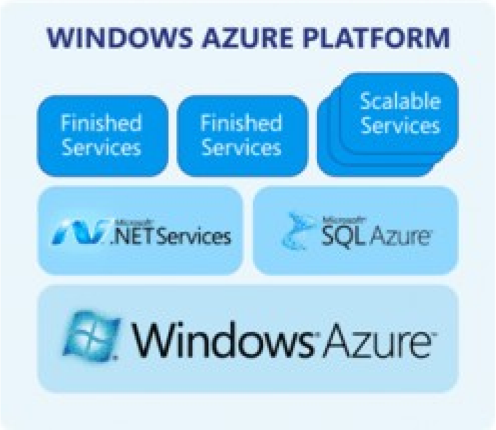
Introduction to Windows AZURE

visit http://worldims.weebly.com



For enterprise architects and developers, the Windows Azure platform does cause a major change in the accepted way of architecting, developing, and deploying software services.
Software development today typically consists one or more of the following types of applications:

- Rich client and Internet applications: Examples are Windows Client, Windows Presentation Foundation, and Silverlight.
- Web services and applications: Examples are ASP.NET, ASP.NET Web Services, and Windows Communications Foundation.
- Server applications: Examples are Windows Services, WCF, message queuing, and database development.
- Mobile application: Examples are .NET Compact Framework and placeholder.

Most of these application types are on-premise enterprise applications or consumer applications hosted in data centers. The Windows Azure platform adds a new cloud services type to the list. During planning and architecture phases of a project, architects can now choose to include the Windows Azure platform in the overall service architecture.

The Windows Azure platform supports development and deployment of different types of applications and services, not only in the cloud but also on-premise. It is a collection of building blocks of platform, middleware, enterprise, and consumer services for developers to build cloud services. It provides developers with a cloud operating system called Windows Azure, a cloud database called SQL Azure, infrastructure middleware component called .NET Services and a consumer services component called Live Services. Developers can either build services that span across all these components, or pick and choose the components as needed by the service architecture. The overall concept of Windows Azure platform is to offer developers the flexibility to plug in to the cloud environment as per the architectural requirements of the service.

Development tools like Visual Studio .NET have matured enough in the past decade to increase developer productivity several folds in application design, development and testing, but there has been little improvement in the infrastructural challenges involved in deploying distributed applications.

Organizations frequently require months of planning and coordinated efforts between multiple internal groups like Directory Services, Database Management, Platform Services and Security for deploying distributed enterprise applications. Ultimately, organizations end up spending more time and resources in coordinating activities across multiple groups than the delivery of the application itself.
Windows Azure platform readily provides internet scale infrastructure for deploying distributed applications and services. You can develop a cloud service in Visual Studio .NET and deploy it into the Azure cloud right from on-premise tools. This frees up critical project resources to focus on solution design and delivery instead of managing internal infrastructure dependencies.

Windows Azure Platform Overview

The Windows Azure platform is an end-to-end development and deployment platform for building cloud services. Each component of the Windows Azure platform is designed to provide a specific functionality to cloud services. In this section, we will look at the high-level architecture of the Windows Azure platform.

Windows Azure Platform Architecture

The Windows Azure Platform consists of three main components – Windows Azure, SQL Azure, and .AppFabric. . Figure 1 shows a simple illustration of the four Windows Azure platform components.

