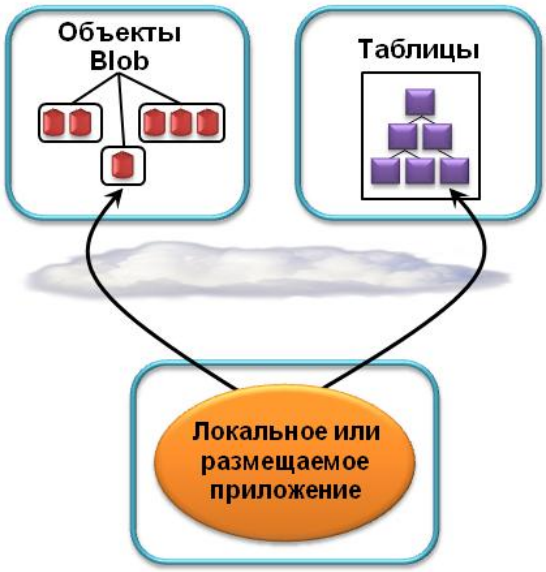

Рис. 13. Локальное или размещаемое приложение может использовать объекты BLOB или таблицы Windows Azure для хранения данных в облаке.

Как упоминалось ранее, приложение, размещаемое локально или на стороннем хостинге, может напрямую осуществлять доступ к хранилищу Windows Azure. Хотя такой доступ может быть медленнее, чем использование локального хранилища, этот вариант, скорее всего, будет дешевле, а масштабируемость и надежность — выше. Для некоторых приложений такой компромисс определенно имеет смысл. На рисунке это не показано, но приложения могут по схожей схеме использовать SQL Azure.

ПОДРОБНОЕ РАССМОТРЕНИЕ ВОЗМОЖНОСТЕЙ WINDOWS AZURE

Для понимания возможностей Windows Azure мы рассмотрели базовую информацию об этой платформе и базовые типовые сценарии применения. Однако возможности этой технологии намного шире. И в этом разделе более подробно рассматриваются некоторые интересные аспекты.

СОЗДАНИЕ ПРИЛОЖЕНИЙ WINDOWS AZURE

Для разработчиков создание приложений Windows Azure в основном напоминает разработку обычных приложений Windows. Поскольку платформа поддерживает и приложения .NET, и приложения, созданные с помощью неуправляемого кода, разработчики могут использовать средства, наиболее удобные для решения их задач. Чтобы сделать разработку удобнее, в Visual Studio предоставляются шаблоны проектов для создания приложений Windows Azure. Также есть возможность напрямую загружать приложения из Visual Studio в Windows Azure.

Одно очевидное различие между облачными и локальными технологиями заключается в том, что приложения Windows Azure выполняются не локально. Потенциально это может несколько усложнить разработку. Чтобы облегчить эту задачу, Microsoft предоставляет *фабрику развертывания* — версию среды Windows Azure, которая выполняется на компьютере разработчика.

Фабрика развертывания выполняется на одном настольном или серверном компьютере. Это эмулятор функциональных возможностей Windows Azure в облаке, включая Web-роли, Worker-роли, VM-роли и все три варианта хранилища Windows Azure. Разработчик может создать приложение Windows Azure, развернуть его в фабрике развертывания и выполнять практически так же, как в реальной среде Windows Azure. Например, можно указывать количество выполняемых экземпляров для каждой роли, использовать очереди для связи между экземплярами и делать практические все, что можно делать в реальной среде Windows Azure. После того как приложение создано и протестировано локально, разработчик может загрузить код и сведения о конфигурации, а затем запустить это приложение.

Независимо от способа разработки, обычно используется процесс из двух шагов, чтобы сделать приложение доступным в облаке Windows Azure. Сначала разработчик загружает приложение в промежуточную область платформы. По готовности запустить приложение в эксплуатацию, он с помощью портала Windows Azure отправляет запрос на помещение приложения в рабочую среду. Такое перемещение можно выполнять без простоя, так что версию выполняемого приложения можно обновлять, не доставляя неудобств пользователям.

