Basics of Cloud Computing – Lecture 6

Platform as a Service (PaaS)

Pelle Jakovits
Satish Srirama
Outline

• Introduction to PaaS Cloud model
• Types of PaaS
• Google App Engine & other examples
• Advantages & disadvantages
Previous lectures have discussed mostly IaaS

IaaS provides computing resources
  – Virtual machines, storage, network.

User do not need to purchase hardware themselves

IaaS can utilize resources more efficiently

You have worked with OpenStack instances
Cloud Models

SaaS

PaaS

IaaS

End Users

Application Developers

Network Architects

http://nolegendhere.blogspot.com.ee/2012/06/presentation-4-5-7.html
IaaS Issues

• To deploy an application in IaaS, need to choose and set up:
  – Computing infrastructure
  – Software environment

• User is responsible for:
  – System administration, backups
  – Monitoring, log analysis
  – Managing software updates
  – Stability & scalability of the software environment
Platform as a Service - PaaS

- Complete platform for hosting applications in Cloud
- The underlying infrastructure & software environment is managed for you
- Enables businesses to build and run web-based, custom applications in an on-demand fashion
- Eliminates the complexity of selecting, purchasing, configuring, and managing hardware and software
- Dramatically decreases upfront costs
PaaS Characteristics

• Multi-tenant architecture
• Built-in scalability of deployed software
• Integrated with cloud services and databases
• Simplifies prototyping and deploying startup solutions
• More fine grained cost model
  – Generally do not pay for unused resources
  – Users only pay for services they use
• Typically introduces vendor lock-in
Types of PaaS

1. PaaS for online applications
2. Function as a Service (FaaS)
3. Data Processing as a Service
PaaS for online applications

• Typically built on top of existing IaaS Cloud
• Provides and manages all computing resources and services needed for running applications
• **Google App Engine**, AWS Beanstalk
• Open-Computing Platforms, which are not tied to a single IaaS provider
  – Interoperability and open-source tools
  – E.g. Cloud Foundry, Red Hat OpenShift etc.
• **Social App Platforms**
  – Develop add-ons for SaaS, such as Google+ or Facebook
  – Integrated API with the social website platform
Function as a Service

- Platform for hosting microservices
- Also called **Serverless**
- "**APP**" is a callable function
- Event driven execution based on triggers and preconditions

**Example:**
- Trigger Event: New image uploaded to S3
- Precondition: File size is larger than 10 MB
- Execute: `Resize_image(filePath)`
FaaS

• **AWS Lambda**
  - Run code/functions in AWS without managing infrastructure or software
  - Pricing is based on number of requests and GB-Sec "Memory-Duration"
  - Free: 1M requests a month. After: $0.20 per 1M
  - Free: 400,000 GB-Sec. After: $0.000017 per 1 GB-SsC

• **Apache OpenWhisk**
  - Open Source serverless cloud platform
  - Used also by IBM Bluemix PaaS
  - Event, trigger & rule based execution
  - Supports any language*
    • JavaScript, Swift, Python or PHP function, Java or any binary executable.
Data Processing as a Service

• Platform for deploying data processing applications
• Service provider manages the data processing environment and cluster
• User only needs to upload the application and set resource limits.
• Typically consist of Hadoop MapReduce ja Apache Spark
• AWS Elastic MapReduce
• Google Cloud DataProc
Google App Engine
Google App Engine

- PaaS for developing and hosting web applications in Google-managed data centers
- Easy to build, maintain, and scale applications
- No servers to maintain or configure by yourself
- Upload & Go
- Was created before Google Cloud became available for public use
- Supported languages
  - Python, Java, PHP, Go
App Engine Characteristics

- **Persistent storage** with queries, sorting, and transactions
- App Engine distributes user requests across multiple servers and *scales servers* to meet *dynamic traffic* demands
- **Asynchronous task queues** for performing work outside the scope of a request
- **Scheduled tasks** for triggering events at specified times or regular intervals
- **Integration** with all other Google Cloud services and APIs
- Your application runs within its own *secure, sandboxed and reliable* environment
  - Independent of the hardware, operating system, or physical location of the server
Deployment Life Cycle

- Write Code
- Test Locally
- Admin via Web Console
- Push to Google servers
App Engine Environment

User Application

- Cloud File Storage
- Big Query
- User Auth.
- Datastore, Cloud SQL
- Compute
- Task queue, schedule
Integration with Cloud services
Available cloud services

• **Google Cloud SQL** - A fully-managed web service that allows you to create, configure, and use relational databases in Google's cloud

• **Datastore** - A schemaless object datastore providing robust, scalable storage for your web application, a rich data modeling API, and a SQL-like query language called GQL

• **Blobstore** - Allows your application to serve large data objects, such as video or image files, that are too large for storage in the Datastore service
Other data services

• **Search** - Allows your application to perform Google-like searches over structured data such as: plain text, HTML and geographic locations.

• **Memcache** - A distributed, in-memory data cache to improve application performance

• **Logs** - Provides programmatic access to application and request logs from within your application
Process management

• **Task Queue** - Allows applications to perform work outside of a user request, and organize that work into small, discrete units, called "tasks," to be executed later

• **Scheduled Tasks** - Allows applications to configure regularly scheduled tasks that operate at defined times or regular intervals

• **Remote** - Lets external applications transparently access App Engine services. For example, you can use Remote API to access a production datastore from an app running on your local machine.
Other services

• **Users** - Allows applications to sign in users with Google Accounts or OpenID, and address these users with unique identifiers.

