第5章

Windows Azure云计算平台

微软公司的模式和实践团队的资深项目经理 Eugenio Pace 在为《云应用开发》一书(本作者翻译,已经由清华大学出版社出版)所写的序里面写道:“3 月 4 日,我收到了公司首席执行官 Steve Ballmer 的一封电子邮件。由于平时并不经常收到他的邮件,我对这封邮件给予了高度关注。邮件的标题是‘我们全身心投入(we are all in)’。邮件开宗明义地强调了微软公司对云计算的投入。如果还需要任何证据来证明微软对云计算的投入是认真的话,这封邮件就是了。”

由此开始,微软全力转舵,向云计算发起了全面进攻。一直以来,微软的在线产品丰富多样,但在云计算的核心领域一直有些犹豫不前,也许是担心云计算是一场大忽悠,也许是担心云计算并不会被用户所接纳,也许是担心云计算的技术还不够成熟。不管是什么原因,事实是,在云计算领域,微软似乎滞后了。

不过,瘦死的骆驼比马大,微软作为软件行业的老大,其积累深厚,一旦将注意力专注起来,其在云计算领域的推进就异常的迅速,Windows Azure 从一些零散的组件开发开始,到整体平台的推出只不过用了 3 年时间,并且在用户响应时间方面取得了领先于其他所有的商用云计算平台的佳绩。考虑到云计算平台的复杂性,这个时间长度不能不算是迅速。

本章就以 Windows Azure 作为例子来阐述云计算平台的各种相关原理和概念。如果在前面几章的讨论中,读者对云计算的概念还有不少模糊或不解之处,在阅读完本章之后,读者对云计算的理解一定会清晰起来。

5.1 Windows Azure:云的操作系统

根据微软模式和实践团队的资深项目经理 Eugenio Pace 的话,他在 2007 年着手研究将软件作为服务发布的可能性,并在同年晚些时候开发了一个模型,称为北风托管(Northwind Hosting)。该模型演示了 Windows Azure 平台在今天所提供的许多能力。或者说,Northwind 就是 Windows Azure 的前世今生。

但 Windows Azure 到底是什么,不同的人经常会给出不同的解答。有的人说 Windows Azure 是微软的云计算战略,有的人说是微软的云平台,有的人说 Windows Azure 是公有云,有的人说 Windows Azure 是构建在微软数据中心上的云环境。所有这些说法不能说不对,有的说法甚至是完全正确,但却怎么也不能释疑解惑,听了这些说法的人还是不清楚,Windows Azure 具体为何物。因为这些描述都非常抽象,难以落到某个实处,让人感觉摸不到一个抓手,从而难以理解。

而要让读者理解 Windows Azure,就要从非常具体的方面进行阐述,并与传统的、大家

所熟悉的概念进行关联。其实，从 Windows Azure 这个名字来看，读者应该可以猜出来一点名堂，那就是 Windows Azure 似乎是一个操作系统，因为 Windows 不是微软的操作系统的名字吗？微软的操作系统都称为 Windows XX，如 Windows NT/95/98/2000/2008/XP/7/8。根据这个观察，Windows Azure 似乎也应该是一个操作系统。这个判断一点都不错。

而操作系统是什么，学习过操作系统的人都知道：资源的管理者和魔幻师。操作系统扮演的角色是魔幻和管理。因此，如果 Windows Azure 是一个操作系统，则它就是一个魔幻师和管理者，只不过其管理的资源与传统 Windows 操作系统管理的资源有较大的不同，这些资源在性质、构成单元和规模上都不相同。简单来说，Windows Azure 管理的资源包括云体（云基础设施）、计算资源、存储资源、各种配置文件，还有用户的应用程序，如图 5-1 所示。

