Cloud Computing for Developers

Uli Hitzel | Singapore Spring User Group Meeting, March 2014

spring by Pivotal

CloudFX

Agenda

OpenStack

What is it | Why is it important | Who is using it

Cloud Computing

Cloud in 2014 | Developers as the consumer | Architect Cloud Applications

About Uli

- Senior Architect at CloudFX

- previously working as software developer, engineer, project manager and consultant for companies including

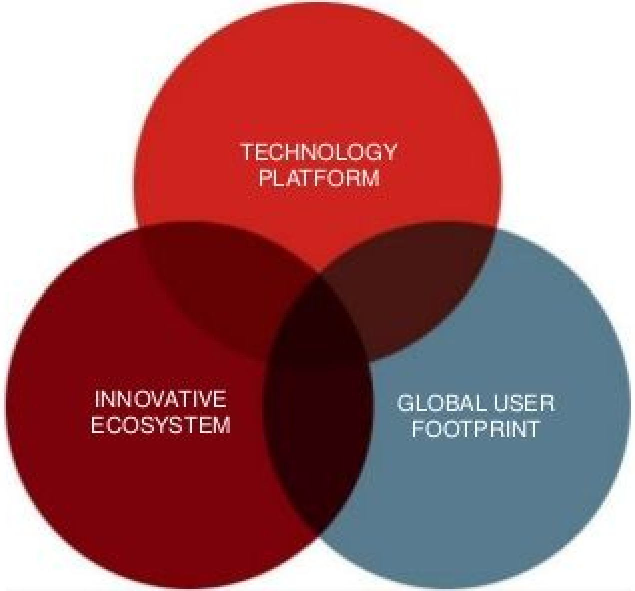
Y!™ redhat. IBM® hp

Part 1: OpenStack

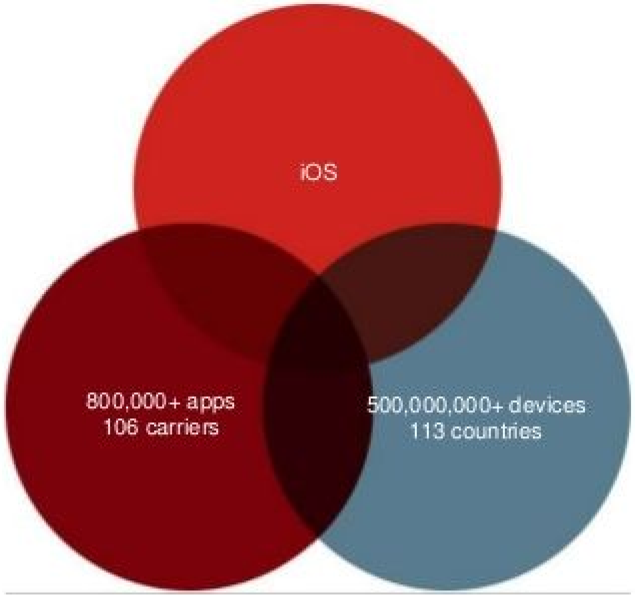
What’s OpenStack

- Cloud Infrastructure Software
- Global open source community, founded by Rackspace & NASA
- Collaboration between technology vendors including Red Hat, IBM, Cisco and many others

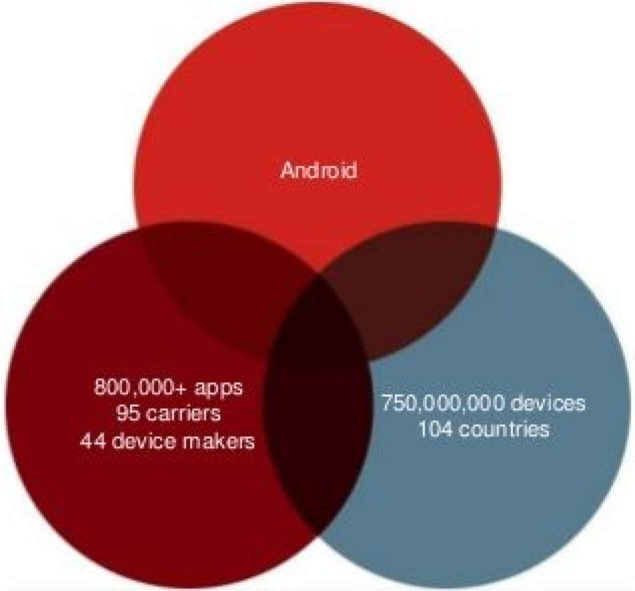
General Success Factors



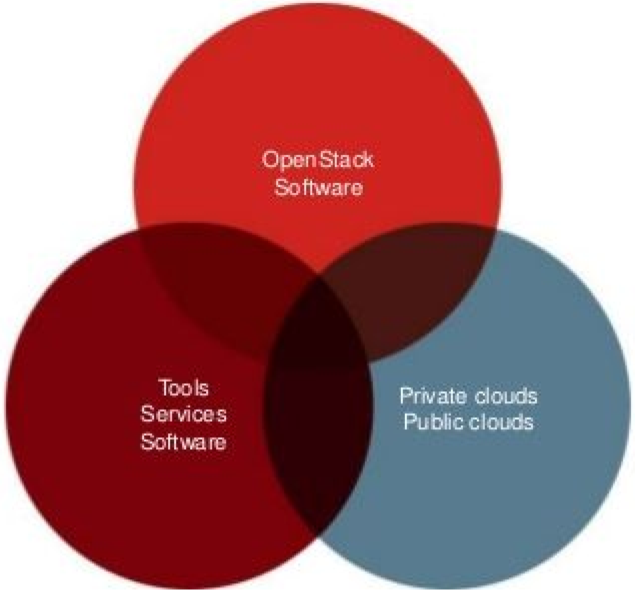
Apple


iOS
800,000+ apps
106 carriers
500,000,000+ devices
113 countries

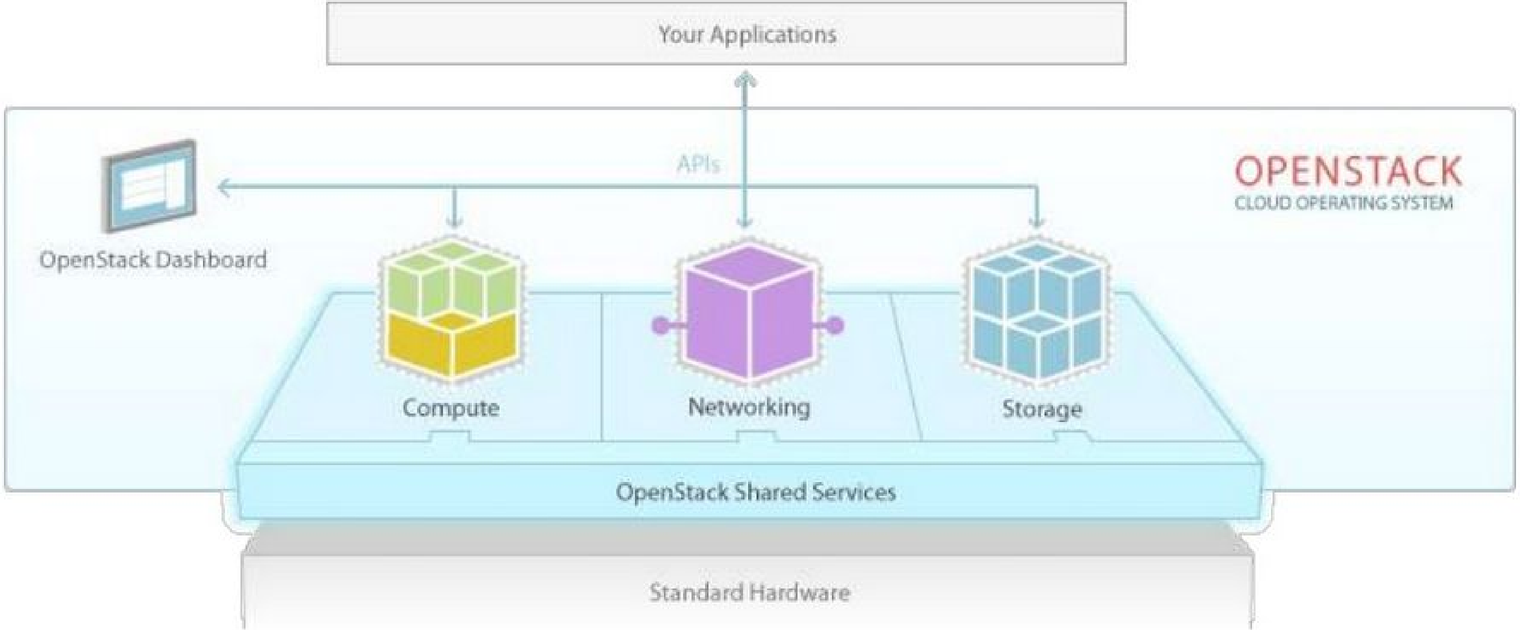
Android


Android
800,000+ apps
95 carriers
44 device makers
750,000,000 devices
104 countries