• **Capabilities** - Provides detection of outages and scheduled maintenance for specific APIs and services, so that your application may bypass them or inform your users.

• **Multitenancy** - Makes it easy to compartmentalize your data to serve many client organizations from a single instance of your application.

• **PageSpeed** - A family of tools that automatically optimizes the performance of your application.
Google App Engine Daily Free Quotas

<table>
<thead>
<tr>
<th>Requests</th>
<th>Daily quota</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource</td>
<td></td>
</tr>
<tr>
<td>Outgoing Bandwidth</td>
<td>1 GB</td>
</tr>
<tr>
<td>Incoming Bandwidth</td>
<td>1 GB</td>
</tr>
<tr>
<td>Frontend Instance Hours</td>
<td>28 Instance Hours</td>
</tr>
<tr>
<td>Backend Instance Hours</td>
<td>9 Instance Hours</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Storage</th>
<th>Daily quota</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource</td>
<td></td>
</tr>
<tr>
<td>Cloud Datastore Read Operations</td>
<td>0.05 Million Ops</td>
</tr>
<tr>
<td>Cloud Datastore Small Operations</td>
<td>0.05 Million Ops</td>
</tr>
<tr>
<td>Cloud Datastore Stored Data</td>
<td>1 GB</td>
</tr>
<tr>
<td>Blobstore Stored Data</td>
<td>5 GB</td>
</tr>
<tr>
<td>Number of Indexes</td>
<td>200</td>
</tr>
</tbody>
</table>
Free Google Cloud Account

• **NB!** Not required to try Google App Engine
• $300 Credit for 12 months
• Maximum limit of 8 cores at a time for instances

• **Need credit card to sign up**
  – Won't be billed until upgrade to paid account
  – Only the portion of usage above the App Engine free daily quota is charged against credit
AWS Elastic Beanstalk

- Platform as a Service for web applications
- Languages: Java, .NET, PHP, Node.js, Python, Ruby, Go
- Platforms: Docker, Apache, Nginx, Passenger, and IIS
- Upload code & deploy
- Automatically handles deployment, capacity provisioning, load balancing, auto-scaling, application health monitoring
- More manual control available (and required)

https://aws.amazon.com/elasticbeanstalk/
Windows Azure

• Hybrid PaaS & IaaS cloud platform
• Designed more for enterprise applications
• Programming languages
  – .NET, Java, PHP, Node.js, Python, or Ruby
• Datastores
  – Azure SQL database and NoSQL storage

https://azure.microsoft.com/
Windows Azure

• BigCompute
  – HPC on demand
  – MPI applications with Remote Direct Memory Access (RDMA)
• HDInsight
  – Setting up dynamic Hadoop clusters for Data Analysis
• Examples of applications running in Azure:
  – Office 365, Skype, Bing, and Xbox
  – WebZen, Toyota Gazoo.com, HALO
Other PaaS Examples

- **AppScale** [https://www.appscale.com/](https://www.appscale.com/)
  - Open-Source framework
  - Supports Google App Engine Applications
  - Supports MySQL Clusters, HBase, Hypertable, and Apache Cassandra
  - Python, Go, and Java applications

- **Cloud Foundry** [https://www.cloudfoundry.org/](https://www.cloudfoundry.org/)
  - Open-source PaaS platform that supports programming languages Java and Scala

- **AppFog** [https://www.ctl.io/appfog/](https://www.ctl.io/appfog/)
  - Supports Java, Node, .Net, Ruby, PHP, MySQL, Mongo, PostgreSQL
  - Choose your own Cloud Provider.

- **Heroku: Cloud Application Platform** [https://www.heroku.com/](https://www.heroku.com/)
  - Was one of the first PaaS services offered on the market
  - Supports Ruby, Java, Scala, and Python applications
IaaS vs PaaS Pricing Model

• IaaS
  – Per instance hour
  – Per storage volume/month
  – Per network bandwidth/month
  – Per additional services (Static IP, Autoscaling)

• PaaS
  – Individual prices for each Cloud Service
  – Small applications can be hosted for free
PaaS Advantages

- User does not have to manage low level computing resources and services
- Many services ready to use in a plug-in fashion without any configuration or setup
- **Provider handles most of the non functional requirements of your applications**
- **Scaling is automatically managed** by the platform
- Easier and more agile application deployment
  - Simplifies prototyping and application startups
- **Lower costs**
  - Pay for only for resource which are used
  - More fine-grained cost model than in IaaS
- Platform provider has the best knowledge to optimize the services running on the underlying hardware
Disadvantages of PaaS

• **Not in full control** over:
  – Computing resources (Intel vs AMD, GPU’s, FPGA, …)
  – Software and library versions
  – Service configuration

• Available programming languages are sometimes limited

• **Vendor lock-in**

• Offered services may not be flexible enough for user needs

• Have to **fully trust** in the PaaS provider
  – Billing accuracy
  – Security
  – Reliability
  – Data ownership

• What happens when application exceeds billing quotas? What happens when payments fail?
That's All

• This week's practice session is:
  – Google App Engine: Creating and deploying applications

• Next lecture
  – Other Cloud Services & Cloud based research