У приложения, помещенного в промежуточную область, есть DNS-имя в виде *<GUID>*.cloudapp.net, где в качестве *<GUID>* используется глобальный уникальный идентификатор, назначенный платформой Windows Azure. Для рабочей среды разработчик выбирает DNS-имя в том же домене, например myazureservice.cloudapp.net. Чтобы использовать настраиваемый домен, а не домен cloudapp.net от Microsoft, владелец приложения может создать DNS-псевдоним с помощью стандартной записи CNAME.

Когда приложение будет доступно из внешнего мира, пользователям, скорее всего, понадобится каким-то образом идентифицировать себя. Для этой цели Windows Azure позволяет разработчикам использовать любой механизм проверки подлинности на основе HTTP. Например, в приложении ASP.NET можно использовать поставщика членства для хранения собственного идентификатора пользователя и пароля или какой-нибудь другой способ вроде сервиса Microsoft Windows Live ID. Приложения Windows Azure могут также использовать Windows Identity Foundation (WIF) для реализации удостоверения, основанного на утверждениях. Выбор варианта — полностью на усмотрение разработчика приложения.

ИЗУЧЕНИЕ СЕРВИСА ВЫЧИСЛЕНИЙ

Как и большинство технологий, вычисления в Windows Azure эволюционировали с момента первого выпуска. Например, в первоначальном варианте код в Web-ролях и Worker-ролях можно было выполнять только в пользовательском режиме. Однако в настоящее время роли обоих типов предоставляют возможность использования повышенных привилегий, благодаря чему приложения могут выполняться в административном режиме. Это может пригодиться для приложений, которым, например, требуется установить компонент COM — в первом выпуске Windows Azure это было бы проблематично.

Но при этом нужно иметь в виду, что каждый экземпляр Web-роли или Worker-роли запускается «с чистого листа». Операционная система в виртуальной машине — это стандартный образ, определяемый платформой Windows Azure. Следовательно, все установки ПО, выполняемые ролью, должны совершаться при каждом создании нового экземпляра. Это не создает трудностей при простых установках, таких как добавление одного компонента COM. Но, предположим, экземпляру нужно установить много программных продуктов для выполнения своих задач. Если выполнять такие действия при каждом создании нового экземпляра роли, это занимало бы слишком много времени.

Исключение таких задержек — одна из главных задач VM-ролей. Вместо того чтобы требовать установки ПО при каждом создании экземпляра, все необходимое ПО можно включить в виртуальный жесткий диск, с которым затем будет создаваться экземпляр VM-роли. Такой вариант может выполняться значительно быстрее, чем при использовании Web-ролей и Worker-ролей с повышенными привилегиями. Такое решение может также пригодиться, если в процессе установки необходимо ручное вмешательство — ведь оно не допускается в Windows Azure.

Еще одно изменение по сравнению с первоначальной версией Windows Azure заключается в том, что теперь платформа поддерживает доступ по протоколу удаленного рабочего стола. Например, это может пригодиться при отладке, чтобы разработчик мог напрямую получить доступ к определенному экземпляру. Однако не следует ожидать поддержки инфраструктуры виртуальных рабочих столов — в Windows Azure (по крайней мере, в настоящее время) не предусмотрена поддержка такого сценария.

Другие важные возможности вычислений Windows Azure были доступны уже с первого выпуска этой технологии. Например, Windows Azure позволяет разработчику указывать, в каком центре обработки данных должно выполняться приложение и где должны храниться его данные. Разработчик может также указать, что определенная группа приложений и данных (в том числе данные в SQL Azure) должна располагаться в одном и том же центре обработки. С самого начала Microsoft предоставляет центры обработки данных Windows Azure в Соединенных Штатах, Европе и Азии, и в дальнейшем их география будет расширяться.

ИЗУЧЕНИЕ СЕРВИСА ХРАНИЛИЩА

Чтобы использовать хранилище Windows Azure, разработчик должен сначала создать *учетную запись хранения*. Для управления доступом к информации в такой учетной записи платформа Windows Azure выдает создателю секретный ключ. Каждый запрос, совершаемый приложением для получения информации в учетной записи хранения — больших двоичных объектов, таблиц и очередей, — должен иметь подпись, созданную с этим секретным ключом. Другими словами, авторизация осуществляется на уровне учетной записи (хотя для больших двоичных объектов есть еще один вариант, который описывается далее). Хранилище Windows Azure не предоставляет списков управления доступом и других более детальных способов управления доступа к данным.