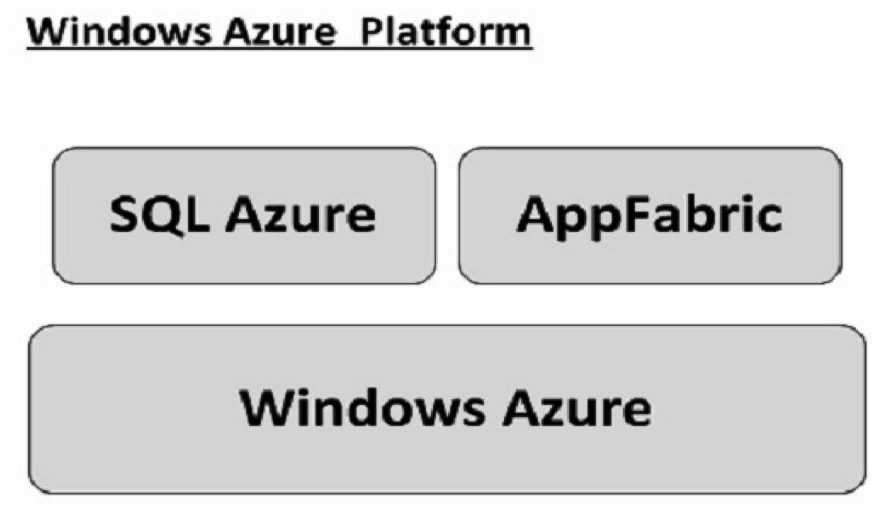


Figure 1: Windows Azure Platform

The Windows Azure Operating System

Windows Azure is the underlying operating system for running your cloud services on the Windows Azure platform. Microsoft brands Windows Azure as the Operating System in the cloud, because it provides all the necessary features for hosting your services in the cloud. It provides a runtime environment that includes a web server, computational services, basic storage, queues, management services, and load-balancers. Windows Azure also provides developers with a local development fabric for building and testing services before they are deployed to Windows Azure in the cloud. Figure 2 illustrates the three core services of Windows Azure.

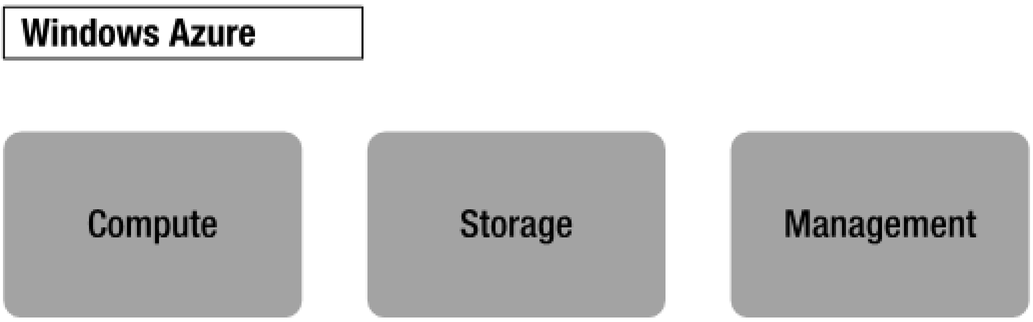


Figure 2: Windows Azure Core Services

The three core services of Windows Azure are as follows:

Compute: The compute service offers scalable hosting of services on 64-bit Windows Server 2008 platform with Hyper-V support. The platform is virtualized and designed to scale dynamically based on demand. The platform runs Internet Information Server (IIS) version 7 enabled for ASP.NET Web applications. The abstraction is at the operating system layer. Developers can write managed and unmanaged services for hosting in the Windows Azure Compute cloud without worrying about the underlying operating systems infrastructure.

Storage: There are three types of storage supported in Windows Azure: tables, blobs, and queues. These storage types support REST-based direct access through REST APIs. Windows Azure tables are not traditional relational database tables like SQL Server tables. Instead, they provide structured data storage capabilities. They have independent data model popularly known as the entity model. Tables are designed for storing terabytes of highly available data like user profiles in a high-volume ecommerce site. Windows Azure blobs are designed to store large sets of binary data like videos, images, and music in the cloud. The maximum allowable size per blob item is 50GB. Windows Azure queues are the asynchronous communication channels for connecting between services and applications not only in Windows Azure but also from on-premise applications. You can also use queues to communicate across multiple Windows Azure role instances. The queue infrastructure is designed to support unlimited number of messages, but the maximum size of each message cannot exceed 8KB. Any account with access to storage can access tables, blobs, and queues.

Management: The management service supports automated infrastructure and service management capabilities to Windows Azure cloud services. These capabilities include automatic commissioning of virtual machines and deploying services in them, as well as configuring switches, access routers, and load balancers for maintaining the user defined state of the service. The management services consist of a fabric controller responsible for maintaining the health of the service. The fabric controller abstracts the underlying virtualized platform infrastructure from the compute and storage services. The fabric controller supports dynamic upgrade of services without incurring any downtime or degradation. Windows Azure management service also supports custom logging and tracing and service usage monitoring.

SQL Azure

SQL Azure is the relational database in the Windows Azure platform. It provides core relational database management system (RDBMS) capabilities as a service, and it is built on the core SQL Server product code base. In the current version (CTP), developers can access SQL Azure using tabular data stream (TDS), which is the standard mechanism for accessing on-premise SQL Server instances through SQL client today. The SQL client can be any client, like ADO.NET, LINQ, ODBC, JDBC, ADO.NET Entity Framework, or ADO.NET Data Services.

