David Chappell
& Associates

THE WINDOWS AZURE PLATFORM AND ISVS

A GUIDE FOR DECISION MAKERS

DAVID CHAPPELL

JULY 2010

SPONSORED BY MICROSOFT CORPORATION

CONTENTS

ISVS AND CLOUD COMPUTING

Why should an independent software vendor (ISV) care about cloud computing? The answer is simple: Using the cloud has the potential to increase an ISV's revenues and/or decrease its costs. Running code and storing data on computers in large Internet-accessible data centers owned by somebody else can offer compelling advantages. Anyone responsible for charting the course of an ISV ought to be thinking seriously about how cloud computing will affect their business.

One option for an ISV looking to benefit from the cloud is to exploit the Windows Azure platform. A fundamental part of Microsoft's cloud offerings, this group of technologies is designed to support highly scalable and available applications. The goal of this paper is to make clear why cloud computing is important for ISVs and to illustrate how they might use this new platform.

Before diving into the topic, it's worth summarizing up front some of the main ideas. Here are the key points to understand:

- A primary goal of the Windows Azure platform is to be a foundation on which ISVs can create Software as a Service (SaaS) applications. Customers are increasingly interested in having a SaaS option for the software they buy. To satisfy this demand and to keep pace with the competition, many ISVs will choose to offer a SaaS version of their current or future products. Creating a SaaS application requires building a highly scalable, highly available cloud-based service that can be used simultaneously by many customer organizations. Building your own foundation for this makes no more sense than would writing your own operating system for an on-premises application. Just as Windows provides a platform for traditional on-premises applications, the Windows Azure platform can support SaaS applications.

- Cloud computing needn't be an all-or-nothing proposition, and so SaaS applications aren't an ISV's only choice. Another option is to enhance an existing on-premises application with cloud-based functionality, such as running some code or storing a subset of data on the Windows Azure platform. This incremental approach to the cloud can save money and improve a current application's functionality. It can also provide a low-risk way to gain experience with this new kind of technology.

- Cloud platforms aren't useful only for firms that create end-user applications. If you're an ISV that provides infrastructure add-ons or developer aids for the on-premises Windows environment, it's likely that you can also find value-added products to create for the Windows Azure platform. As more computing moves into the cloud, finding these new offerings is likely to be an important way to maintain your revenue stream.

- Cloud platforms are different from traditional hosting. From a technical perspective, the Windows Azure platform provides a programming model expressly designed to create scalable and available applications, as well as simpler administration. The business differences include minimal up-front commitment and easier ways to increase and decrease the computing resources your application uses. These differences mean that the Windows Azure platform can potentially provide better technology and lower costs for ISV applications.

Initially, the Windows Azure platform is likely to be used to support today's applications in the cloud. It's worth pointing out, however, that cloud platforms offer services we haven't seen before, such as access to large numbers of cheap CPUs and massively scalable data storage. Along with support for the world we

already know, we should expect to see creative ISVs finding ways to do wholly new things with this new kind of platform.

A BRIEF OVERVIEW OF THE WINDOWS AZURE PLATFORM

Making good decisions about how to use the Windows Azure platform requires a basic understanding of the platform itself. This section provides an overview of the technology and its associated business model.

TECHNOLOGY

The Windows Azure platform has three components:

- Windows Azure, providing compute and storage services in the cloud.
- SQL Azure, providing a cloud-based relational database.
- Windows Azure AppFabric, providing cloud-based infrastructure services.

Working out how your firm might use the platform requires understanding at least the basics of all three.

Windows Azure

Like the Windows Azure platform as a whole, Windows Azure can be viewed in three parts: a Compute service that runs applications, a Storage service that stores data, and a Fabric that supports the Compute and Storage services. Figure 1 shows this breakdown.

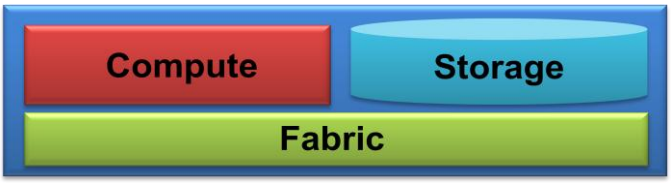

Figure 1: Windows Azure has three main parts: The Compute service, the Storage service, and the Fabric upon which both depend.