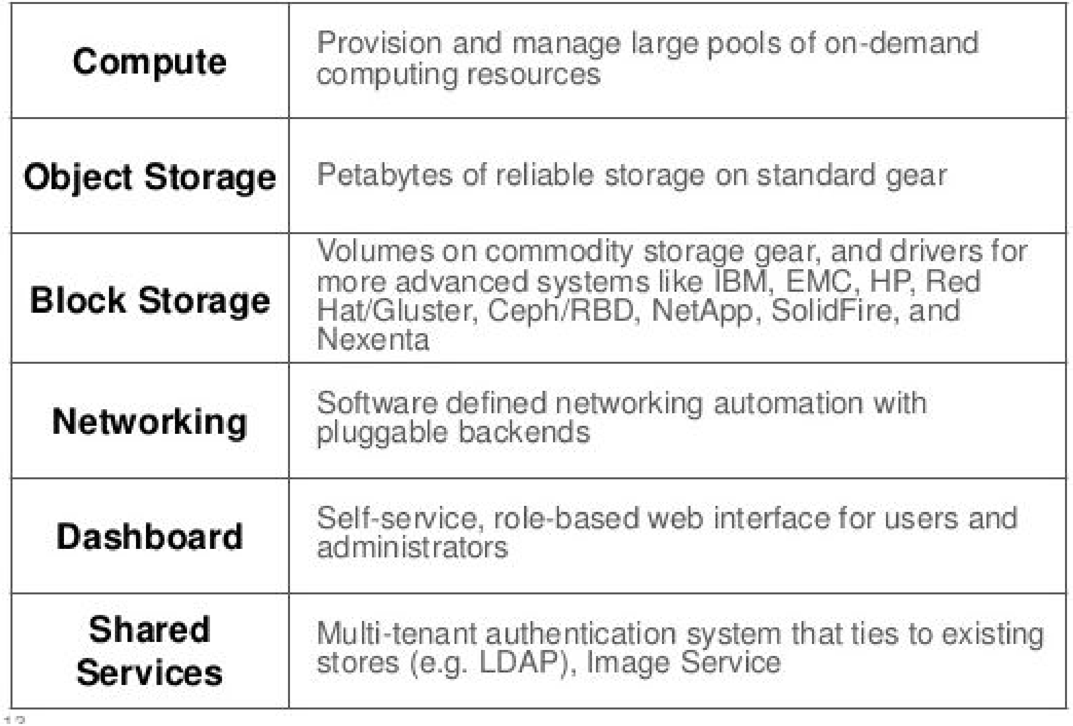
OpenStack


OpenStack
Software
Tools
Services
Software
Private clouds
Public clouds

OpenStack - Cloud Infrastructure Software

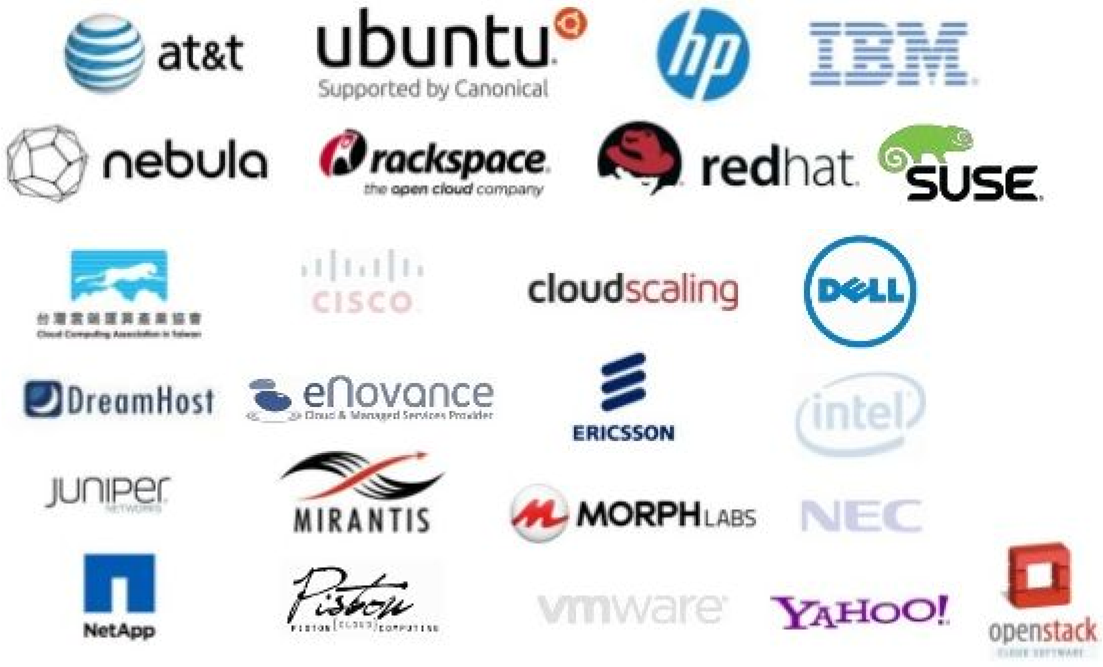


OpenStack - Features



Compute	Provision and manage large pools of on-demand computing resources
Object Storage	Petabytes of reliable storage on standard gear
Block Storage	Volumes on commodity storage gear, and drivers for more advanced systems like IBM, EMC, HP, Red Hat/Gluster, Ceph/RBD, NetApp, SolidFire, and Nexenta
Networking	Software defined networking automation with pluggable backends
Dashboard	Self-service, role-based web interface for users and administrators
Shared Services	Multi-tenant authentication system that ties to existing stores (e.g. LDAP), Image Service

OpenStack Ecosystem



The need for Open Standards

iPhone 5

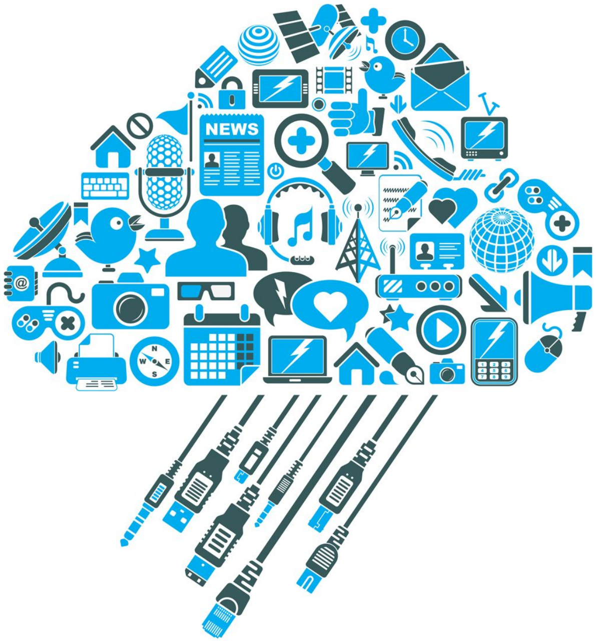
Micro USB

OpenStack Implementations

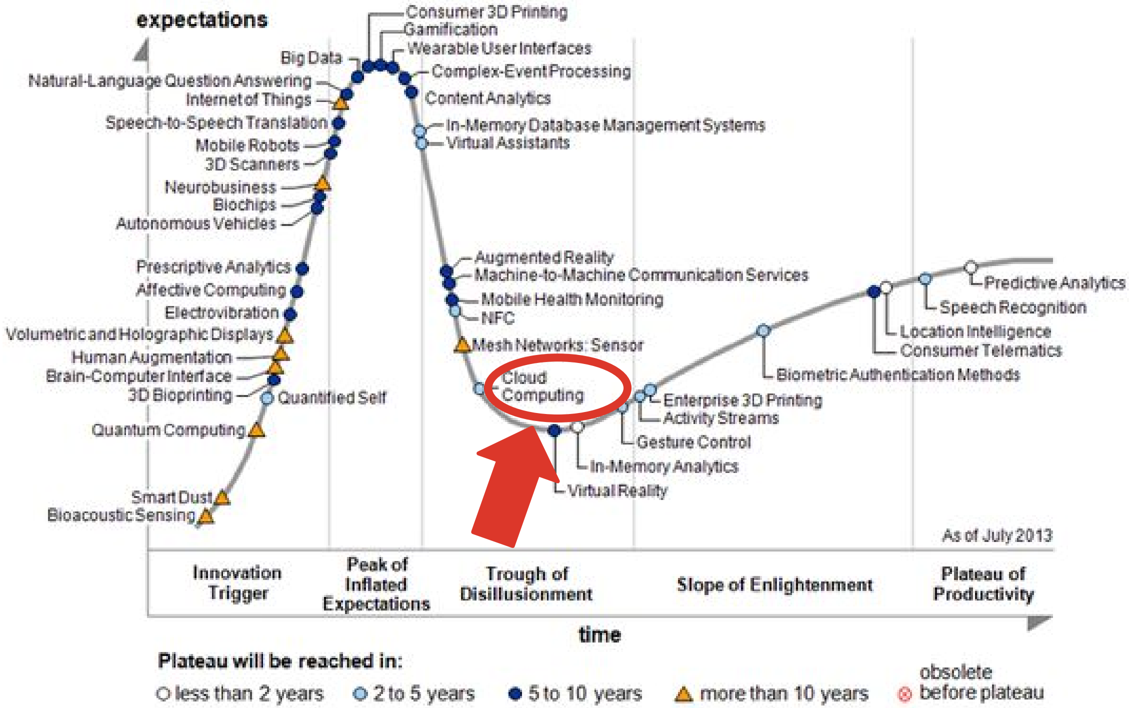
PayPal
Cisco webex
ebay
CERN
TELVENT
T
AT&T
NASA
rackspace HOSTING
acens
mercado Libre.com
INFORMATION SUPERGRID
aptira
x.commerce
DreamHost
NCI
SDSC SAN DIEGO SUPERCOMPUTER CENTER
STACKSCALE
WIKIMEDIA FOUNDATION
kt
MEMSET HOSTING
VEXXHOST
CANONICAL
CYBERA
Argonne NATIONAL LABORATORY
INTERNAP
SONY COMPUTER ENTERTAINMENT
hp
Dualtec Cloud Builders
ISI Information Sciences Institute
YAHOO!
globo.com
eduserv
choopa.com
ITRI Industrial Technology Research Institute
eNovance Cloud & Managed Services Provider
sina 新浪