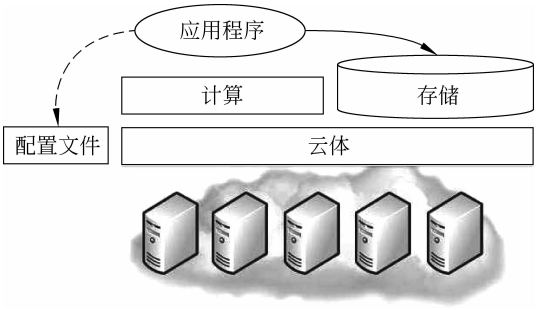

图 5-1 Windows Azure 云计算平台涉及的资源

乍一看，Windows Azure 管理的资源似乎与传统操作系统所管理的资源一样：计算不就是 CPU 吗？存储不就是磁盘吗？云体不就是各种其他计算机硬件码？配置文件在传统操作系统上不也存在吗？是的，从表面上看，Windows Azure 管理的资源和传统操作系统所管理的资源是类似的，但在底下却存在很大的区别。最重要的是 Windows Azure 所管理的资源规模巨大，云体可能由成千上万的服务器组成，计算和存储也是无数服务器计算能力或存储容量的抽象。此外，云体和云计算资源的拥有者不是用户，而是云供应商。俗话说，量变导致质变，这种数量上的变化及拥有者的不同导致 Windows Azure 和传统操作系统的诸多不同，这些不同至少包括如下几点：

（1）Windows Azure 不需要安装，而传统操作系统需要自己安装；

（2）其所用的物理资源由微软公司提供，而不是用户自己购买的台式机或个人电脑；

（3）需要联网才能使用，而传统操作系统无须联网就可使用；

（4）Windows Azure 能够伸缩，传统操作系统几乎不具备真正意义上的伸缩能力；

（5）Windows Azure 是按照用户的用量计费，而传统操作系统必须一次买断。

上述这些不同点在本章的讨论中将逐步显现出来。

5.2 Windows Azure 概览

既然 Windows Azure 是一个操作系统，就意味着可以在其上面运转用户程序。任何可以在 Windows 操作系统上运行的程序都可以在 Windows Azure 上运行。从传统操作系统

环境来推断，要运行应用程序必须有计算能力、存储能力和调度能力。Windows Azure 也不例外，也有这三种能力。

5.2.1　计算能力

Windows Azure 首先具有的是进行计算的能力。这种能力体现在 Windows Azure 里面许多许多的服务器上，这些服务器构成 Windows Azure 的云体(Fabric)，如图 5-2 所示。

对于云体里面的每台服务器来说，一般会被切割成多个虚拟机，每个虚拟机上运行的是传统的操作系统，这个系统在本书写就时是 Windows 2008 操作系统(这点可以根据用户需求发生改变)，如图 5-3 所示。

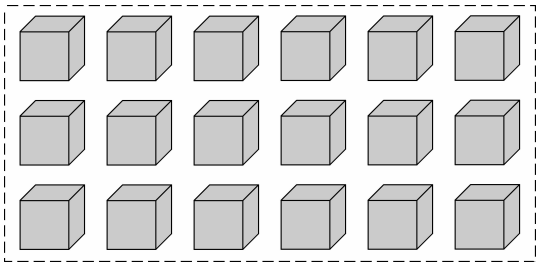

图 5-2　Windows Azure 的云体，由许许多多的服务器所构成(立方体代表服务器)

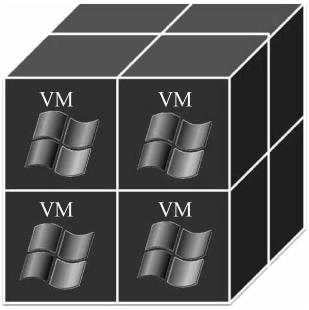

图 5-3　Windows Azure 云体里的每台服务器被切割成多个虚拟机，每个虚拟机上运行着 Windows 2008(当然，也可以运行其他的传统操作系统如 Linux)

众所周知，Windows 2008 上面是能够跑应用程序的。这就是说，用户的云应用程序可以在任意一台虚拟机上运行。不过，与部署在传统操作系统平台上有所不同的是，用户的云应用程序一般不止部署在一台服务器上，而是跑在云体里面的多个服务器上，如图 5-4 所示。

