Cloud computing with the Azure platform

Jouni Mäenpää
Helsinki University of Technology
jouni.maenpaa@tkk.fi

Abstract

Software industry is heading towards centralized computing. Due to this trend data and programs are being taken away from traditional desktop PCs and placed in compute clouds instead. Compute clouds are enormous server farms packed with computing power and storage space accessible through the Internet. Instead of having to manage one's own infrastructure to run applications, server time and storage space can can be bought from an external service provider. From the customers' point of view the benefit behind this idea is to be able to dynamically adjust computing power up or down to meet the demand for that power at a particular moment. This kind of flexibility not only ensures that no costs are incurred by excess processing capacity, but also enables hardware infrastructure to scale up with business growth. Because of growing interest in taking advantage of cloud computing a number of service providers are working on providing cloud services. As stated in [7], Amazon, Salerforce.com and Google are examples of firms that already have working solutions on the market. Recently also Microsoft released a preview version of its cloud platform called the Azure. Early adopters can test the platform and development tools free of charge.[2, 3, 4]

The main purpose of this paper is to shed light on the internals of Microsoft's Azure platform. In addition to examining how Azure platform works, the benefits of Azure platform are explored. The most important benefit in Microsoft's solution is that it resembles existing Windows environment a lot. Developers can use the same application programming interfaces (APIs) and development tools they are already used to. The second benefit is that migrating applications to cloud is easy. This partially stems from the fact that Azure's services can be exploited by an application whether it is run locally or in the cloud.

KEYWORDS: Azure platform, cloud computing, centralized computing, SaaS

1 Introduction to cloud computing

A company deploying an Internet service needs to invest huge amounts of money upfront on infrastructure needed to serve possible users. Also estimating the success of the service can be hard - what is the correct amount of network bandwith, computation capacity and storage space? However, instead of trying to answer these questions why not move infrastructure costs directly into operating costs - pay only for what is needed. Cloud computing is the ultimate solution to the problem just described. Rather than buying, installing, and operating its own system, an organization can rely on a cloud provider to do this for them. As computation keeps centralizing the economy of scale enable cloud providers to keep pricing competitive.[3, 5]

There isn't a single definition of cloud computing. Partly because cloud computing means different things to different people. For some, as Hakan Erdogmus [2] states in recent IEEE software magazine, "cloud computing can mean everything new, cool, and trendy on the Web". Others think cloud computing as scalable Internet-based IT-services and resources. Examples of such resources are computing time and data storage. Whatever the exact definition, one feature is common to all such new technologies - a shift in the geography of computation. [2]

Although traditional software dominates the market and is nowhere near disappearing, the focus of innovation seems to be targeting on centralized computing. The cloud model together with the possibility to offer software to customers as a service (SaaS) clearly paints a new kind of future for software companies. This change is about to affect not only software companies, but all levels of the IT ecosystem, from casual users of software to hardware manufacturers and software developers. End users will see the change in the form of simplified terminal equipment. Also, software is no longer owned or bought in a traditional way. Instead, it is payed for on a basis of use. Developers, on the other hand, will feel the change as increasing complexity in parallel and distributed backend systems. [3, 5]

In the following chapters this paper discusses a particular cloud computing system, namely Microsoft's Azure Services Platform. To begin with, an overview of this platform is given. After that, the components of Azure platform are described thoroughly. To complete the big picture, a simple walkthrough on implementing a sample solution follows. The paper ends with a recap on the advantages and disadvantages involved in cloud computing.

2 The Azure Platform overview

Broadly speaking, the main function of an application platform is to provide developer-accessible services for creating applications. In a traditional situation where applications are hosted on-premises, an application platform includes components such as a development framework and database. This kind of model can easily be extended to the distributed cloud computing world. To exploit computing power and resources located in computing clouds, a cloud application platform is required.

Microsoft's Azure platform is a group of cloud technologies, currently under development and being engineered to enable usage of massive computing power packed within Microsoft's data centers around the world. As stated in [1], these technologies are grouped into four main components of the Azure Platform:

- *Windows Azure*: A Windows based cloud computing operating environment for running applications and storing their data in Microsoft's data centers. Cloud applications can be managed through a browser-accessible portal.

