


Advanced Continuous Deployment with Jenkins

Andrew Phillips

XebiaLabs
the deployment automation company

http://www.xebialabs.com


Advanced Continuous Deployment with Jenkins

with special thanks to

Benoit Moussaud

@bmoussaud

Continuous Integration

- Emerged at the end of the ‘90s as one of the XP practices
- By continuously building and testing software quality should improve
- Tests often limited to unit tests (e.g. JUnit)
- Sometimes also functional tests (e.g. Selenium)

CI Shortcomings

- Deployment to the target platform often not part of the CI cycle

→ *Deployment procedures not tested!*

→ *Application not tested on ultimate target platform!*

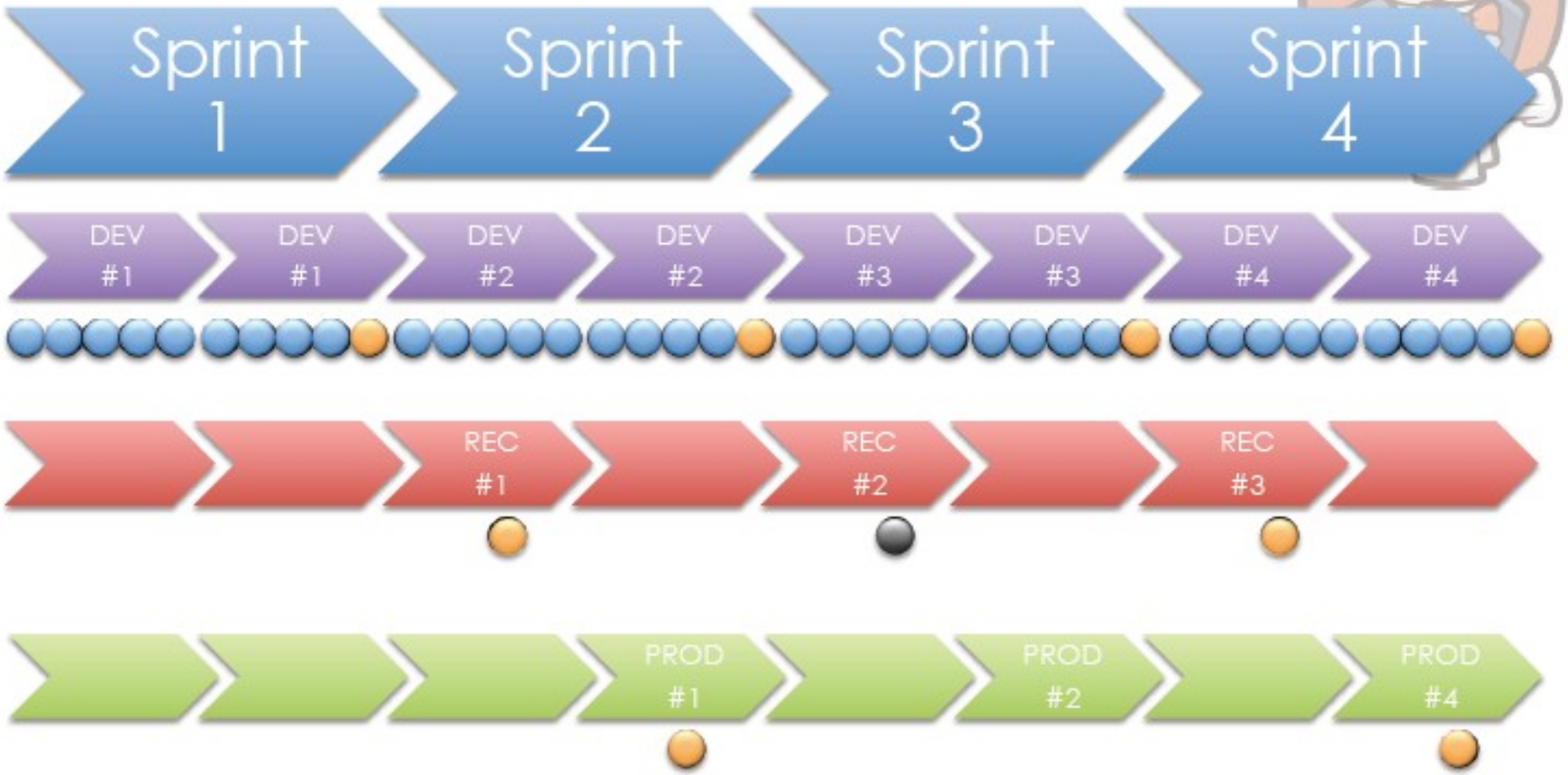
Enter Continuous Deployment

- Strictest definition: Every (tagged) version goes to production
  - Used by LinkedIn amongst others.
- Less strict: Include deployment in the CI cycle to test the deployed artifacts on the target platform

With Continuous Deployment...