The core services offered by SQL Azure are as follows: (see figure 3)

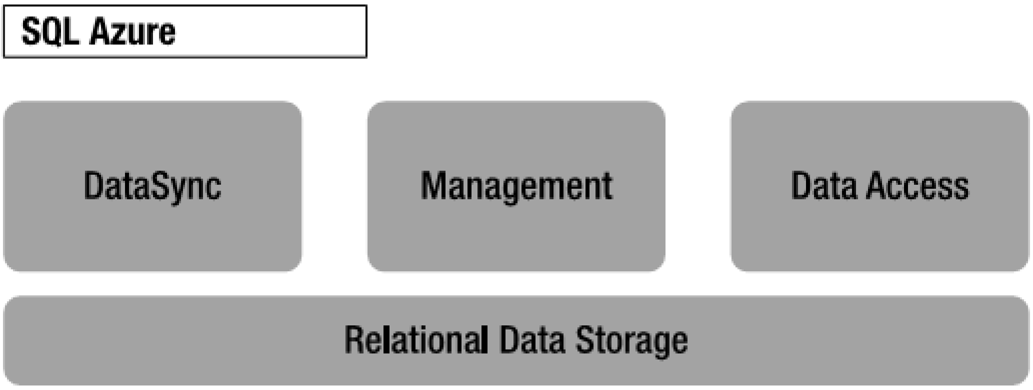


Figure 3: SQL Azure Core components

Relational Data Storage: The relational data storage engine is the backbone of the SQL Azure and is based on the core SQL Server code base. This component exposes the traditional SQL Server capabilities like the tables, indexes, views, stored procedures, and triggers.

Data Sync: The Data Sync capabilities provide the synchronization and aggregation of data to and from SQL Azure to enterprise, workstations, partners and consumers devices using the Microsoft Sync Framework.

Management: The management component provides automatic provisioning, metering, billing, load-balancing, failover and security capabilities to SQL Azure. Depending on the SLA, each database is replicated to one primary and two secondary servers. In case of a failover, the switching between the primary and the secondary server is automatic without interruptions.

Data Access: The Data Access component defines different methods of accessing SQL Azure programmatically. Currently, SQL Azure will support Tabular Data Stream (TDS), which includes ADO.NET, Entity Framework, ADO.NET Data Services, ODBC, JDBC, and LINQ clients. Developers can access SQL Azure either directly from on-premise applications or through cloud services deployed in Windows Azure. You can also locate a Windows Azure compute cluster and a SQL Azure instance together for faster data access.

.NET Services

.NET Services is the middleware engine of Windows Azure platform providing access control service and service bus.

AppFabrichas a service-oriented architecture and allows the creation of federated access control and distributed messaging across clouds and enterprises. I consider AppFabricto be the integration backbone of the Windows Azure platform, because it provides connectivity and messaging capabilities among distributed applications. It also provides capabilities for integrating applications and business processes not only between cloud services but also between cloud services and onpremise applications.

AppFabricalso provides a development environment integrated into Visual Studio .NET 2008 SP1 and beyond. Developers can build WCF-like services in Visual Studio .NET and publish endpoints to the cloud from within Visual Studio .NET design environment. Figure 4 illustrates the two core services of .NET Services.

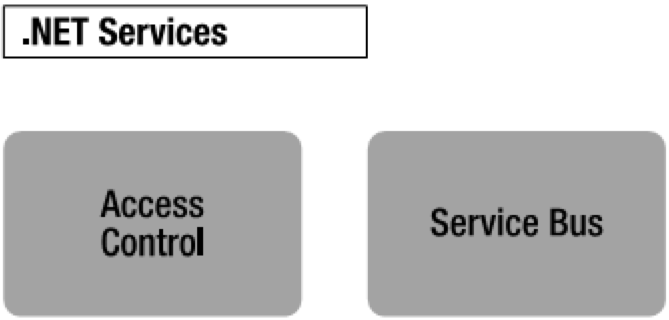


Figure 4: AppFabriccore services

The two core services of AppFabricare as follows:
Access Control: The access control component provides rules-driven, claims based access control for distributed applications. You could define claims based rules and authorization roles in the cloud for accessing on-premise as well as cloud services.

