Taking Hudson to the Next Level
From CI to CD

Winston Prakash, Duncan Mills
Hudson Development Team
Sept 30, 2014
Program Agenda

1. Hudson in a nutshell
2. Latest changes in Hudson
3. Hudson for continuous delivery
4. Case studies - Hudson @ Oracle
Hudson in a Nutshell
Hudson in a Nutshell

Continuous Integration Server

• Manage the essential flow of a modern development env.
• Heterogeneous technologies and operating systems
• Extensible
Hudson in a Nutshell

Has all the Right Bits for Continuous Integration

• Runs automatically
  – SCM: GIT, SVN, CVS, Perforce...
  – Builds: Maven, Ant, scripts...
  – Testing: Junit, Selenium, Abbot...
  – Deployment: app servers, web...

• Feedback
  – Email, IM, RSS, Lava Lamp...

• Analysis
  – Clover, Sonar, PMD...
Hudson in a Nutshell

The Project

• Mature Technology Project at the Eclipse Foundation
  
  www.eclipse.org/hudson  ||  www.hudson-ci-org
  – Hudson Core is EPL licensed
  – Plug-ins mixed licenses; usually MIT

• Current release 3.2.1 (Sept 2014)
  – Project plan available @ Eclipse
  – http://projects.eclipse.org/projects/technology.hudson

• Open (non-Eclipse) plug-in community, GitHub, java.net etc.
What's been Happening in Hudson?
From CI for an Individual / Small Team

What Matters?

• Saving Time
• Agility
• Flexibility
To CD in the Enterprise

What Matters?

- Manageability
- Security
- Performance and Scalability
- Visibility
- Legal / IP
Hudson – Embracing the Enterprise

Core Theme to make Hudson more Enterprise Friendly

• IP – Hudson 3.0 cleaned up and documented all deps.
  – 109 libraries reduced to 85
  – LGPL and unprovenanced libraries removed / replaced

• Plug-in management
  – Switchable Update Centers and initial setup capabilities

• More complex pipelines
  – Cascading templates
Hudson – Address Scalability

Hudson 3.1 – Yet More...

• Big memory optimization exercise
  – Can reduce required heap by between 50-75% for the same set of jobs
  – The more builds the bigger the savings

• Team Concept
  – Allowing sandboxed use of a single infrastructure
Hudson Teams in 3.1

Team Structure

* A single user or group can be present in more than one team with different rights in each case
Hudson – Latest

Some Housekeeping and More...

• Team Concept
  – Per-team views
  – Per-team slaves

• Spring Upgrade (3.1.2)
  – Includes upgrade to the security layer
  – Major internal re-working

• Plugins
  – Needed reworking to support Spring changes
  – Now have multiple plugin centers
Hudson For Continuous Delivery
Development Lifecycle Evolution

Until 2000

Waterfall Method

Typical turn around time is 6 months to 1 year

2000 - 2012

Agile Method

Sprint cycle is typically 2 weeks to 4 weeks

2013 -

Continuous Delivery

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.
But it's More than Continuous Integration

CI is but the First Step

• Continuous Delivery
  – A set of practices and principles aimed at building, testing and releasing software faster and frequently
  – Produce a deployable-to-production build regularly, probably on each commit.
  – Every build is a potential release.
Hudson for CI

Hudson is mostly tuned to focus on development teams
Typical Sprint Cycle

- **Product Backlog**
- **Sprint Backlog**
- **Sprint**
- **Working Increment of the software**

**Sprint 1**
- Plan → Develop → Test → Stage

**Sprint 2**
- Plan → Develop → Test → Stage

**Sprint 3**
- Plan → Develop → Test → Stage

Reasonably working product
Commit to deploy

In a **Test Driven Development** build pipeline, **Continuous Integration** is the first step and the end result is the **Continuous Delivery**.