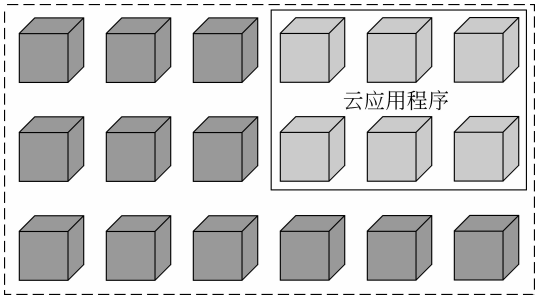

图 5-4　云应用程序通常运行在 Windows Azure 云体里面的多台服务器上

需要指出的是，云应用程序可以用到的服务器数量是没有限制的(指理论上，实际上云供应商通常会对此加以约束)，这里隐含的意思是 Windows Azure 是可以随意扩展的，这种随意扩展的能力正是云计算平台和普通操作系统平台的一个重大不同之处。对于 Windows Azure 来说，其采取的扩展模式是横向扩展，而不是纵向扩展。可以伸缩的云应

用程序则通过无状态连接和持久存储机制来予以实现。

5.2.2 Windows Azure 的持久存储

对于一个完整的应用软件运行环境来说，仅有计算能力是不够的，还需要提供某种数据存储的能力。对于 Windows Azure 来说，这种存储能力由 Windows Azure 存储服务提供。也许“存储服务”这个名词听上去有点晦涩，但实际上很简单，因为 Windows Azure 的存储服务只不过是一个云计算应用程序而已，它也运行在云体上，负责管理云体里面的存储资源，并把它们整合起来提供给应用程序或者用户使用。目前，Windows Azure 存储服务提供三种持久的存储机制，分别是块(Blob)、表(Table)和队列(Queue)，如图 5-5 所示。

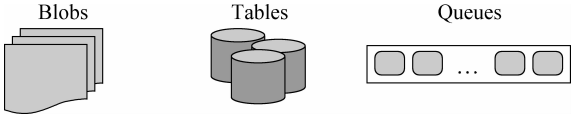

图 5-5 Windows Azure 云计算平台里的持久存储机制

这三种存储机制都是简单且基本的存储抽象。本书前面讲过，Blob 是二进制大数据块的英文缩写，用来存放规模很大的数据实体(如文件流)，表格一般用来保存服务状态(如缓存)，队列则用于服务之间的通信和同步。这三种存储机制都着眼于海量伸缩、高可用性和高持久性，都可以横跨地理位置，并且所有的数据都复制三份，这是 Windows Azure 所内置的数据保护机制。对数据的所有访问均使用 REST API 进行。需要再次强调指出的是，这里所说的表格不是关系数据库的表格，而是所谓的键值对(Key-Value Pair)表格。这里的表格并不支持关系数据库的操作，事实上，这几种存储机制都不支持关系数据库。

5.2.3 Windows Azure 云体控制器

有了运转传统操作系统的虚拟机，有了云体里面的持久存储机制，就有了运行应用程序的基本环境。不过，云应用程序要真正在 Windows Azure 上跑起来，要真正用到 Azure 上面的各种资源，还需要一个关键的组件，这个组件就是云体控制器(Fabric Controller)。云体控制器是 Windows Azure 的大脑，可以看作云操作系统的内核。Windows Azure 里面的所有管理任务都由云体控制器进行自动管理。例如，用户程序对云资源的需求均需要通过云体控制器进行，此外，云体控制器还负责云计算平台的安全、访问授权、资源配置、负载均衡、自动伸缩、应用紧缩、应用启动和关闭、故障排查和容错等一些必要的管理任务。