Service bus: The service bus is a generic .NET Internet service bus. It is analogous to the Enterprise Service Bus (ESB) popularly seen in large enterprises. Unlike the ESB, the AppFabricServiceBus is designed for Internet scale and messaging with cross-enterprise and cross-cloud scenarios in mind. The service bus provides key messaging patterns like publish/subscribe, point to-point, and queues for message exchanges across distributed applications in the cloud as well as on-premise.

Windows Azure Platform for Developers

Windows Azure, SQL Azure, .NET Services, and Live Services all have separate software development kits (SDKs), but Visual Studio .NET and the .NET Framework are the common programming tools used for building applications for all the Windows Azure components. Windows Azure SDK and the Live Framework SDK have local development fabrics that emulate the cloud environment at a miniature scale. Developers can utilize their existing .NET development skills for developing services for Windows Azure platform. In this section.

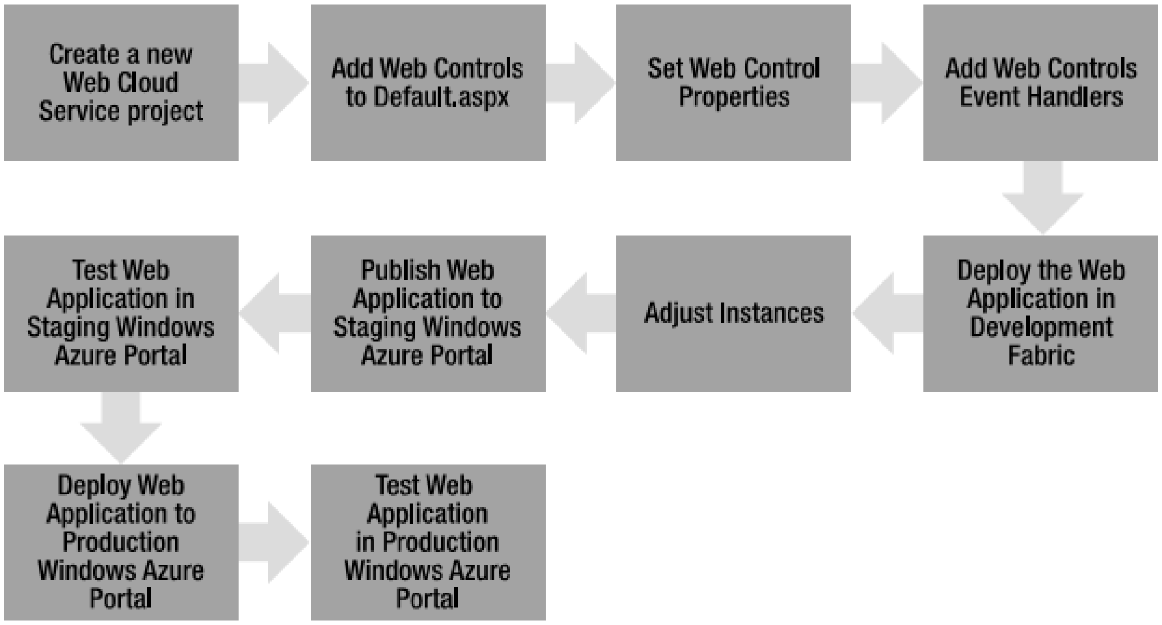
Developer Roles The types of developers that may be interested in developing Windows Azure applications follow:

Enterprise developers: Enterprise developers typically work either in the IT or Business departments of an enterprise. They are responsible for developing applications that improve business productivity. The business requirements come directly from the business groups within the enterprise or from strategic IT initiatives. Enterprise developers can then create cloud services on Windows Azure platform for consumption either by the local business units or cross-enterprise business units.

ISV developers: Independent software vendors (ISVs) develop business solutions on an existing platform. ISVs are more focused in vertical markets like financials, manufacturing, oil, gas, and healthcare. ISVs can leverage the Windows Azure platform for deploying services that are targeted for multiple customers in vertical markets. They can design multitenant architectures on top of Windows Azure platform, specifically dedicated for vertical industries they specialize in.

Consumer developers: Consumer developers work for online service companies like MSN Live, Yahoo, Apple, Google, Facebook, and MySpace, and they offer software services like mail, collaboration, social networking, and mobile services directly to the consumers. Consumer developers can build consumer services on Windows Azure platform and offer them to consumers around in the world.