- *.NET Services*: Cloud-based services to address infrastructure challenges in creating distributed applications. The .NET component includes services for access control, Internet accessible application deployment and easy application composition with the aid of workflows.

- *SQL Services*: A data storage facility for many kinds of data. Although the name would suggest a relational interface, it isn't. Data is to be queried with the REST- or the SOAP-protocol.

- *Live Services*: An access to data from Microsoft's Live applications. This data can optionally be synchronized between clients automatically.

Each technology under the Azure umbrella provides a specific set of services to application developers. Regardless of these new technologies, cloud application development with the Azure platform is still very much like the traditional Windows application development. Microsoft even provides ready-to-use templates and necessary plugins for the Visual Studio development environment to make existing software developers feel at home. In a Community Technology Preview (CTP) version of Windows Azure, made public in the fall of 2008, it's only possible to run applications built on the .NET framework. However, there are plans to support unmanaged code, i.e. non-Microsoft based technology, in form of PHP and Python.

In addition to making it possible to run applications in Microsoft's server farms, the Azure platform enables not only cloud applications, but also on-premises applications to use cloud services. Azure's services are offered through industry standard SOAP, REST and XML protocols, thus using them won't be a problem whatever the operating system used. The fact that services provided by the Azure platform can be used both by applications running in the cloud and by applications running on local systems makes the Azure platform a very flexible solution to use. The common code base and versatile possibilities of using Azure's services paves the way for easy migration of existing local applications to clouds. The overal structure of the Azure Platform is illustrated in Figure 1. [1]

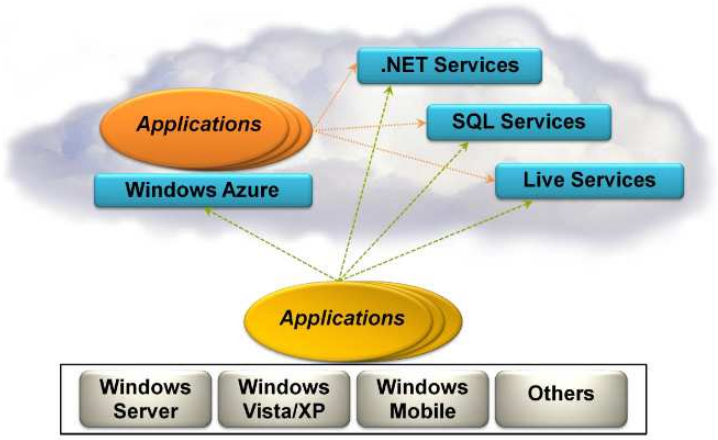

Figure 1: Azure Platform overview. [1]

3 The components of the Azure platform

3.1 Windows Azure

Windows Azure is a Windows based cloud computing operating system that is run on a large number machines. All of these machines are located in Microsoft's data centers around the world and accessible via the Internet. A component called Azure Fabric is constructed on top of these machines. The goal of the Fabric is to put together the enormous distributed processing power and show it as a unified whole to the layers above. Windows Azure's two main services compute and storage are built on top of the Azure Fabric. The architecture just described is illustrated in Figure 2. To support efficient management each application running in Windows Azure has a configuration file. By changing information in this file, the owner of the application can control various aspects of that application's behaviour. The owner can for example change the number of physical processor cores dedicated to the application. [1]

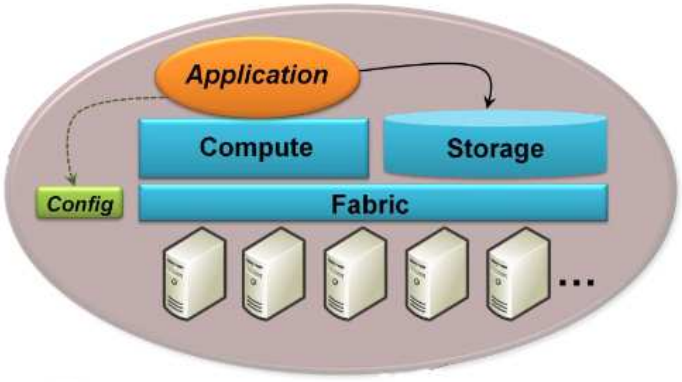

Figure 2: Windows Azure architecture. [1]