5.2.4 Windows Azure 的鸟瞰图

将本章到目前为止所讨论过的各种内容合并起来就可以获得一幅 Windows Azure 云计算平台的鸟瞰图，如图 5-6 所示。

在该鸟瞰图中，大量的服务器、存储器及相应的网络设备构成云计算平台的云体。云体控制器自己则在多台服务器上运行，云体控制器所占用的服务器数量通常取决于云体里面的服务器数量。在所有服务器上面都运行一个云体代理，该代理负责相关服务器对云体环境的感应，并为该服务器上所运行的应用程序提供一个访问云体功能的通道。

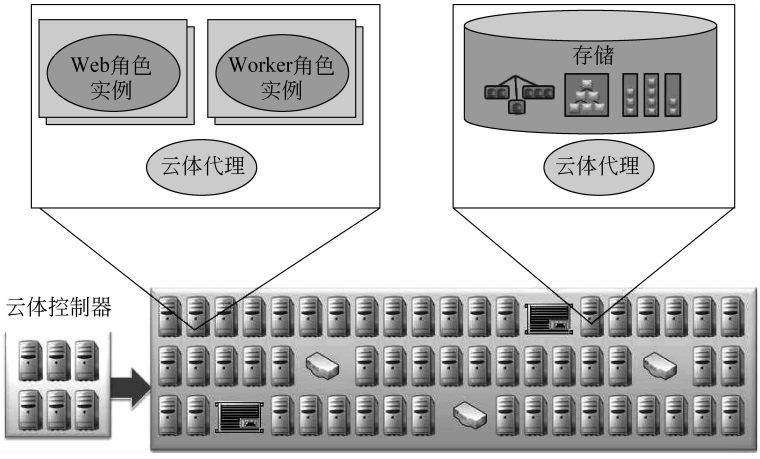

图 5-6　Windows Azure 云计算平台的全景图

5.2.5　Windows Azure 的故障域与升级域

为了避免云计算平台出现单点故障，云体通常被划分为故障域和升级域。故障域是系统失效的基本单元，也就是说，失效的发生是按照单元进行，例如，一个计算节点或者一个机架就是一个失效单元。尽管一个计算节点上可能存在多个虚拟机，但计算节点的失效将导致上面的所有虚拟机都发生失效。机架的电源供应出现问题则可导致机架上的所有机器出现故障。

为了提高应用程序的容错能力，在分配服务角色时需要考虑故障域，不能将所有的角色分配在同一个机架上，当然更不能分配在同一个计算节点上。用户在发布应用程序时，可以向 Windows Azure 指明特定的故障域数量，例如，用户可以要求 Windows Azure 给自己的应用程序分配 10 个 Web 角色，分布在两个不同的故障域上。

升级域则是为了保障在对系统或应用程序进行升级时，应用程序仍然处于持续可用的状态。升级域在本质上是对软件或者配置进行升级的基本单元，如一组节点。在对系统进行升级时，Windows Azure 是针对升级域逐个展开，在一个升级域里面的所有节点都升级成功后，才展开对另一个升级域的升级。开发人员可以在部署应用程序时向 Windows Azure 提出要求，指定应用程序所占用的升级域数量。例如，用户完全可以要求 Windows Azure 为自己的应用程序分配 10 个 Web 角色，分布在 5 个升级域上。

图 5-7 描述的是 Windows Azure 平台上的故障域和升级域。

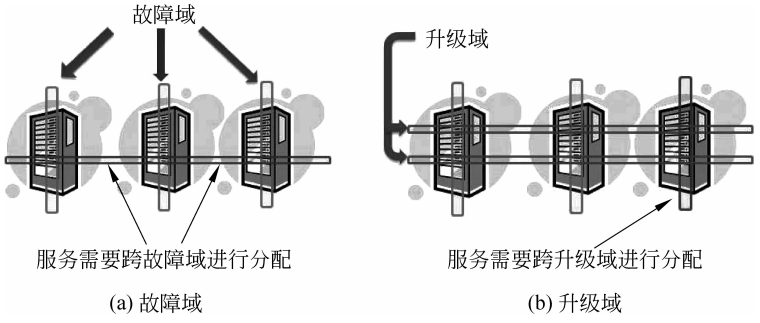

图 5-7　Windows Azure 云平台上的故障域和升级域

在对云计算平台进行运维时，如果需要对虚拟机打安全补丁，也必须逐个进行，在对一台虚拟机打完补丁并确认无误后，才会对下一个虚拟机进行安全补丁。这种逐个打补丁的模式也称为滚动补丁。同理，对虚拟机上的操作系统进行的任何升级也是采用滚动模式进行。

