Smalltalk in the Cloud

ESUG 2012, Friday, 31 August 2012, 11:15 – 11:45 AM, Gent, Belgium

James Foster, Sr. Member Technical Staff, VMware

Abstract

Cloud Foundry is the open "Platform as a Service" (PaaS) project initiated by VMware. It can support multiple frameworks, multiple cloud providers, and multiple application services all on a cloud scale platform.

In this presentation we show how Smalltalk can fit in this environment.

Agenda

- Overview of Cloud Foundry (corporate marketing slides)
- Smalltalk use described with demo
- Experience report from Tim Felgentreff at HPI

Hosting Options

What is provided by vendor:

Type	Utilities	Hardware	Stack	Applications
Self-hosting				
Data center	✔			
IaaS	✔	✔		
PaaS	✔	✔	✔	
SaaS	✔	✔	✔	✔

Utilities: Network, power, A/C

Hardware: CPU, RAM, disk

Stack: operating system, web server, database, runtime, framework

Application: this is the programmer's concern

Introducing Cloud Foundry™
The Open Platform as a Service

March 2012

www.cloudfoundry.com

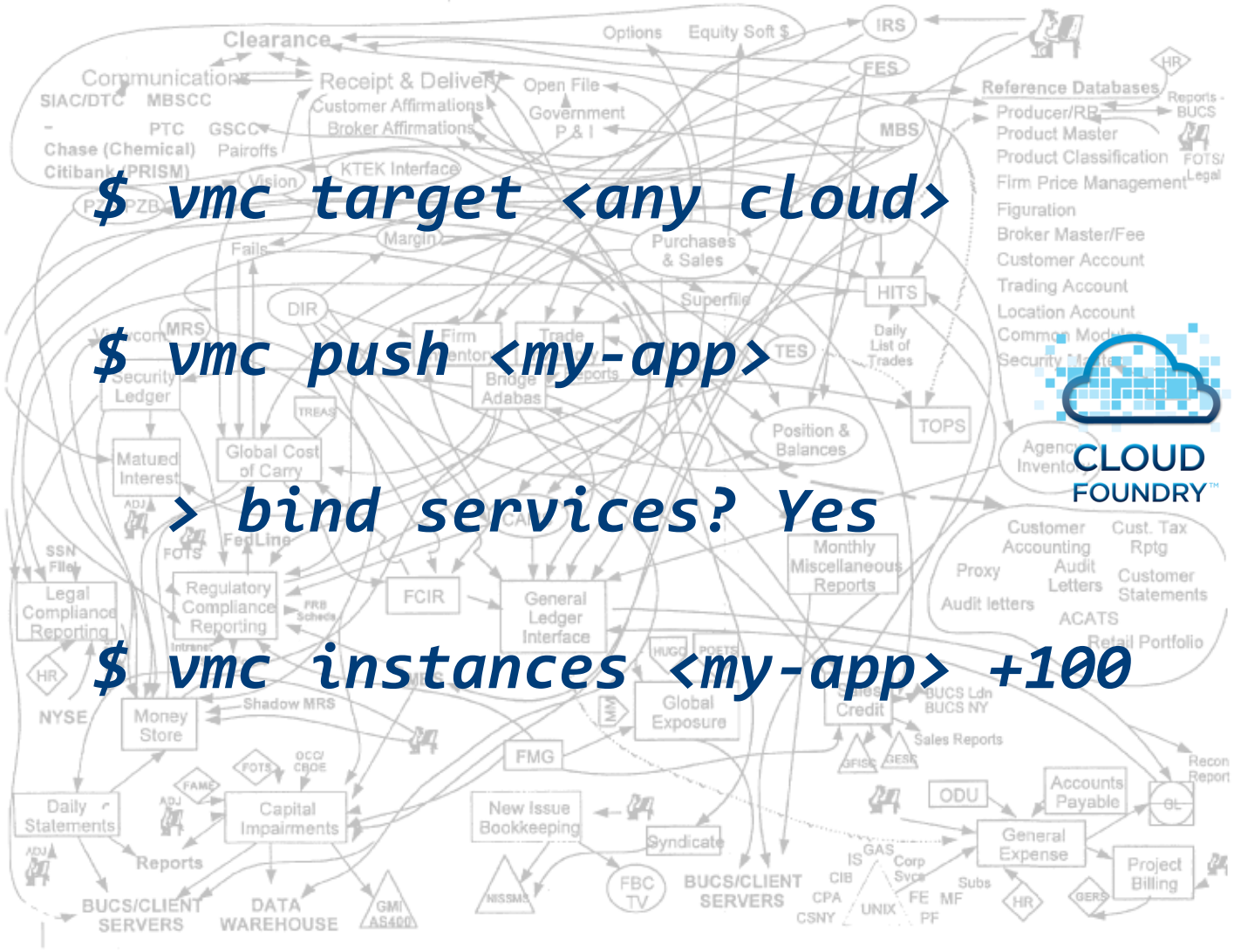
vmware

The Open Platform as a Service

Deploy and scale applications in seconds, without locking yourself into a single cloud

Simple, Open, Flexible, Scalable

What if…



vmware

Cloud Foundry open Platform as a Service

The PaaS of choice for the Cloud era

Simple

- Lets developers focus on their code and not wiring middleware

Open

- Avoid lock-in to specific cloud, frameworks or service
- Completely open source from day one

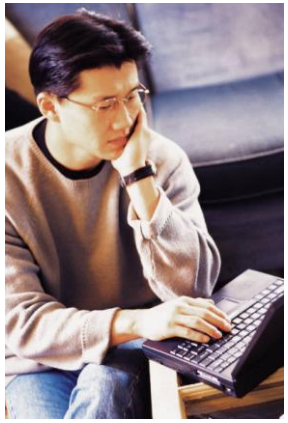
Flexible and Scalable

- Self service, deploy and scale your applications in seconds
- Extensible architecture to “digest” future cloud innovation

Cloud Foundry – key audiences

IT Developers

"Write code, not tickets"