- Smoke tests
  - Landing page
  - *Line of Life*
- Functional tests on target platform (e.g. Selenium)
  - Content of the landing page
  - Typical run
- Performance tests (e.g. JMeter)
  - Response time of the landing page
  - Response time of the simple / complex path

Continuous Deployment Flow



Continuous Deployment Flow

- Dev Team
  - Full nightly build
  - Tag package as “released”/”ready”
- Acceptance/QA
  - Deploy “released” package to Test environment
  - Perform tests
  - If OK, tag package as “accepted”
- Production/Ops
  - Deploy “accepted” packages to Production

The “Dev Commandments”

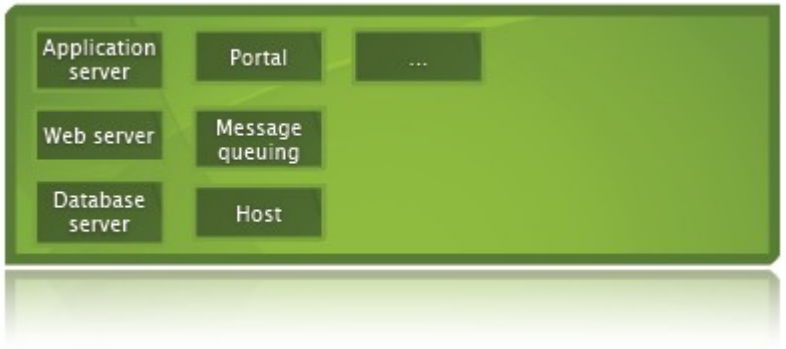
*For each version of the application, we shall provide one single package containing all the artifacts **and** resource definitions*

The package shall be independent of the target environment

The “Ops Commandments”

We shall provide fully configured infrastructure items (hosts, application servers, web servers, databases etc.) grouped by environment

We shall associate configured environment variable values to all environments



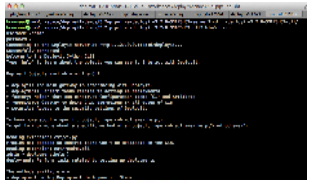
DIY with Jenkins

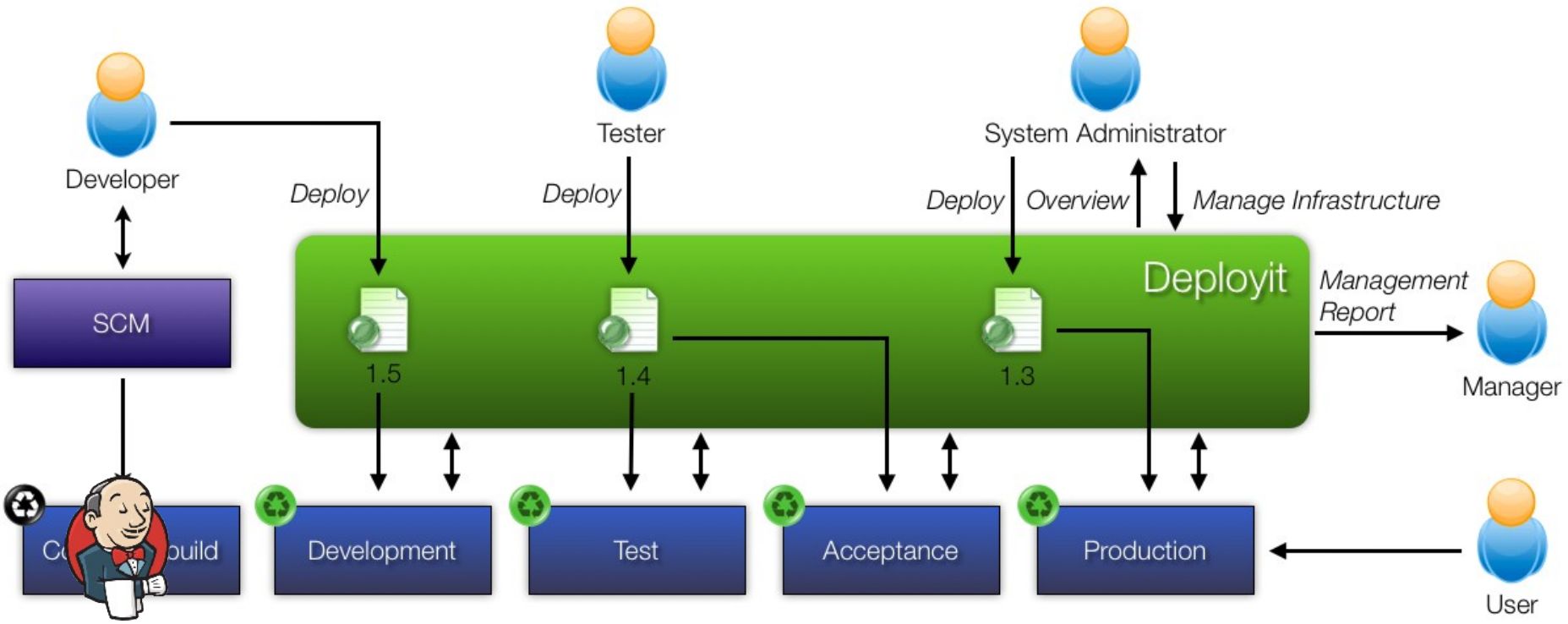
- Maven / Ant
- Cargo / SSH plugins
- Middleware scripting
- Maven profiles
- or...?