Большие двоичные объекты

Большие двоичные объекты (BLOB) зачастую очень необходимы в приложении. В них можно хранить самые различные данные — видео, аудио, заархивированные сообщения электронной почты или любую другую информацию. Чтобы использовать большие двоичные объекты, разработчик должен сначала создать один или несколько контейнеров в какой-либо учетной записи хранения. В каждом из таких *контейнеров* можно будет хранить один или несколько больших двоичных объектов.

Чтобы идентифицировать определенный большой двоичный объект, приложение может предоставлять универсальный код ресурса (URI) в следующем виде:

http://*<StorageAccount>*.blob.core.windows.net/*<Container>*/*<BlobName>*

где <StorageAccount> — это уникальный идентификатор, присвоенный при создании учетной записи хранения, а *<Container>* и *<BlobName>* — это наименования конкретного контейнера и большого двоичного объекта в этом контейнере.

Большие двоичные объекты бывают двух видов:

- Блочные BLOB-объекты, которые могут вмещать в себя до 200 гигабайт данных. Для более эффективной передачи они делятся на блоки. Если происходит сбой, отправку можно возобновить с последнего по времени блока, вместо того чтобы отправлять весь большой двоичный объект заново. После того как все блоки большого двоичного объекта загружены, его можно зафиксировать целиком.

- Страничные BLOB-объекты, каждый из которых может иметь размер до одного терабайта. Страничный BLOB-объект делится на 512-байтовые страницы, любые из которых может записывать и считывать приложение.

Независимо от типа хранящихся в них больших двоичных объектов, контейнеры могут быть помечены как частные или «публичные». Для больших двоичных объектов в частном контейнере запросы на чтение и запросы на запись должны быть подписаны ключом для учетной записи хранения таких объектов. Для больших двоичных объектов в «публичном» контейнере должны быть подписаны только запросы на запись, а чтение разрешено любому приложению. Это может пригодиться, например, если нужно открыть в Интернете общий доступ к видеоматериалам, фотографиям и другим неструктурированным данным. В действительности сеть доставки контента Windows Azure работает только с данными, хранящимися в «публичных» контейнерах больших двоичных объектов.

Еще один важный аспект больших двоичных объектов — это выполняемая ими функция в поддержке дисков Windows Azure. Чтобы лучше понять смысл выполняемой ими задачи, сначала нужно отметить, что экземпляры ролей могут по своему усмотрению осуществлять доступ к своей локальной файловой системе. Однако по умолчанию это хранилище не является постоянным. Когда экземпляр прекращает работу, виртуальная машина и ее локальное хранилище уничтожаются. Однако если смонтировать диск Windows Azure для экземпляра, то можно сделать так, чтобы страничный BLOB-объект выглядел, как локальный диск с файловой системой NTFS. Данные, записываемые на диск, могут тут же записываться в базовый большой двоичный объект. Когда экземпляр не выполняется, эти данные постоянно хранятся в страничном BLOB-объекте и готовы для монтирования. Вот несколько примеров использования таких дисков:

- Разработчик может загрузить виртуальный жесткий диск с файловой системой NTFS, а затем смонтировать его в качестве диска Windows Azure. Это простой способ перемещения данных файловой системы между Windows Azure и локальной системой Windows Server.

- Разработчик Windows Azure может установить и запустить систему базы данных MySQL в экземпляре роли Windows Azure, используя диск Windows Azure в качестве базового хранилища.

Таблицы

Большой двоичный объект прост для понимания — это всего лишь множество байтов. А вот с таблицами все немного сложнее. На рис. 14 наглядно показано, как соотносятся между собой части таблицы.

