UNIVERSITY OF TARTU

INSTITUTE OF COMPUTER SCIENCE

Basics of Cloud Computing – Lecture 6

Platform as a Service (PaaS)

Pelle Jakovits

Outline

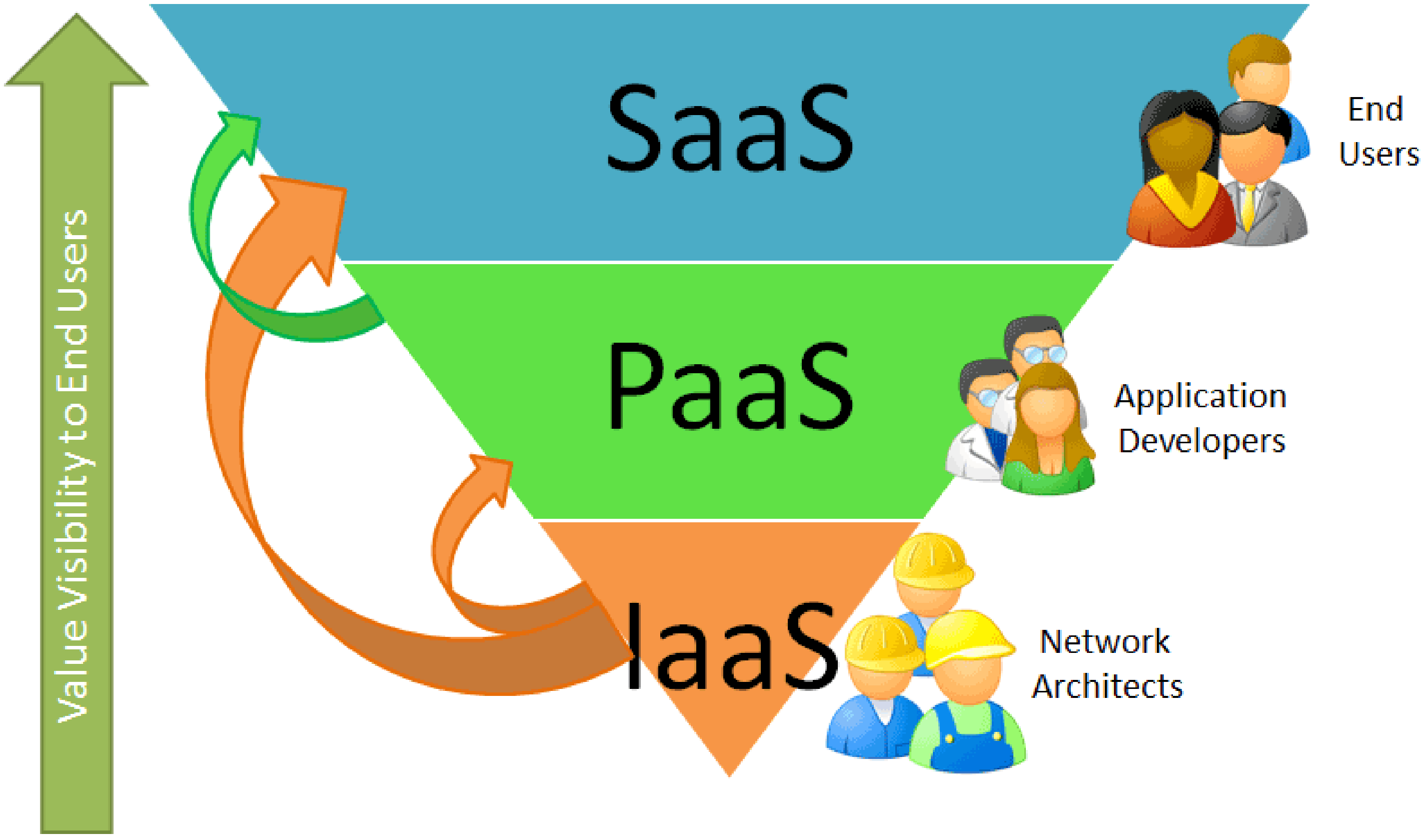
- Introduction to PaaS Cloud model
- Types of PaaS
- Google App Engine & other examples
- Advantages & disadvantages