5.2.6 地理分布

云计算平台的一大特点是地理分布广泛，即云体里面的计算节点和存储节点一般分布在不同的地理位置上。Windows Azure 也不例外，Windows Azure 的云体横跨多个地理位置上的多个数据中心。用户在部署应用程序时，可以根据自己的客户聚集地情况指定特定的地理位置，还可以通过创建关联组来将相关的应用程序部署在同一个地理位置上。地理位置指定和关联组的指定均只能通过开发人员的门户进行操作。

5.3 在 Windows Azure 上运行用户程序

有了计算、存储和调度，就具备了运行应用程序的基本能力。那么如何才能在 Windows Azure 上运转一个应用程序呢？这需要多个步骤才能完成。先要开发应用程序，然后是定义自己的服务，再然后是发布应用程序到云端，最后是对云应用程序进行控制。

5.3.1 开发云应用程序

要运转一个云应用程序，当然先得开发一个云应用程序（购买也可以，只不过真正的云应用程序是很难被买断的）。云应用程序并不是简单的传统操作系统上的应用程序，应当是根据具体云平台的特征而有的放矢所开发的软件，这样才能够充分利用云平台的各种不同寻常的能力。在开发云应用时，开发人员的经验非常重要。一开始，没有经验，开发过程会比较周折。但随着对云应用开发过程的熟悉，情况就会好转。对于每个云应用程序来说，其生命周期可以分为三个界限非常分明的阶段：开发阶段、维护阶段、管理阶段。

在开发阶段，开发人员在自己的本地机器上进行开发，但需要下载一个云应用的本地开发模拟器。该模拟器能够在本地模拟一个云环境，并能够与现有的开发工具如 Visual Studio 进行集成。这样，用户就可以在 Visual Studio 里面编码、编译和运行云应用程序。

一旦应用程序通过编译，接下来的工作就是调试，鉴于云端的动态随机特性，调试一般只能在本地的模拟器上进行。但在本地调试得差不多时，还是需要将应用程序部署在云体上运行，进行最后的实战调试。云端调试很受限制，不能在云体里面设置断点和进行跟踪执行，但可以通过云平台的 API 调用获得应用程序运行时的一些日志信息，以协助对程序进行分析。

不管是因为最后的实战调试，还是一切都已经就绪，下一步是将应用程序发布到云端。不过，在发布之前，需要先向云体控制器定义自己的服务。

5.3.2 向云体控制器定义自己的服务

要想将应用程序发布到云端，就需要云体控制器来分配资源和进行调度。而这就需要

告诉云体控制器有关应用程序的信息，有了相关的信息，云体控制器才能决定如何实现用户的请求。对于 Windows Azure 来说，用户需要告诉云体控制器关于应用程序的信息包括三个方面：该服务由哪些角色组成？这些角色之间如何交互？在什么规模上进行交互？

对于 Windows Azure 来说，应用程序里面可以使用的角色为 Web 角色和 Worker 角色。简单来说，Web 角色是应用程序的用户界面部分，用来将服务呈现给用户，用户通过这个界面与应用程序打交道。Worker 角色是应用程序的业务逻辑部分，用来实现实际的工作处理。当然，如果用户愿意，Web 角色里面也可以包括部分甚至全部的业务逻辑。因此，一个 Windows Azure 应用程序必须包含至少一个 Web 角色，但却可以没有 Worker 角色。图 5-8 给出的是 Windows Azure 云应用程序的构成示意图。

