



Red Hat and the story behind <any middleware> to JBoss migrations

Roel Hodzelmans
Cojan van Ballegooijen

Agenda

- Migrations
- JBoss Windup
- JBoss Tattletale
- Labs



So Red Hat, how do you do migrations?



Migration Strategies – **Where to Start**

- Bring together your
 - Source code
 - EARs/WARs
 - Application Architecture
 - Infrastructure Architecture
 - Interfacing Applications
 - Current App & Web Server architecture
 - Desired App & Web Server architecture
- Collecting this information will help get you to your end goal

Migration Strategies – **Multi-Pronged Approach**

- Analyze 1-2 of your largest, most convoluted applications
- Analyze 1-2 of your average size applications
- Analyze 1-2 of your smallest, most basic applications
- Why?
 - It will give you a good idea of your whole organization's ability to move to JBoss

Migration Strategies – **Multi-Pronged Approach**

- Create a spreadsheet to list
 - Each Application
 - Rated Easy, Medium, Hard in terms of
 - Size
 - Proprietary code
 - Knowledge of code base
 - Testing Ability
 - Subject Matter Expert (SME) Application Contact
 - Testing Team Contact
 - Date of Migration

Migration Strategies – **What you need to know**

- Are there Subject Matter Experts (SMEs) for each application?
- Build Process
 - Are there a lot of Application Server-specific Ant build processes?
 - Is there a standard way of building the application?
 - Is it a self-contained WAR or EAR?

Migration Strategies – **What you need to know**

- Testing
 - What are your current testing standards?
 - Is regression testing necessary?
 - Do you have JUnit or a similar testing platform?
 - What is the support's staff availability?

Migration Strategies – **Project Plan**

- Create a project plan based on spreadsheet
- After using JBoss Tattletale and JBoss Windup
 - Projections based on LOEs
 - Testing takes roughly the same amount of time
- Ensure all necessary resources will be available during this time

But is it all manual labour?



PRESENTING



windup

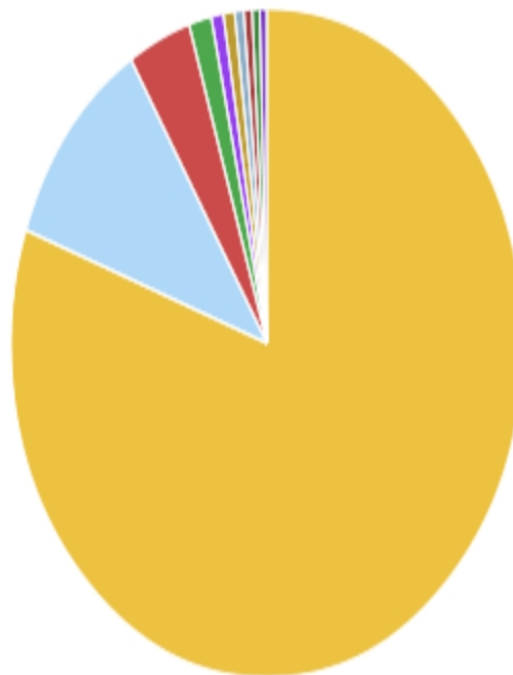












Windup

- Takes One or More Deployables (EAR/WAR/JAR)
- Recursively profiles resources
- Analyzes JSP, XML, and Java Classes
- Uses extensible rules
- Match resources against “known blacklist”
- Provide 10,000 Foot View
- Provide level of effort (LOE) estimation for migration
- Provide development team responsible for delivery a guided experience
- Provide a platform to make migrations easier

Windup – Story Points

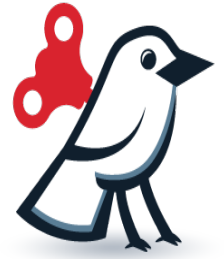
266
Story Points



	javax.persistence.* - 3387
	org.jboss.* - 446
	javax.ejb.* - 168
	org.jbpm.* - 60
	oracle.sql.* - 32
	org.springframework.* - 30
	javax.naming.* - 24
	javax.sql.* - 22
	java.sql.* - 20
	Other - 19



Windup - Classifications



Classification

- JBoss EJB Deployment Descriptor (prior to AS7/EAP6)

Highlights

- If migrating to JBoss AS7 or EAP6 the "jboss.xml" descriptor is ignored in deployments. Replace with "jboss-ejb3.xml"

```
01.  <jboss>
02.  <enterprise-beans>
03.    <session>
04.      <ejb-name>AddressServiceBean</ejb-name>
05.      <jndi-name>address-service-ear-1/AddressServiceBean/remote</jndi-name>
06.    </session>
07.  </enterprise-beans>
08. </jboss>
```