OpenStack.org/User-Stories

Part 2: Cloud Computing

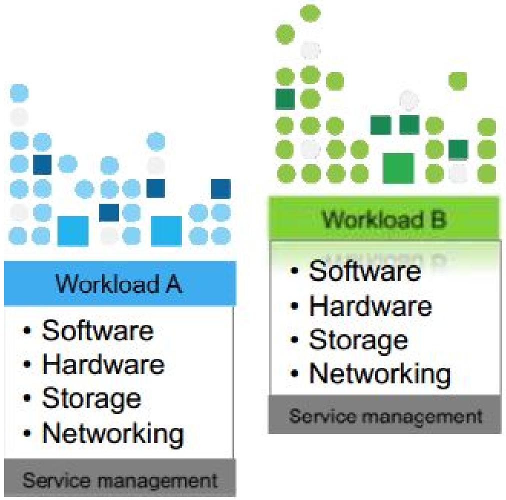


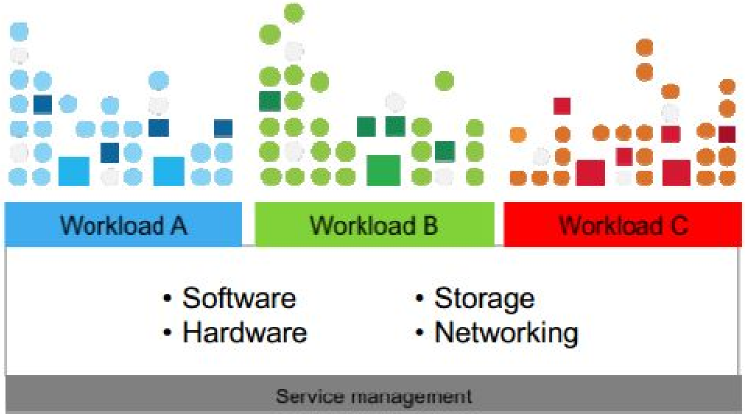
Cloud Computing in 2014



Hype Cycle for Emerging Technologies 2013 Source: Gartner

Traditional Workloads vs Cloud Workloads





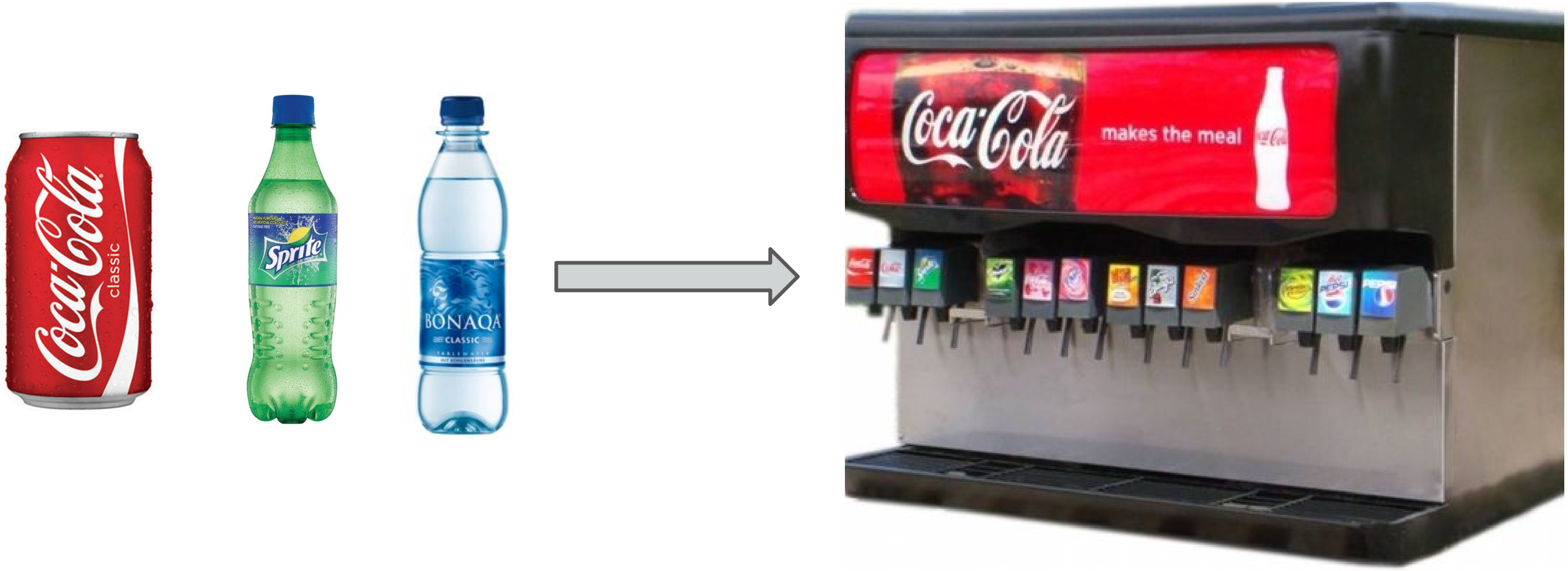
Traditional

Dedicated Resources for each workload

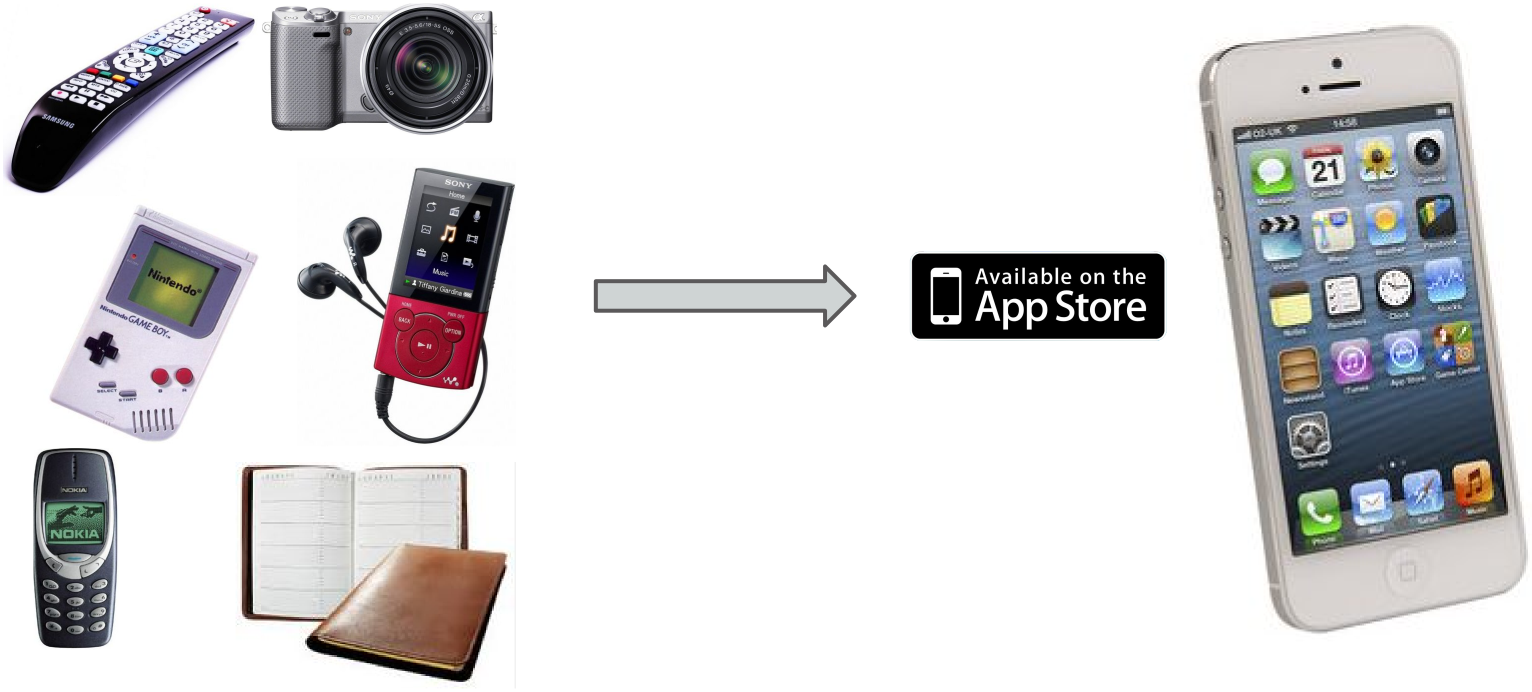
Cloud

Virtualized & Shared & Standardized Resources
Scalability & Elasticity
Automated Service Management

