PaaS - Platform as a Service
Google App Engine

Pelle Jakovits
14 April, 2015, Tartu
Outline

• Introduction to PaaS
• Google Cloud
• **Google AppEngine**
  – DEMO - Creating applications
  – Available Google Services
  – Costs & Quotas
• Amazon Beanstalk& Elastic MapReduce
• Windows Azure PaaS
• PaaS Advantages & Disadvantages
Cloud Services

- **SaaS**
- **PaaS**
- **IaaS**

End Users

Application Developers

Network Architects

Value Visibility to End Users
Platform as a Service - PaaS

• Users are provided with a full platform for their applications
• Enables businesses to build and run web-based, custom applications in an on-demand fashion
• Eliminates the expense and complexity of selecting, purchasing, configuring, and managing hardware and software.
• Provides access to unlimited computing power
• Decreasing upfront costs dramatically
PaaS Characteristics

• Multi-tenant architecture
• Built-in scalability of deployed software
• Integrated with web services and databases
• Users are provided with tools to simplify creating and deploying applications
• Simplifies prototyping and deploying startup solutions
PaaS Characteristics

• Users only pay for services they use
• More fine grained cost model
• Provides tools to handle billing and subscription management
• Typically introduces vendor lock-in
Types of PaaS

• **Stand Alone Application Platforms**
  – Typically built on top of an existing IaaS
  – Provides development tools for designing and deploying software.
  – Provide all required computing resources and services needed for hosted applications

• **Social Application Development Platforms**
  – Used to develop addons and internal applications for social websites like Google+ and Facebook.
  – Integrated API with the social website platform.
  – Can be seen as extending a SaaS

• **Open-Computing Platforms**
  – Not tied to a single IaaS provider.
  – Supports applications that are written in numerous languages and that use any type of database, operating system, and server.
Google Cloud

• IaaS & PaaS services
• All services built on top of Google hardware located across the globe
  • Global redundancy and service locality
• Utilizes Google’s Global network
  – Dedicated fiber optic networks
  – Even between continents
• Google Managed Services
  – Database administration, server configuration, sharding and load balancing managed for the user
Google Cloud Services

• **Compute Engine** - IaaS
  - **App Engine** - PaaS
  - **Storage**
    - **Cloud SQL** – Fully managed Relational MySQL
    - **Cloud Storage** – Object storage for applications
    - **Cloud Datastore** – Automatically scaled NoSQL storage
  - **Big Data**
    - **Big Query** – SQL like queries against multi-terabyte datasets
Google Cloud Tools

• **Cloud Endpoints**
  – RESTful services to make your code accessible from iOS, Android and Javascript clients.

• **Google Cloud SDK**
  – For creating and managing Google cloud services

• **Push-To-Deploy** – Git repository to access your code

• **Cloud Playground** – Browser access to all Google Services
Google Cloud Product Lines

• **Mobile**
  – Automatic scaling and load balancing
    • From hundreds to millions of users
    • Scaling up to 6 billion requests a day
  – Dedicated hosting for serving static content

• **Hadoop Data Processing**
  – Run at Scale – MapR used Google Compute Engine to set a new world record for MinuteSort, sorting 1.5 trillion bytes in 60 seconds.

• **Gaming**
  – Synchronize game state across multiple devices
  – Analyze user engagement
  – Distribute and monetize your game with Google AdSense and Google In-App Payments.
Google App Engine

Try it out yourself DEMO @

https://developers.google.com/appengine
Google AppEngine

• Easy to build, maintain, and scale applications
• No servers to maintain yourself
• Upload & Go
• Was created before Google Cloud became available for public use
• Supported languages
  – Python, Java, PHP (preview), Go (Experimental)
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 that is independent of the hardware, operating system, or physical location of the server.
Use Cases

• Rovio
  – Transition “Angry Birds” games to an online environment
  – Use a platform that could support explosive demand
  – Easily add new features to improve the user experience

• Ubisoft
  – Develop a web-based version of a console game “From Dust”
  – Find an easy-to-use system so developers focus on the user experience
  – Scale effortlessly to accommodate a fast-growing number of players

• Best Buy
  – Quickly develop and deploy apps to achieve its business goals
  – Create apps that are scalable and low-maintenance
  – Dramatic time and cost savings in app development, which has led to the creation of many more apps
Datastore

• **Google Cloud SQL** - A fully-managed web service that allows you to create, configure, and use relational databases that live 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.
Datastore

• **Search** - Allows your application to perform Google-like searches over structured data such as: plain text, HTML, atom, numbers, dates, 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.
Communication

- **Channel** - Creates a persistent connection between your application and Google servers, so you can send messages to JavaScript clients in real time without "polling."

- **Google Cloud Endpoints** - Enables automatic generation of APIs, making it easier to create a web backend for web clients and mobile clients.

- **Mail** - Sends email messages on behalf of administrators and users with Google Accounts, and receives mail at various addresses.

- **URL Fetch** - Uses Google's networking infrastructure to issue HTTP and HTTPS requests to URLs on the web.

- **XMPP** - Enables an application to send and receive chat messages to and from any XMPP-compatible chat messaging service.