To use the Compute service, a developer creates a Windows application. This application might be written using C# and the .NET Framework, using C++, or in some other way. However it's built, the application is implemented as some combination of *Web roles* and/or *Worker roles*. Figure 2 illustrates this idea.

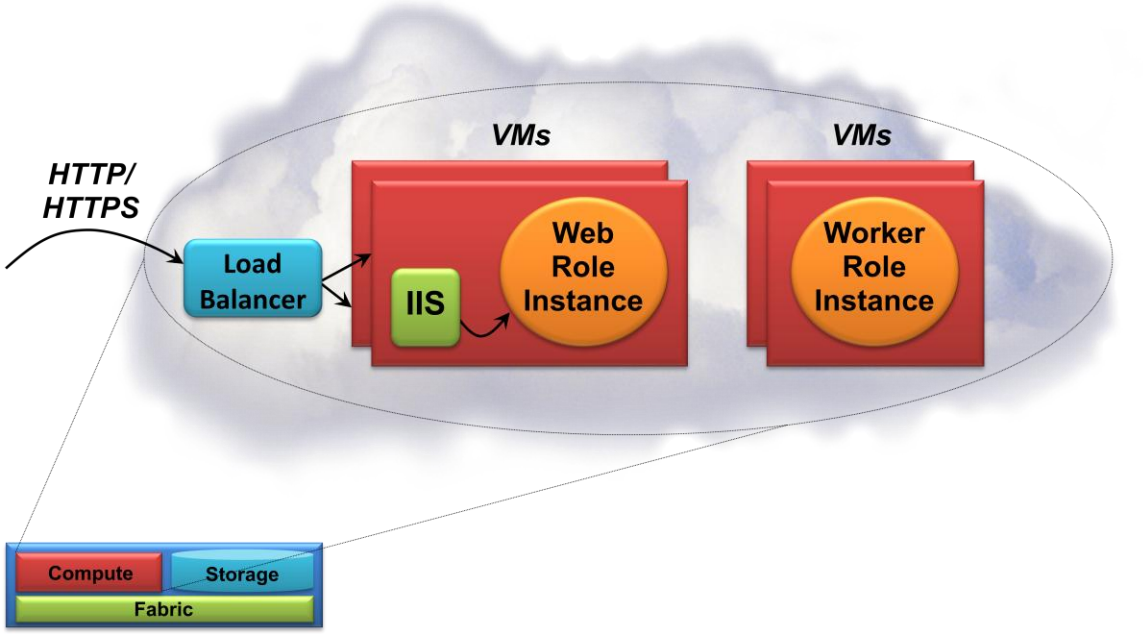

Figure 2: Applications built on the Windows Azure Compute service can consist of Web role instances, Worker role instances, or both.

As the name suggests, a Web role instance accepts Web requests. It can be created using ASP.NET or another technology that works with Internet Information Services (IIS). Whatever technology is used, Windows Azure provides built-in hardware load balancing across all of an application's Web role instances.

For functions that aren't solely intended to respond directly to Web requests, a Windows Azure application can also contain Worker role instances. The main difference between a Web role and a Worker role is that Web roles always have IIS running in them, while Worker roles don't. Among other things, this two-part design allows creating scalable applications where Web role instances accept requests, then pass them to Worker role instances to be processed.

As Figure 2 shows, each instance—Web role or Worker role—runs inside its own virtual machine (VM). This provides isolation, letting Windows Azure applications run with full trust, and it also allows a clear view into application performance, since there's a defined mapping between VMs and processor cores. But a developer doesn't explicitly create VMs. Instead, she uploads an application to Windows Azure, together with an XML configuration file that specifies how many Web role instances and Worker role instances should run. Once this is done, the Windows Azure Fabric creates the required number of VMs, then monitors their execution. If an instance fails, Windows Azure will start a new one, making sure that the specified number of Web role and Worker role instances are always running.