A new consumption & delivery model



Smart Phones - Turn devices into apps

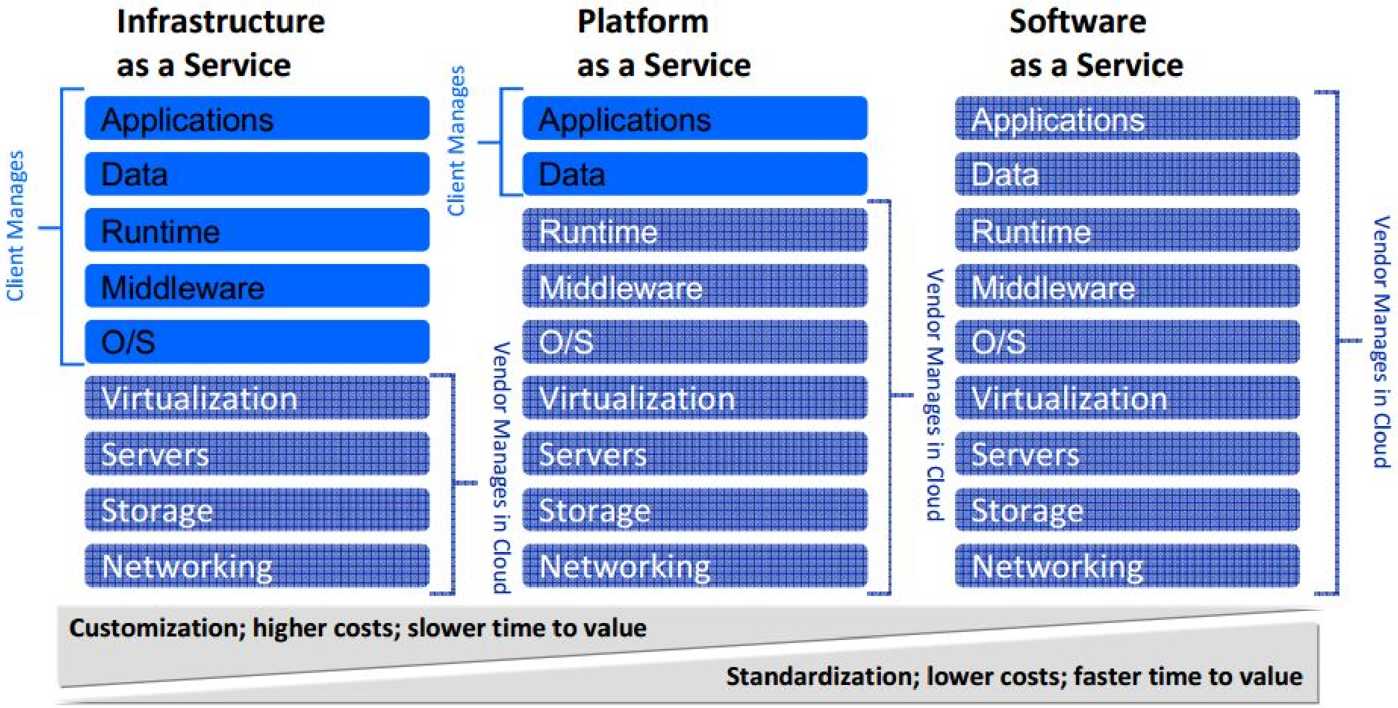


The Developer as the Cloud Consumer

Developer
“I want to develop my application”

IT Administrator
“I manage servers, storage and networks”

Virtualize the Application Stack



Infrastructure as a Service

Client Manages: Applications, Data, Runtime, Middleware, O/S

Vendor Manages in Cloud: Virtualization, Servers, Storage, Networking

Platform as a Service

Client Manages: Applications, Data

Vendor Manages in Cloud: Runtime, Middleware, O/S, Virtualization, Servers, Storage, Networking

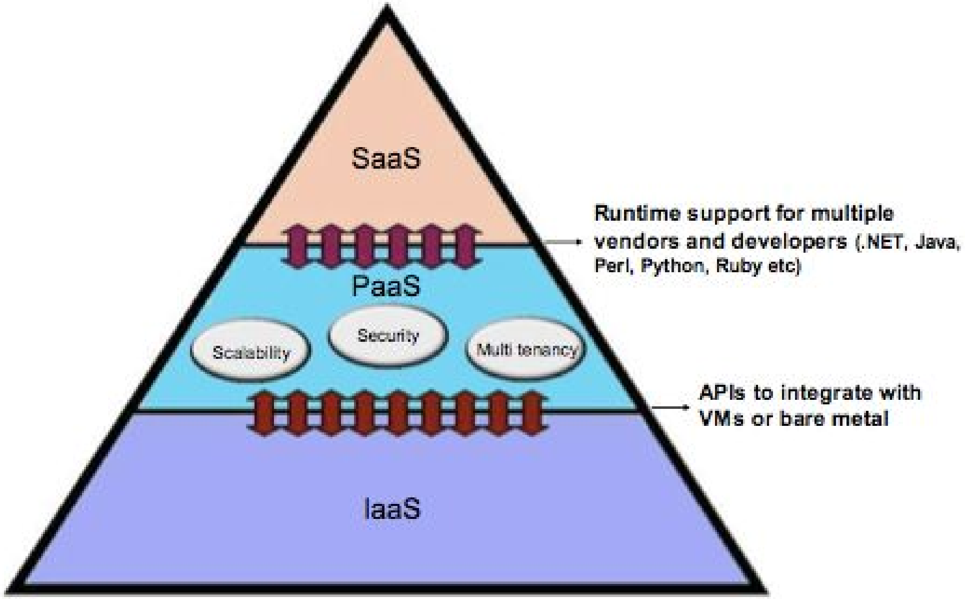
Software as a Service

Vendor Manages in Cloud: Applications, Data, Runtime, Middleware, O/S, Virtualization, Servers, Storage, Networking

Customization; higher costs; slower time to value

Standardization; lower costs; faster time to value

Platforms are your Friend

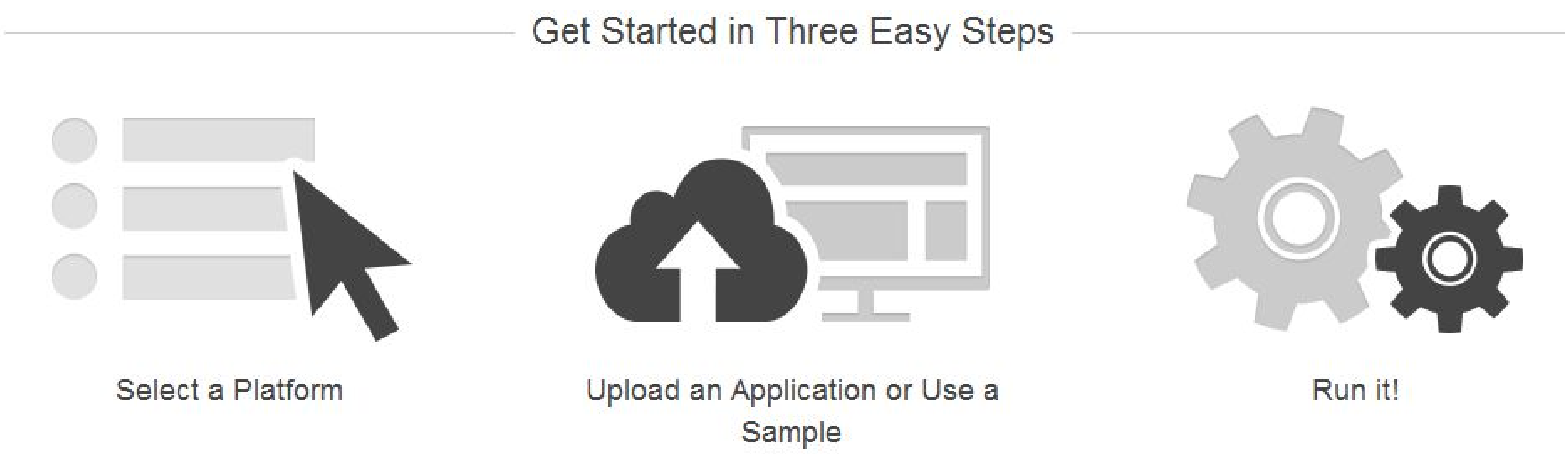


Runtime support for multiple vendors and developers (.NET, Java, Perl, Python, Ruby etc)