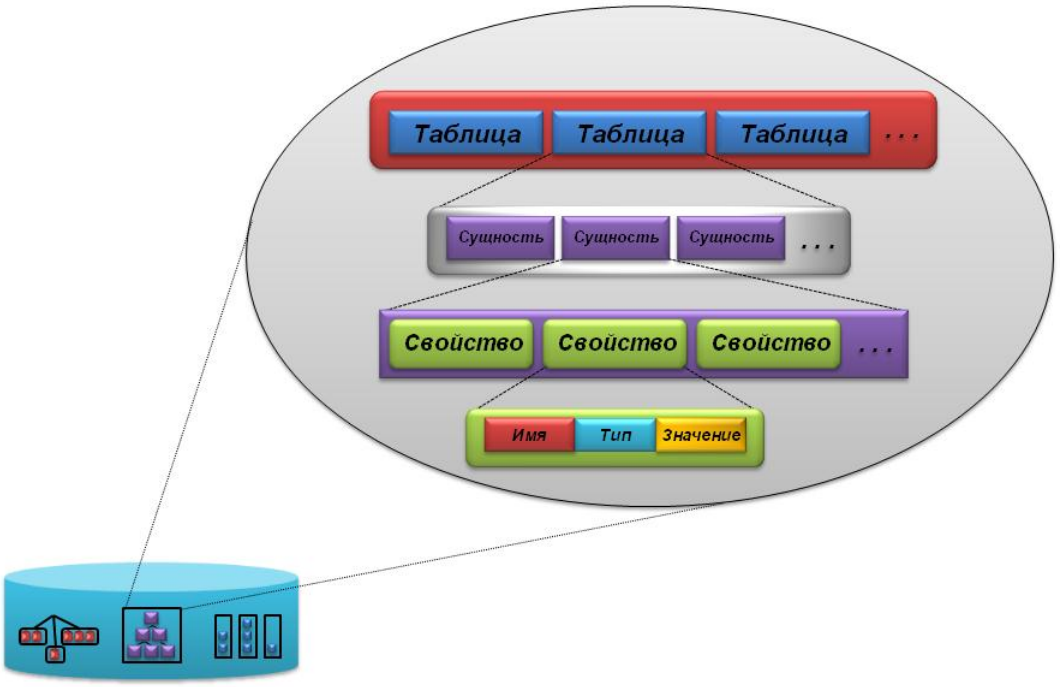

Рис. 14. Таблицы предоставляют хранилище, основанное на сущностях.

Как показано на рисунке, в каждой таблице есть некоторое количество сущностей. У сущности может быть одно или несколько свойств, у каждого из которых есть имя, тип и значение. Поддерживаются разнообразные типы, в том числе Binary, Bool, DateTime, Double, GUID, Int, Int64 и String. Свойство может в разные моменты иметь разный тип в зависимости от хранящегося в нем значения. И не требуется, чтобы все свойства в одной сущности были одинакового типа, — разработчик может по своему усмотрению делать так, как удобно в разрабатываемом приложении.

Вне зависимости от содержимого сущности, ее размер может достигать 1 мегабайта, а доступ к ней всегда осуществляется как к единому целому. При чтении сущности возвращаются все ее свойства, а при записи — заменяются значения всех свойств. Можно автоматически обновлять группы сущностей в одной таблице, при этом либо все обновления будут выполнены успешно, либо все они будут отменены.

Между таблицами хранилища Windows Azure и реляционными есть несколько отличий. Во-первых, они не являются таблицами в обычном смысле этого слова. Во-вторых, к ним нельзя осуществлять доступ с помощью обычных средств ADO.NET, и они не поддерживают запросы SQL. Таблицы в хранилище Windows Azure не обязаны придерживаться жесткой схемы — свойства одной сущности могут быть различных типов, причем они могут меняться со временем. Возникает естественный вопрос: зачем? почему бы просто не поддерживать обычные реляционные таблицы со стандартными запросами SQL?

Ответ кроется в главной задаче Windows Azure — поддержке приложений с высокой степенью масштабируемости. Традиционные реляционные базы данных могут масштабироваться, обслуживая все большее число пользователей, и для этого нужно, чтобы СУБД выполнялась на более мощных компьютерах. Однако для одновременной поддержки действительно огромного числа пользователей хранилище должно масштабироваться горизонтально, а не вертикально. Для этого механизм хранилища должен стать проще, а традиционные реляционные таблицы со стандартным SQL не очень для этого подходят. Решение данной задачи — это структура такого типа, как в таблицах Windows Azure.