图 5-8　Windows Azure 云应用的组成示意

Web 角色和 Worker 角色在运行过程中，必定要对某些数据进行处理，也就是要与存储机制打交道。用户必须告诉云体控制器，它们要与哪些存储实体交互。一般来说，云应用程序的不同角色会通过队列存储机制进行交互和信息沟通。此外，如果一个云应用程序有多个 Web 角色，则需要将这个信息告诉云体控制器，以便 Windows Azure 为该应用程序配备负载平衡器，以便将用户的请求均衡地分布到这多个 Web 角色上。

上述这些信息加上 Windows Azure 所提供的门户、云体控制器和负载平衡器，就形成了云应用程序运行时的典型场景，如图 5-9 所示。

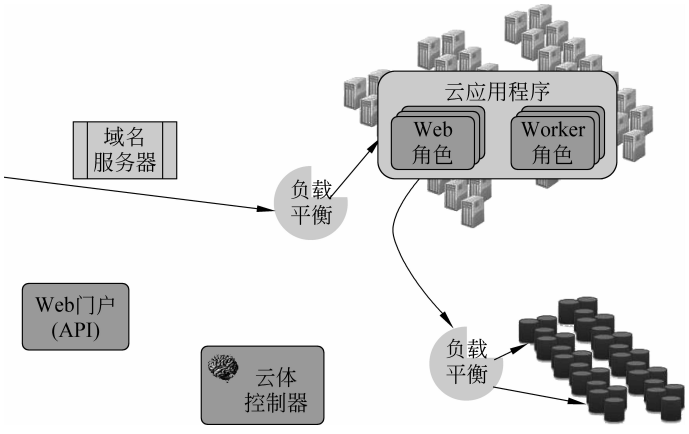

图 5-9　云应用程序在 Windows Azure 云计算平台上运行时的场景

用户告诉云体控制器上述信息的媒介是配置文件。用户在发布应用程序到 Windows Azure 平台上的时候将配置文件一同发布上去。在云应用程序运转的中间还可以上传更新的配置文件来对应用程序的运行态势进行控制。具体的配置文件名称及其语义请参看本作者翻译的微软公司撰写的云计算系列丛书(已由清华大学出版社出版)。

5.3.3 将应用程序发布到云端

定义好软件的各种信息后，就可以发布到云体上运行了。具体的发布方式有三种：可以通过 Windows Azure 提供的 Web 门户来发布，可以通过 Windows Azure 平台的 API 来发布，还可以通过命令行来发布。用户可以根据自己的需要选择发布的方式。

不管采用哪种发布方式，发布应用程序一般需要考虑如下几点。第一点是分配节点，即在哪些节点上运行该应用程序。根据前面讨论过的内容，节点的分配应该横跨故障域和升级域(图 5-10)。第二点是将需要的操作系统放置在分配的节点上。第三点是将角色的程序代码放置在分配的节点上。第四点是对配置文件进行设置，用以对应用程序的执行进行动态控制。第五点是启动角色实例，应用程序开始运行。第六点是对负载平衡器进行配置，使其将用户请求均衡分配到多个 Web 角色中。当然，如果只有 1 个 Web 角色，则不需要进行这步操作。