APIs to integrate with VMs or bare metal

Platform as a Service Offerings

AWS Elastic Beanstalk Example



Get Started in Three Easy Steps

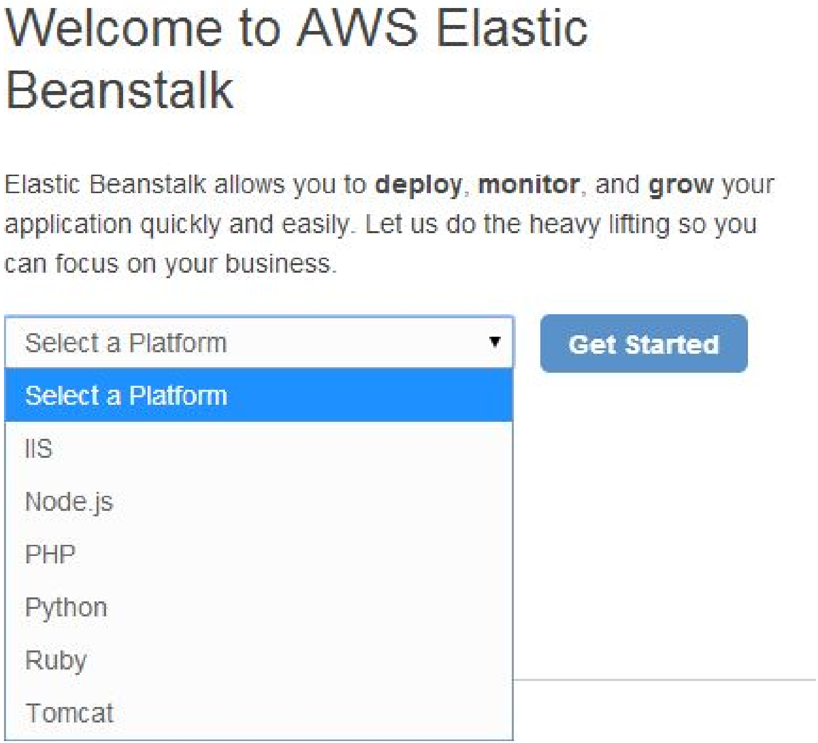
Select a Platform

Upload an Application or Use a Sample

Run it!

AWS: Choose your Platform

amazon web services
Elastic Beanstalk

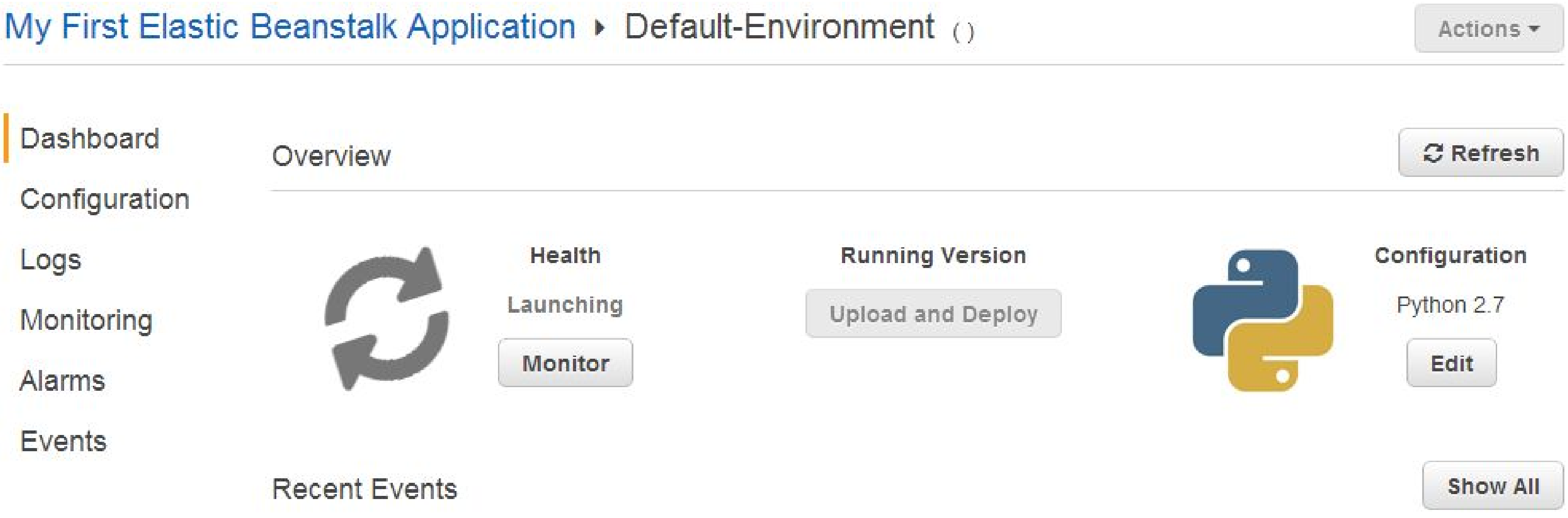


Welcome to AWS Elastic Beanstalk

Elastic Beanstalk allows you to **deploy**, **monitor**, and **grow** your application quickly and easily. Let us do the heavy lifting so you can focus on your business.

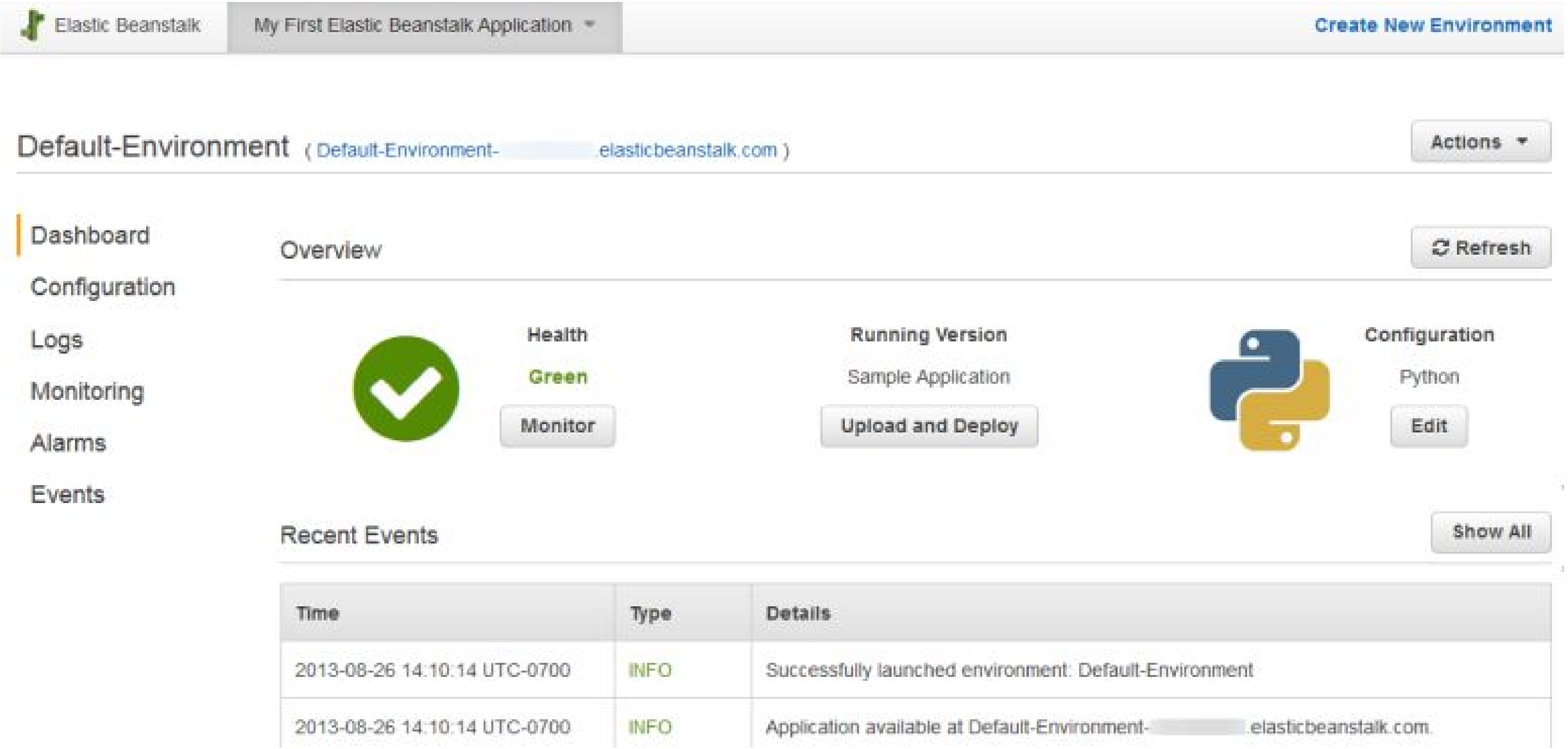
AWS: Application Environment

amazon web services
Elastic Beanstalk



My First Elastic Beanstalk Application ▸ Default-Environment ( )

AWS: Application Environment



Default-Environment (Default-Environment-[redacted].elasticbeanstalk.com)