Background

- 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

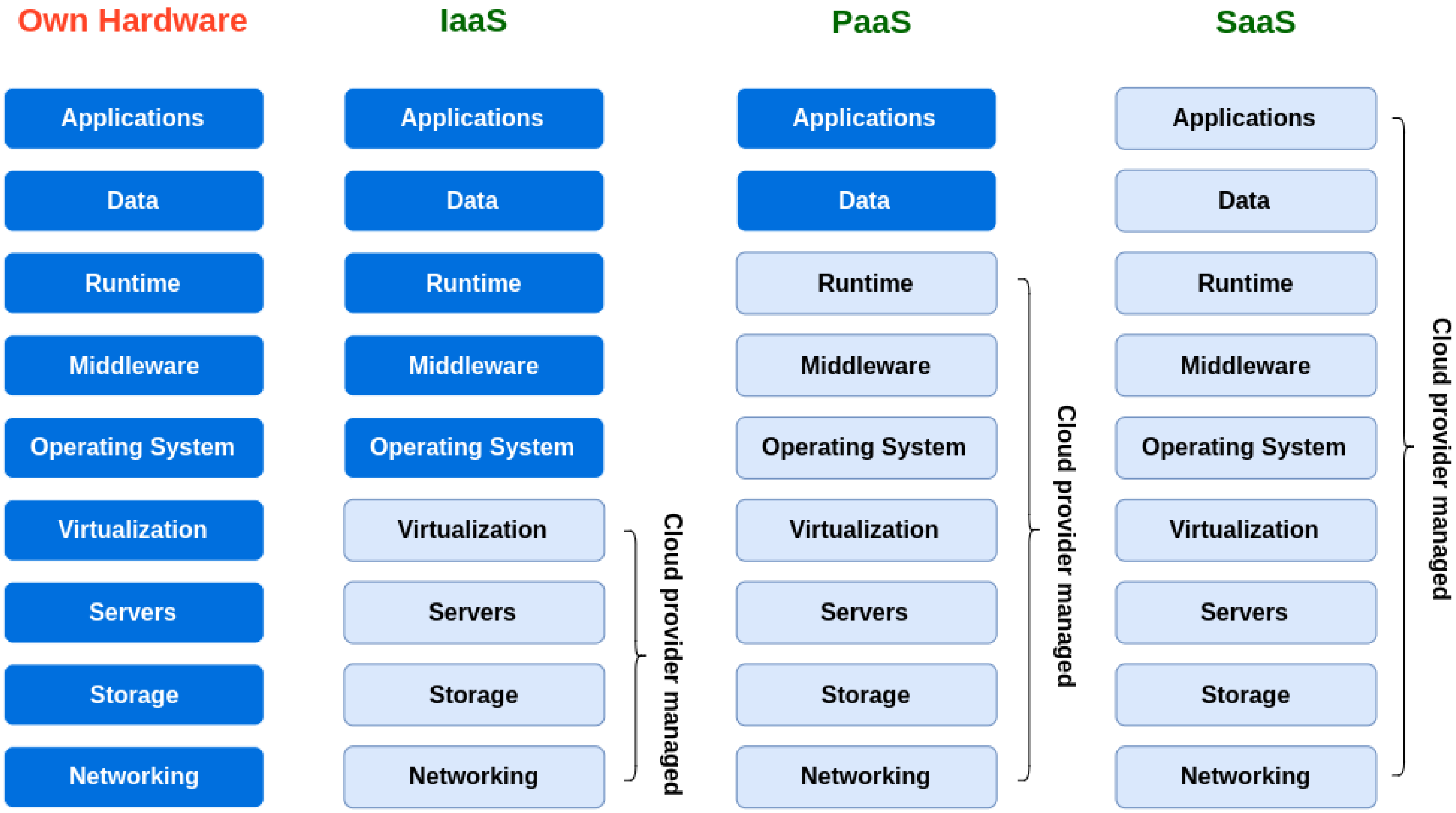


http://nolegendhere.blogspot.com.ee/2012/06/presentation-4-5-7.html

IaaS Issues

- To deploy an application in IaaS, need to 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

Cloud Model complexity



Own Hardware	IaaS	PaaS	SaaS
Applications	Applications	Applications	Applications
Data	Data	Data	Data
Runtime	Runtime	Runtime	Runtime
Middleware	Middleware	Middleware	Middleware
Operating System	Operating System	Operating System	Operating System
Virtualization	Virtualization	Virtualization	Virtualization
Servers	Servers	Servers	Servers
Storage	Storage	Storage	Storage
Networking	Networking	Networking	Networking

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 web 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 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

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
- 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
- Application runs within its own **secure, sandboxed and reliable** environment
  - Independent of hardware, OS or physical location of the server