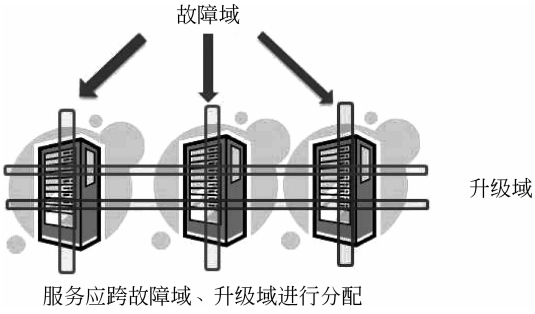

图 5-10 部署应用程序时应尽量横跨故障域和升级域

图 5-11 描述的是通过 Windows Azure 的 Web 门户进行应用发布的场景。

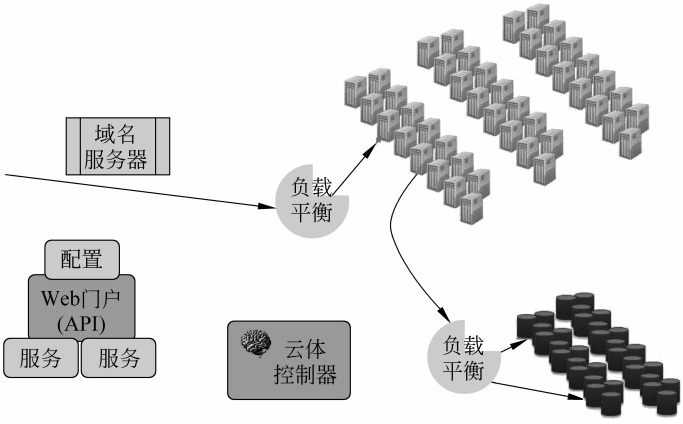

图 5-11 通过 Web 门户发布应用程序：发布之前

在发布之前，需要先将配置文件和应用程序准备好，然后通过 Windows Azure 的 Web 门户将应用程序和配置文件一同上传到 Windows Azure 上，如图 5-12 所示。

注：用户在使用 Windows Azure 之前需要先注册一个账号，之后用经过批准的账号和密码登录 Windows Azure，登录之后即可以见到 Windows Azure 的 Web 门户。本章最后

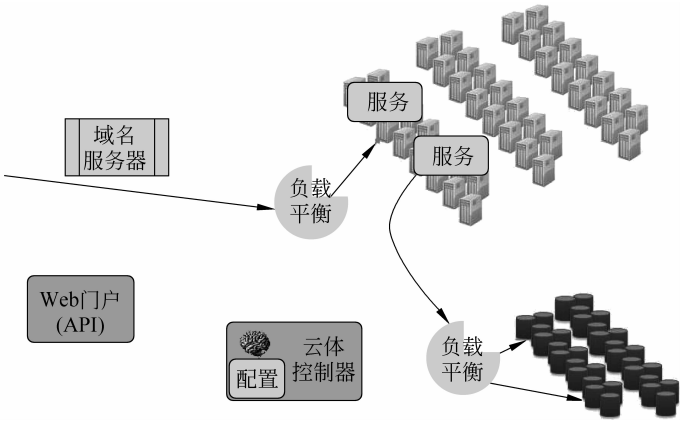

图 5-12　通过 Web 门户发布应用程序：发布之后

一节将对在 Windows Azure 上发布应用程序进行更详细的讨论。

5.3.4　监控和管理云应用程序

与传统平台上的应用程序不同的是，传统平台上的程序在运行过程中，应用程序提供商不会对其进行任何管理，而在云平台上，应用程序的拥有者需要不断监控和管理应用程序的运行，也就是对应用程序进行运维。运维的主要任务是维持足够数量的角色实例，并根据需求的变化来对实例数量进行调整。如果出现角色失效，则需要进行重启，如果出现节点失效，则需要自动分配新的节点并予以启动。具体来说，运维的任务包括对应用程序进行伸缩、对应用功能进行紧缩、对失效的应用实例进行重启或恢复等。

所有运维和管理任务可以通过 Windows Azure 提供的 Web 门户进行，也可以通过 Windows Azure 提供的 RESTful API 来进行，还可以通过命令行命令来进行。