Для применения таблиц разработчикам придется немного пересмотреть концепции, поскольку привычные реляционные принципы не получится применить без корректировок. И все же для создания хорошо масштабируемых приложений такой подход является оптимальным. Он позволяет разработчикам не беспокоиться о масштабах — они просто могут создавать новые таблицы, добавлять новые сущности, а об остальном позаботится Windows Azure. Также исключается значительная часть работы по поддержке СУБД, поскольку эту задачу Windows Azure берет на себя. В конечном итоге все это позволяет разработчикам сосредоточиться на создаваемом приложении, не отвлекаясь на хранение и администрирование больших объемов данных.

Как и для всего остального в хранилище Windows Azure, для доступа к таблицам используется подход RESTful. Для этого приложение .NET может использовать сервисы данных WCF, скрывая базовые запросы HTTP. Любое приложение, созданное на основе .NET или других средств, также может по своему усмотрению использовать такие запросы напрямую. Например, запрос в отношении конкретной таблицы выражается как HTTP GET в отношении универсального кода ресурса (URI), форматированного следующим образом:

http://*<StorageAccount>*.table.core.windows.net/*<TableName>*?$filter=*<Query>*

где *<TableName>* обозначает запрашиваемую таблицу, а *<Query>* содержит запрос, который должен быть выполнен в отношении этой таблицы. Если запрос возвращает большое количество результатов, разработчик может получить маркер продолжения, который можно передать следующему запросу. Это позволяет поэтапно, фрагментами извлекать полный набор результатов.

Таблицы Windows Azure могут не подходить для некоторых сценариев хранения, и для их использования разработчикам нужно будет освоить некоторые новшества. Однако для приложений, которым требуется высокая степень масштабируемости, таблицы являются оптимальным решением.

Очереди

В то время как таблицы и большие двоичные объекты предназначены прежде всего для хранения данных и доступа к данным, главной задачей очередей является обеспечение связи между различными частями приложения Windows Azure. Как и для всего остального в хранилище Windows Azure, для доступа к очередям используется подход RESTful. И приложения Windows Azure, и внешние приложения ссылаются на очередь, используя универсальный код ресурса (URI) в следующем формате:

http://<StorageAccount>.queue.core.windows.net/<QueueName>

Как описывалось ранее, очереди используются в основном для обеспечения взаимодействия между экземплярами Web-ролей и Worker-ролей. На рис. 15 показано, как это выглядит.