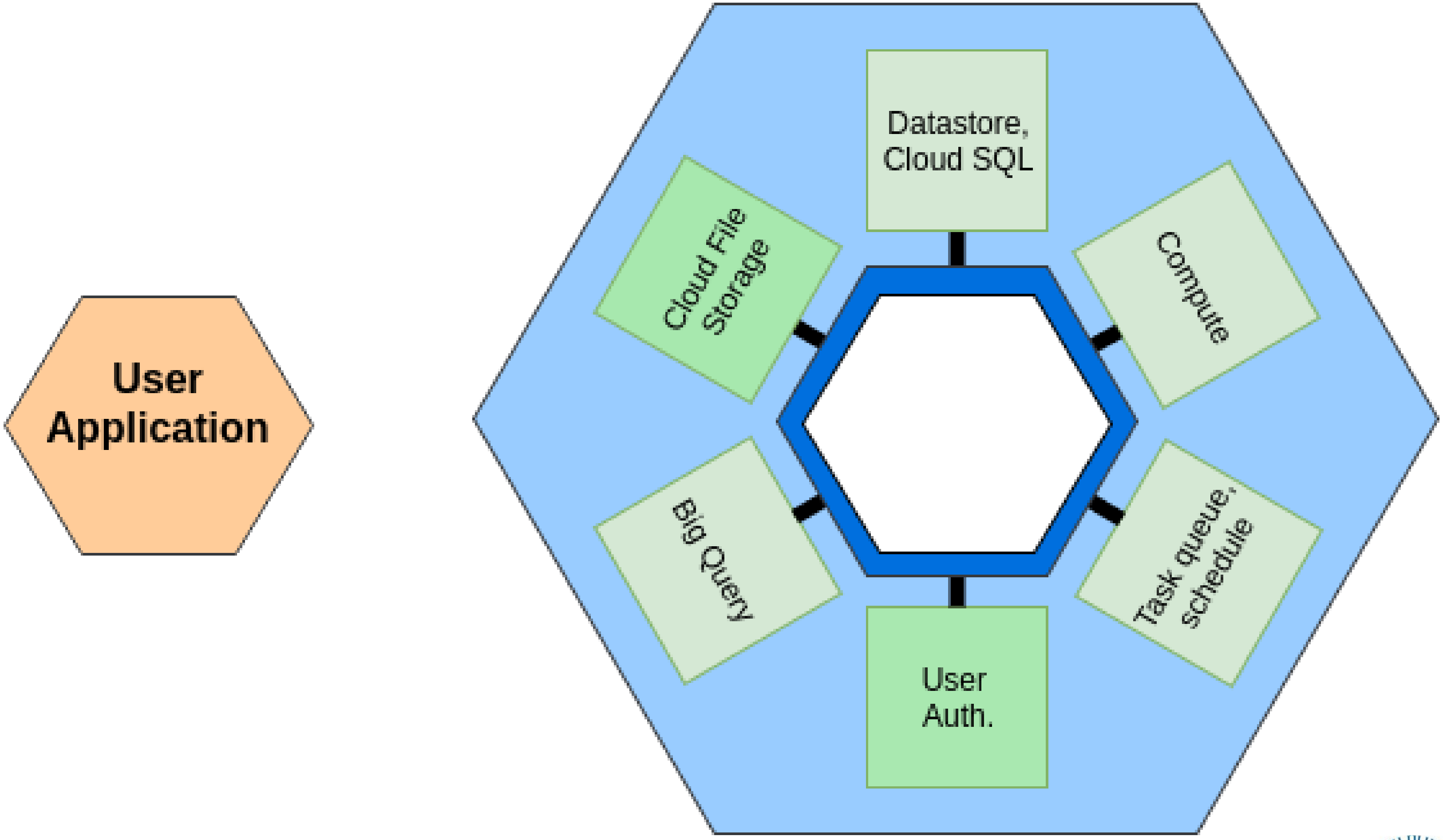
Available cloud services

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

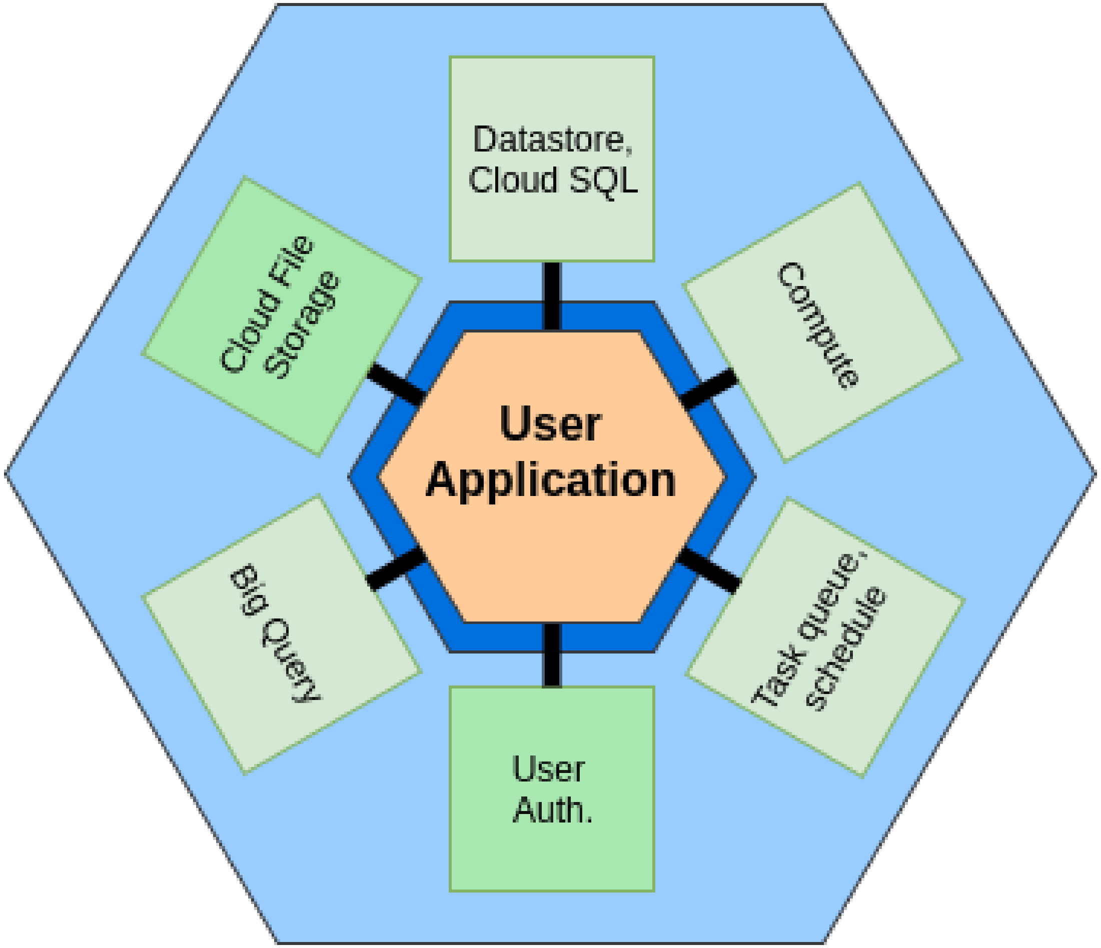
Available services

- **Search** - Allows your application to perform Google-like searches over structured data: 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
- **Remote** - Lets external applications transparently access App Engine services. For example, to access a production datastore from an app running on local machine.

