Apache Stratos
Building a PaaS using OSGi and Equinox

Paul Fremantle
CTO and Co-Founder, WSO2
Committer, Apache Stratos

@pzfreo #wso2 #apache
paul@wso2.com
pzf@apache.org
About me

- CTO and Co-Founder WSO2
- Working in Apache for 14 years
- Working with Cloud, SOA, APIs, MQTT, IoT
- Also presenting tomorrow about IoT security (M2M day 14:15)
Disclaimer

Apache Stratos is an effort undergoing incubation at The Apache Software Foundation (ASF). Incubation is required of all newly accepted projects until a further review indicates that the infrastructure, communications, and decision making process have stabilized in a manner consistent with other successful ASF projects. While incubation status is not necessarily a reflection of the completeness or stability of the code, it does indicate that the project has yet to be fully endorsed by the ASF.
I hired a consultant to help us evolve our products to cloud computing.

Blah blah cloud.
Blah blah cloud.
Blah blah cloud.

It's as if you're a technologist and a philosopher all in one!

Blah blah platform.
Today’s Buzzword Bingo!

<table>
<thead>
<tr>
<th>PaaS</th>
<th>Cloud</th>
<th>Elastic</th>
<th>Policy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto-Scale</td>
<td>DevOps</td>
<td>jClouds</td>
<td>Puppet</td>
</tr>
<tr>
<td>Platform</td>
<td>IaaS</td>
<td>Chef</td>
<td>Cassandra</td>
</tr>
<tr>
<td>Multi-tenant</td>
<td>Load Balancer</td>
<td>Governance</td>
<td>Monitoring</td>
</tr>
</tbody>
</table>
What is a PaaS anyway?
Gartner Reference Model for Platform as a Service

**PaaS Service Offerings**
- Application Platform
- Integration Platform
- Business Process Management Platform
- Cloud Database Platform
- User Experience Platform
- Other

**PaaS Technology Core**
- Cloud Value Foundation
  - Shared Resources, Multitenancy, Self-Service, Elasticity, Real-Time Versioning, Metadata Management, Subscription/Use Billing
- Cloud Performance Foundation
  - In-Memory Computing, Grid/Massive Scale, Auto-Scaling, SLA Enforcement, Use Tracking, High Availability, Security, Data Integrity, Parallel Processing

**System Infrastructure or System Infrastructure Services (IaaS)**

**Hardware**
What is Cloud?

- Depends who you are
  - My daughter: iCloud (her music in the cloud)
  - My mum: gmail (her email in the cloud)
  - My VP sales: Salesforce (his prospects in the cloud)
  - Sysadmin: Amazon/Rackspace/etc (his infrastructure in the cloud)
  - *: what you care about, self-provisioned, managed, metered and paid per use, in the cloud
PaaS is a cloud for developers

- Provisioning stuff developers care about:
  - Not VMs, networks and disk (though that might still be useful)
  - Application environments, databases, queues
  - Dev / Staging / Production
  - Different runtimes:
    - Java Servlet/Tomcat, Node.js, PHP, Python
    - MySQL, MongoDB, Cassandra
    - Etc
  - Working with Git, SVN, Jenkins, etc
Who needs a PaaS anyway?

- Anyone who wants to provision application environments on-the-fly
- A basis for multi-tenant SaaS applications
- Elastic scaling and management of the infrastructure
- Application based monitoring of the environment
- More intelligent use of resources
What is Apache Stratos?

- A Platform-as-a-Service (PaaS) Framework
- Built on OSGi and Equinox
- Donated to Apache Software Foundation
  - Committers from Cisco, Citrix, SunGard, Indiana U, NASA JPL, etc
- Deploys onto an Infrastructure-as-a-Service (IaaS)
  - Including Amazon EC2, VMWare vCloud, OpenStack...etc
- Creates a secure, multi-tenant, elastic, metered, billed PaaS
  - Supports private, public or hybrid PaaS
Quick Demo

- Warning: this is alpha software
- Running on Amazon
- Accessed via Conference wifi
- Hmmm? Ok who’s idea was that
The most comprehensive enterprise grade PaaS Framework

Login To Create Cartridges

paul@freo.me

Password

Sign In

Stay Signed In

Create an account
Why OSGi?

- A set of servers that need to be deployed in different architectures:
  - Single JVM for developer testing / small footprint
  - Scale-out for large scale deployment
- Re-usable components from a wider set of middleware
  - Message Broker
  - HTTP Load Balancer
  - Identity Services
  - Logging and Metering
  - etc
Carbon Component Manager

<table>
<thead>
<tr>
<th>Name</th>
<th>version</th>
</tr>
</thead>
<tbody>
<tr>
<td>WSO2 Carbon - Logging Management Feature</td>
<td>3.0.0.SNAPSHOT</td>
</tr>
<tr>
<td>WSO2 Carbon - MXM Module Feature</td>
<td>3.0.0.SNAPSHOT</td>
</tr>
<tr>
<td>WSO2 Carbon - Security Management Feature</td>
<td>3.0.0.SNAPSHOT</td>
</tr>
<tr>
<td>WSO2 Carbon - Service Management Feature</td>
<td>3.0.0.SNAPSHOT</td>
</tr>
<tr>
<td>Service Hosting Feature</td>
<td>3.0.0.SNAPSHOT</td>
</tr>
<tr>
<td>WSO2 Carbon - Statistics Feature</td>
<td>3.0.0.SNAPSHOT</td>
</tr>
<tr>
<td>WSO2 Carbon - Tools Feature</td>
<td>3.0.0.SNAPSHOT</td>
</tr>
<tr>
<td>WSO2 Carbon - JMS Transport Core Feature</td>
<td>3.0.0.SNAPSHOT</td>
</tr>
<tr>
<td>WSO2 Carbon - Mail Transport Core Feature</td>
<td>3.0.0.SNAPSHOT</td>
</tr>
<tr>
<td>WSO2 Carbon - Transport Management Feature</td>
<td>3.0.0.SNAPSHOT</td>
</tr>
</tbody>
</table>
More on Carbon