Developer WorkFlow


Create a new Web Cloud Service project
Add Web Controls to Default.aspx
Set Web Control Properties
Add Web Controls Event Handlers
Deploy the Web Application in Development Fabric
Adjust Instances
Publish Web Application to Staging Windows Azure Portal
Test Web Application in Staging Windows Azure Portal
Deploy Web Application to Production Windows Azure Portal
Test Web Application in Production Windows Azure Portal

The developer workflow illustrates the steps required to develop, build, and deploy a simple web role cloud service to the cloud.
I have divided the remainder of this example into two separate sections: “Developing the Service” and “Deploying the Service.” In the development section, you will develop and test your service locally in the Windows Azure development fabric.

More about Architecture.

Windows Azure is the operating system that manages not only servers but also services. Under the hood, Windows Azure runs on 64-bit Windows Server 2008 R2 operating systems with Hyper V support. You can think of Windows Azure as a virtual operating system composed of multiple virtualized servers running on massively scalable but abstracted hardware. The abstraction between the Windows Azure core services and the hardware is managed by Fabric Controller. Fabric Controller manages en-to-end automation of Windows Azure services, from hardware provisioning to maintaining service availability.
Fabric Controller reads the configuration information of your services and adjusts the deployment profile accordingly. Figure 5 illustrates the role of Fabric Controller in Windows Azure architecture.

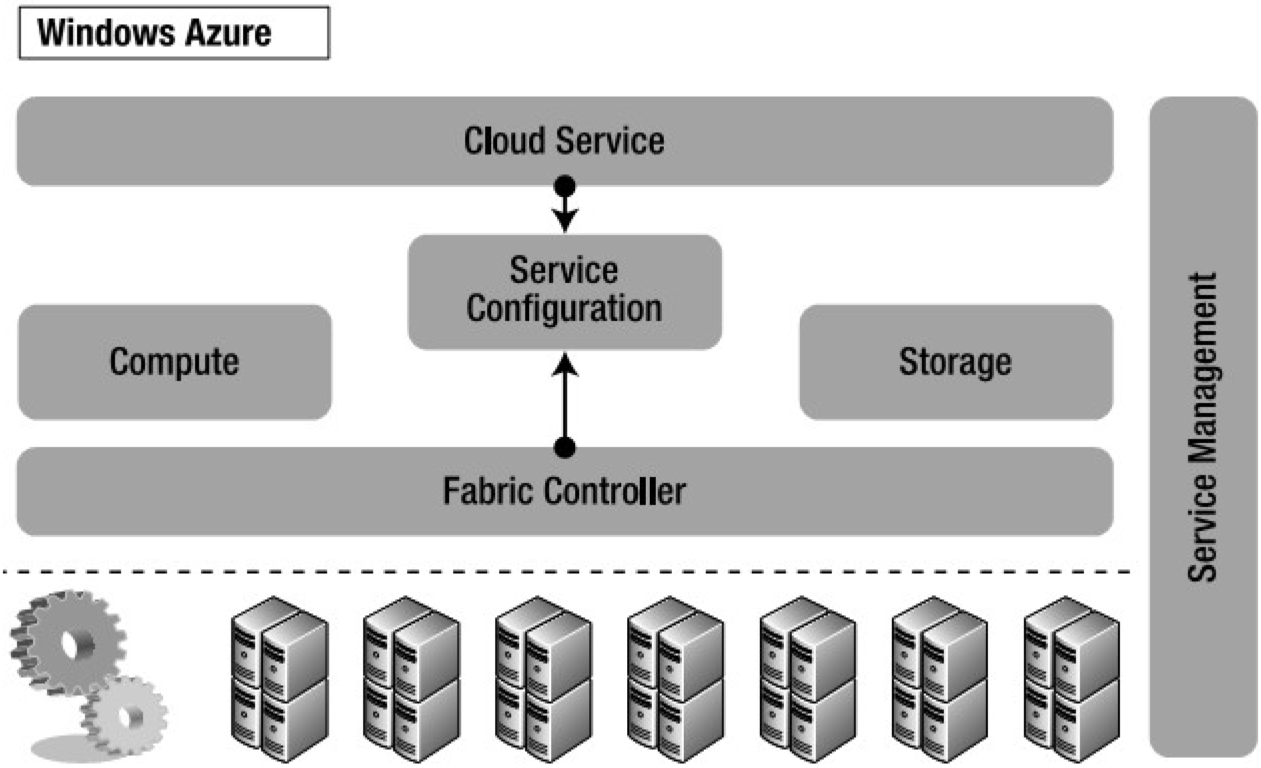
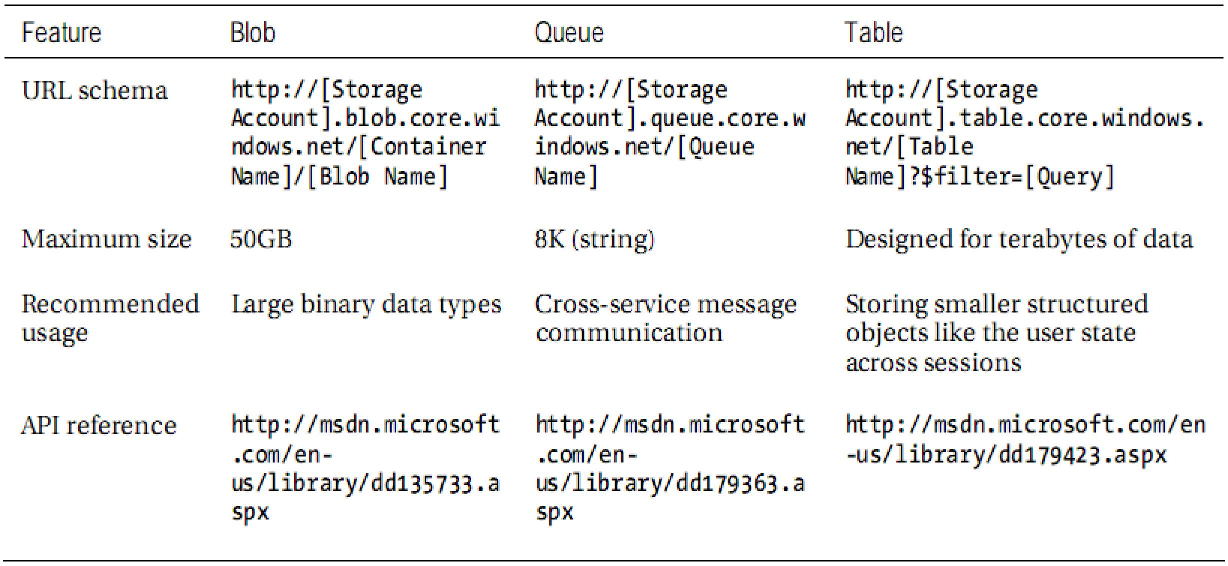