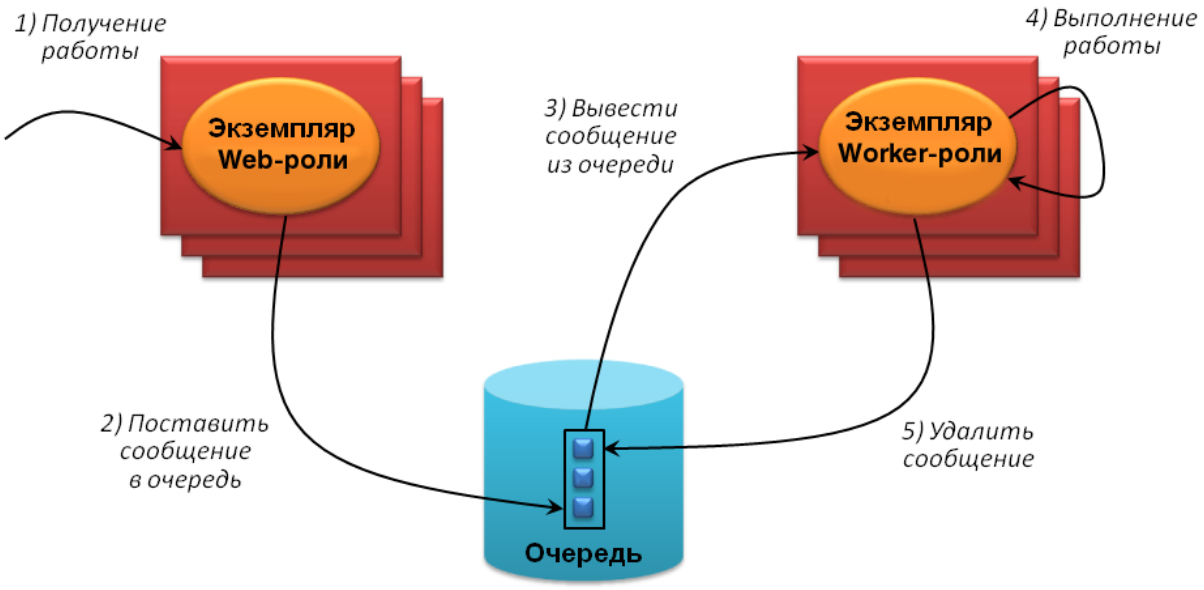

Рис. 15. Сообщения ставятся в очередь, выводятся из очереди, обрабатываются, а затем явным образом удаляются из очереди.

В типовом сценарии выполняется несколько экземпляров Web-роли, каждый из которых принимает задания от пользователей (шаг 1). Чтобы передать задание экземплярам Worker-роли, экземпляр Web-роли записывает сообщение в очередь (шаг 2). В таком сообщении, размер которого может доходить до 8 килобайт, может содержаться универсальный код ресурса (URI), указывающий на большой двоичный объект, сущность в таблице или на что-то еще — на усмотрение приложения. Экземпляры Worker-роли считывают сообщения из очереди (шаг 3) и затем выполняют задание, запрашиваемое в сообщении (шаг 4). Важно отметить, что при чтении сообщения из очереди сообщение фактически не удаляется. Вместо этого сообщение на определенный период времени (по умолчанию — на 30 секунд) становится невидимым для других читателей. Когда экземпляр Worker-роли завершает выполнение запрошенного в сообщении задания, он должен явным образом удалить это сообщение из очереди (шаг 5).

Разделение экземпляров Web-ролей и экземпляров Worker-ролей оправдано. За счет этого пользователю не придется ждать, пока будет обработано большое задание, а также упрощается масштабируемость — для этого достаточно просто добавить дополнительные экземпляры Web-роли или Worker-роли. Но зачем сделано так, что экземпляры должны явным образом удалять сообщения? Дело в том, что такой подход позволяет справляться со сбоями. Если экземпляр Worker-роли, который извлекает сообщение, обрабатывает его успешно, то он удалит это сообщение, пока оно еще невидимо, то есть в течение 30-секундного промежутка. Но если экземпляр Worker-роли выводит сообщение из очереди, а затем происходит сбой, прежде чем указанная в этом сообщении работа выполнена, то он не удалит сообщение из очереди. Когда истечет срок невидимости сообщения, оно снова появится в очереди и будет прочитано другим экземпляром Worker-роли. Таким образом гарантируется, что каждое сообщение будет обработано хотя бы один раз.

Как следует из этого описания, семантика очередей хранилища Windows Azure отличается от очереди сообщений Microsoft Message Queuing (MSMQ) и других привычных технологий очередей. Например, традиционная система очередей может обеспечивать семантику «первым пришел — первым обслужен», гарантируя, что каждое сообщение будет доставлено в точности один раз. Очереди хранилища Windows Azure не дают таких обещаний. Как описывалось выше, сообщение может быть доставлено несколько раз и нет гарантии, что сообщения будут доставлены в каком-то определенном порядке. Работа в облаке имеет свои особенности, и разработчикам нужно к ним приспособиться.

ИЗУЧЕНИЕ FABRIC CONTROLLER

Для разработчика приложений самыми важными компонентами Windows Azure являются вычисления и хранилище. Но ни тот ни другой компонент не смогли бы функционировать без Fabric Controller. Связывая воедино множество компьютеров центра обработки данных, он создает основу для всех остальных составляющих частей.

Как описывалось ранее, Fabric Controller принадлежат все ресурсы определенного центра обработки данных Windows Azure. Он также отвечает за назначение приложений физическим компьютерам. Очень важно, чтобы это действие выполнялось оптимальным образом. Для примера представим, что разработчик запрашивает для приложения пять экземпляров Web-роли и четыре экземпляра Worker-роли. При упрощенном назначении все эти экземпляры могли бы быть размещены в одной и той же стойке, обслуживаемой одним и тем же сетевым коммутатором. Если выйдет из строя стойка или коммутатор, все приложение станет недоступным. Учитывая, что задачей Windows Azure является обеспечение высокого уровня доступности, было бы крайне нежелательно делать приложение зависимым от единой точки отказа, как в представленном выше примере.