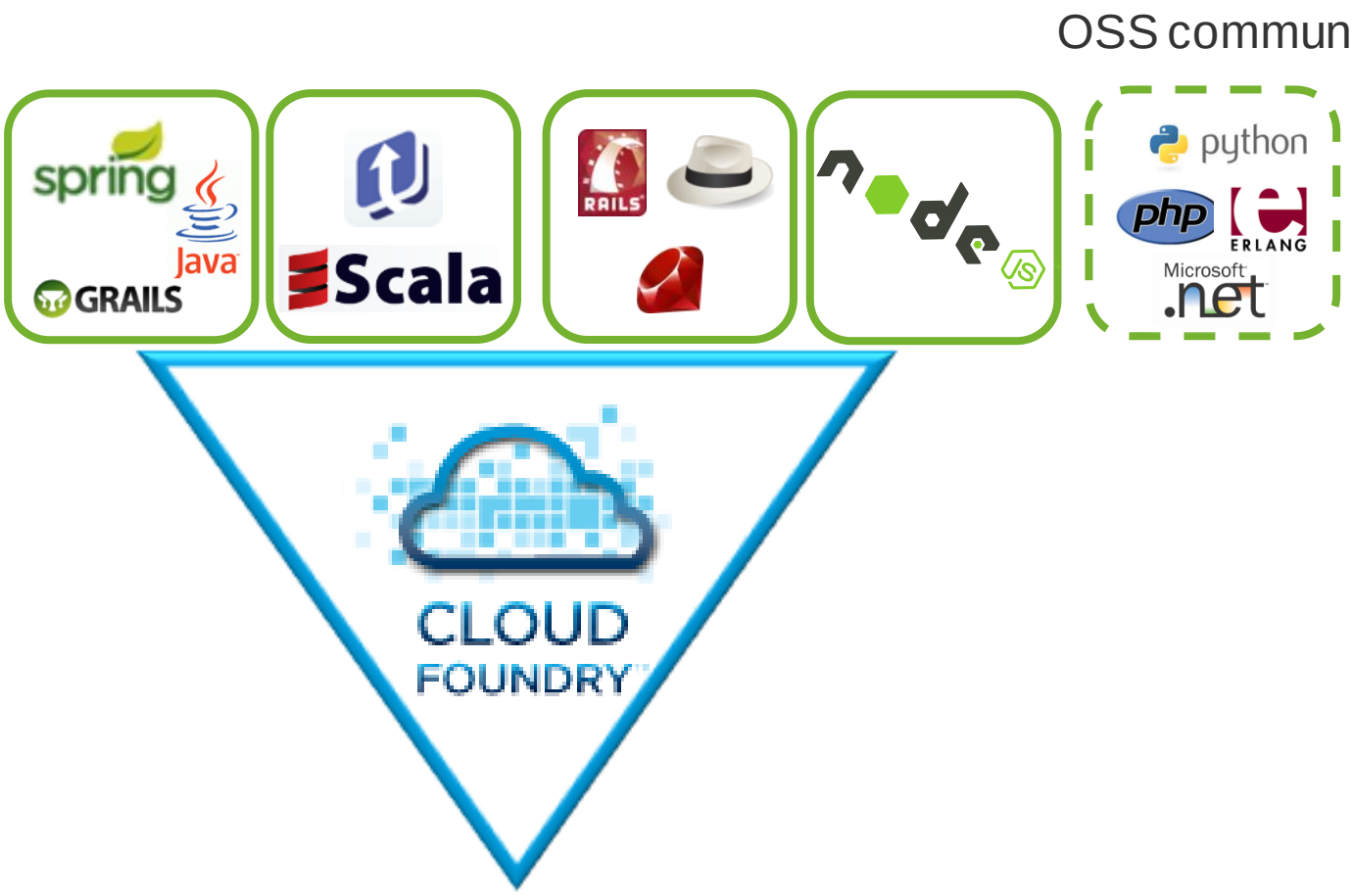
- ✓ Friction-free deployment
- ✓ No machines or middleware to manage
- ✓ Latest high productivity frameworks
- ✓ Choice of application services
- ✓ Cloud portability

IT Operations

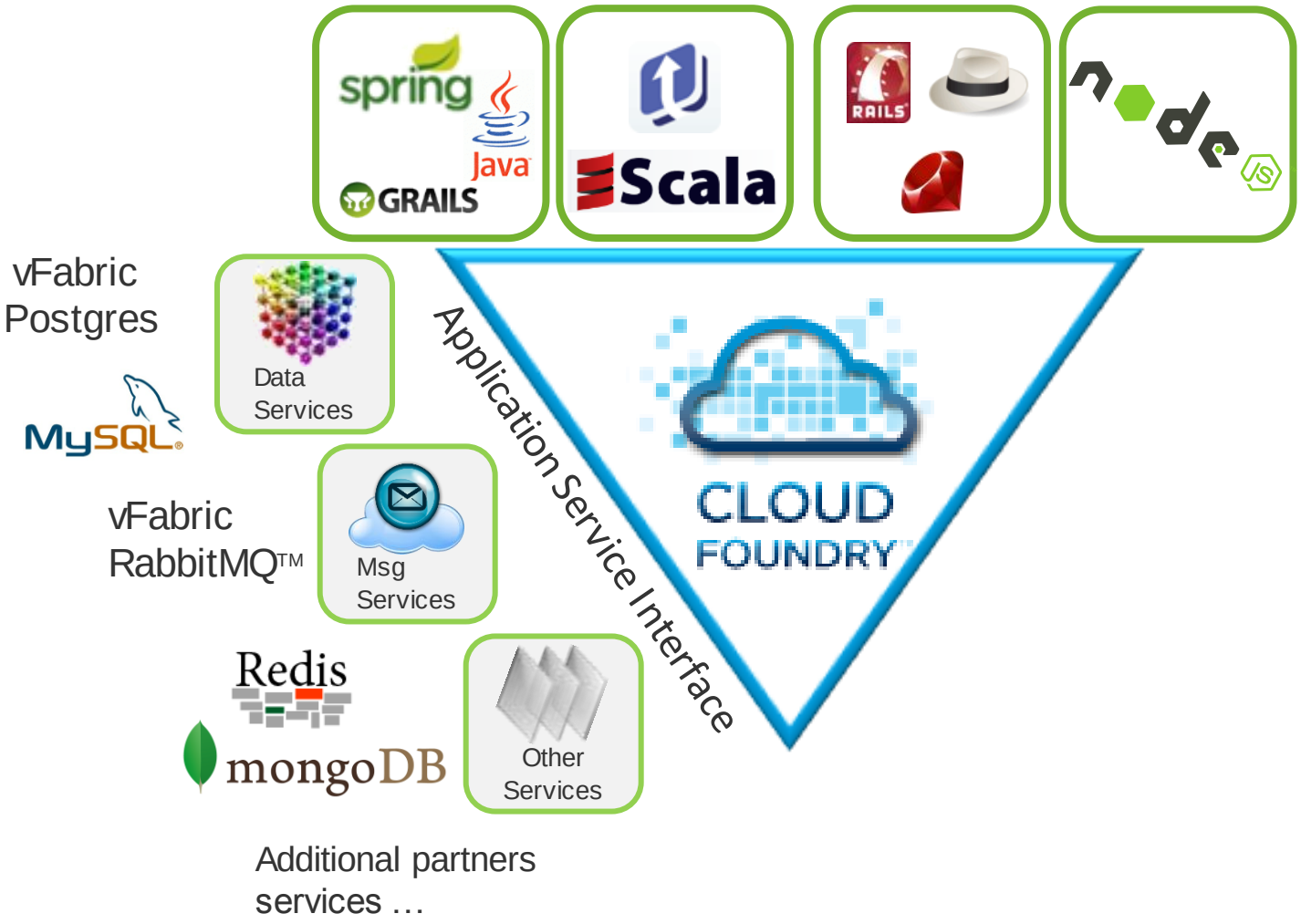
"IT as a service provider"