While **Continuous Delivery** promotes the concept of keeping your product in a **deliverable state** on each commit, **Continuous Deployment** takes it further. On each commit, the deliverable can be **deployed** to a production environment.
Moving on to Continuous Delivery

Continuous Integration

Dev

Build and Unit Test

Source Repository

Acceptance Test

Functional Test

Integration Test

Automated Tests

Continuous Testing

Package Repository

Continuous Delivery

Automated Deployment

Deploy-QA

Deploy-production

Deploy-performance

Resources Repository

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.
Think in terms of Build Pipeline and promotion

- **Commit**

- **Build and Test**
  - Unit tests
  - Static code coverage
  - Packaging
  - Integration tests
  - UI tests
  - Performance tests
  - Regression tests
  - Deployment tests (install, uninstall etc.)
  - Manual exploratory tests
  - Regulatory, compliance checks
  - Clearance from UAT

- **Stage and Deploy**
Team Management (3.2) for CD environment
Tenant enabled

Hudson 3.2 enables per tenant resource isolation and sharing. This brings the Dev-Ops (dev-qa-ops) team together, but with resource isolation to play nicely in a Continuous Delivery Environment.
CD Best Practice: Monitor Quality Metrics Trend

Code quality measurement is important in Continuous delivery.
Improves the confidence of the product being in a deliverable state.
CD Best practice: Monitor Test Trends

In a CI build, the unit tests should never fail. During the initial stage of the project, the integration test may be in flux.
Choosing Plugins for the Practicing of CD

http://wiki.eclipse.org/Hudson-ci#Hudson_Plugins
Practicing Continuous Delivery using Hudson

Case Study – Developer Cloud Service
Case Study: Oracle Developer Cloud Service

- Development Platform provided as a Service
- Application Lifecycle Management
- Team Management

- Source Control Management
- Issue Tracking
- Hudson Continuous Integration
- Wiki Collaboration
Oracle Developer Cloud Service

Hudson 3.2 - Stability and Density in Multi-tenancy

• ODCS Projects organized by Hudson Teams
  – Single Hudson Master per Tenant

• Memory and Stability Improvements
  – Stability improves predictability
  – Removed regular Master restart scripts
  – 75% savings on Heap allocation per master
  – One order of magnitude more tenants per hardware allocation

• On-demand provisioning of Hudson Slaves
Case Study - Delivering Oracle Fusion Middleware with Hudson
Solving a Problem

• Oracle's development org is big – very big (and heterogeneous)
  – Mix of SCMs
  – Mix of build methodology
  – Mix of testing, quality and security tools
  – Multiple bug, task and requirement systems
  – Mix of deliverables
  – Infrastructure!

• Huge complexity in release promotion and consumption

• Fortunately Hudson gives us common ground and the adaptability we need
  – But it needs a little help...
Meeting Challenges of CD at Scale

• Core activities need to be managed centrally
  – Pushing new plugins / maintaining local plugin center
  – Tracking plugin usage by job and plugin version
  – SSH key management across Hudson executors and SCM systems
  – Management of local tools installation
  – Managing the recipes for Hudson slave machines

• Security
  – In our case, a single SSO system used across the farm
  – Organization specific policies for control and sharing
Hudson at Scale

• The numbers
  – 87 Hudson masters (allocated by organizations within development)
  – ~1200 slaves
  – More than 25,000 jobs

• Coordinated by "Carson" message bus
  – Manages event driven continuous delivery pipeline
  – Bus handles ~ 1,000,000 messages / week
  – Manages retry in case of infrastructure failure
  – Provides management UI and additional reporting on pipelines
Overview

HTTP Management Interface & REST APIs

Orchestration

Plugins
artifactory

Executor Recipes
CHEF

Configuration  Identity  Executors  Storage

git  key  nimbula  

Shared Resources

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.
What's Next for Hudson?

3.3 And Beyond?

• We're thinking about
  – Modernizing the UI
  – ...

• What about You?
  – Join the Birds-of-a-Feather Session @ 7PM right here!