Windup - Hints

```
30. @WebContext(contextRoot="/svc/AddressService",urlPattern="/1")
```

❗ Annotation 'org.jboss.ws.spi.annotation.WebContext' at line 30



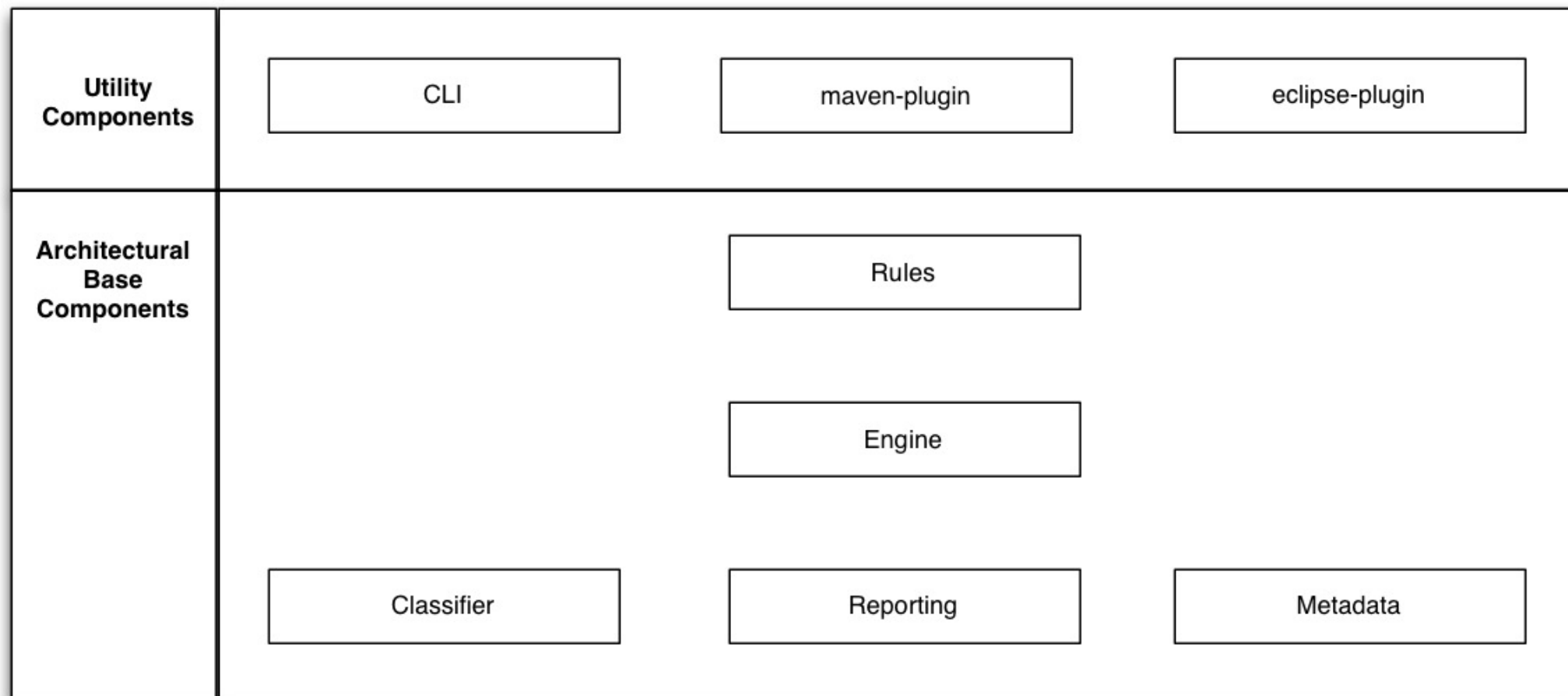
Windup – Workings

- Source Code and Binary Analysis
- Eclipse
- Maven
- CLI
- Forge Plugin
- SOA-Migrations





Windup – Architecture



Windup – Workings

- Source Code and Binary Analysis
- Eclipse
- Maven
- CLI
- Forge Plugin
- SOA-Migrations



Ok, that helps a lot actually – but is it all?

NOPE

- [nəʊp], *adv (inf) ne(e)*
(dial), no.