- ✓ More responsive to developers
- ✓ Elastic and dynamically scalable
- ✓ Improved efficiency
- ✓ Digest future cloud advances
- ✓ Cloud portability

Cloud Foundry open PaaS - Choice of frameworks

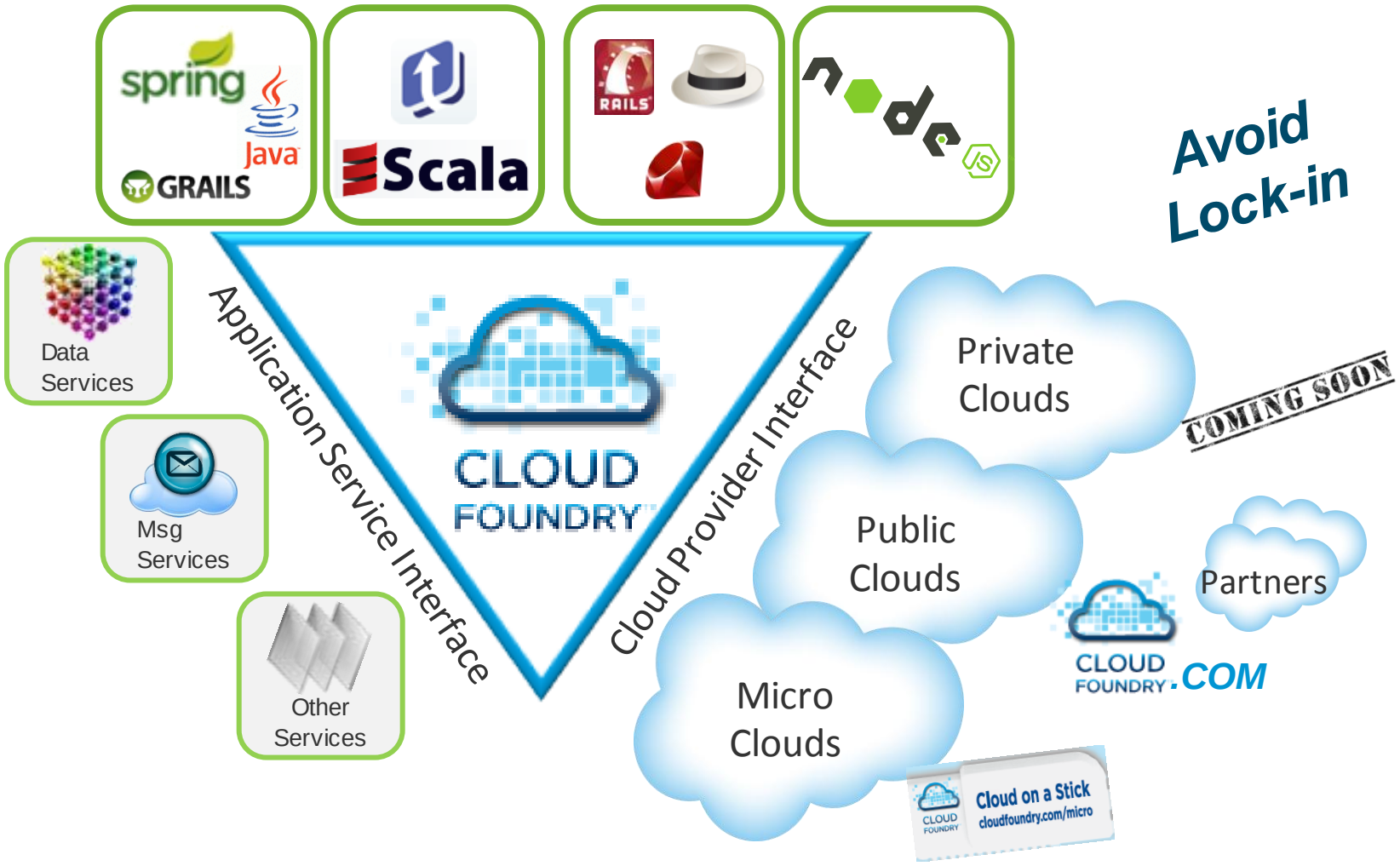


Cloud Foundry open PaaS - Choice of application services



vmware

Cloud Foundry open PaaS - Choice of clouds



vmware

Multi-cloud flexibility is critical to your long-term success

- Make use of both public and private clouds without rewriting your applications
- Protect against vendor lock-in
- Meet different compliance and geographical needs
- Accommodate peak loads while optimizing costs
- Manage your growth and changing needs over time

Cloud Foundry: Making multi-cloud a reality

Choice of Private Cloud Distributions

ActiveState

DELL

Cloud on a Stick
cloudfoundry.com/micro

CANONICAL

ServiceMesh

nimbula

OPSCODE

SCALR

Choice of Public Cloud Providers

appfog

CLOUD FOUNDRY.COM

ENSTRATUS

RIGHT SCALE

bluelock.

Tier 3
Tech Support

virtacore SYSTEMS

Choice of Cloud Infrastructure

openstack

amazon web services

the rackspace cloud

vmware vCLOUD POWERED

VMware vSphere

Bare metal

Eucalyptus

CloudFoundry.COM - Multi-tenant PaaS operated by VMware

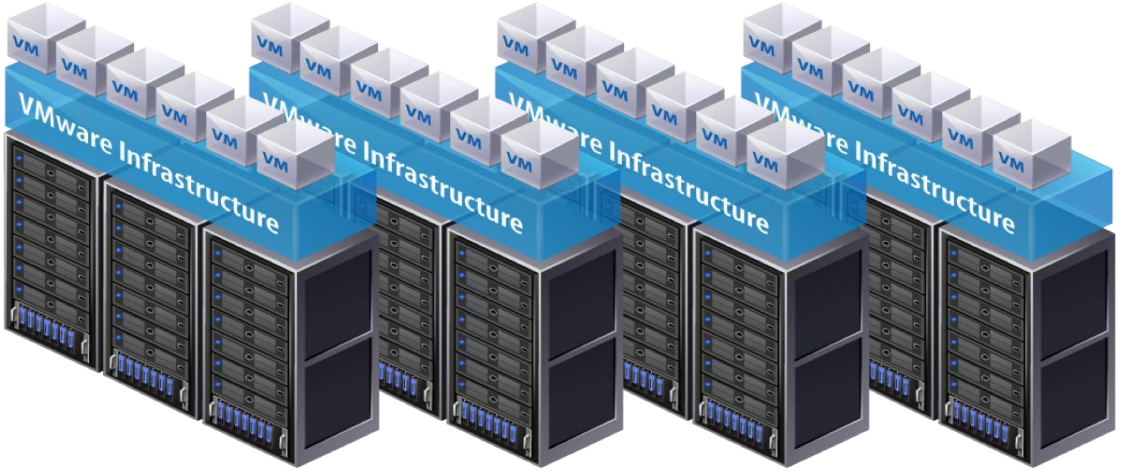
CloudFoundry.COM (beta)