Challenges

- Middleware-specific manual effort
- Exposed security credentials
- How to secure pipeline to later-stage environments ?
- How to separate deployment and build audits?
- Time Machine (what was the state of my environment at a certain time)?

Continuous Deployment with Deployit

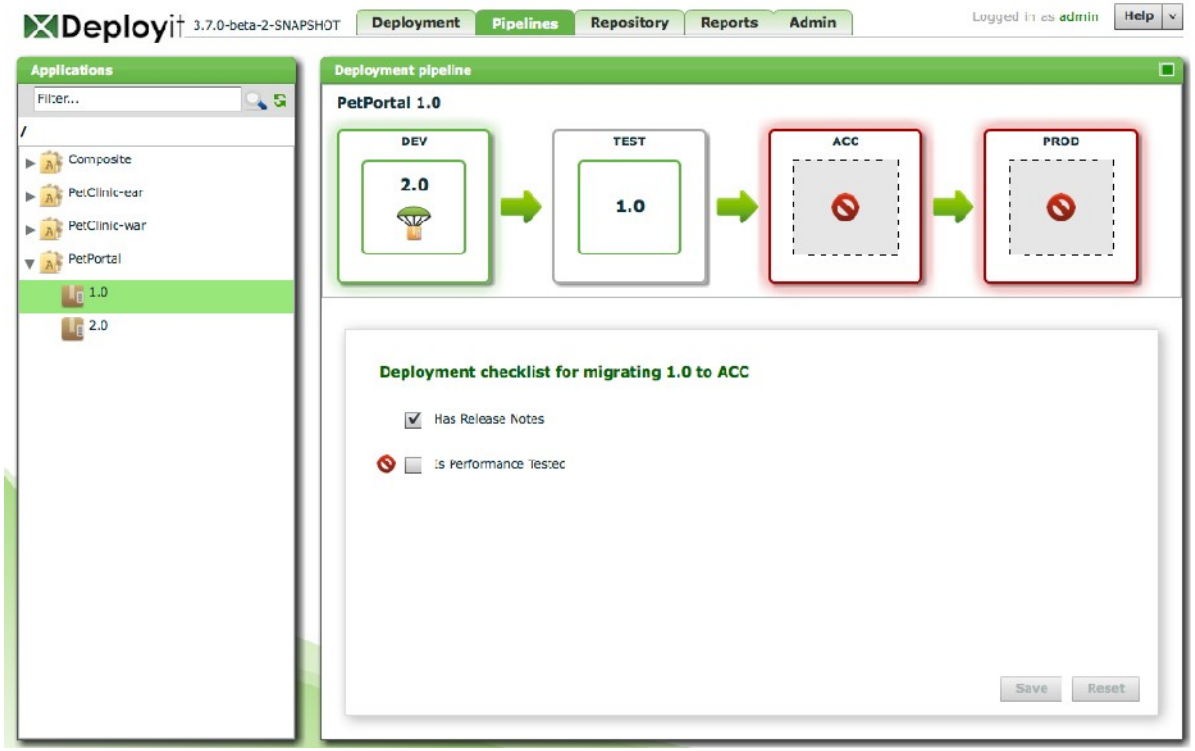




Continuous Deployment Flow

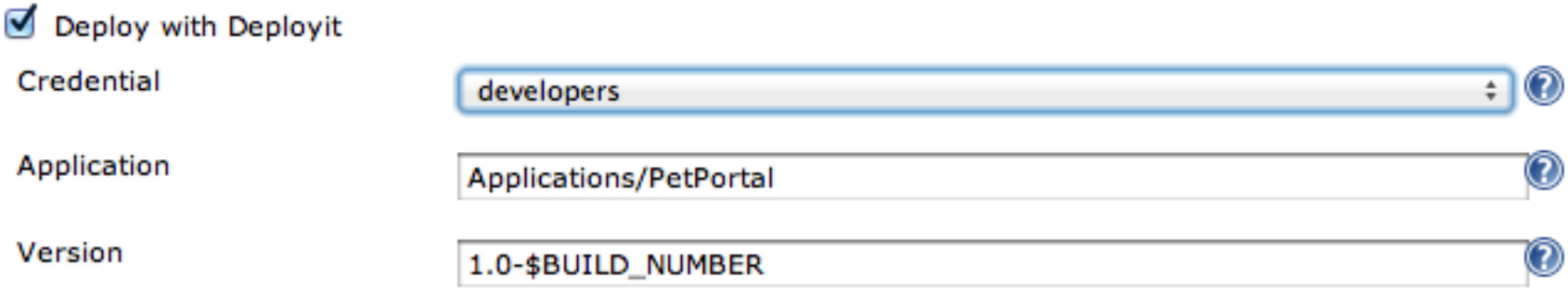
- Dev Team
  - Jenkins CI drives Deployit
  - Creates package, publishes to Deployit and triggers deployment to the development environment
- Acceptance/QA
  - Uses UI to deploy selected tested version to QA
  - Tags the version as “accepted” if all tests pass
- Production/Ops
  - Automates deployment of accepted versions to production environments via CLI

Continuous Deployment Flow



- Dev → Test → QA → Prod pipeline conditions are checked by Deployit

Deployit Jenkins Plugin



Deployit Jenkins Plugin: Package

Package your application

Deployables

Artifact

Type: jee.War

Name: petclinic

Tags:

Location: PetClinic/target/PetClinic.war

Delete

Artifact

Type: jee.War

Name: petclinic-backend

Tags:

Location: PetClinic-Backend/target/*.war

Delete

Resource

Type: tomcat.DataSourceSpec

Name: petDataSource

Tags:

Properties: Name: jndiName — Value: jdbc/petds

Supprimer

Ajouter

Properties

Delete

Artifact

Type: sql.SqlScripts

Name: sql

Tags:

Delete

Resource

Type: tests2.HttpRequestTest

Name: testPage

Tags:

Properties: Ajouter

Properties

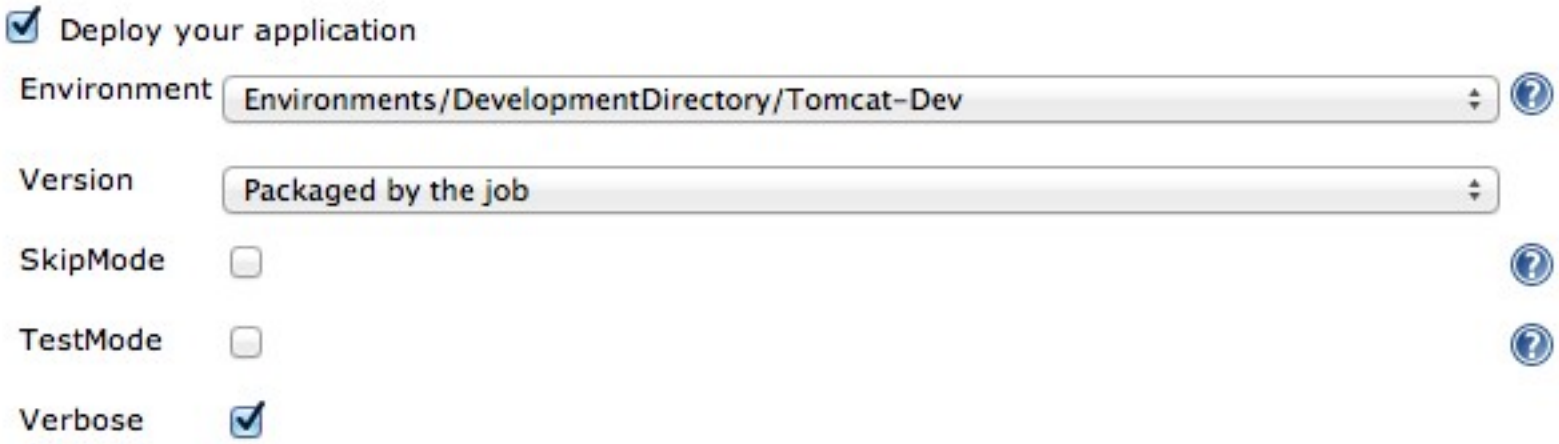
Delete

Add Deployables

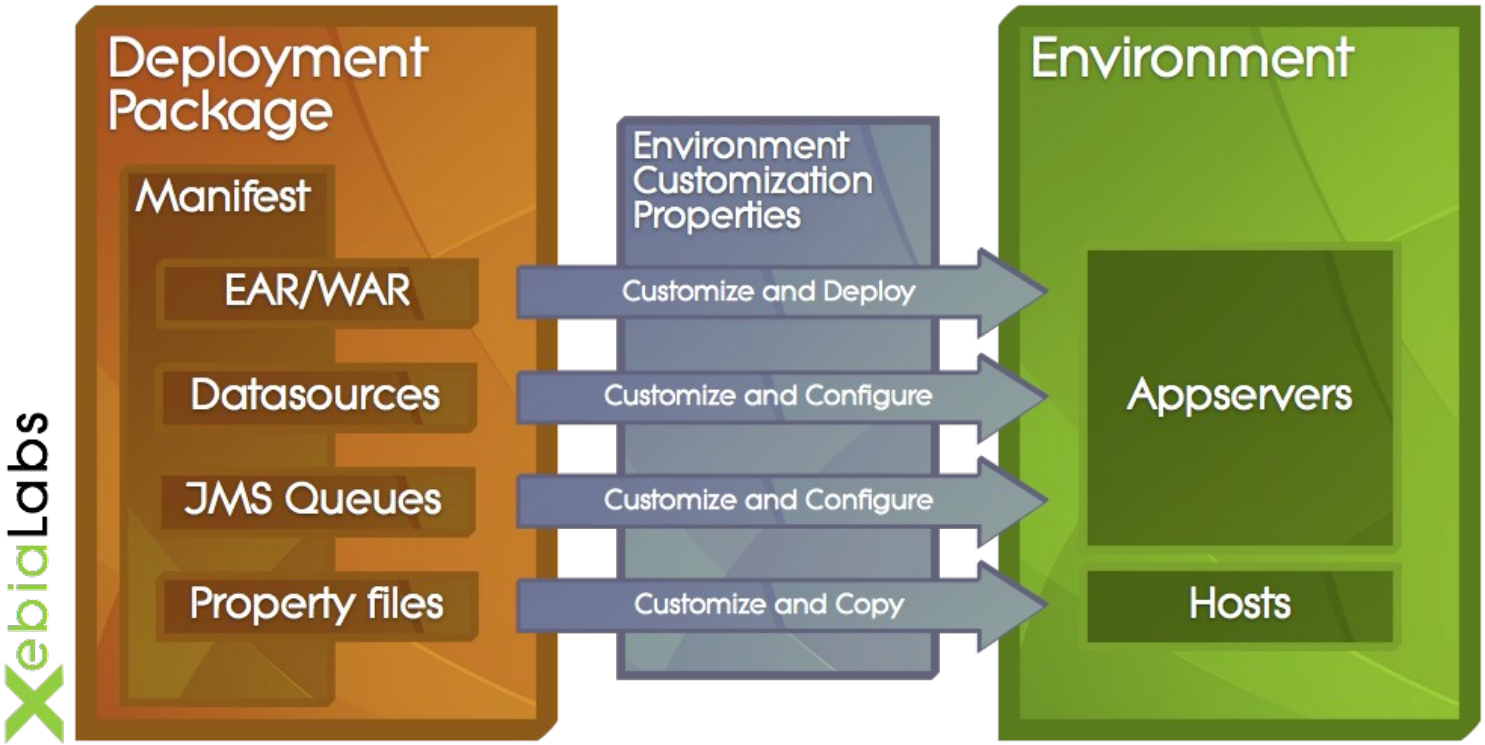
Timestamped

Add Jenkins Build Number

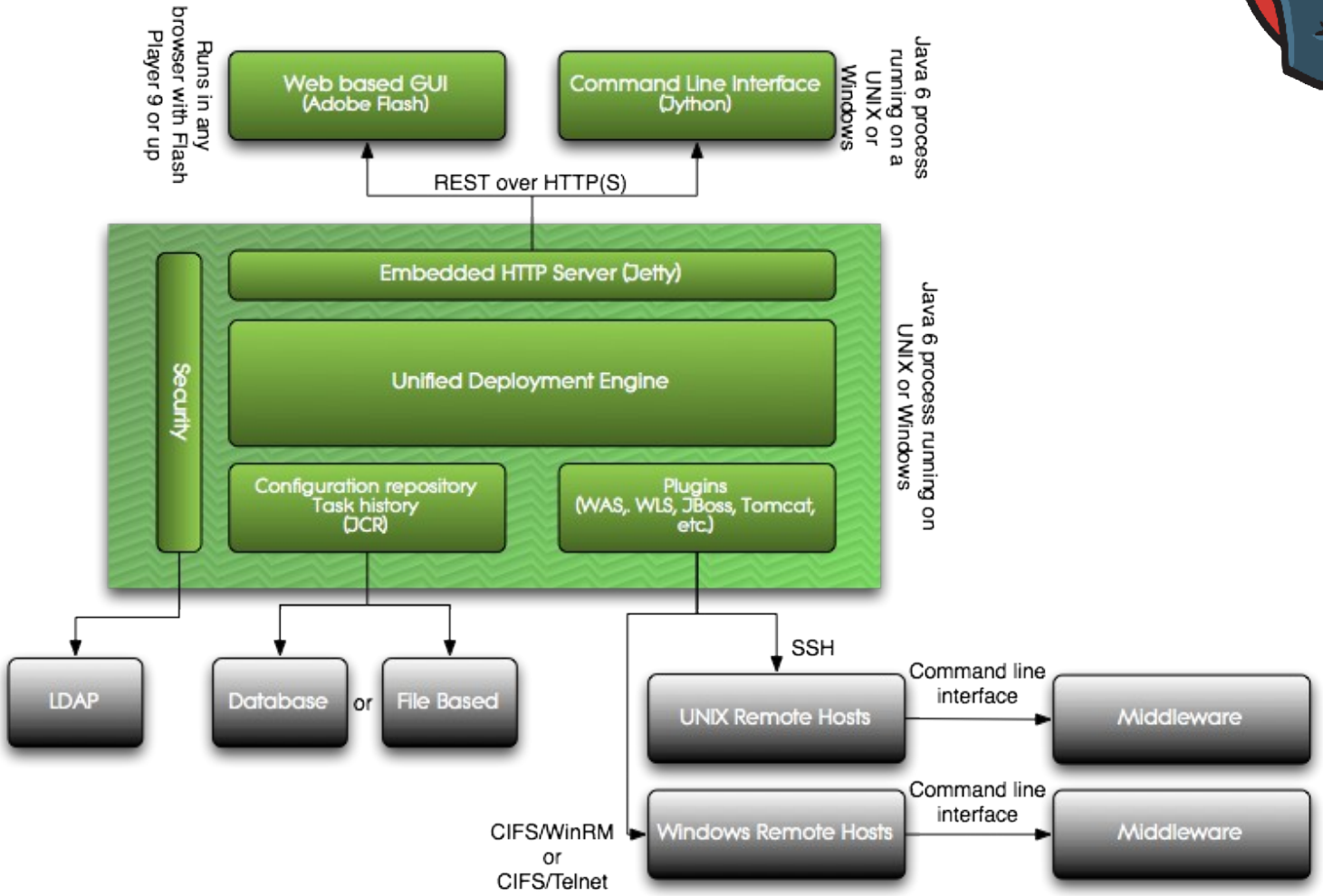
Deploying Jenkins Plugin: Deploy

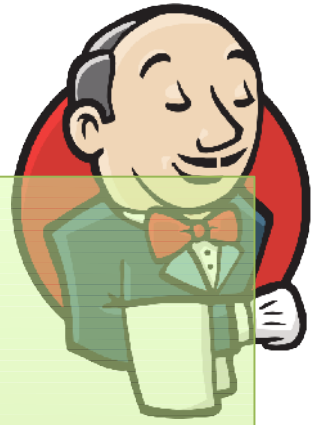


The Unified Deployment Model



Deployit Architecture





Live Demo

Security & Audit

- Lock down access to target systems:
  only Deployit needs to know the credentials
- Control deployment capabilities
  - *permissions to deploy* this *on that*
- Track deployment activities
  - who *deployed* what where *and* when*?*
- Audit past events
  - *what happened during that deployment x months ago?*

One Best Practice for Application Deployments