- **Traffic Splitting** - Allows you to roll out features for your app slowly over a period of time. Traffic Splitting works by splitting incoming requests to different versions of your app.
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.
Computation

- **Backends** - Instances of your application that are exempt from request deadlines and have access to more memory and CPU resources.
- **Images** - Manipulates, combines, and enhances images, converts images between formats, and queries image metadata such as height and frequency of colors.
- **MapReduce** - An optimized adaptation of the MapReduce computing model for efficient distributed computing over large data sets.
Other services

• **App Identity** - Gives code access to the application identity; provides framework to assert this identity over OAuth.

• **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.
AppEngine for Mobiles

• Define business logic on App Engine and access them via RESTful APIs on multiple platforms including Android, IOS and JavaScript.
• Deploy an app in 300 seconds
• Automatically generate strongly-typed client libraries for iOS, Android, and JavaScript.
• Geo-proximity search
• Push Notifications to iOS and Android
• Simple access to data storage and processing
• Might not make sense if your mobile application does not need to be connected to internet
AppEngine for Mobiles
IaaS Pricing Model

- Per instance hour
- Per storage amount/month
- Per Network bandwidth/month
- Per additional services
  - Static IP
  - Autoscaling/load balancing
# PaaS Pricing Model

<table>
<thead>
<tr>
<th>Service</th>
<th>Free quota per app per day</th>
<th>Pricing if you exceed your free quota</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instances</td>
<td>28 instance hours</td>
<td>$0.05 / instance / hour</td>
</tr>
<tr>
<td>Cloud Datastore (NoSQL Database)</td>
<td>50k read/write/small 1 GB storage</td>
<td>$0.06 / 100k read or write ops Small operations free* $0.18 / GB / month</td>
</tr>
<tr>
<td>Outgoing Network Traffic</td>
<td>1 GB</td>
<td>$0.12 / GB</td>
</tr>
<tr>
<td>Incoming Network Traffic</td>
<td>1 GB</td>
<td>Free</td>
</tr>
<tr>
<td>Cloud Storage</td>
<td>5 GB</td>
<td>$0.026 / GB / month</td>
</tr>
<tr>
<td>Memcache</td>
<td>Free Usage of Shared Pool No free quota for Dedicated Pool</td>
<td>Free Usage of Shared Pool Dedicated Pool: $0.06 / GB / hour</td>
</tr>
</tbody>
</table>
# PaaS Pricing Model

<table>
<thead>
<tr>
<th>Service</th>
<th>Free quota per app per day</th>
<th>Pricing if you exceed your free quota</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search</td>
<td>1000 basic operations</td>
<td>$0.50 / 10k searches</td>
</tr>
<tr>
<td></td>
<td>0.01 GB indexing documents</td>
<td>$2.00 / GB indexing documents</td>
</tr>
<tr>
<td></td>
<td>0.25 GB document storage</td>
<td>$0.18 / GB / month Storage</td>
</tr>
<tr>
<td></td>
<td>100 searches</td>
<td></td>
</tr>
<tr>
<td>Email API</td>
<td>100 recipients</td>
<td>Contact Sales</td>
</tr>
<tr>
<td>Logs API</td>
<td>100 MB</td>
<td>$0.12 per GB</td>
</tr>
<tr>
<td>Task Queue and Logs Storage</td>
<td>5 GB</td>
<td>$0.026 / GB / month</td>
</tr>
<tr>
<td></td>
<td>1 GB</td>
<td></td>
</tr>
<tr>
<td>SSL Virtual IPs</td>
<td>No free quota</td>
<td>$39 / virtual IP / month</td>
</tr>
</tbody>
</table>
AWS Elastic Beanstalk

• Languages: Java, .NET, PHP, Node.js, Python, Ruby, Go

• Platforms: Docker, Apache, Nginx, Passenger, and IIS.

• Automatically handles deployment, capacity provisioning, load balancing, auto-scaling, application health monitoring.

• More manual control available (and required)
Amazon Elastic MapReduce

• Managed Hadoop Platform
  – Resource provisioning, cluster setup, tuning
  – Guarantees reliability and scalability
  – Still flexible with full access to cluster if needed

• MapReduce, Spark, or other Hadoop supported engines

• Input and output into S3, DynamoDB or other AWS datastores

• Cheaper thanks to spot and reserved instances
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
• Mobiles
  – Native SDK’s for Windows, iOS, Android and HTML5
  – Notification hubs for push notifications (Windows, iOS, Android)
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**
  – Open-Source framework
  – Supports Google App Engine Applications.
  – Supports MySQL Clusters, HBase, Hypertable, and Apache Cassandra
  – Python, Go, and Java applications.

• **Cloud Foundry**
  – Cloud Foundry is an open-source PaaS platform that supports programming languages Java and Scala.

• **AppFog**
  – Supports Java, Node, .Net, Ruby, PHP, MySQL, Mongo, PostgreSQL
  – Choose your own Cloud Provider.

• **Heroku**
  – Was one of the first PaaS services offered on the market
  – Supports Ruby, Java, Scala, and Python applications
PaaS Advantages

• User does not have to manage low level computing resources and services
• Provider handles most of the non functional requirements of your applications
• **Scaling is automatically managed** by the platform
• Easier and Agile application deployment
  – Simplifies prototyping and application startups
• **Lower costs**
  – Pay for only what is used
  – More fine-grained cost model than in IaaS
• Platform provider has the best knowledge to optimize the services running on the underlying hardware
PaaS Disadvantages

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

• Available programming languages are typically 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?
That's All

• This week's practice session is:
  – Creating and deploying Google AppEngine applications using python

• Next lecture: Other Cloud Services