Runtimes & Frameworks

Services

vCenter / vSphere

Infrastructure



vmware

Micro Cloud Foundry™ – Industry first downloadable PaaS

Micro Cloud Foundry

Runtimes & Frameworks

Services

Your Laptop/PC

Single VM instance of
Cloud Foundry
that runs on a developer's
MAC or PC

CloudFoundry.ORG - Community open-source project

CloudFoundry.ORG

Java spring node.js RAILS python php ERLANG Microsoft .net Scala RabbitMQ Messaging that just works mongoDB redis MySQL PostgreSQL

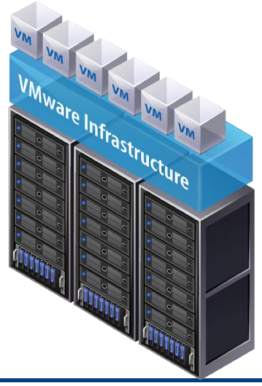
Your Infrastructure

Download Code

github SOCIAL CODING

Apache2 license

Setup Environment



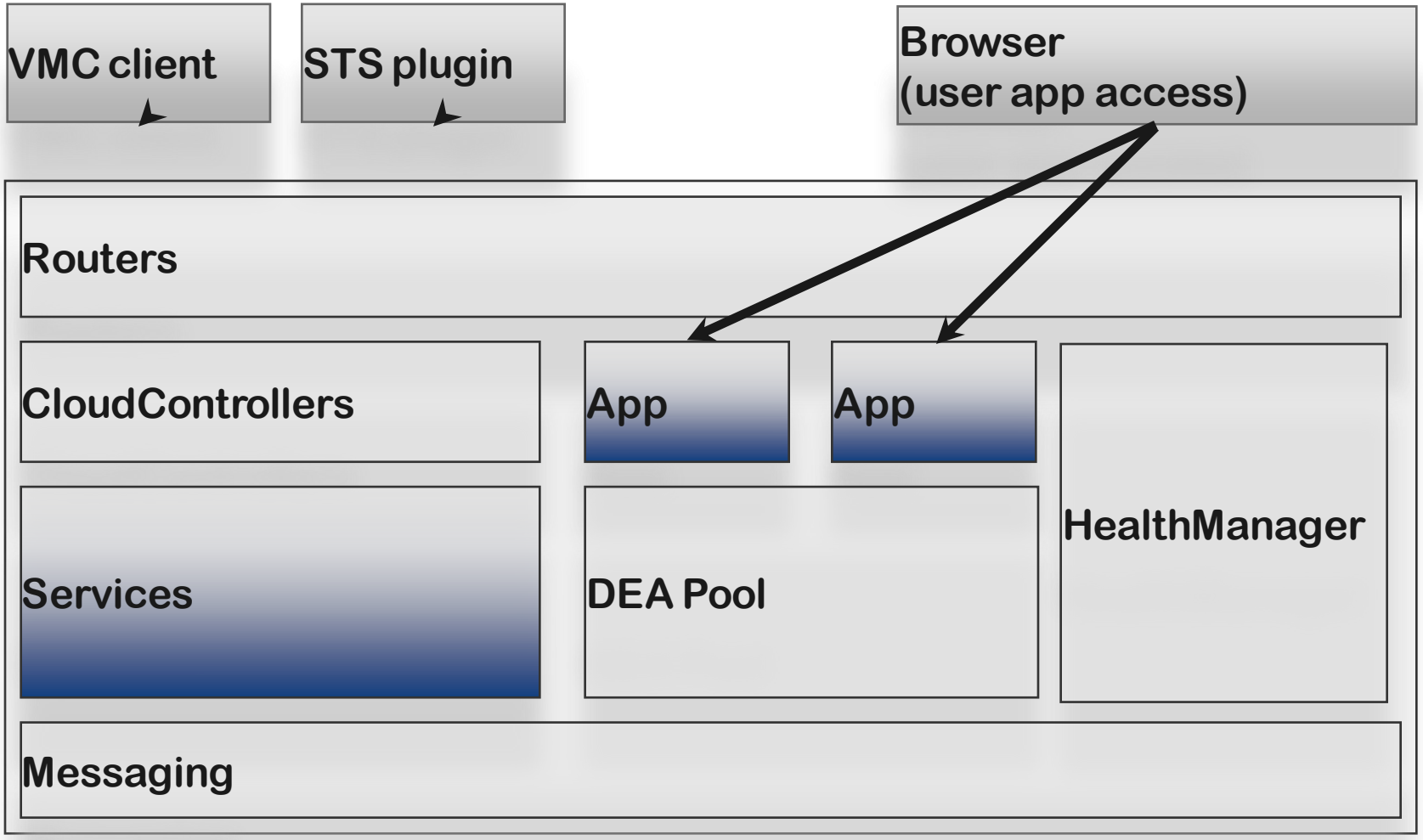
Setup Scripts

Chef

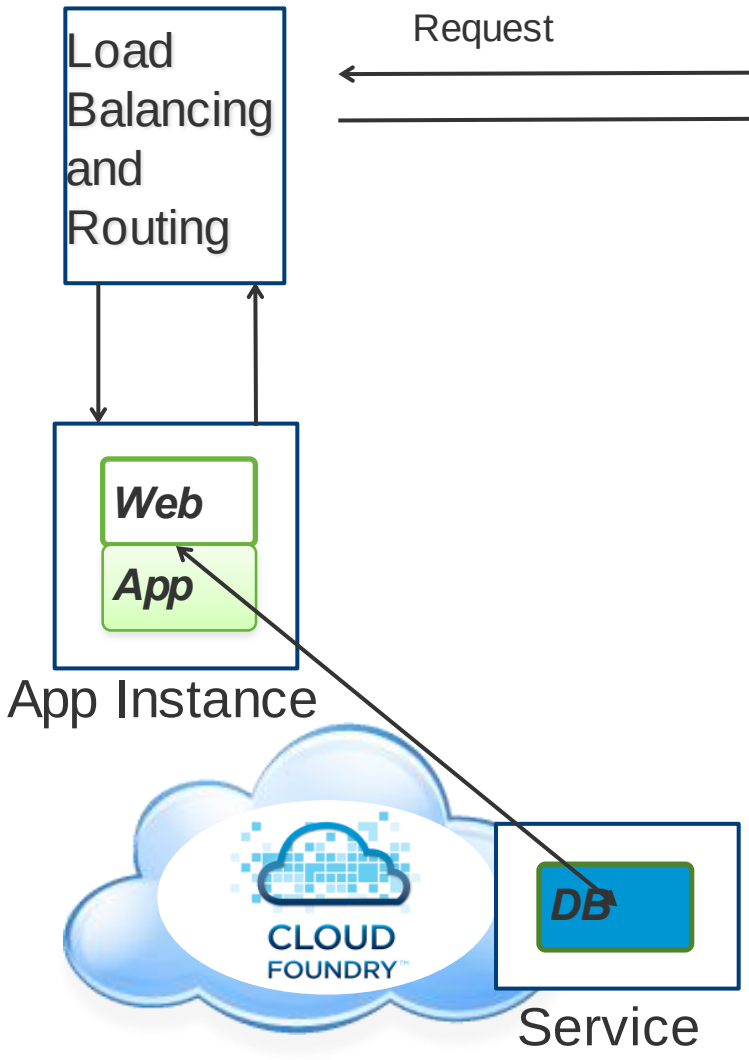
Deploy Behind Firewall

vmware

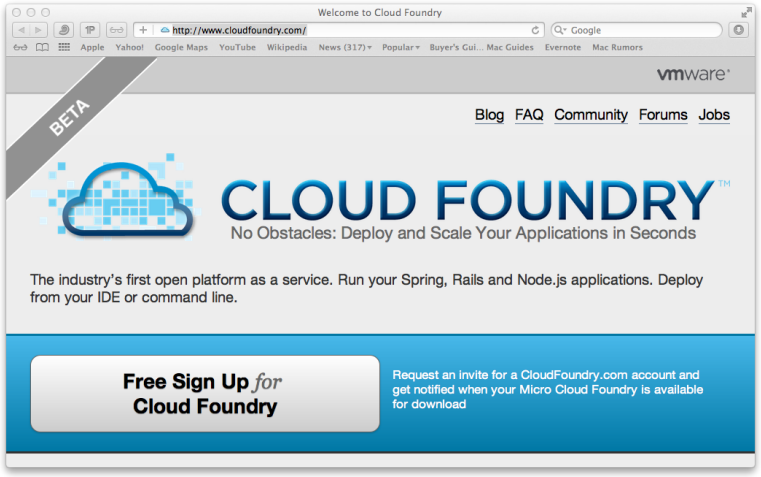
Cloud Foundry Logical View



How Apps are Accessed on Cloud Foundry

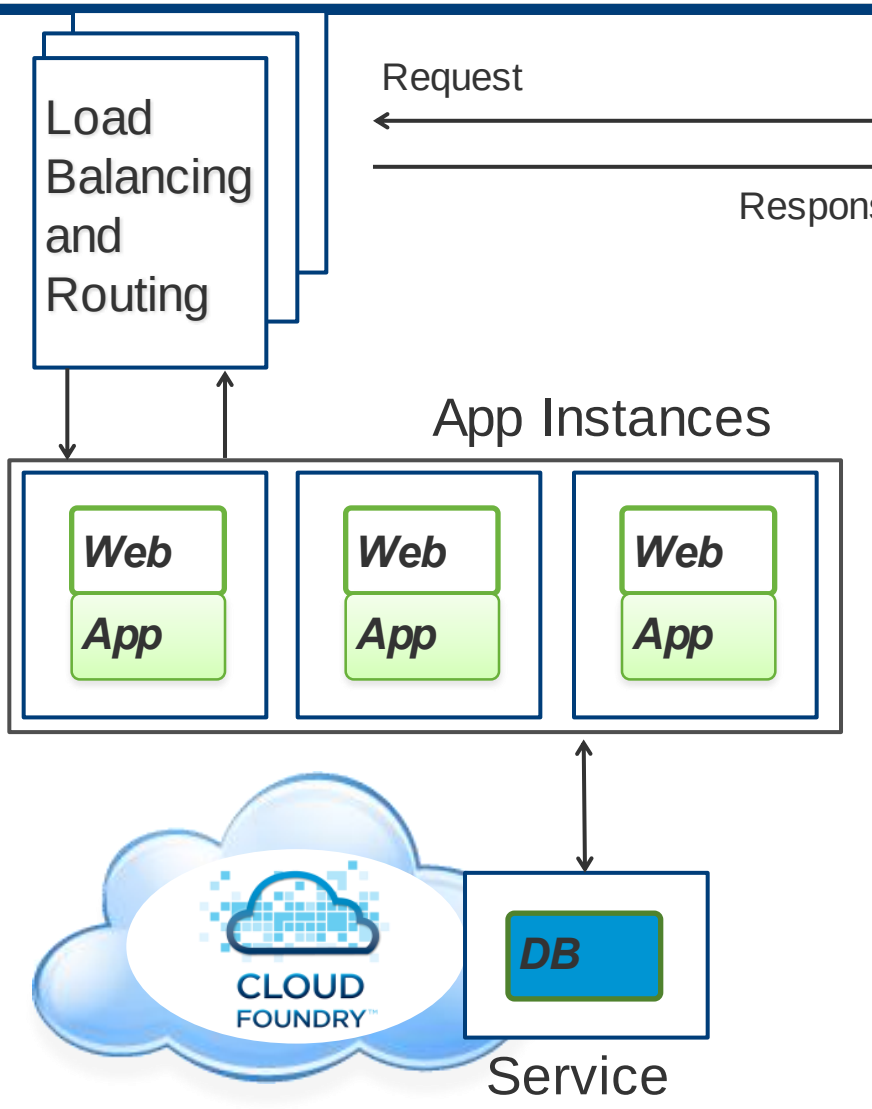


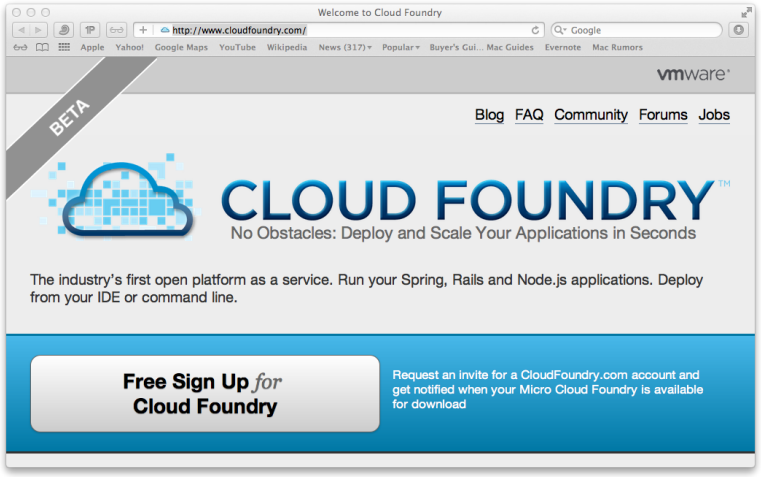
Web Interface



"vmc push MyApp"