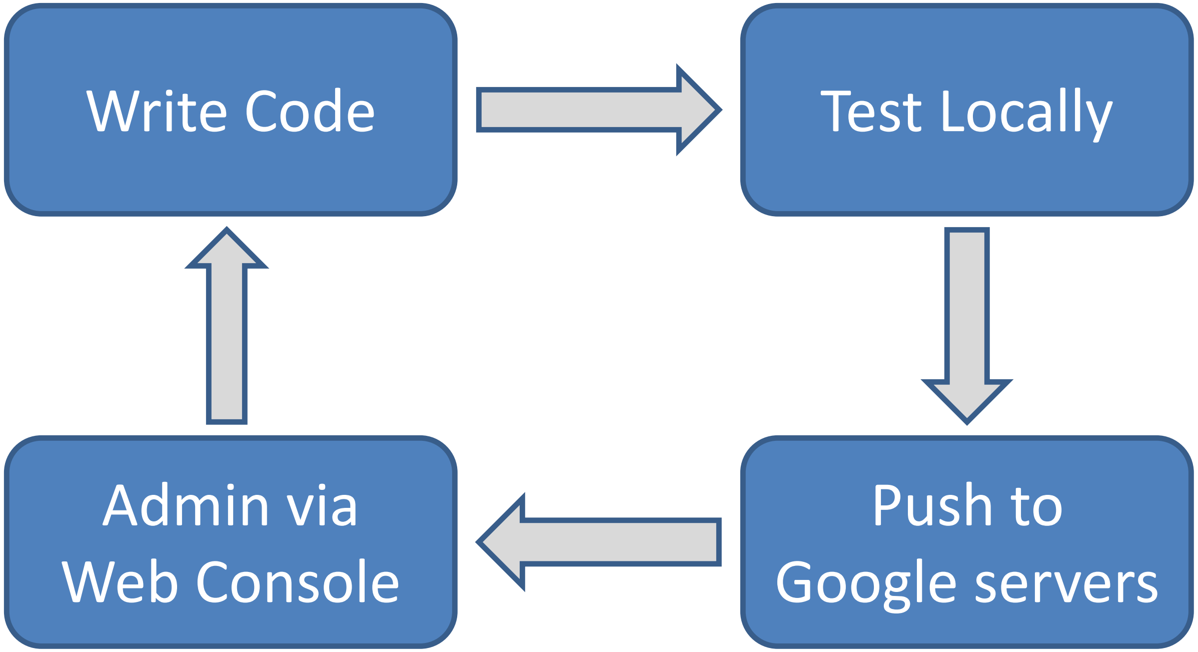
App Engine Environment



Integration with Cloud services



Deployment Life Cycle



Google App Engine Daily Free Quotas

Requests	
Resource	Daily quota
Outgoing Bandwidth	1 GB
Incoming Bandwidth	1 GB
Frontend Instance Hours	28 Instance Hours
Backend Instance Hours	9 Instance Hours

Storage	
Resource	Daily quota
Cloud Datastore Read Operations	0.05 Million Ops
Cloud Datastore Small Operations	0.05 Million Ops
Cloud Datastore Stored Data	1 GB
Blobstore Stored Data	5 GB
Number of Indexes	200

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
- **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, Xbox, WebZen, HALO

Other PaaS Examples

- **AppScale** https://www.appscale.com/
  - Open-Source PaaS framework for **Vendor Agnostic** cloud applications
  - Can deploy in any of the supported public clouds
    - Google App Engine, AWS, IBM Cloud, etc.
  - Supports Python, Java, PHP and Go
  - Can deploy existing Google App Engine applications
- **Heroku: Cloud Application Platform** https://www.heroku.com/
  - Was one of the first PaaS services offered on the market
  - Provides fully managed reliable containers for applications
  - Supports Node, Ruby, Java, PHP, Python, Go, Scala or Clojure

Function as a Service

- Often also referred to as **Serverless**
- Each deployed "application" is a single Function
- Functions are independent from each other
  - Scaled, managed and billed separately
  - Can be written in different languages
- Event driven execution based on triggers and preconditions:
  - **`Trigger Event:`** `New image uploaded to S3`
  - **`Precondition:`** `File size is larger than 10 MB`
  - **`Execute:`** `Resize_image(filePath)`
- Functions can be composed into larger applications

FaaS

- **AWS Lambda**
  - Run code/functions in AWS without managing infrastructure or software environment
  - 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

- PaaS for data processing applications
- Service provider manages the data processing cluster
- User specifies required computing resources and uploads the application
- Typically consist of a Hadoop cluster and includes a selection of Hadoop Ecosystem frameworks and tools

- AWS Elastic MapReduce
- Google Cloud DataProc

Amazon Elastic MapReduce

- Service for requesting Amazon-customized Hadoop clusters on-demand
- Supports MapReduce, Spark, Hive, Pig, Flink and Hadoop ecosystem tools
- Web interface and command-line tools for running Hadoop jobs on EC2
- Data can be stored in HDFS, Amazon S3, Dynamo DB, Redshift
- Monitors job and shuts machines after use
- Running a job
  - Upload job jar & input data to S3
  - Create the cluster
  - Create a Job Flow as steps
  - Wait for the completion and examine the results

http://aws.amazon.com/elasticmapreduce/

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 weeks practice session is:
  - Google App Engine: Creating and deploying applications
- Next lecture
  - Other Cloud Services

References

- Rajkumar Buyya, Christian Vecchiola, S.Thamarai Selvi, Mastering Cloud Computing: Foundations and Applications Programming
- Google Cloud Platform https://cloud.google.com/
- Chakkrit Tantithamthavorn, Introduction to Google App Engine, https://www.slideshare.net/klainfo/introduction-to-google-app-engine-13223789