Given that Windows Azure applications are essentially Windows apps, it shouldn't be surprising that developers can create them with Visual Studio. This tool provides templates for creating cloud applications as Web roles, Worker roles, or both. Windows Azure also provides a *Development Fabric*, which is a facsimile of Windows Azure that runs on a local machine. Developers can use this to create their code and do initial testing, then upload the app to Windows Azure when it's ready.

Applications usually need persistent storage, and so Windows Azure provides its own cloud-based mechanisms for storing and retrieving data. The technology offers three storage options, all accessed via standard HTTP GETs, PUTs, and DELETEs. Figure 3 illustrates this.

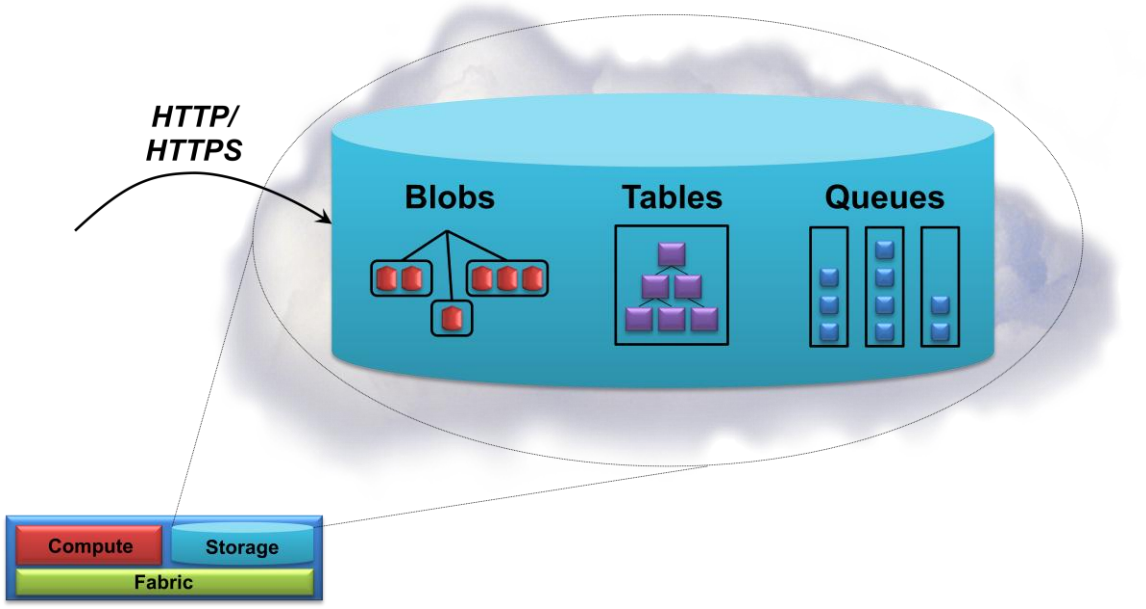

Figure 3: Windows Azure Storage can be accessed by Windows Azure applications or by applications running elsewhere.

The three kinds of Windows Azure storage are:

- Blobs: allow storing large binary objects, such as videos and images. Blobs can also be used as persistent storage for the file system in Web role or Worker role VMs, an option known as Windows Azure drives. The platform also provides a content delivery network (CDN) for blobs, which can speed up access to frequently accessed data.
- Tables: provide highly scalable entity-based storage (not relational tables).
- Queues: allow sending and receiving messages, such as between an application's Web role instances and Worker role instances.

It's important to point out that all three of these can also be accessed by applications that aren't running on the Windows Azure Compute service. For example, an on-premises or hosted application might choose to store large video files as Windows Azure blobs.

SQL Azure

While the storage mechanisms in Windows Azure are certainly useful, most applications today use relational storage. To provide a cloud-based option for this, the Windows Azure platform includes SQL Azure. According to Microsoft, this part of the Windows Azure platform will eventually include a range of functions, including reporting services and more. Today, though, it contains one component, called SQL

Azure Database, that's focused on providing the fundamentals of a relational database. Figure 4 illustrates the idea.