Overview

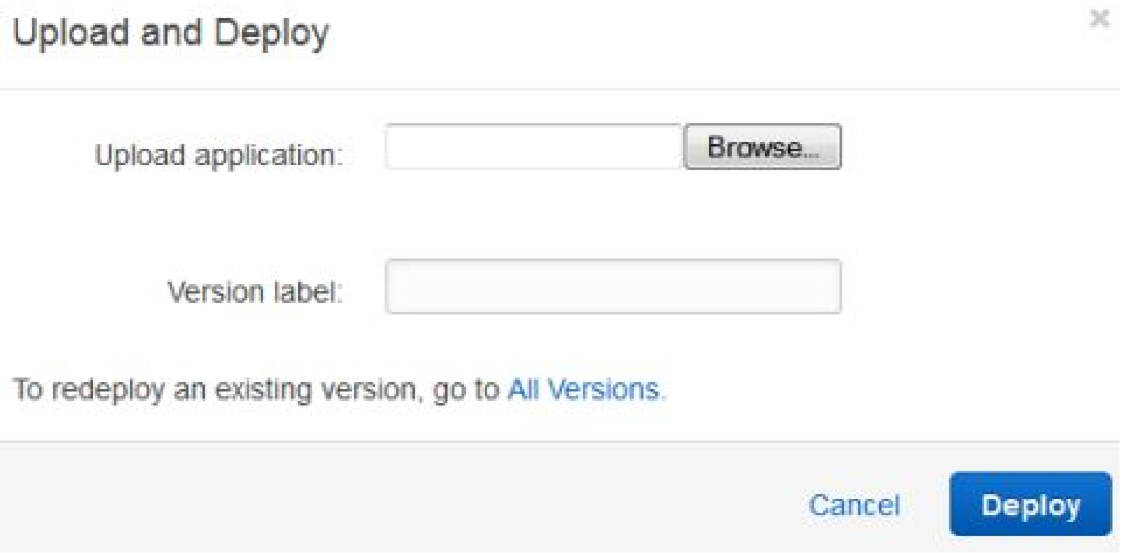
Health	Running Version	Configuration
Green	Sample Application	Python

Recent Events

Time	Type	Details
2013-08-26 14:10:14 UTC-0700	INFO	Successfully launched environment: Default-Environment
2013-08-26 14:10:14 UTC-0700	INFO	Application available at Default-Environment-[redacted].elasticbeanstalk.com.

AWS: Deploy Application

amazon web services
Elastic Beanstalk



Upload and Deploy

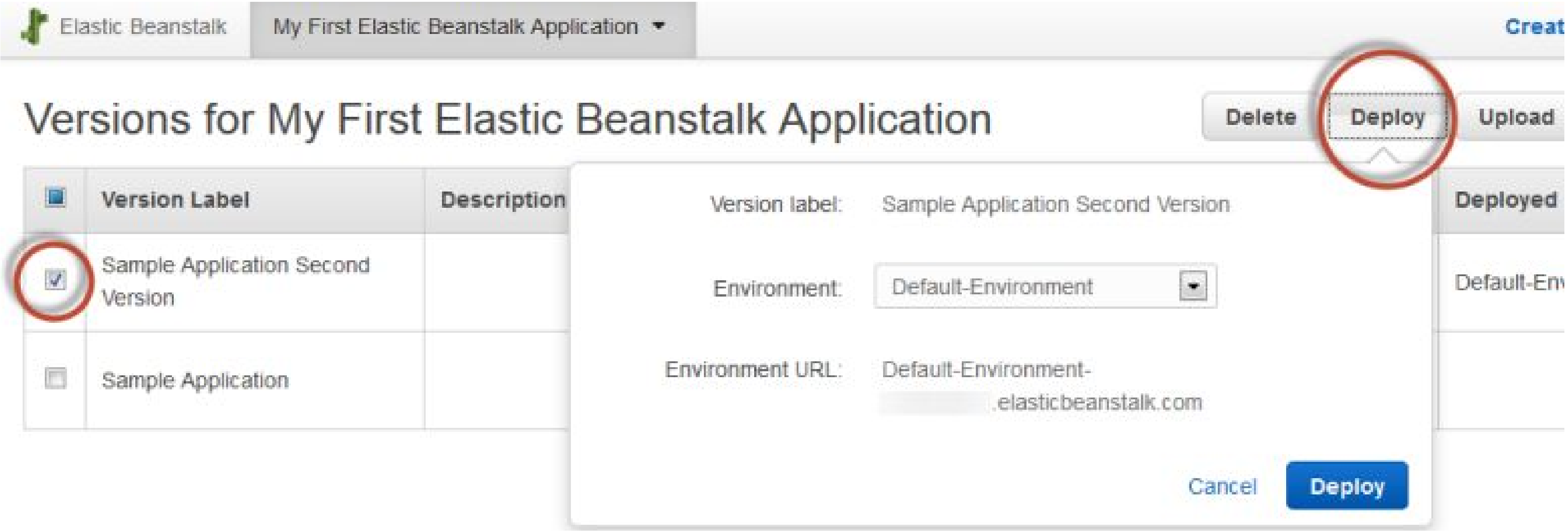
Upload application:

Version label:

To redeploy an existing version, go to All Versions.

AWS: Deploy Application

amazon web services
Elastic Beanstalk



Versions for My First Elastic Beanstalk Application

■	Version Label	Description	Deployed
☑	Sample Application Second Version		Default-Env
□	Sample Application		

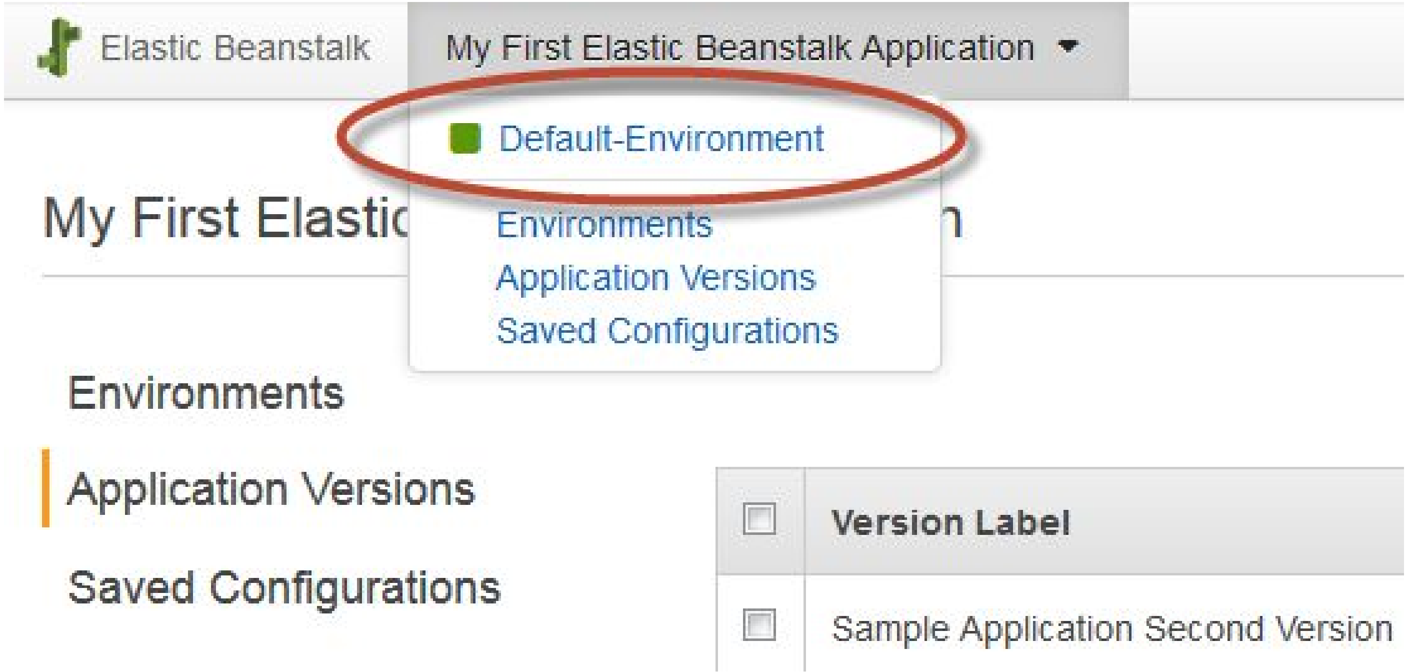
Version label: Sample Application Second Version