- Based on Equinox
- Using p2 feature model
  - Allows components to be managed and deployed from either local or remote repo
  - Handles versions and dependencies
  - Supports rollback
- Command line p2 management and OSGi Console
- Consistent model for scaling
  - Master/Slave + Deployment Synchronizer
- Consistent Registry/Repository
- Security model
Lessons learnt from Carbon

- OSGi is a bit of a pain sometimes
  - We hide as much as possible from end-users
- Used properly it enables massive productivity gains
  - Small company with > 15 products and world class deployments
- Small teams can be productive
- End users can enhance the products with their own components
- Minification
- Scaling out
What does a PaaS look like

> stratos subscribe-cartridge php myphp
--repo-url https://github.com/pzfreo/php-s4
--deployment-policy economy

  Subscribes to a runtime (e.g. php)

> git push php-s4

  Auto deploys the code into the cartridge
Stratos Layered Architecture

Apache Stratos 4.0.0 Layered Architecture

Cartridges
- Ruby and Rails
- Java Tomcat Cartridge
- MySQL Cartridge
- PHP Cartridge
- Any Pluggable Cartridge

Stratos Load Balancer
- Other Load Balancers
- Identity Service
- Logging Service
- Metering Service

Stratos PaaS Framework
- Stratos Manager
- Cloud Controller
- Auto Scaler
- CLI/Web UI
- Artifact Distribution Coordinator
- Complex Event Processor

Message Broker

jclouds

IaaS
- Infrastructure as a Service (OpenStack, vCloud, EC2 etc.)
**Stratos Cartridges**

- A component which can be plugged into Stratos, so that it can use Stratos core services of the framework layer

- Cloud-aware platform environment extending legacy technologies into the cloud and delivering cloud benefits

- Users/DevOps may create custom cartridges and host any application, container, or framework in a Stratos Cloud. (Eg: node.js, Ruby, MongoDB, Wordpress, .net, Oracle WebLogic or JBoss SOA Platform..etc)

- Single tenant or multi-tenanted
Stratos Cartridges

VM Cartridge
- Service / Framework / Application
- Guest OS

LXC Cartridge
- Service / Framework / Application
- Stratos Agent

Host Operating System
Puppet

- Puppet
  - Powerful devops and provisioning tool
  - Used by the cartridges to auto-provision
  - Today to create a cartridge you basically build a puppet script
  - Next step is that will be a generic Puppet cartridge
Multi-factored Auto Scaling

- Integrate both real-time and rule-based decision making
- Scaling algorithm can use multiple-factors
- Capable of predicting future load
Scalable and Dynamic Load Balancing

- Comes with an HTTP Load Balancer as a cartridge
- Service level load balancing
- Auto Scaling load balancers
- External load balancer integration support
  - Supports HAProxy for non-HTTP loads
Cloud Bursting

Cloud Controller
  jclouds

Message Broker
  Topology

Autoscaler
  Rules Engine

EC2
  Load balancer - PHP - EC2
  Cartridge Instance 1 PHP
  Cartridge Instance 2 PHP

OpenStack
  Load balancer - PHP - OpenStack
  Cartridge Instance n PHP

new topology

create/destroy instances

pub
Controlling IaaS Resources

Partitions

- Logically group IaaS resource locations
- Partitions are important to make application high availability
- Cartridge instances are spawned inside these partitions
- Partitions are defined by DevOps

Network Partitions

- Logical groups of multiple partitions, that are in the same network.
- Stratos will spawn Load Balancers per network partition.
- Since LB instances and cartridge instances reside in same network, they can communicate using private IP addresses.
- Used in deployment policies.
Smart Policies

Auto Scaling Policy

- Defines threshold values pertaining to scale up/down decision
- Auto scaler will refer this policies
- Defined by DevOps
- During cartridge subscription user specifies the auto scaling policy to be used

Deployment Policy

- Defines how and where to spawn cartridge instances
- Defines minimum and maximum instances in a selected service cluster
- Defined by DevOps based on deployment patterns
- Guarantee SLA of cloud applications
More highlights...

- Unified communication
- Centralized monitoring and metering
- Web UI and CLI
- REST API for integration with external PaaS management interface for both DevOps and user interaction
- Artifact distribution coordinator
- Persistence volume support for cartridges
- Gracefully shutdown instance when scale down
App Factory
Apache License Open Source
Why Stratos?

- Inherently Multi-tenant
- Highly Extensible
  - OSGi components, AMQP-based architecture, Policies, IaaS, etc
- Built in Single Sign On/Identity management model
- Policy based
- Supports a simple model of partitions
- Smart Load Balancing
- Non-HTTP Load balancing
- LXC support
- Apache Foundation
- App Factory takes it to the next level
Want to help?

- Feedback and testing
- Creating Cartridges
  - Existing apps too (e.g. Drupal, Wordpress, etc)
- Porting to Google Compute
- Autoscaling policies and rules
- Ease of use – creating a Vagrant/LXC distro
- Help with App Factory
Stratos @ Google Summer of Code

6 proposals
- Python Agent
- AWS LB support
- GCE LB support
- New scaling algorithms
- New LB algorithms
- GCE support
Resources

- Hangout archives (and future hangouts)
- Carbon:
  - [http://www.slideshare.net/wso2.org/introduction-to-the-wso2-carbon-platform-webinar](http://www.slideshare.net/wso2.org/introduction-to-the-wso2-carbon-platform-webinar)
- Stratos