Figure 5: Fabric Controller

Windows Azure is designed to be massively scalable and available. In Figure 5, the Fabric Controller reads the service configuration information provided by the cloud service and accordingly spawns the server virtual machines required to deploy the cloud service. The deployment of cloud services and spawning of virtual instances of servers are transparent to the developer. The developer just sees the status of the cloud service deployment on the Windows Azure developer portal. Once the cloud service is deployed, it is managed entirely by Windows Azure. You just have to specify the end state of the cloud service in its configuration file, and Windows Azure will provision the necessary hardware and software to achieve it. Deployment, scalability, availability, upgrades, and hardware server configurations are managed by Windows Azure for the cloud service.

We saw that Windows Azure consists of three main services: Compute, Storage, and Management. The Compute service provides scalable hosting for IIS web applications and .NET background processes. The web application role is called the Web role, and the background process role is called the Worker role. The Worker role is analogous to Windows Services and is designed specifically for background processing. A Windows Azure cloud service comprises of a Web role and/or a Worker role and service definition of the service.

The Storage service in Windows Azure supports three types of services: blobs, queues, and tables. these storage types support local as well as direct access through a REST API. Figure 6 illustrates the commonalities and differences among the three storage types in Windows Azure.



Feature	Blob	Queue	Table
URL schema	http://[Storage Account].blob.core.windows.net/[Container Name]/[Blob Name]	http://[Storage Account].queue.core.windows.net/[Queue Name]	http://[Storage Account].table.core.windows.net/[Table Name]?$filter=[Query]
Maximum size	50GB	8K (string)	Designed for terabytes of data
Recommended usage	Large binary data types	Cross-service message communication	Storing smaller structured objects like the user state across sessions
API reference	http://msdn.microsoft.com/en-us/library/dd135733.aspx	http://msdn.microsoft.com/en-us/library/dd179363.aspx	http://msdn.microsoft.com/en-us/library/dd179423.aspx

Figure 6: Windows Azure Storage

Even though the Storage service makes it easy for Windows Azure cloud services to store data within the cloud, you can also access data directly from client applications using the REST API. For example, you could write a music storage application that uploads all your MP3 files from you client machine to the blob storage, completely bypassing the Windows Azure Compute service. Compute and Storage services can be used independently of each other in Windows Azure.
The Management service offers the features offered by Windows Azure developer portal as REST API's. So, you can manage your applications and storage in Windows Azure dynamically by calling the vice Management API over REST interface.