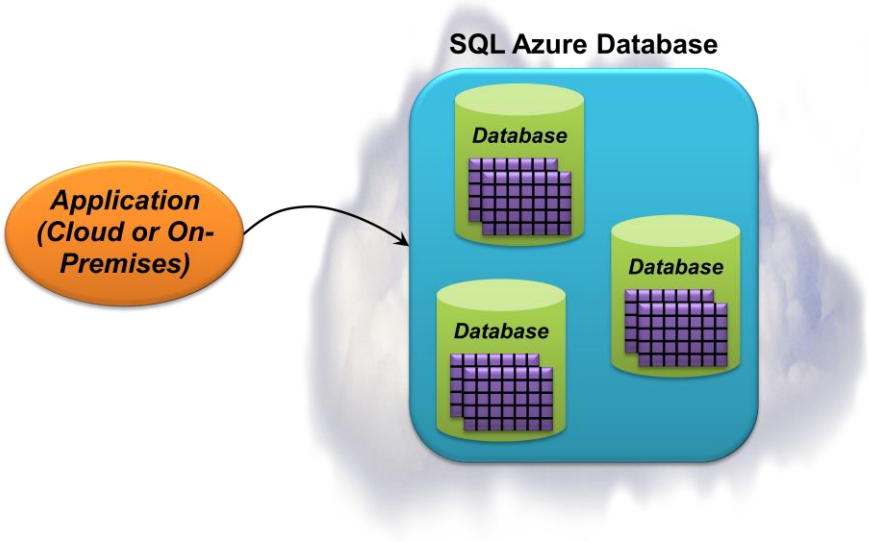

Figure 4: SQL Azure Database provides cloud-based relational storage for applications running on-premises or in the cloud.

An application using SQL Azure Database can run in the cloud, such as on Windows Azure, or on premises. Wherever it runs, the application accesses this cloud service just as it would SQL Server. In fact, SQL Azure Database provides essentially a SQL Server environment in the cloud, complete with indexes, views, stored procedures, triggers, and more. Applications that today access SQL Server locally will largely work unchanged with data in SQL Azure Database. Customers can also use on-premises software such as SQL Server Reporting Services to work with data stored in this cloud-based service.

While an application using SQL Azure Database sees a familiar SQL Server environment, there are some differences. For example, as you'd expect in a shared environment, a query can run for only a limited time—no single request can take up more than a pre-defined amount of resources. Also, a single database can't exceed 50 gigabytes. An application with more data will need to create multiple databases, dividing its data across them.

Windows Azure AppFabric

Along with running applications and storing data, it can also be useful to run some kinds of infrastructure in the cloud. This is exactly what's done with Windows Azure AppFabric. This part of the Windows Azure platform has two components:

- Service Bus: Provides a secure way for an on-premises application to expose services built using Windows Communication Foundation (WCF) to clients outside the firewall. Each exposed service establishes a connection with Service Bus, through which clients can access the service.

- Access Control: Provides identity services in the cloud. Among other things, Access Control is used to control access to Service Bus.

And finally, don't confuse Windows Azure AppFabric with the fabric component of Windows Azure itself. Even though both contain the term "fabric", they're wholly separate technologies addressing quite distinct problems.

BUSINESS MODEL

A primary attraction of a cloud platform, and one of its biggest differences from traditional hosting, is consumption-based pricing. With traditional hosting or an in-house data center, the owner of an application typically pays for a specific set of machines for a fixed amount of time. For applications with variable load, this fixed set must be large enough to handle the highest peak, which means that capacity goes unused at non-peak times.

Cloud platforms aren't like this. With the Windows Azure platform, for example, the owner of an application pays for the compute and storage resources he uses. When the application's load is light, he might request, say, three Web role instances and two Worker role instances. When the load is heavy, he might up his request to six Web role instances and four Worker role instances. In both cases, the application's owner pays only for the resources his application is using.

Here's a summary of standard consumption pricing for the Windows Azure platform. (For more details, see http://www.microsoft.com/windowsazure/offers/popup.aspx?lang=en&locale=en-US&offer=MS-AZR-0003P.)

- For compute services, customers pay $0.12 to $0.96/hour for each Windows Azure role instance depending on the instance's size. Note that this charge is per hour of wall-clock time, not CPU time.

- For storage services, the charge for Windows Azure tables and blobs is $0.15/gigabyte per month, along with an access charge of $0.01/10,000 operations (i.e., GETs and PUTs on the data). The relational storage provided by SQL Azure Database costs $9.99/gigabyte per month, and databases can be purchased in various sizes up to 50 gigabytes.