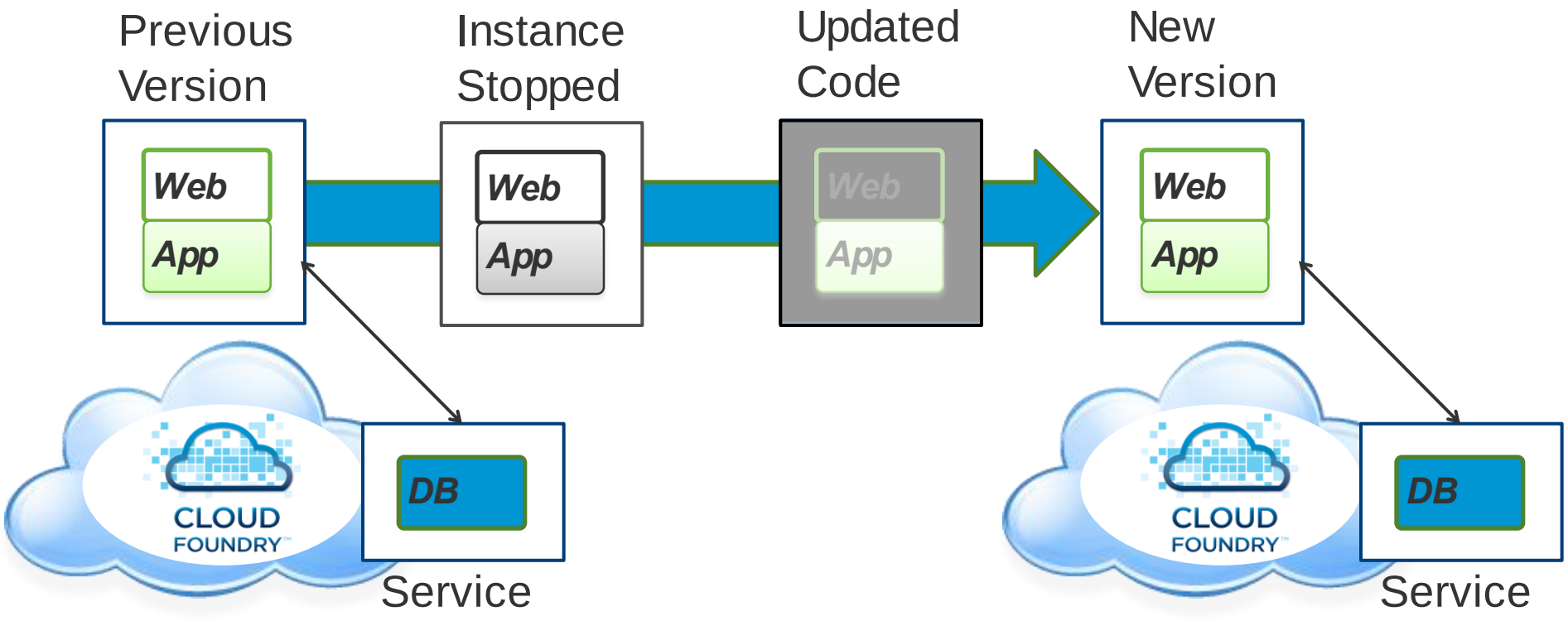
How Apps are Scaled on Cloud Foundry





"vmc instances MyApp 3"

How Apps are Updated on Cloud Foundry



"vmc update MyApp"

Create a Trivial Ruby Application

- In a new directory, create 'env.rb':

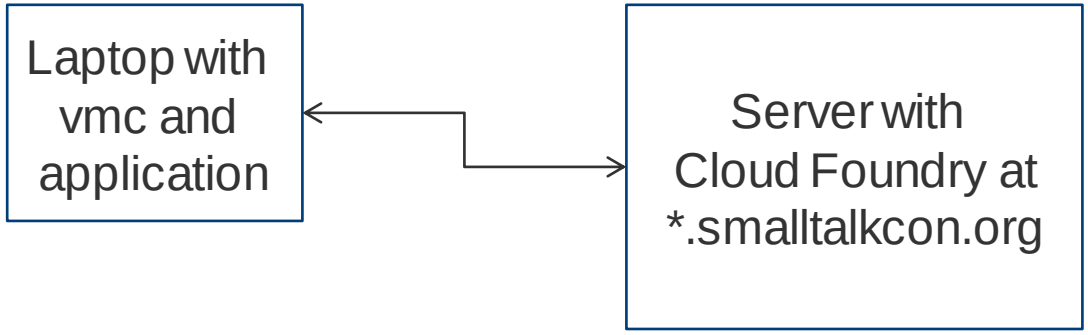
```
require 'rubygems'
require 'sinatra'
configure do
  disable :protection
end
get '/' do
  host = ENV['VMC_APP_HOST']
  port = ENV['VMC_APP_PORT']
  "<h1>Hello ESUG! via: #{host}:#{port}</h1>"
end
get '/env' do
  res = ''
  ENV.each do |k, v|
       res << "#{k}: #{v}<br/>"
  end
  res
end
```

Deploy the Ruby Application

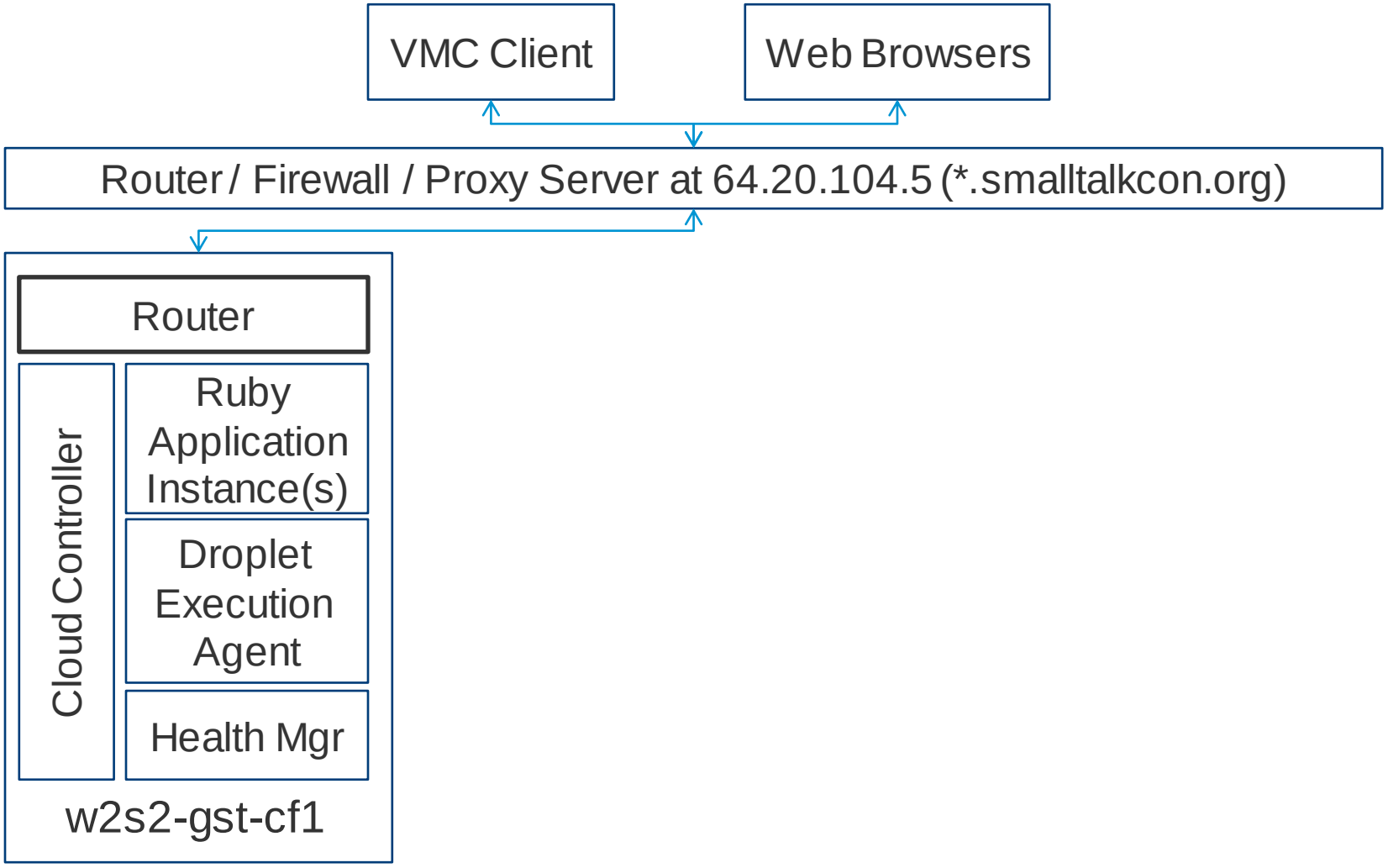
- **Try the application locally**
  - `ruby env.rb`
- **Push the application to the cloud (with a unique name):**
  - `vmc push MYAPP -n`
- **View the application:**
  - http://myapp.smalltalkcon.org and http://myapp.smalltalkcon.org/env
- **Increase the number of instances:**
  - `vmc instances MYAPP +1`
- **View the application again and note differences**
  - Port number and directory may change
- **Delete your application:**
  - `vmc delete MYAPP`