Environment: Default-Environment ▾

Environment URL: Default-Environment-[redacted].elasticbeanstalk.com

AWS: Modify Configuration

amazon web services
Elastic Beanstalk



Elastic Beanstalk

My First Elastic Beanstalk Application ▾

🟩 Default-Environment

Environments

Application Versions

Saved Configurations

My First Elastic ... n

Environments

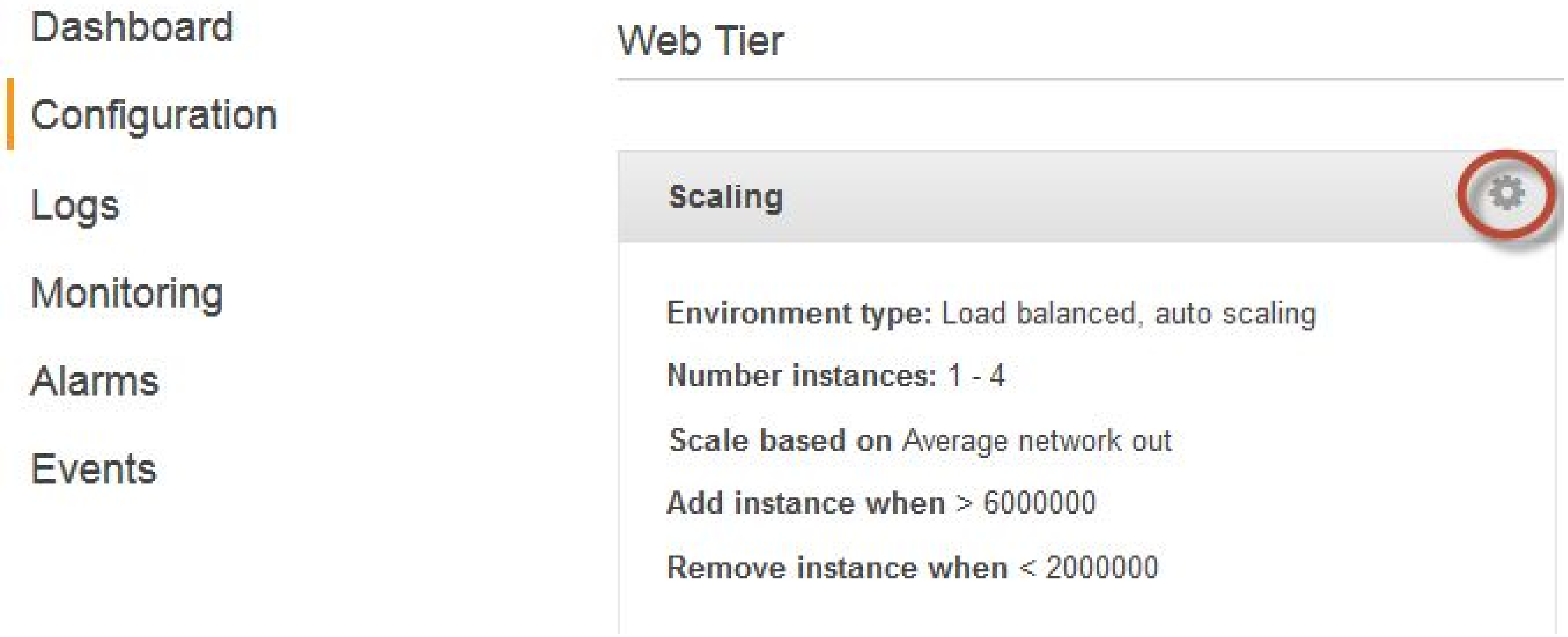
Application Versions

Saved Configurations

☐	Version Label
☐	Sample Application Second Version

AWS: Modify Configuration

amazon web services
Elastic Beanstalk



Web Tier

Scaling

Environment type: Load balanced, auto scaling

Number instances: 1 - 4

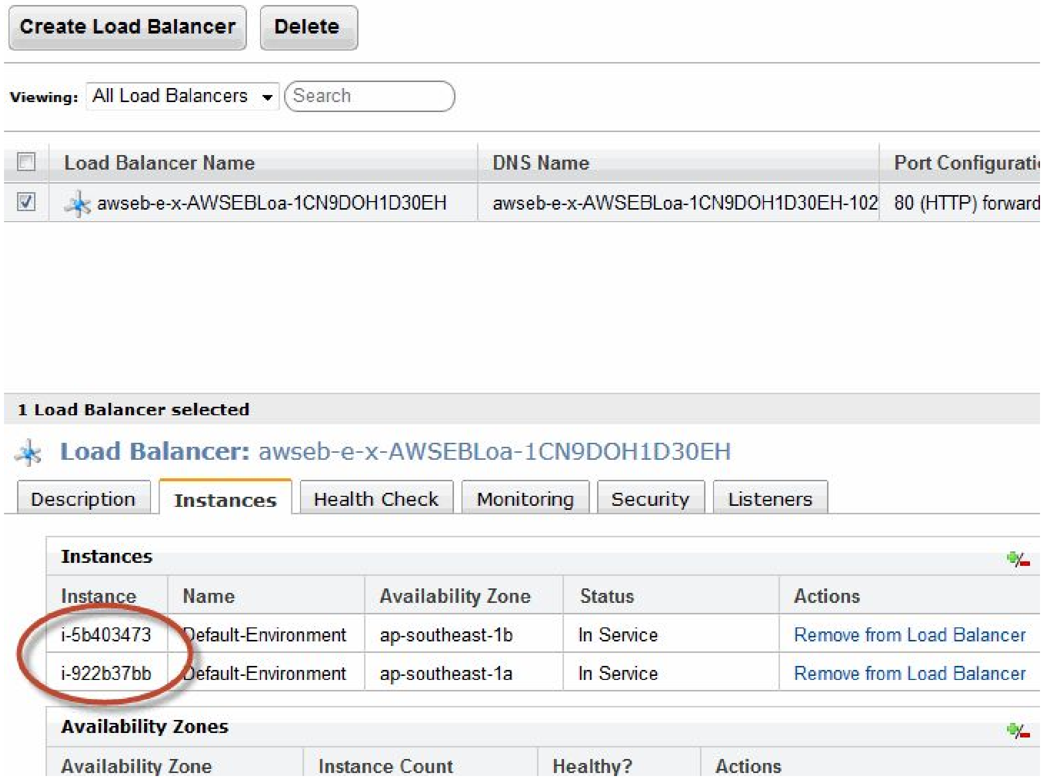
Scale based on Average network out

Add instance when > 6000000

Remove instance when < 2000000

AWS: Modify Configuration

amazon web services
Elastic Beanstalk



□	Load Balancer Name	DNS Name	Port Configurati
☑	awseb-e-x-AWSEBLoa-1CN9DOH1D30EH	awseb-e-x-AWSEBLoa-1CN9DOH1D30EH-102	80 (HTTP) forward

1 Load Balancer selected

Load Balancer: awseb-e-x-AWSEBLoa-1CN9DOH1D30EH

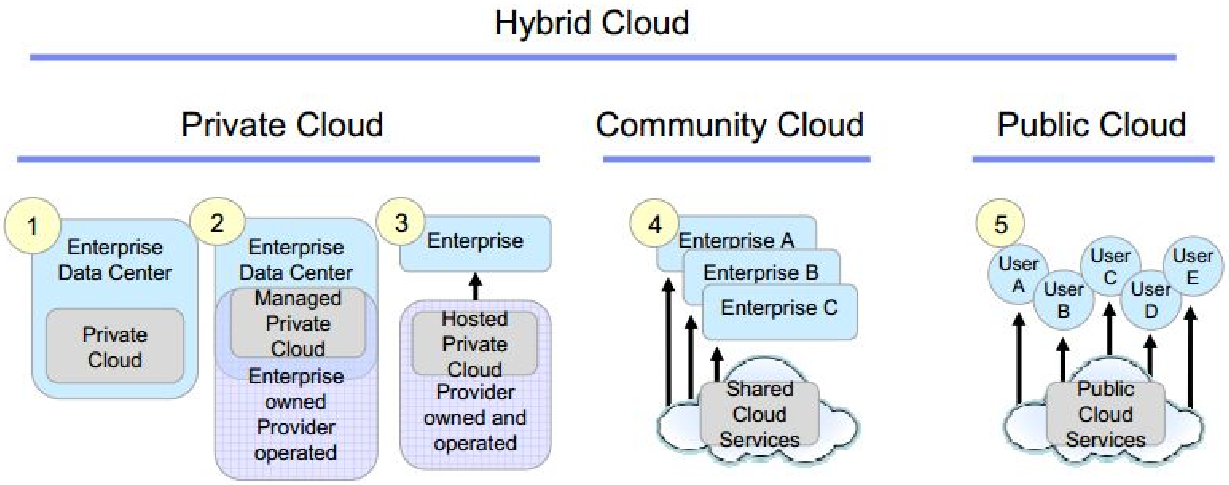
Instances

Instance	Name	Availability Zone	Status	Actions
i-5b403473	Default-Environment	ap-southeast-1b	In Service	Remove from Load Balancer
i-922b37bb	Default-Environment	ap-southeast-1a	In Service	Remove from Load Balancer