图 5-13 描述的是通过 Web 门户进行应用伸缩的过程示意。

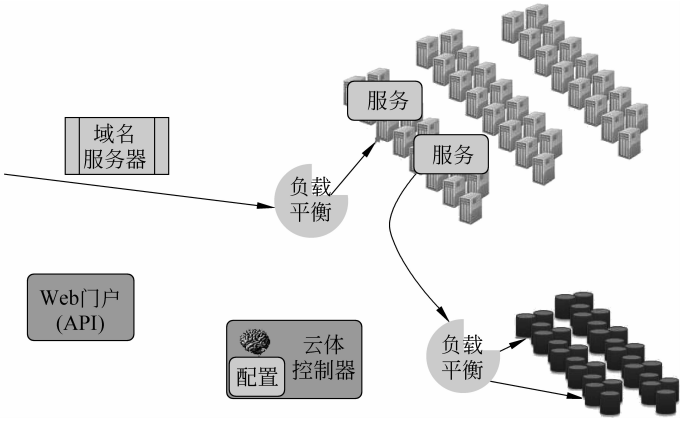

图 5-13　Windows Azure 云平台里面的应用伸缩：伸缩之前，应用程序只有 2 个实例

从图 5-13 可以看出，在进行伸缩之前，应用程序有 2 个实例。在应用程序执行过程中，可能出现用户的需求暴增，对应用进行监控的系统管理员在发现后，通过 Web 门户启动应用伸展，即增加应用程序的运行实例，如图 5-14 所示。

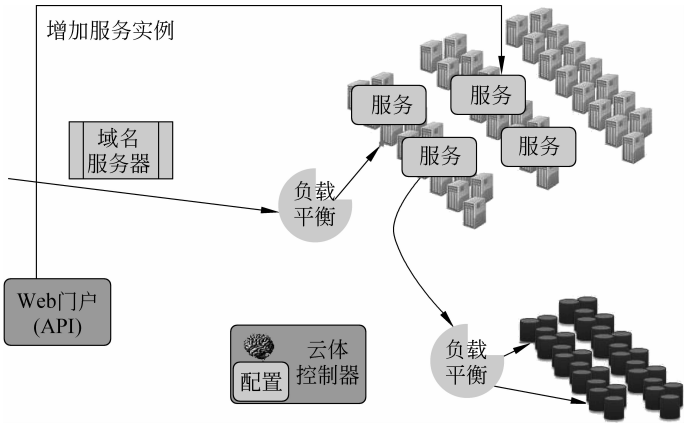

图 5-14 应用伸缩：通过 Web 门户进行伸展后，应用程序实例增加到 4 个

此外，在应用程序的运行过程中，有可能出现因各种原因而导致的服务实例崩溃或其他故障，此时就需要重启该失效的实例或者启动一个新的实例予以替换。而重启或更换实例均可以通过 Web 门户或者 API 来进行。图 5-15 描述的是用新实例来替换故障实例的情形。

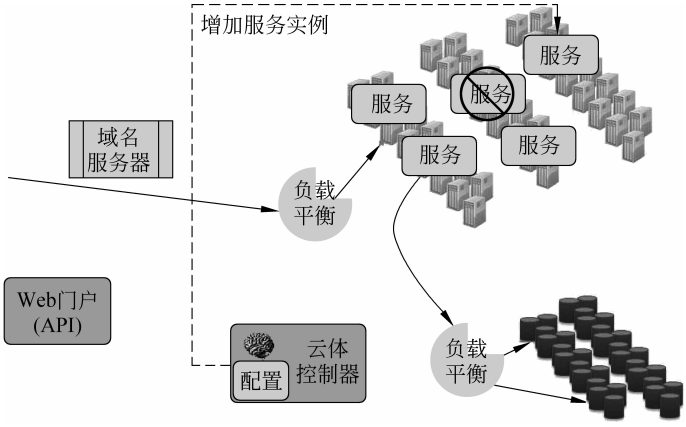

图 5-15 Windows Azure 云平台里面的故障处理：通过 Web 门户替换失效的实例

在图中，应用程序的其中一个实例出现故障（上面有红圈的服务），云体控制器（应该）很快就侦测到这个问题。之后将试图重启有故障的服务实例，只有在重启不成功后，才会启动一个新的实例来予以替换。

5.4 服务隔离和安全

作为多租户架构，云计算平台上一般运行有很多用户的很多应用程序，不同用户和不同应用程序之间的隔离显然非常重要，就如传统操作系统需要隔离不同的用户程序一样。在传统操作系统平台上，用户程序之间的隔离是通过地址翻译机制来实现，在云计算平台上则使用服务模型来进行应用程序之间的隔离。每一个服务（应用程序）都有自己的特定模型，