- Windows Azure AppFabric services are each priced separately. Access Control is $1.99/100,000 transactions, while Service Bus is $3.99/connection per month, with discounts for groups of connections.

- Customers are also charged for bandwidth, defined as all data moved into and out of a Windows Azure datacenter. (There's no charge for accessing data within the same datacenter.) In the Americas and Europe, those charges are $0.10/gigabyte in and $0.15/gigabyte out. In the Asia/Pacific region, the charges are $0.30/gigabyte in and $0.45/gigabyte out

These prices are in US dollars; prices outside the US are those listed here converted into local currency.

Microsoft also provides various discounts, such as a Development Accelerator offer that gives ISVs lower prices for developer use of the Windows Azure platform. And although it's not yet available, Microsoft says that the platform will eventually offer commitment-based pricing. Customers willing to commit to specific minimum usage levels will see their charges reduced.

As with its other offerings, Microsoft also offers Windows Azure platform support through partner programs. There are three main offerings:

- Microsoft Platform Ready: Focused on ISVs, this program provides services specifically designed to help partners build new applications on the Windows Azure platform and port existing applications to this new environment.
- BizSpark: Open solely to software start-ups with annual revenues under $1 million, this program provides MSDN Premium subscriptions to a start-up's developers. Each subscription includes hundreds of hours of free Windows Azure instances, free Windows Azure and SQL Azure Database storage, and more.
- Microsoft Partner Network: Open to any Microsoft partner, this program offers complimentary MSDN Premium subscriptions (including the Windows Azure platform benefits just described) to members in the program's top tiers. Members also get a 5% discount on Windows Azure platform usage (not including charges for storage and data transfer).

All three of these options allow access to the Partner Learning Center, an online portal that provides Windows Azure training for various roles and other information. All three also give partners access to support services, including partner-only forums moderated by Microsoft product engineers and phone support.

USING THE WINDOWS AZURE PLATFORM: SOME OPTIONS FOR ISVS

Working out whether a cloud platform can improve your business requires thinking through how you might use one. Accordingly, this section looks at some of the primary ways in which ISVs can use the Windows Azure platform.

USING CLOUD STORAGE FROM YOUR ON-PREMISES APPLICATION

Perhaps the simplest way an ISV application might use the Windows Azure platform is to store data. As mentioned earlier, both Windows Azure Storage and SQL Azure Database can be accessed from on-premises applications as well as from Windows Azure applications. For example, an application that currently does back-ups to an on-premises storage system might instead choose to use Windows Azure Storage blobs. This can improve backup reliability, since like everything else in Windows Azure Storage, blobs are replicated at least three times. It might also lower costs, given the economies of scale provided by Microsoft's very large data centers. Or think of an application that provides large amounts of data to its users: videos, audio files, or something else. Rather than storing this data locally, the application might choose to use Windows Azure Storage blobs for higher availability, lower costs, or both.

Alternatively, an application might use SQL Azure Database to store relational data in the cloud. For example, an application that needs to share a set of relational tables across multiple instances running in different locations might benefit from having that data accessible in one place. Much of the work required to administer a database goes away with SQL Azure Database, since Microsoft does it for you. The availability of your data also goes up, since like Windows Azure Storage, SQL Azure Database stores three copies of all data to protect against hardware failures. The cost of this storage might even be lower than on-premises storage, especially for smaller organizations.

Using Microsoft's cloud platform to store your data requires trusting that platform. The best way to build trust is to start small, then if warranted, expand from that base. An ISV that's considering using Windows Azure or SQL Azure Database might well find that using these storage services from an on-premises application lets them gain experience with this new approach before making a bigger commitment. It can be a good way to dip your toe into the water of cloud computing.

