Programming the Cloud

Tony Hey, Corporate Vice President
External Research, Microsoft Research
Microsoft Corporation

Abstract

Cloud computing uses data centers to provide on-demand access to services such as data storage and hosted applications that provide scalable web services and large scale data analysis. The massive scale of today's data centers and network bandwidth to these facilities make it possible for users and application service providers to '*pay as they go*', paying only for resources consumed. In this talk we cover basic cloud computing system architectures from the perspective of application programming models that are emerging as central elements of cloud computing. We also illustrate the types of applications that can take advantage of these distinct models.

The Rise of Cloud Computing

The coming shift to cloud computing is a major change in our industry. One of the most important parts of that shift is the advent of cloud platforms. As its name suggests, this kind of platform lets developers write applications that run in the cloud, or to use services provided from the cloud, or both. But the transformation that cloud computing makes possible goes beyond simply running applications on someone else's platform. It extends to utilizing a platform that the provider has created which, to some degree, abstracts the essence of scalability and distributed processing. This offers developers unique capabilities not previously available.

First, cloud computing enables a "pay only for use" strategy where users bear no cost unless they use the cloud services, and then pay only for the number of service units consumed. Next, groups can deploy and expand services rapidly, in minutes, rather than the weeks or months needed to procure and install local infrastructure, to meet rising demand or to address time-critical needs. Finally, the elasticity of cloud services means that time and computing are interchangeable – the user cost to use 10,000 processors for one hour is the same as using ten processors for 1,000 hours. This is a transformative equivalence; even individuals and small companies can exploit computing resources at a scale heretofore accessible only to large companies, university research labs, and governments.

Cloud programming is about knowing what and how to program on cloud platforms. We examine the capabilities (and differences) of today's most visible cloud platform technologies from the perspective of application programming models that are emerging to build on these platforms.

Cloud Application Programming Models

Like their on-premises cousins, cloud platforms provide the basic local functions that an application program requires. These can include an underlying operating system and local support such as deployment, management, and monitoring. Yet how current cloud platforms provide these functions differs from what we are used to, as we illustrate in our talk. There are currently three main approaches to providing user/customer application support for data centers. Each approach has strengths and limitations. We will refer to these as Infrastructure as a Service, Platform as a Service, and Software as a Service [Gannon09]. As we shall see throughout the presentation, these are really three points in a space of possible programming model architectures.

Infrastructure as a Service (IaaS). In this model the cloud provider simply provides a scalable platform for hosting client VMs. By far the most well-known example is Amazon's Elastic Compute Cloud (EC2), as associated services such as the Simple Storage Service (S3). Amazon provides a basic set of web services that can be used to deploy the VM, create an instance and secure it. The application programmer selects a favorite VM from a list of available ones. There are various versions of Linux, Windows Server, and Solaris instances configured with different web servers and databases. The application programmer picks the system that best meets his or her needs and deploys the new application in the VM. A running instance has full network access. Multiple instances can be created to support demand as needed. Amazon S3 can be used as a persistent store for data and state. From a technical perspective, it might be more accurate to think of EC2 as a platform for VMs rather than a virtualized OS or virtualized infrastructure.

Each development team is free to use whatever local support it likes in this VM. The creators of one application might choose a Java EE app server and MySQL, for example, while another group might go with Ruby on Rails. EC2 customers are even free to create many VM instances, and then distribute large workloads across them in parallel, such as for scientific applications. While the service EC2 provides is quite basic, it's also very general, and so it can be used in many different ways. One of the challenges with this approach is that it requires a substantial systems admin burden on the application designer. As a result, third party application hosting companies have emerged to provide higher-level tools on top of Amazon Web Services.

Platform as a Service (PaaS) refers to a class of cloud platform application support tools that can be used to directly support highly parallel data analysis. The way these cloud platforms operate is to use a software design pattern known as inversion of control. The idea is that the application programmer supplies the kernel of the computation and the framework invokes many copies of the kernels in parallel and controls the overall execution.

Perhaps the most well know example is the Google MapReduce framework. This is an implementation of an old idea from parallel computing and programming languages. Old, but as Google has shown, it is very effective. It works as follows. A map operator takes a function and a set of values and applies the function to each of the values. A reduce takes a set of values and a combining operation and reduces the set of values to a single value. The Google technology has been reproduced as an open source tool, Hadoop, which is now running on research clusters on behalf of NSF supported researchers. Dryad from Microsoft Research generalizes MapReduce, allowing more complex computation graphs and operators. Application programmers can use any .Net programming language to invoke Dryad through the use of DryadLINQ. Dryad and DryadLINQ were recently been released for use by academic researchers.

From the applications perspective, the critical feature is fact that a re-usable software platform can allow the developer to plug-in application specific components of a data analysis computation and the framework automatically scales the computation for parallel execution. MapReduce and Dryad are only instances of many possible parallel execution templates. It is entirely reasonable to envision a more general parallel composition framework derived from a combination of workflow, large-grain dataflow and systolic concepts. The data analysis applications can be expressed in a combination of a high-level language describing the concurrency pattern to be used and whatever language is appropriate for the computational kernels. A final advantage of this model is the fact that it can be tuned to work well with manycore server processors.

Software as a Service (SaaS). Another approach to present a programmable framework on cloud platforms is to provide a remote language execution environment, such as a Java VM, the Microsoft CLR or a language interpreter and a library of useful services. For example, to support cloud platform application development Google offers AppEngine, which is basically a way to run remote Python apps on the data center, which provides local support for running Python Web applications. Along with a standard Python runtime, AppEngine also includes a hierarchical data store with its own query language. Microsoft offer Windows Azure, which is a host for ASP.Net applications, running on Windows Server and backed by a range of services covering file storage, database access (SQL Server), identity and access control, and links to Sharepoint document management. Another example of a cloud platform providing local support is Force.com, offered by Salesforce.com.

Using these remote execution environments an application programmer can build a data access or analysis service on a local host and then push it to the data center for deployment and remote invocation as a service. A key property here is being able to expose remote data or computation as a Web Service (either pure WS or REST style). For example Astoria provides the tools to expose any data object from a collection,

stored in a database or other form, as a URI to an encoded form using a standard such JSON or ATOM representation. The Google AppEngine provides a way to deploy a remote Python script that becomes a web service that can access BigTable data. Microsoft Azure allows application developers to deploy an ASP.Net front-end to a service with worker nodes that carry out the actual computation, and a variety of persistent data storage options including tables, queues, blobs, and a relational store. A local version of the same program can have some functionality when the client is off-line if a limited local version of the cloud API is available.

The examples above are implementation of the third category, which is often referred to as "Software as a Service" (SaaS). The key idea is that the cloud provider presents a stable platform where application providers can deploy their applications for use by customers. The application provider does not need to be concerned about acquiring the hardware and networking resources to support the application, because the data center provides this and is capable of scaling it to meet the needed application user-load requirements. However, to make this work, the cloud platform must provide a set of core services and APIs to the application developer.

Cloud Comparisons

There are substantial differences between today's cloud platform technologies, as we will see during the presentation on application programming models. Other differences we will discuss in the presentation is i) extent to which resources are shared, ii) language choices, iii) persistent data storage options, and iv) pricing models. It is clear that these cloud platforms, their features and support for application programming are still in flux. However, a new kind of application platform doesn't come along very often. But when a successful platform innovation does appear, it can have enormous impact so we can expect this to be a lively topic for years to come.

References

[Gannon09] The Computational Data Center – A Science Cloud, Dennis Gannon, *available from* http://www.extreme.indiana.edu/~gannon/science-data-center.pdf.