- Targets
  - Java Middleware (incl. Tomcat, WebSphere, JBoss, WebLogic)
  - .NET & IIS
  - System: Files & Folders
  - Database: SQL (incl. rollback)
  - Web Servers: PHP, images, video, JavaScript
  - Load Balancers
- Supports heterogeneous packages & environments

Forging your Deployment Patterns

- Server side
- Configure
  - Application level: Manifest
  - Environment level: Dictionaries
  - Global level: *deployit-default.properties*

Forging your Deployment Patterns

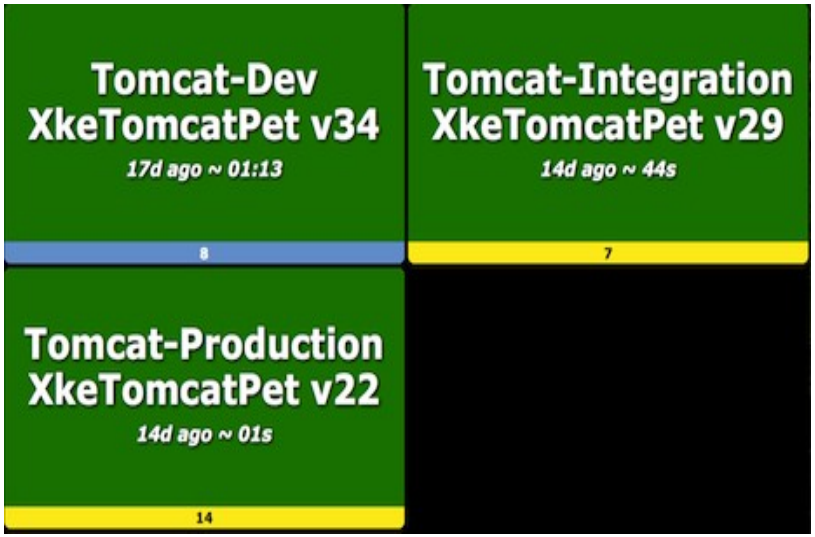
- Extend
  - Plugins
    - Extend existing types, e.g. *google.TomcatDataSource*
    - Modify existing types
  - Generic plugin
    - Create your own model
    - Support additional middleware platforms
  - Command plugin
    - Levarage existing scripts

Advanced Continuous Deployment Best Practices

1. Build complete packages
2. “Dev commandments”
3. “Ops commandments”
4. Lock down credentials
5. Forge & share your deployment patterns

An aside: Visuwall

- ‘Live’ Build Wall
  - Build time
  - Test coverage
  - LOC
  - …
- Connectors for
  - Jenkins
  - Deployit
  - …
- http://awired.github.com/visuwall/



Thank you!

and...

Thank You To Our Sponsors

Platinum Sponsor	CloudBees
Gold Sponsors	JFrog, CLOUDANT
Silver Sponsors	LIFERAY., SendGrid (Email Delivery. Simplified.), CLOUDSMITH
Bronze Sponsors	WAN DISCO

Get in touch!

Andrew Phillips

XebiaLabs
the deployment automation company

aphillips (at) xebialabs (dot) com
http://www.xebialabs.com