COMBINING CLOUD COMPUTING WITH YOUR ON-PREMISES APPLICATION

If putting an application's data in the cloud can make sense, so can using the cloud to run some of your app's code. For example, think of an ISV with an application that could benefit from creating an on-line marketplace among all of the companies that use this solution. Building this functionality on a cloud platform such as Windows Azure should be significantly faster and cheaper than building it entirely from scratch. Or suppose an on-premises application could sometimes benefit from more processor cores to run CPU-intensive loads. This application could create a number of Windows Azure Worker role instances to do this work, then shut them down when they're no longer needed. As always, the customer would be charged only for the resources they use, i.e., the hours these Worker role instances were actually running.

There are plenty of situations in which moving an existing application entirely into the cloud isn't practical. Porting millions of lines of code to a cloud platform might be too risky, too expensive, or just not worthwhile. In cases like this, adding new functionality that runs in the cloud can make more sense. Because Windows Azure provides a pre-built platform for running cloud applications, it can make creating this kind of code easier.

CREATING A SAAS VERSION OF YOUR APPLICATION

Being an ISV has long meant installing software directly on your customers' machines. With SaaS applications, this is no longer true. Unlike conventional packaged software, a SaaS application runs in an Internet-accessible data center, and it's typically delivered to customers via the Web.

SaaS applications can offer real benefits for customers, including the following:

- Less risk: Unlike conventional packaged software, SaaS applications needn't require a large up-front investment. Instead, customers can typically try the application for no charge before purchasing it, letting them have more confidence that the application will provide business value.
- A more attractive price structure: SaaS applications commonly provide usage-based pricing, such as a charge per user per month. This lets customers start small, then add users as needed. It also allows replacing a capital expense—buying software—with an operating expense, which can be a draw for some organizations.
- Faster and cheaper deployment: Rather than installing software on local machines, the users of a SaaS application typically access the app via an ordinary Web browser.
- Easier upgrades: Instead of upgrading its own copies of a purchased software package, the customer of a SaaS application can rely on the SaaS provider to upgrade the application centrally.

The advantages of SaaS are certainly appealing for some customers and some kinds of applications. This approach also has drawbacks, however, and so it's not right for every application. The biggest challenges that SaaS customers face include:

- Trust: Can a customer really trust the provider of a SaaS application? Will the application always be available? Can the application provider be trusted with sensitive corporate data? Trust in the provider is the single biggest barrier that most customers face with SaaS applications. It's worth pointing out that this trust must also extend to the cloud platform a SaaS application is built on.

- Regulatory and compliance issues: Many businesses are required by governments to conform to a variety of standards, such as Sarbanes-Oxley, Basel II, and others. If a SaaS application (and the cloud platform it's built on) can't meet these obligations, the business can't use it.

- Customization: Traditional on-premises software packages can usually be customized in various ways. SaaS applications, by contrast, are typically *multi-tenant*, which means that a single copy of the software is shared by all users. While customization is still possible, it's often more limited than with traditional packaged software.

- Integration with on-premises applications: A SaaS application must provide some way to integrate its code and data with on-premises application. This includes identity integration, since customers typically want single sign-on.

- Management: Most on-premises tools for managing and monitoring applications don't work well with SaaS applications today. While this will likely change, the situation today is off-putting for some customers.

Just as SaaS applications have pros and cons for organizations that buy software, they also offer advantages and disadvantages to companies that sell software. The benefits of SaaS applications for ISVs include:

- The potential for more sales: Since customers face less risk and a smaller up-front financial commitment, making initial sales can be faster and cheaper. SaaS applications can also be attractive to new categories of customers, such as smaller organizations, because they require less in-house IT expertise.

- Easier customer upgrades: Rather than convincing each customer to replace an on-premises package, an ISV can upgrade all users of its SaaS application at once. This can significantly reduce an ISV's support costs, since there's no longer a requirement to support many old versions of an app.

Nothing is free, however, and so the move to a SaaS world also presents some drawbacks for ISVs. Some of the most visible are the following:

- A different sales and revenue model: Money trickles in rather than coming in up-front license fees. At least initially, this may mean lower margins.

- Less customer lock-in: Without the sunk costs of a purchased and perhaps extensively customized on-premises application, customers might find it easier to switch to a competitor.

- Less services revenue from customization: ISVs that derive a significant share of their income from customization services might find this revenue reduced for SaaS applications.

- Technical challenges: Creating a SaaS application requires a different technical skill set from the one most ISVs already have. Rather than creating familiar on-premises software, your development organization must now build a highly scalable and highly available shared application.