Чтобы исключить подобные ситуации, Fabric Controller группирует принадлежащие ему компьютеры в несколько *доменов отказа*. Каждый домен отказа — это часть центра обработки данных, где один сбой может закрыть доступ ко всему, что находится в этом домене. На рис. 16 это наглядно показано.

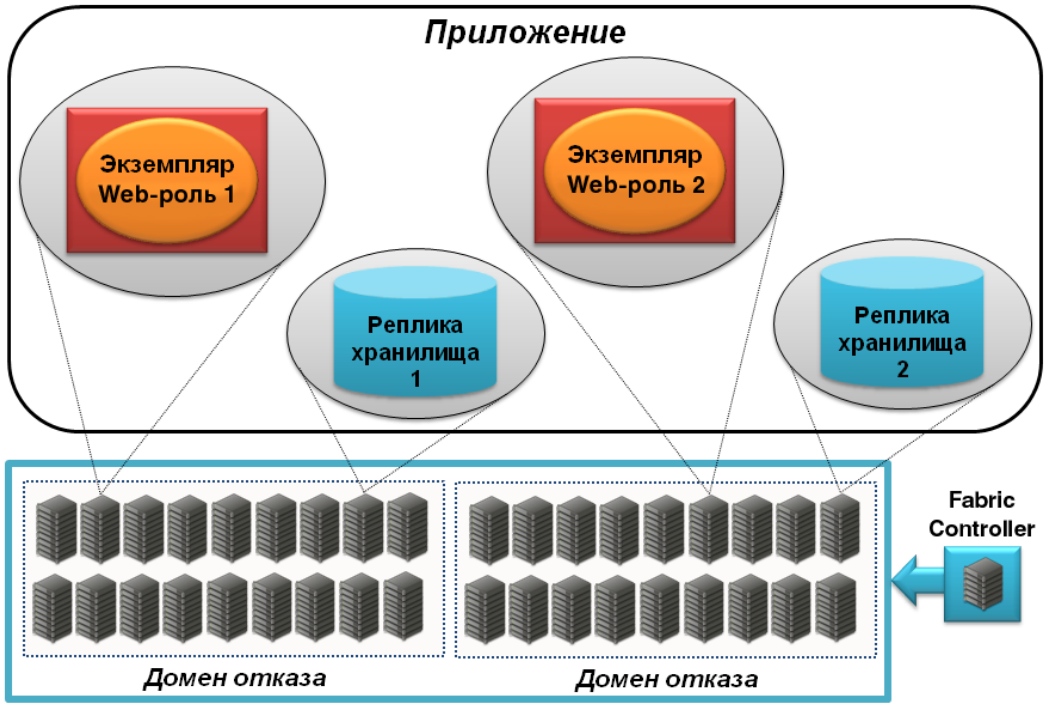

Рис. 16. Fabric Controller помещает разные экземпляры приложения в разные домены отказа.

В этом простом примере приложение использует только два экземпляра Web-роли, а центр обработки данных разделен на два домена отказа. Когда Fabric Controller развертывает такое приложение, он помещает по одному экземпляру Web-роли в каждый домен отказа. Такая схема означает, что из-за одного сбоя оборудования в центре обработки данных не будет прекращена работа всего приложения. Также напомним, что для Fabric Controller хранилище Windows Azure выглядит просто как еще одно приложение, то есть он не занимается репликацией данных. Вместо него эту задачу выполняет само приложение хранилища, заботясь о том, чтобы реплики всех используемых больших двоичных объектов, таблиц и очередей помещались в разные домены отказа.