As already mentioned, the primary function of Windows Azure is to run applications. A running application typically has multiple instances. Each instance includes a copy of all

or a part of the application's code and is executed in its own virtual machine (VM). These VMs each run 64bit Windows Server 2008 (see Figure 3). To guarantee the performance of an application, each VM and the instance running in it gets a dedicated physical processor core. To meet the demand of processing power during well known peak load times, additional instances of the application can be initiated. Efficient switching between configured processor cores is handled by Windows Azure component called the Load balancer. [1]

Each running application instance is either a Web role instance or a Worker role instance. The difference between them is that only Web role instances can accept incoming HTTP requests via a Internet Information Services (IIS) web server. Each VM running a Web role instance contains its own IIS server (see Figure 3). A Worker role instance, by contrast, gets input via a queue from Windows Azure storage. The Worker role instances are typically created for running batch jobs and can exist indefinitely. Whether an application runs a Web role instance or a Worker role instance, each VM must include a Windows Azure Agent to interact with the Azure Fabric.[1]

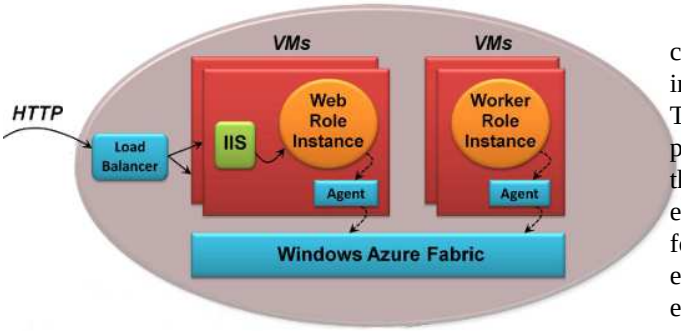

Figure 3: Windows Azure in detail. [1]

The second function of Windows Azure is to address the storage requirements of applications. Windows Azure's data storage has three different classes of data containers: blobs, tables and queues. These containers are all accessed in RESTful style via the HTTP protocol. Using blobs as a container type is the simplest, unstructured way to store and retrieve data. Blobs are suitable for example for single files. To allow applications to work with data in a more fine-grained way, Windows Azure provides tables. A table isn't a traditional relational table, but rather a structured set of typed variables. The third option for storage in Windows Azure, namely queues, has a quite different purpose. Queues primarily provide a way for Web role instances to communicate with Worker role instances. What makes Windows Azure's data storage special is that to achieve fault tolerance all data held in Windows Azure isn't replicated only twice, but three times. The system also guarantees consistency of stored data. An other important feature in Windows Azure storage is that all data is available for non-cloud applications, as well.[1]

3.2 .NET Services

The second component of the Azure Platform is .NET services. This service package shouldn't be confused with Microsoft's .NET framework. The purpose of the .NET services is to address common infrastructure challenges in creating distributed applications. The .NET services component can further be devided into following subcomponents:

- *Access Control Service*
- *Service Bus*
- *Workflow Service*

The Access Control Service is a security token service (STS) running in the cloud. The purpose of an STS is to act as a layer between an application and its users. The Access Control Service deals with tokens defined using the Security Assertion Markup Language (SAML). A security token contain claims each of which carry a piece of information about the user. With the aid of rules set by application's owner these claims are transformed to a form required by the application. Later on a token can be analyzed to determine what the user is allowed to do. Again, all communication with the STS relies on standard procols.[1]

In addition to Access Contol Service, .NET services include the Service Bus. The Service Bus addresses challenges in discovering and exposing service endpoints in the Internet. To begin, an application needs to register one or more endpoints with the Service Bus. The Service Bus then exposes them on application's behalf. To support service discovery, each organization (registered user) is assigned a root Uniform Resource Identifier (URI), below which any naming hierarchy can be created. Along with making communication easier, the Service Bus can also improve security. Because clients now see only an IP address provided by the Service Bus, there is no need to expose any IP address from within organization's network. The actual application using the Service Bus can even reside behind Network Address Translated (NAT) private network.[1]