The truth is evident: SaaS applications have a significant role to play, but they won't obliterate on-premises software. Every ISV that provides a packaged application must look at its offerings and make a decision: What makes sense to provide as SaaS? Especially if your competitors are going down this path, creating a SaaS version of a current on-premises application might be the right option.

As mentioned earlier, a primary goal of the Windows Azure platform is to support SaaS applications. Many aspects of its design reflect this goal. For example, a SaaS application must be more scalable than an on-premises application because it will support multiple customers simultaneously. The Web role/Worker role division in Windows Azure is intended to help create massively scalable applications by dividing work between a Web front end and Worker back end. Similarly, Windows Azure Storage tables are expressly designed to hold very large amounts of data, much more than can be managed by a single relational database management system.

A SaaS application must also be more reliable than an on-premises application, since a failure affects all customers, not just one. By monitoring every running instance of an application, the Windows Azure Fabric can help achieve this. If an instance or a machine fails, the Fabric will restart another one to take its place. The Fabric also allows upgrading a running application without shutting it down, an important service for a SaaS app that must be continuously available.

One more important attribute for an effective SaaS application is elasticity, i.e., the ability to handle peaks in demand. With a conventional application, a data center must be able to support the maximum load this application will ever see. This is certainly possible, but it's expensive and wasteful—most of the data center's capacity will probably lie unused most of the time. With Windows Azure, however, addressing this problem is simpler. Recall that the owner of an application can change the number of running instances on the fly, relying on the Fabric Controller to create or shut down VMs as needed. Since Windows Azure customers are charged only for the resources they use, this lets them pay for a large number of computing resources only when those resources are really needed. When the load diminishes, the application can shrink back to its normal size.

Still, it's worth pointing out that the Windows Azure platform isn't identical to the Windows Server environment. For example, code running in Web roles and Worker roles must run in user mode—admin mode isn't allowed. Because of this, some functions that are possible on-premises aren't available in the cloud. Also, while Windows Azure load balances requests across an application's Web role instances, it doesn't support session affinity. This allows better scalability, but it might not be the way your application is currently designed. The point is that moving an application to the Windows Azure platform may well require some changes to its code.

SaaS applications have different requirements from traditional on-premises applications. Supporting these differences is a big part of why Windows Azure is designed the way it is. An ISV that creates a SaaS application is free to build its own platform, and originally some did—there was no alternative. Yet with

the rise of cloud platforms, ISVs creating SaaS applications can now focus on their business logic instead of infrastructure.

PROVIDING SUPPORTING SERVICES FOR CLOUD PLATFORMS

Many ISVs today provide infrastructure and management services for on-premises environments. Many others provide software that makes life easier for developers. While some of these solutions aren't relevant in the cloud, others are. And new opportunities exist as well, places where ISVs can make money by adding value to the Windows Azure platform.

For example, Windows Azure provides access to performance data about running applications. A tool that aggregated this data, then presented it through an effective user interface could help Windows Azure customers manage their cloud applications. Windows Azure also provides APIs that let an application change the number of running Web role instances and Worker role instances on the fly rather than relying on a person to do this. The platform does not, however, provide software that monitors the application's load, then uses these APIs to adjust the number of running instances accordingly. An ISV might fill this hole, providing code that lets developers easily add this behavior to their applications.

The advent of cloud platforms is a big change for ISVs. With any big change, it often makes sense to do things a piece at a time, such as using the Windows Azure platform's cloud storage or offloading a part of your app to the cloud. If this works well—and makes financial sense—you can then move on to bigger steps when they're justified, such as creating a full SaaS version of your application.

COMPARING THE WINDOWS AZURE PLATFORM WITH ALTERNATIVES

The Windows Azure platform doesn't exist in a vacuum—there are other approaches. This section compares this platform with two of its most obvious alternatives: traditional hosting and cloud platforms that offer VMs on demand.

TRADITIONAL HOSTING

The first stop for most people looking for an outsourced place to run their applications is a hosting provider. In traditional hosting, a customer requests a fixed set of resources and commits to pay for those resources for a defined period of time. For example, an ISV wishing to run a SaaS application might contract with a hoster to provider six Windows servers for a year, paying a pre-defined amount for this service.