Availability Zones

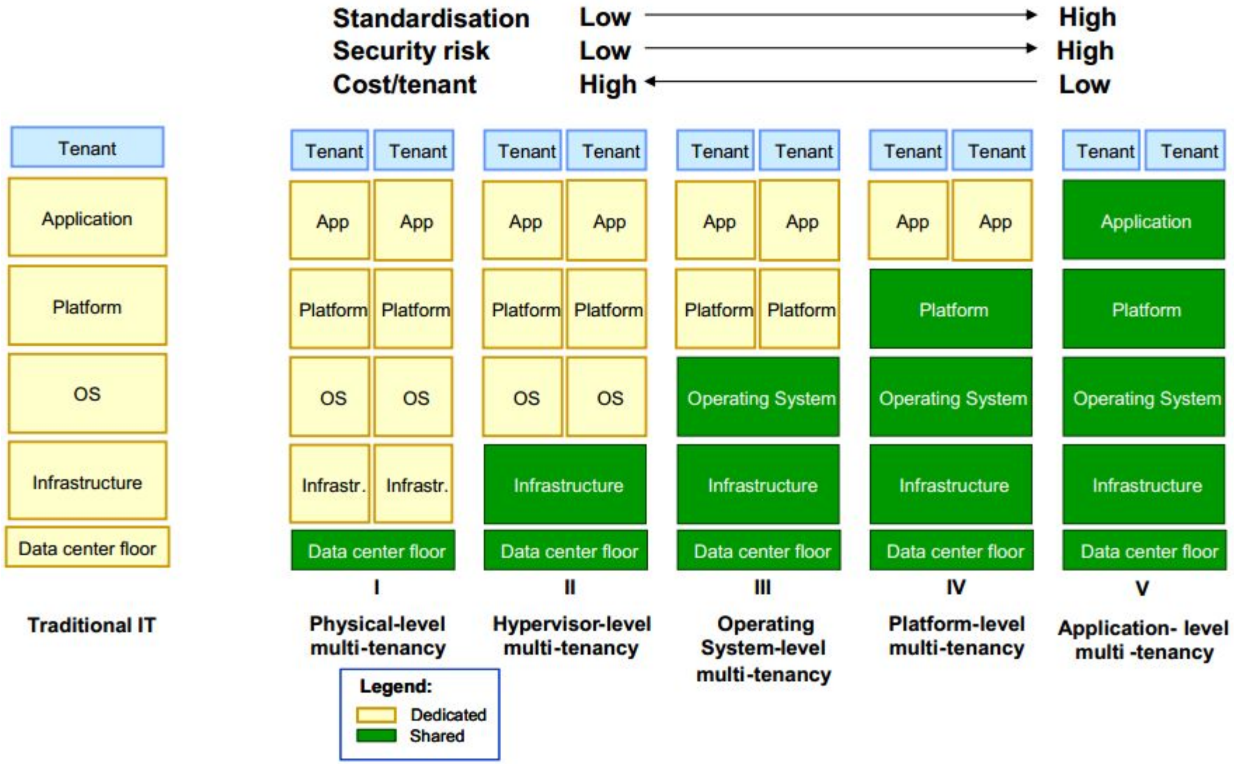
Availability Zone	Instance Count	Healthy?	Actions

AWS: Deployment Models



1. Customer managed private Cloud
2. Customer premise, provider operated private Cloud
3. Provider premise, provider operated private Cloud
4. Provider premise, provider managed, public Cloud
5. Provider premise, provider managed, provider applications, public Cloud

Multitenancy Considerations



Standardisation Low → High

Security risk Low → High

Cost/tenant High ← Low

Traditional IT	I	II	III	IV	V
Tenant	Tenant Tenant	Tenant Tenant	Tenant Tenant	Tenant Tenant	Tenant Tenant
Application	App App	App App	App App	App App	Application
Platform	Platform Platform	Platform Platform	Platform Platform	Platform	Platform
OS	OS OS	OS OS	Operating System	Operating System	Operating System
Infrastructure	Infrastr. Infrastr.	Infrastructure	Infrastructure	Infrastructure	Infrastructure
Data center floor	Data center floor	Data center floor	Data center floor	Data center floor	Data center floor
Traditional IT	Physical-level multi-tenancy	Hypervisor-level multi-tenancy	Operating System-level multi-tenancy	Platform-level multi-tenancy	Application- level multi -tenancy

Legend:

🟨 Dedicated

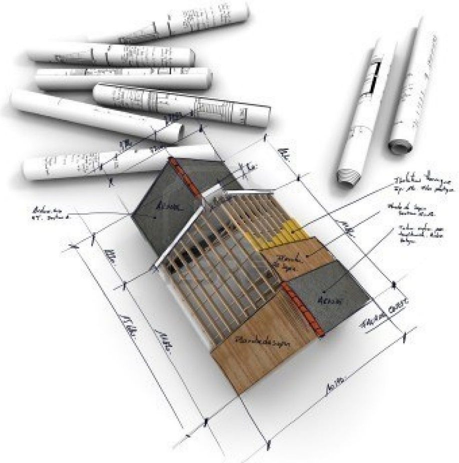
🟩 Shared

Architect Applications for the Cloud

1. Virtualize the Application Stack

2. Componentize, decouple & design all components as a 'black box'

3. Design for Scalability



Design for Scalability

Traditional way

- add more RAM
- use faster servers
- expensive ‘micro-optimization’
- complex caching
- faster hard disks

Cloud Applications

- minimize mutable state
- create asynchronous services
- alternative data stores
- automate deployment

Design for Failure

"Everything fails, all the time“

Werner Vogels, CTO Amazon.com

- find single point of failures
- evaluate scenarios. What levels of risk is acceptable?
- failure tolerance

Minimize Mutable State

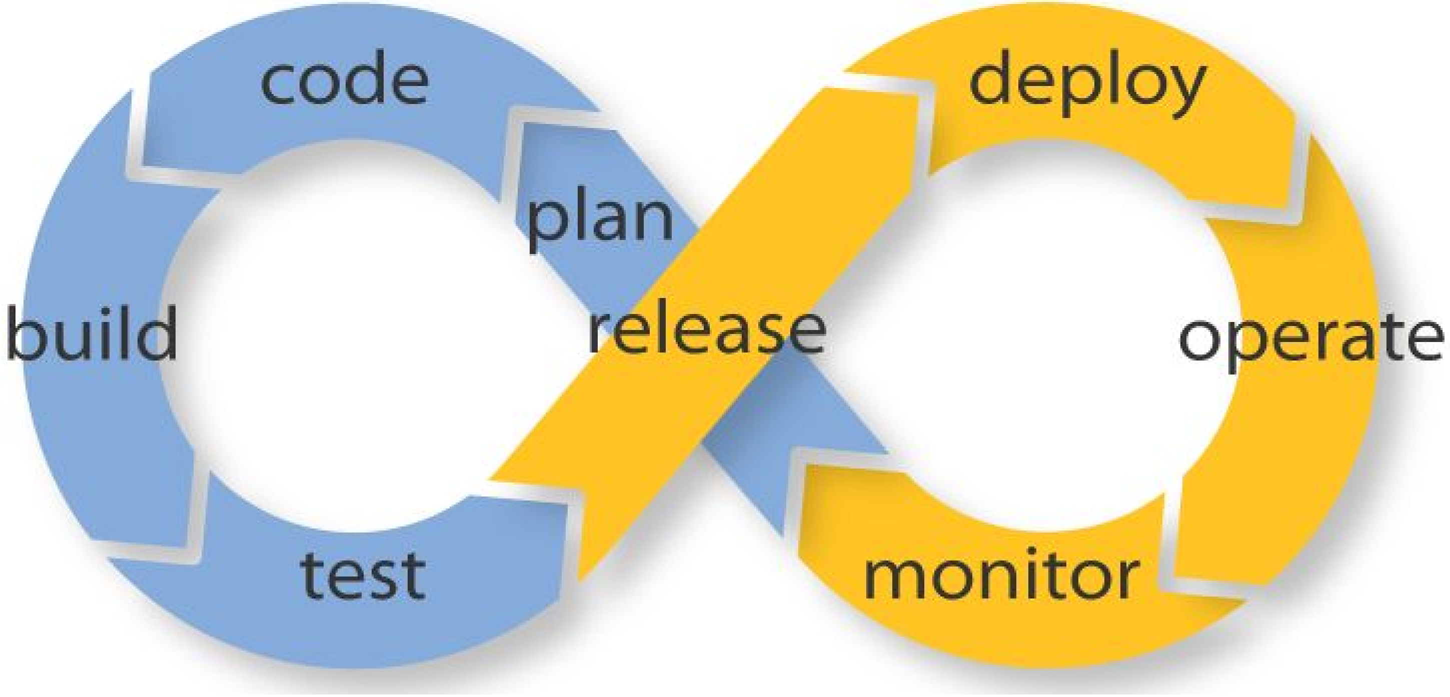
Variables shared across application

- Multiple servers and processes trying to update the same variables at the same time result in deadlocks, time-outs, and failed transactions
- minimize or eliminate those in webservers, application and the database
- specific considerations for filesystems, applications and datastores
- look at cluster filesystems, object stores, NoSQL / CouchDB, MongoDB – asynchronous ‘fire & forget’ updates

Components & Asynchronous Services

- Offload work from main application servers – Web 2.0
- Break tasks into separate services, run by different components
- Scale independently
- Use message queues for guaranteed delivery

Automate Deployment - DevOps



Key Takeaways

1. **OpenStack** deals with Cloud Infrastructure

2. As a developer, your friends are **platform services**

3. **Design Applications** for the cloud - scalability & anticipate failure

Thank you.