Push Process

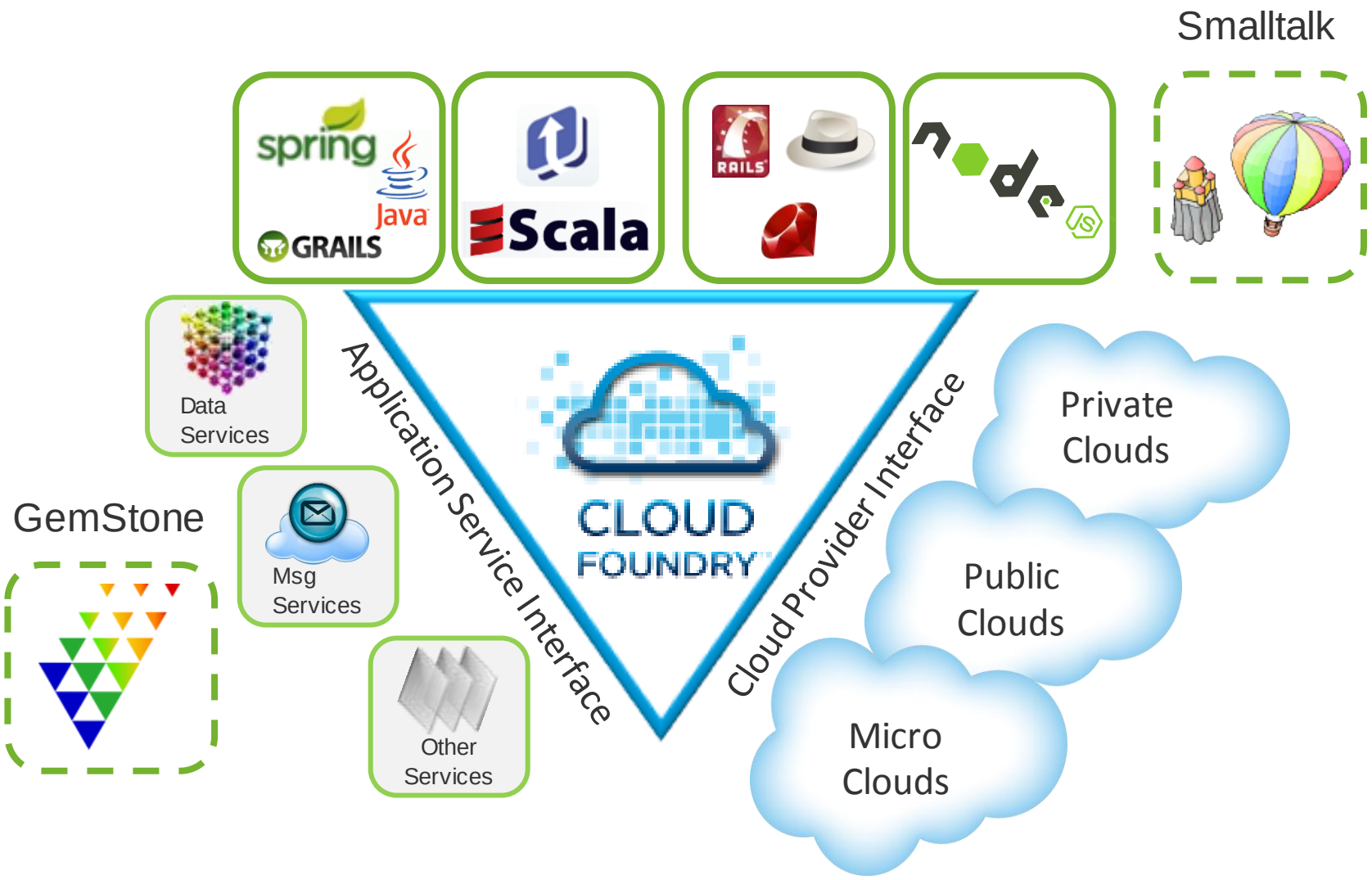
```
Creating Application: OK
Uploading Application:
  Checking for available resources: OK
  Packing application: OK
  Uploading (0K): OK
Push Status: OK
Staging Application 'james-ruby': OK
Starting Application 'james-ruby': OK
```



Server Architecture



Adding Smalltalk to Cloud Foundry



Smalltalk image from http://www.freudenbergs.de/bert/balloon.html

Support for "Traditional" Programming Model

- **Ruby**
  - Interpreter works with text files
  - Application code "includes" external libraries
  - Frameworks (Rails, Sinatra) can be pre-loaded on server
- **Java / Scala / Groovy**
  - Runtime VM works with "jar" files
  - Application code "includes" external libraries
  - Frameworks (Spring, Lift, Grails) can be pre-loaded on server

- **How is Smalltalk different?**
  - Let me count the ways!
  - Monolithic image with full application and framework

Smalltalk can Fit!