Hosting has plenty of advantages. Using a hoster is frequently cheaper than running an in-house data center, especially for smaller organizations. It also lets the customer avoid the complexity of running its own data center while still having total control over the machines it's using. While the advent of cloud platforms will likely cut into the traditional hosting business, this model isn't going away—it still makes sense in many situations.

Yet at least for some applications, the Windows Azure platform is a better choice. The advantages include:

- The ability to quickly increase the number of servers in use: While a traditional hoster might take days to make a new machine available, a Windows Azure application can get a new instance up and running in minutes.

- The ability to quickly decrease the number of servers in use: Hosters commonly require a commitment to a fixed set of servers that are provisioned just for you. With Windows Azure, an application can reduce the number of VMs it's using—and thus the cost of running this application—by decreasing the number of Web role and/or Worker role instances. There's no up-front commitment to a minimum number of servers.

- The ability to provide services explicitly designed for highly scalable, highly available applications: Hosters generally provide standard Windows systems, leaving it up to their customers to do whatever else is necessary to run their applications successfully. As described earlier, the Windows Azure platform is explicitly designed to support applications with very high scalability and availability requirements.

- Less administrative overhead: Unlike Windows Azure, hosters commonly give customers full administrative access to their machines. The trade-off is that more administrative work is required, everything from patching operating systems to managing database management systems. With the Windows Azure platform, most of this work is done for you, saving time and money.

VMS ON DEMAND

A number of vendors, including Amazon, Rackspace, GoGrid, and others, offer virtual machines on demand. Unlike traditional hosters, these vendors typically provide usage-based charging with no required commitment and rapidly available VMs. In other words, they provide cloud platforms.

Windows Azure is also a cloud platform, but even though it uses (and charges via) VMs, it differs in important ways from platforms that offer VMs on demand. With a purely VM-based platform, the situation is in some ways much like hosting: You have complete control, including administrative access to your VMs, but you also bear full responsibility for configuring and managing those VMs and the software they contain. With Windows Azure, you supply only a Windows application, along with instructions about how many instances to run. The platform itself takes care of everything else, including updating system software when required.

Another important difference is in how relational data is handled. With typical VM-based platforms, you can run a relational database in a VM, just as you'd run the same database on premises or at a hoster. This certainly works, but it requires installing, maintaining, and administering this database yourself. Ensuring reliability can also be challenging, since typical shared-disk clusters often aren't possible. In the Windows Azure platform, an application can instead use SQL Azure Database. As described earlier, this technology provides a Microsoft-managed relational store that writes all data multiple times for reliability. Once again, you lose the ability to have total control but gain simplicity and built-in reliability.

CONCLUSIONS

Like all new platforms, the Windows Azure platform will succeed only if ISVs choose to build applications on it. Microsoft clearly understands this, and so making their new cloud platform attractive to this audience is a priority. The core attractions are these:

- Because the Windows Azure platform lets ISVs run applications and store data in a very large data center while paying only for the resources used, it can provide appealing economics.

- By providing a ready-made foundation designed to support scalable and reliable cloud applications, the Windows Azure platform reduces the time and money required to create and run SaaS applications and other cloud-based code.

Cloud computing looks like the next great wave in our industry. Just as ISVs have had to adapt to the changes brought by PCs, mobile devices, and other new platforms, they now need to decide how to exploit cloud platforms. And just as Windows played a significant part in those earlier shifts, the Windows Azure platform is poised to take an important role in this new world. If you're responsible for charting your firm's path, understanding and evaluating this new environment makes good sense.

FOR MORE INFORMATION

Microsoft Partner Network: https://partner.microsoft.com/global/partner

BizSpark: http://www.microsoft.com/bizspark/

Microsoft Platform Ready: http://www.microsoftplatformready.com

Partner Learning Center: https://partner.microsoft.com/global/productssolutions/40111488

ABOUT THE AUTHOR

David Chappell is Principal of Chappell & Associates (www.davidchappell.com) in San Francisco, California. Through his speaking, writing, and consulting, he helps people around the world understand, use, and make better decisions about new technologies.