Figure 7 illustrates the Windows Azure architecture.

In this Figure, Compute and Storage services run as independent services in Windows Azure. The Web and Worker roles run in the Compute Service, and the blob, queue and table services run in the Storage service of Windows Azure. The Fabric Controller abstracts the underlying infrastructure components like virtualized servers, network components, DNS, and load balancers from Compute and Storage services. When a request from the Internet comes for a Windows Azure Web role application, it passes through the load balancer to the Web role of the Compute Service. If a request for a Storage service comes in, it passes through the load balancer to the appropriate Storage service component. Even when a Web or Worker role wants to communicate with the Storage service, it has to use the same REST APIs that other client applications use. Finally, the Compute and Storage services can be managed by the Service Management API.
Let’s consider an example of your own media storage system in the cloud. In the past decade, there has been a data explosion due to exponential rise in the amount of digital assets in an individual’s life.

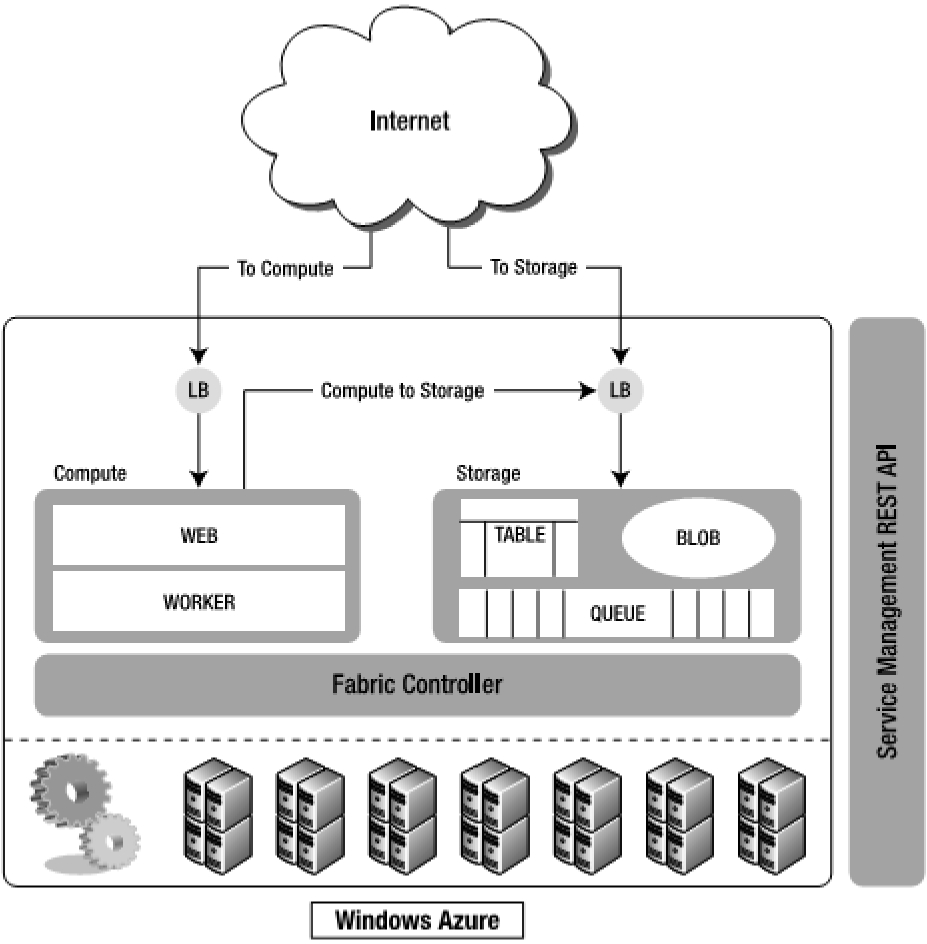


Figure 7: Windows Azure

These assets are in the form of music, video, pictures, and documents. Individuals face the challenge of storing all this content in one place locally. I personally have three hard drives with a combined capacity of 1TB, and I have almost 700GB full. Web sites like Flickr.com and Shutterfly.com can help manage pictures, and sites like YouTube.com and MSN Videos can manage videos. But what if you want a personal hard drive in the cloud with some backup capabilities and functionality so that you don't have to maintain terabytes of digital assets in your house or scattered over multiple web sites. Maybe you would also like to access these assets from anywhere you are. To resolve the digital asset storage problem, you could build a media storage service for yourself on Windows Azure as shown in Figure 8 .