Поддержание работоспособности приложения в случае сбоев оборудования полезно, но не достаточно. По-настоящему надежное приложение — а именно для этого создана платформа Windows Azure — должно обновляться, не вызывая перебоев в работе. Один из способов решения этой задачи — использование ранее описанного подхода для переключения с существующей версии приложения на новую версию. Другой вариант — использовать *домены обновления* Windows Azure. При таком подходе Fabric Controller назначает разные экземпляры ролей приложения разным доменам обновления. Когда нужно развернуть новую версию приложения, Fabric Controller развертывает новый код в одном домене обновления за один шаг. Он останавливает работу экземпляров роли в таком домене, обновляет код для этой роли, а затем запускает новые экземпляры. Смысл в том, чтобы приложение продолжало работать непрерывно — даже во время обновления. Пользователи могут заметить, что происходит обновление, поскольку время отклика приложения, скорее всего, увеличится, когда некоторые его экземпляры прекратили работу. Кроме того, в ходе обновления разным пользователям может предоставляться доступ к разным версиям приложения. И все же с точки зрения пользователя приложение в целом остается доступным непрерывно.

Не следует путать домены обновления, являющиеся свойством приложения, с доменами отказа — свойством центра обработки данных. Однако у этих технологий есть общая цель — помочь Fabric Controller поддерживать непрерывную работу приложений Windows Azure.

ПЕРСПЕКТИВЫ

Корпорация Microsoft объявила о планах по расширению возможностей Windows Azure в 2011 году, в том числе планируется следующее:

- Аппаратно-программный комплекс для платформы Windows Azure Platform, который позволит поставщикам услуг размещения и предприятиям запускать Windows Azure в собственных центрах обработки данных.
- Динамическое кэширование контента в сети доставки контента. В настоящее время сеть доставки контента Windows Azure работает только с данными больших двоичных объектов. Планируемые новые возможности позволят сети доставки также кэшировать контент, динамически создаваемый приложением Windows Azure.
- Создание снимков VM-ролей. В первом выпуске Windows Azure не сохраняла никаких изменений, вносимых в том операционной системы в ходе выполнения. После введения этой функции появится возможность периодически сохранять состояние тома в постоянном хранилище.
- **Улучшение поддержки Java.** Хотя в Windows Azure уже могут выполняться приложения Java, корпорация Microsoft планирует расширить их поддержку. Планируется повысить быстродействие Java, усилить поддержку средств на основе Eclipse, а также предоставить более полный набор библиотек Java для Windows Azure.

Как всегда, целью таких дополнений является расширение спектра применений этой облачной платформы.

ЗАКЛЮЧЕНИЕ

Выполнение приложений и хранение данных в облаке — во многих случаях решение оптимальное. Все это становится возможным благодаря сочетанию различных компонентов Windows Azure. Среда разработки Windows Azure, SQL Azure и другие компоненты платформы Windows Azure — все это облегчает разработчикам переход к облачным технологиям.

В настоящее время облачные платформы все еще несколько непривычны для большинства организаций. Однако по мере более близкого знакомства и получения опыта работы с Windows Azure и другими облачными платформами этот передовой подход уже не будет восприниматься как экзотика. Можно предположить, что со временем облачные приложения и облачные платформы, на которых выполняются такие приложения, будут играть все более значимую роль в сфере программных продуктов.

ДОПОЛНИТЕЛЬНЫЕ МАТЕРИАЛЫ ДЛЯ ЧТЕНИЯ

- Главная страница платформы Windows Azure

  http://www.microsoft.com/windowsazure

- Знакомство с платформой Windows Azure, Дэвид Чеппел

  http://go.microsoft.com/fwlink/?LinkId=158011

- Большие двоичные объекты Windows Azure. Программирование хранилища больших двоичных объектов

  http://go.microsoft.com/fwlink/?LinkId=153400

- Таблицы Windows Azure. Программирование хранилища таблиц

  http://go.microsoft.com/fwlink/?LinkId=153401

- Очереди Windows Azure. Программирование хранилища очередей

  http://go.microsoft.com/fwlink/?LinkId=153402

ОБ АВТОРЕ

Дэвид Чеппел является директором компании Chappell & Associates (www.davidchappell.com), Сан-Франциско, Калифорния. Выступая с лекциями, статьями и консультациями, он помогает людям из разных стран мира больше узнать о новых технологиях, лучше использовать их и принимать более взвешенные решения об их использовании.