The last subcomponent of the .NET services is the Workflow Service. The Workflow Service allows creating workflow-based applications running in the cloud. At the core of Workflow Service is an idea that every workflow can be implemented using some number of base activities. These activities are then tied together with some logic. Communication between hosted applications can be carried out with Hypertext Transfer Protocol (HTTP) or using the Service Bus.[1]

3.3 SQL Services

The goal of SQL Services is to provide a set of cloud-based services for storing and working with many kinds of data. The CTP version of Microsoft's Azure platform only includes a subset of SQL Services, namely SQL Data Services. The SQL Data Services provides a database in the cloud. It lets on-premises and cloud applications to access data on Microsoft's servers housed in data centers. A database in the cloud can be attractive for many reasons. Those reasons include for example: savings in infrastructure costs, lack of

management and backup burden, and reliability guarantees. [1]

Unlike the Windows Azure storage service, SQL Data Services is built on Microsoft SQL Server. Nonetheless, the service doesn't expose a traditional relational interface. Instead, SQL Data Services provides a hierarchical data model. At the highest level the database is divided into authorities, units of geo-location. Each authority is stored in a specific data center. An authority holds containers that act as a unit of replication. Containers are also used for load balancing and availability. At the lowest level data items are stored in properties each of which have its own name, type, and value. To query this data applications can either use SOAP or RESTful approarch. Although SQL Services still lack traditional relational database, Microsoft has announced plans to come up with new features to fill this gap. [1]

3.4 Live Services

Live Services is a group of services wrapped together to allow access to data stored and used in varius ways by Microsoft's Live application family. Live applications include such well known software as Windows Live Messenger. In addition to being able to access Live Services data, application can also rely on the Live Framework to synchronize this data across different parties involved. At the core of synchtonization is an idea of being part of a mesh. Once joined to a mesh, Live Services data is automatically synchronized between joined parties.[1]

The fundamental component in the Live Framework is the Live Operating Environment. This component runs in the cloud and applications use it for data access. Yet again accessing is done through a standard protocol, namely HTTP. There also exist higher level toolkits implemented on top of the HTTP foundation.[1]

4 Creating a sample application for the cloud

The CTP version of Azure platform only supports .NET applications, but Python and PHP support is also planned. Creating a .NET application is easy with Microsoft's Visual Studio (VS) Integrated Development Environment (IDE). To get application development started a programmer needs to install Windows Azure Software Development Kit (SDK) and Windows Azure tools for Microsoft Visual Studio. The Windows Azure toolkit extends the IDE with support for Azure projects.[6]

After having the required software installed, the IDE is initialized with a new solution of the type Cloud Service. Depending on the feature requirements of the application, the solution is setup with one project for each role needed (Web / Worker). In addition to projects for different role implementations, the solution has to have a general cloud project. The cloud project contains XML formatted configuration files that define the model for the Windows Azure solution. An example of configuration option that can be controlled is the number of running instances per role.

Coding a test application for the cloud showing a simple hello message is as easy as adding an ASP.NET label to a preformatted solution template inside the Web role project. The outcome of the solution can be debugged and tested straight from the IDE. The Windows Azure SDK includes a component called Windows Azure Development Fabric for simulating the Azure environment (devfabric, see Figure 4). Once devfabric is run, applications hosted by it can be connnected through a certain local port. The devfabric has a user interface for debugging.