- **Code can exist outside image**
  - Package or file-in
- **Push**
  - Ship "code" to server during push phase
  - All files under a root directory are sent to server
- **Staging**
  - Launch Smalltalk, load code into image, and save image
- **Starting**
  - Launch saved image
  - Determine port on which to listen (environment variable or command line)
  - Start HTTP server on designated port
- **Consider a Pharo application**
  - Same ideas should work with Squeak, Cincom Smalltalk, and VA Smalltalk

Client file structure

- app/
  - aida.st
- Aida.changes
- Aida.image
- PharoDebug.log
- PharoV10.sources -> …
- start*
  - `open -a /…/CogVM.app/ --args Aida.image start.st $VCAP_APP_PORT`
- start.st

```
| file contents |
FileDirectory setDefaultDirectory:
  (FileDirectory default entryAt: 'app') asFileDirectory fullName.
file := FileStream readOnlyFileNamed: 'aida.st'.
contents := file contentsOfEntireFile.
Compiler evaluate: contents.
```

Special File in Application Directory: aida.st

```
| portString |
portString := SmalltalkImage current getSystemAttribute: 3.
(portString isNil or: [portString isEmpty])
  ifTrue: [portString := '8888'].
AIDASite default
  stop;
  port: portString asNumber;
  start.
Author fullName: 'CloudFoundry'.
WebDemoApp compile: 'introductionElement
  | e |
  e := WebElement new.
  e addText: self observee introduction.
  e addText: ''<p>Listening on port: '' ,
    session parent site port printString , ''</p>''.
  ^e'.
```

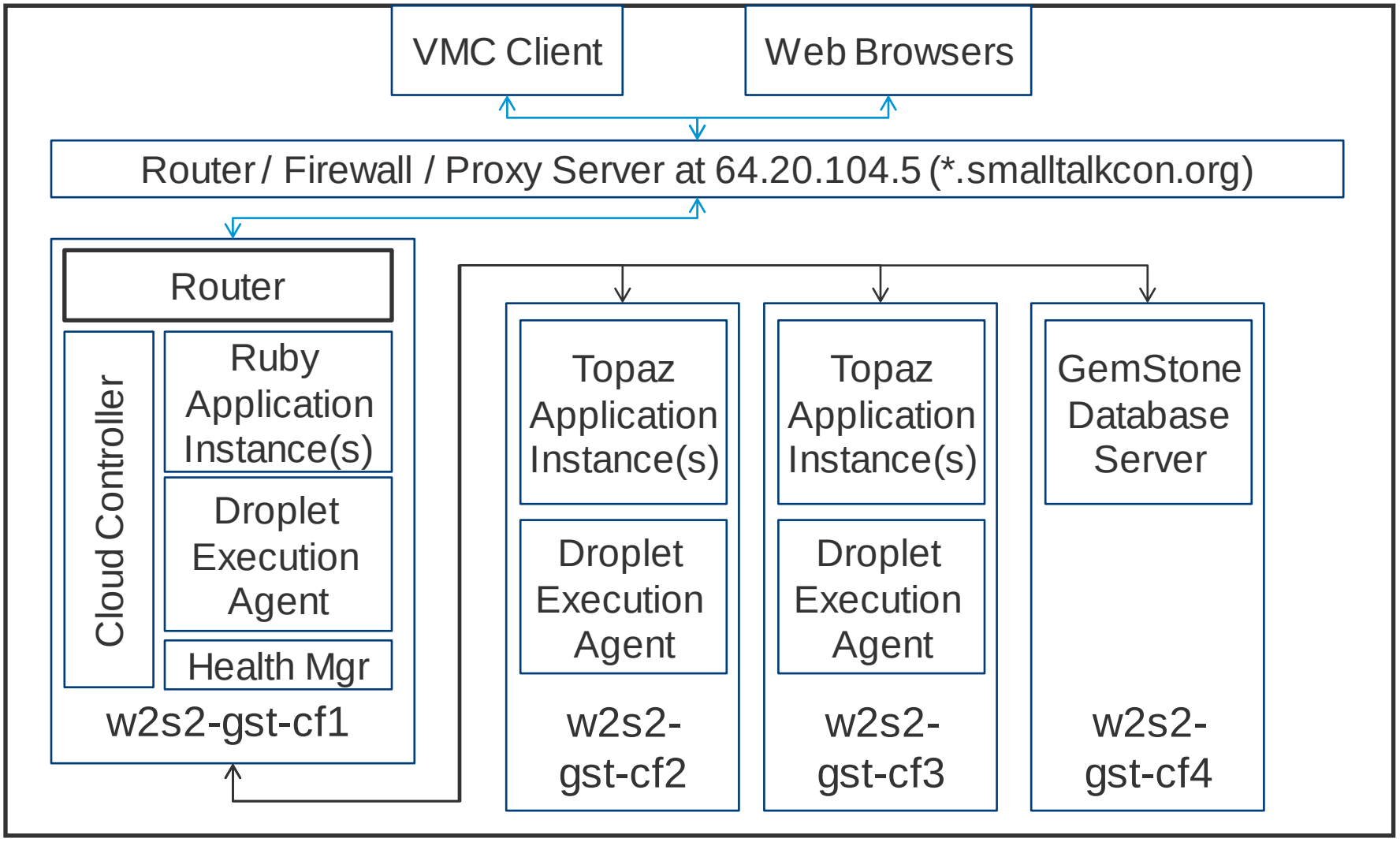
Changes to Cloud Foundry for Aida

- **VMC**
  - Add Aida to list of supported frameworks
  - Auto-detect 'aida.st' as indicating Aida framework and CogVM runtime
- **VCAP (Cloud Foundry Server)**
  - Chef scripts to download and install CogVM and Aida one-click image
  - Staging code to install code and save as new image
  - Startup code to change listening port

GemStone and Cloud Foundry

- **Cloud Foundry Services**
  - MySQL, MongoDB, Redis
  - GemStone/S maps reasonably well to these services
- **Runtime & Framework**
  - Topaz is the command-line executable
  - Support for file-in scripts
  - Startup needs to do lazy-initialization
    - Don't install code if it is already present
- **Changes to VMC for GemStone**
  - vmc-stic is modified to recognize a Topaz application
    - Look for main.tpz in deployment directory
    - Other files will be copied to server as well (e.g., *.mcz)

Server Architecture



Micro Cloud

- **Develop application locally**
- **Deploy it to a public cloud**
- **Intermediate step of testing deployment to a private cloud**
  - You have better access to logs to debug problems!

Experience Report

- **HPI students build Seaside applications in lecture**
- **Internal Cloud Foundry setup for students to show their projects**
  - Described at http://blog.bithug.org/2012/02/cloud-foundry-squeak
  - Cloud Foundry code at https://github.com/timfel/vcap
  - Squeak code at http://ss3.gemstone.com/ss/CloudFoundry.html
- **Will be promoted for all students in upcoming Seaside lecture**

Key takeaways

- **PaaS is the application platform for the Cloud era**
- **Cloud Foundry is the simple, open and flexible PaaS of choice**

- **What's next?**
  - Signup - www.cloudfoundry.com
  - Get the source code - www.cloudfoundry.org
  - Download your Micro Cloud Foundry – my.cloudfoundry.com/micro
  - Learn more on the Cloud Foundry blog - blog.cloudfoundry.com
  - Follow us - @cloudfoundry

Thanks & Questions?

- **Thanks to**
  - Dale Henrichs, Peter McLean, and Monty Williams of VMware
  - John Thornton (JonnyT) for the vmc-stic Ruby Gem and for testing things
  - Norm Green and VMware for the opportunity to work on this project
  - Tim Felgentreff of HPI

- **Contact info for James Foster**
  - jfoster@vmware.com
  - http://programminggems.wordpress.com/