There is:



- Betraying all your project's naughty little secrets





Tattletale

- Helps identify dependencies
 - within the project
 - on standard APIs
- Helps getting an overview of all Java archives
- Helps locate class information
- Generate reports that can help you improve the software quality of your project, e.g.:
 - splitting your project into the right number of archives
 - removing black listed API usage

Tattletale – an example report 1

- Dependants report

Dependants

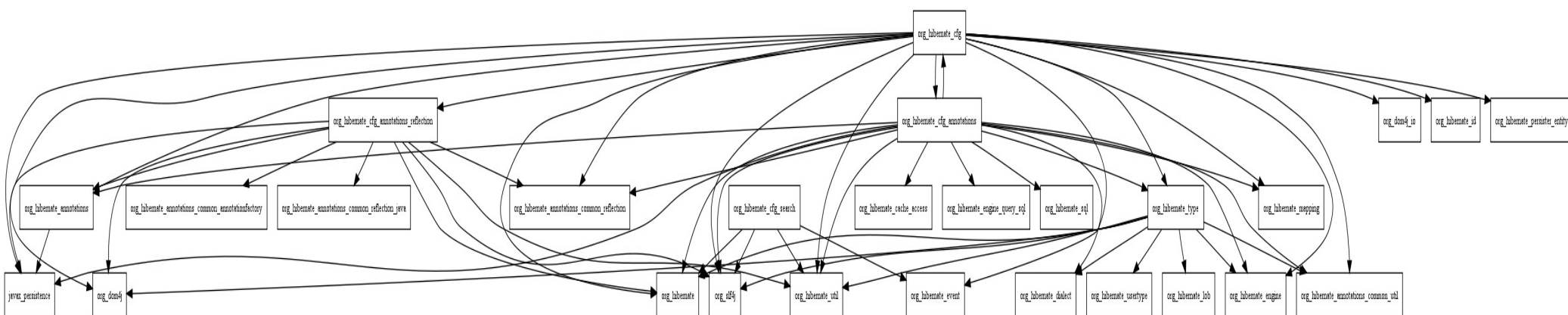
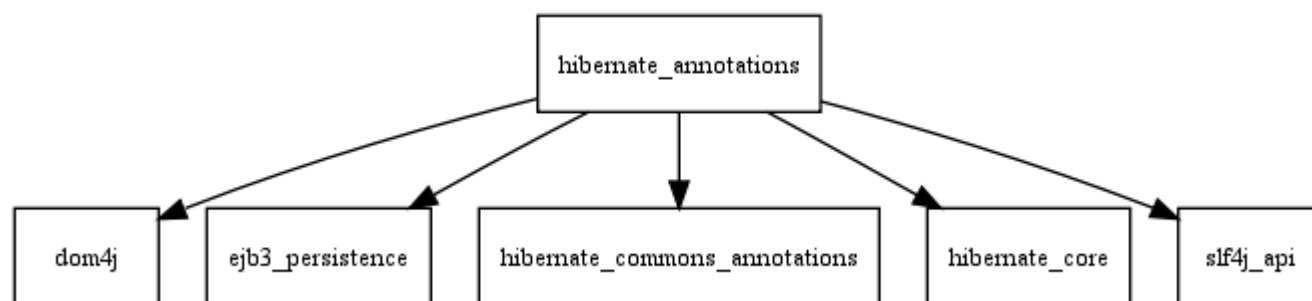
[Main](#)

Archive	Dependants
FastInfoset.jar	jaxb-impl.jar , jaxws-rt.jar , jbossws-native-core.jar
activation.jar	jaxb-api.jar , jaxb-impl.jar , jaxb-xjc.jar , jaxws-rt.jar , jboss-javaee.jar , jboss-xml-binding.jar , jbossws-common.jar , jbossws-native-core.jar , jbossws-native-saaj.jar , mail.jar , scout.jar , stax-ex.jar , streambuffer.jar
antlr.jar	hibernate-core.jar , jacorb.jar , jboss-seam-ui-2.1.0.SP1.jar , org.jboss.seam-jboss-seam-2.1.0.SP1.jar , richfaces-ui-3.3.0.GA.jar
applet.jar	
autonumber-plugin.jar	
avalon-framework.jar	jacorb.jar , jboss-iiop-client.jar
bccl.jar	jboss-mbeans.jar , jboss.jar
bsf.jar	bsh.jar , jboss-monitoring.jar
bsh-deployer.jar	
bsh.jar	applet.jar , bsh-deployer.jar , console-mgr-classes.jar , jboss-aop-aspects.jar , jgroups.jar



