Successful PaaS and CI in the Cloud

Steven G. Harris
steven.g.harris@cloudbees.com
@stevengharris

AgileALM/EclipseCon 2012
Platform as a Service
As-a-Service Examples Today

"Cloud computing is on-demand access to virtualized IT resources that are housed outside of your own data center, shared by others, simple to use, paid for via subscription, and accessed over the Web."

-John Foley, Information Week

©2012 CloudBees, Inc. All Rights Reserved
Traditional Software Stack

- Setup
- Monitor
- Patch
- Update
- Validate

You

Application

LB

AS

JVM

Hypervisor

OS

Server
Cloud Computing: How to do it? Who does what?

Cloud Provider?

- Setup
- Monitor
- Update
- Patch
- Validate

Application
- LB
- AS
- JVM
- OS
- Hypervisor
- Server

©2012 CloudBees, Inc. All Rights Reserved
PaaS in Summary

• Applications and data are at the center of your world!
  – Forget about servers, VMs, load-balancers, etc.
• Cloud concepts are applied to applications as first class citizens
  – On-demand, pay-as-you-go, elasticity, etc.
  – No need to handle updates, patches, scalability, failover, etc.

Delivered as a Service, not as Packaged Software!
What’s So Hard About Building PaaS, Anyway?

• Technology investment to operate at scale
• Intersection of app development/deployment and complex infrastructure
• Cultural shift can be more challenging than technology
• Efficient use of infrastructure resource pools
Internal Overview of CloudBees PaaS
Example of What PaaS and Cloud Enable
Easier Deployment Model

Load balancer

App v7

Easier Deployment Model

Load balancer

App v7

App v8
Easier Deployment Model

Load balancer

App v7

App v8
Easier Deployment Model
Easier Deployment Model

App v7

Load balancer

App v8
Easier Deployment Model

Load balancer

App v7

App v8.1
Easier Deployment Model

Load balancer

App v8.1
Easier Deployment Model

- Doubled resources — briefly & cheaply!
- Reduced risk dramatically
Ci and the Cloud
Development Tradeoffs for the Cloud

- Elasticity
- Ecosystem
- Managed Services
- Scale

Latency

Turnaround

Local speed
Local control
Your environment
Your IDE
Your debugger

©2012 CloudBees, Inc.
All Rights Reserved
Use the Right Tool for the Job

Low Resource → High Resource

Cloud Advantages

Integration Testing

X-Platform Testing

Performance Testing

Regression Testing

Stress Testing

Acceptance Testing

Code Quality

Low Resource

High Resource

Low IT → High IT

Machines

Unit Testing

Functional Testing

Performance Testing

People

Debugging

Cutting Code!

Stress Testing

Low IT

High IT

Low Touch → High Touch

Local Advantages

High Touch

Low Touch

People

High Touch

Low Touch

Machines

Integration Testing

Low Resource

High Resource

©2012 CloudBees, Inc.
All Rights Reserved
CI Resources Over a Few Days

What you would need

What you have (and pay for)

What you consume
CI Resources Over a (Traditional) Product Lifecycle

What you would need

What you consume

- Project start
- Team working
- Public Hollidays
- Team working
- Release!
- Bug fix
- Maintenance

©2012 CloudBees, Inc.
All Rights Reserved
Your CI feels like... but looks like...
Cloud & PaaS – An Abundance of Compute Resource

- How do you get it when you need it?
- Once you have it, how do you harness it?
- Once you’ve harnessed it, how do you keep it control and running at peak efficiency?

What you want

What you want to avoid
Jenkins Has Been Doing Distributed Builds for 5 Yrs

- Easily able to control and manage 100+ computers from a single place
- Plugin mechanism simplifies doing more
Validated Merge of Commits

Taking advantage of an abundance of computing: #1
Is CI Really Helping You as Much as It Could?

• Does your CI server shift work from laptops to servers?
  – You need to commit to have Jenkins test it
  – But if your commit is bad, it blocks others
  – You end up testing locally before committing
  – ... #fail
Mathematics of Large Projects

Every developer makes mistakes once in a while

\[ \downarrow \]

The more developers you add, the less stable the repository gets
Solution: Validated Merge

- Dev commits to branches
- Jenkins tests branches
- If good, Jenkins merges to the trunk
It's Not as Bad as It Might Look

• More realistic commit graph is like this
  – Especially if devs remain close to the tip
  – Your coding is slower than “slow tests”
Advantages