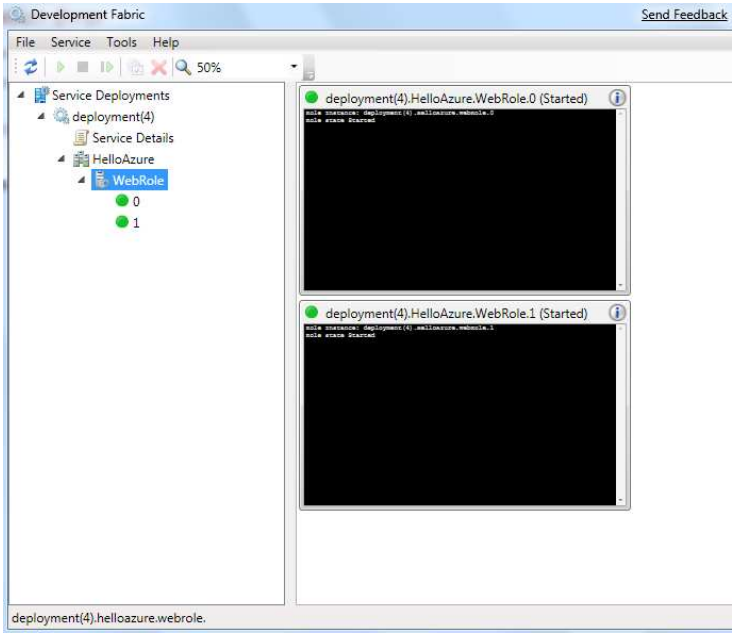

Figure 4: Windows Azure Development Fabric in action.

Once the application is ready to go public, it can be deployed in the cloud. In Microsoft Visual Studio this operation is called publishing. Publishing is a guided action that requires access to the Azure Services Developer Portal. In the deployment process each application is attached a unique URI for Internet wide access. Afterwards cloud applications can be managed and analyzed through the portal. Applications can for example be suspended and started on request. Usage statistics are also gathered on the portal.

5 Conclusion

Executing applications in the clouds offer many advantages over the traditional way of running programs. Firstly, using cloud computing allows rapid service deployment and massive savings upfront because not having to invest in infrastructure. Secondly, cloud computing model allows computing power and storage to scale up with business growth. In addition to this, it's also easy to dynamically adjust computing power up or down. As a customer, you end up paying for the actual usage of resources.

The advantages of using the Azure cloud platform relate to the fact that Microsoft has tried to minimize the changes involved in migrating applications to the cloud. Effort required from developers already familiar with Microsoft's technolo-

gies to utilize the Azure is minimal. In addition to this, upcoming releases of Azure are going to suppport applications written in languages such as Python and PHP. An other advantage in Microsoft's solution is that the services provided can be used in a very flexible fashion. Not only are Azure services available to cloud applications, but also traditional on-premises applications are free to exploit them. What's even better, Microsoft seems to be improving in terms of interoperability. Because all of the services are accessible via industry standard protocols, it is guaranteed exploiting them doesn't force customers to use Microsoft's operating systems on-premises.

Although there are many advantages in cloud computing, there are also disadvantages that shouldn't be ignored. The first and most obvious disadvantage is the fact that by running applications in the cloud you have to hand over your private data. Privacy and security concerns are direct consequences of this. Secondly, although cloud computing relieves customers from the burden of infrastructure management, it also takes away the possibility to be in total control of that infrastructure. In addition to loosing control on hardware, using compute clouds also ties the customer very tightly to the cloud service provider. Data, for example, is usually stored in a proprietary format which makes porting applications to competitors' systems hard. As customers are locked in, they are also at the mercy of that certain service provider's future pricing strategy.

References

[1] D. Chappell. Introducing the Azure Services Platform - an early look at Windows Azure, .NET services, SQL services and Live services, October 2008. `http://download.microsoft.com/download/e/4/3/e43bb484-3b52-4fa8-a9f9-ec60a32954bc/Azure_Services_Platform.pdf`.

[2] H. Erdogmus. Cloud computing: Does Nirvana hide behind the Nebula? *IEEE Software*, 26(2):4–6, 2009.

[3] M. Fitzgerald. When the forecast calls for clouds. *Inc. Boston*, 31(1):100–102, 2009.

[4] B. Hayes. Cloud computing. *Commun. ACM*, 51(7):9–11, 2008.

[5] J. N. Hoover. A stake in the cloud. *InformationWeek*, 26(1209):22–24, 2008.

[6] Microsoft Azure Services Platform documentation, October 2008. `http://www.microsoft.com/azure/default.mspx`.

[7] A. Weiss. Computing in the clouds. *netWorker*, 11(4):16–25, 2007.