Tattletale – an example report 2

- Graphical dependencies report

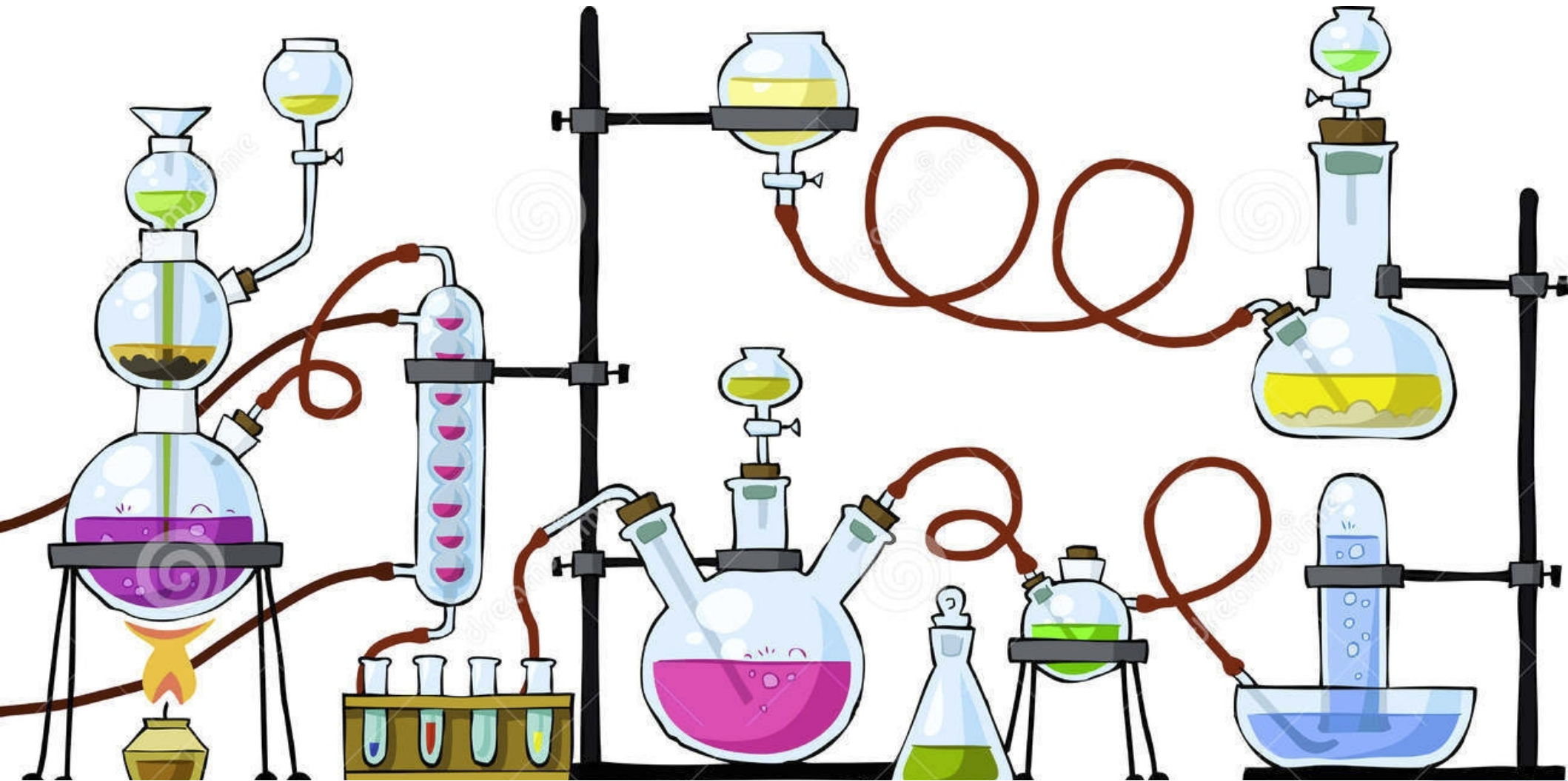


Tattletale - and there is much more

- So we've shown:
 - Dependants report
 - Graphical dependencies report
- But also the following reports:
 - Depends On
 - Transitive dependants
 - Transitive depends on
 - Class location
 - OSGi report
 - Eliminate Jars with different version
 - Invalid version report
 - Multiple Jar files
 - Multiple Locations
 - Black listed
 - No version
 - Archive: Jar



And now its time for **you** to get some work done!



Labs

- They are SOA 5 to Switchyard, but show the principles, so:
- Get hacking!