• Your mistake doesn’t impact others
  – Commit without worrying
• Tests run on servers
  – Large environment-dependent tests are no longer a problem
• Tests run asynchronously
  – Commit, then move on
  – You don’t wait for tests to complete
• Works with other plugins...
  – Subversion Merge Plugin
  – Git Plugin
  – Gerrit Plugin

Hierarchical Validated Merges

- NetBeans team works like this
Automated Deployment

Taking advantage of an abundance of computing: #2
If you build it, run it somewhere...
But You Need To Test Before Deploying

• It compiled ≠ it’s ready to ship

• Hence pipeline
  – Progressively run more expensive tests
  – A failure, and it’s out
  – Avoids wasting computer cycles and improves feedback speed
Build Pipeline: #1

- Waterfall model
- Connect jobs via upstream/downstream
- Copy binaries from upstream
Same Model from Different Angle

Time

Build | Build | Build | Build | Build | Build

Integration | Integration | Integration

Deploy | Deploy
Build Pipeline Plugin

Build Pipeline View:

Configure View
Start Build of Pipeline for Build
View/Hide Build Pipeline Icon Legend

Build #3
Mar 24, 2012
1:41:34 PM PDT
Duration: 13 ms

Integration #3
Mar 24, 2012
1:41:39 PM PDT

Deploy
Pending build of project: Deploy

Build #2
Mar 24, 2012
1:45:37 PM PDT
Duration: 15 ms

Integration #2
Mar 24, 2012
1:46:42 PM PDT
Duration: 26 ms

Deploy #1
Mar 24, 2012
1:49:44 PM PDT
Duration: 15 ms

Build #1
Mar 24, 2012
1:38:10 PM PDT
Duration: 15 ms

Integration #1
Mar 24, 2012
1:39:24 PM PDT

Pending build of project: Deploy

©2012 CloudBees, Inc.
All Rights Reserved
Promoted Builds Plugin

• Think about state flow, not process flow
  – What conditions trigger a transition?

- Good build
  - Compiled OK
  - Unit test passed

- QC Pass
  - Coverage > 60%
  - Int. tests passed

- Stability
  - Ran > 3d in UAT env
Combining Pipeline and Promotions

Developer Org

- [Developer Job]
  - Developer Tests
  - Functional Tests
  - X-Team Tests

- [SQE Job]
  - Security Tests
  - Access Tests

QA Org

- [SQE Job]
  - Database Tests
  - Compliance Tests
  - Stress Tests

Production Org

- [Production Job]
  - Backup Production
  - Deploy
  - Monitor

©2012 CloudBees, Inc. All Rights Reserved
Advantages of Promotion

• Asynchronous
• Retryable
• Flexibility in shape
• More stable
  – Conditions can be tweaked
  – Human teams often interface by such states
• Break free from triggers
  – More flexible conditions
• More opportunity to take advantage of resources
Traceability

Finding the needle in the compute haystack
When bad things happen...
Machine-Assisted Traceability

• Stitch together information in siloed systems
  – Dev: commit ID
  – QA: test execution log
  – Ops: deployment log, running version check
Cl Server Is Big Brother

• Source : commit ID = Binary : checksum
  – In Jenkins, we call it “fingerprints”
• Liberally record fingerprints every time you “use” binaries
  – When you run tests
  – When you deploy
  – When you integrate with something else
• Then cross check where they showed up
Traceability and Pipeline

Usage
This file has been used in the following places:

- Build #4-#7
- Deploy #3 #4
- Integration #7 #10
Major Retailer (courtesy Marco Vermeulen)

• Directive from top management to move to cloud
• Assessment of various solutions
  – Committed to CI
  – Did not want to manage infrastructure, monitoring
  – Grails app
  – Wanted to use standard .war
• Continuous delivery in Dev and Test environments
• Release after each 2-week sprint
• Source code on GitHub
• Plugins...
  – Git, Cobertura, Codenarc, Violations, Grails, Pipeline, Twitter
Production Pipeline View

Build Pipeline View:

- Exploits parallelism
- Mirrored DEV and PRODUCTION pipeline
- Manual trigger to push to production
Enforcing Your Team’s Values

- Code coverage < 80 triggers FAILURE
- Excessive Codenarc violations trigger FAILURE
Summary

• PaaS is serving up platform capabilities to you without the headaches of install, update, patch, and ongoing infrastructure management
• Removing roadblocks to dev/test/production resource availability changes the way we have to work
• Extensible CI server like Jenkins provides hooks to harness the abundance of resources
Thank You!

steven.g.harris@cloudbees.com
@stevengharris