Figure 8 is a solution specific illustration of Figure 7. The Web role is a web application interface for viewing and managing the digital assets stored in Storage services. It also provides some application services like uploading, deleting, updating, and listing of digital assets stored in the Storage service. The Web role application also provides a built-in Silverlight Media Player for accessing the digital assets from your client machine or mobile phone browsers. All the digital assets are stored in the Storage service as blobs. The Worker service does background processing of indexing and cleaning up the digital assets on a periodic basis. Note that this service does not use either tables or queues because the service does not need them.

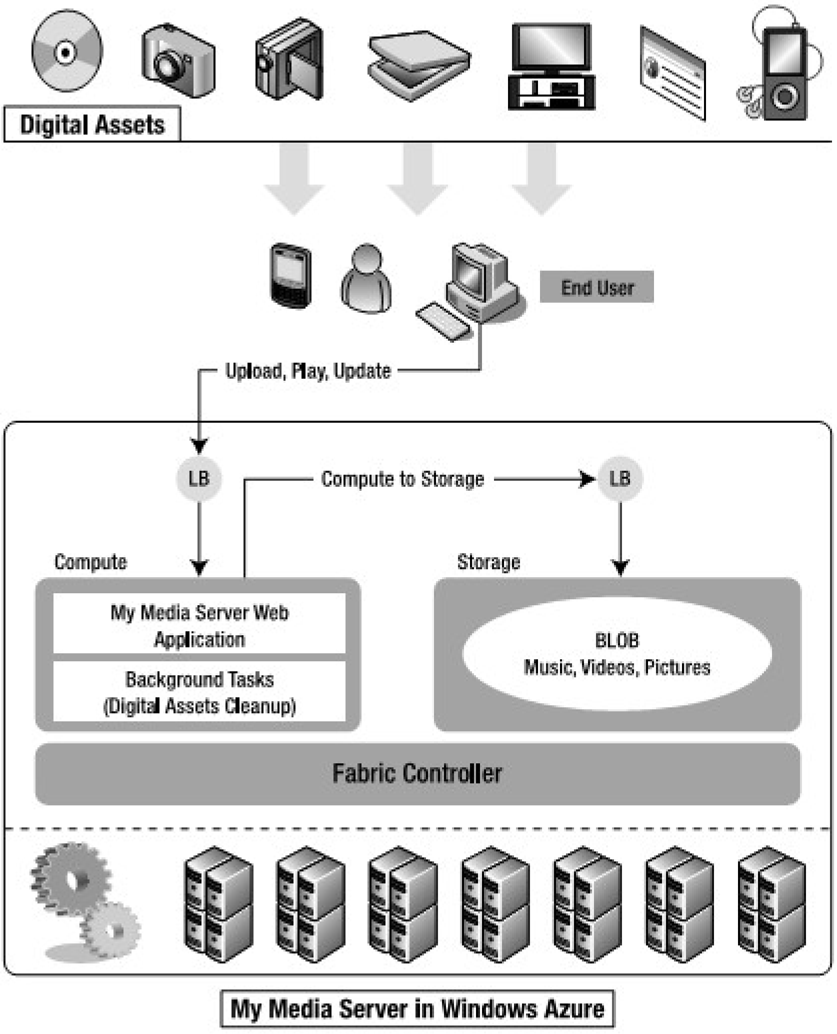


Figure 8: Windows Azure Media Server

Again, the three core services Compute, Storage, and Management combined form the Windows Azure cloud operating system. All the three services abstract the underlying hardware and the operating system infrastructure required for deploying applications in the cloud. The Compute service provides Web and Worker roles that enable running of web and background process applications respectively in Windows Azure. The Storage service offers blob storage, queuing, and table storage capabilities for storing any kind of files, messages, and structured storage in the cloud respectively. The service management interface provides management capabilities to all of your Windows Azure deployments through a single interface. From an architect's perspective, Windows Azure provides most of the features required for designing distributed applications in the cloud.

For more information visit the site http://worldims.weebly.com or send me a mail to dansebus@gmail.com