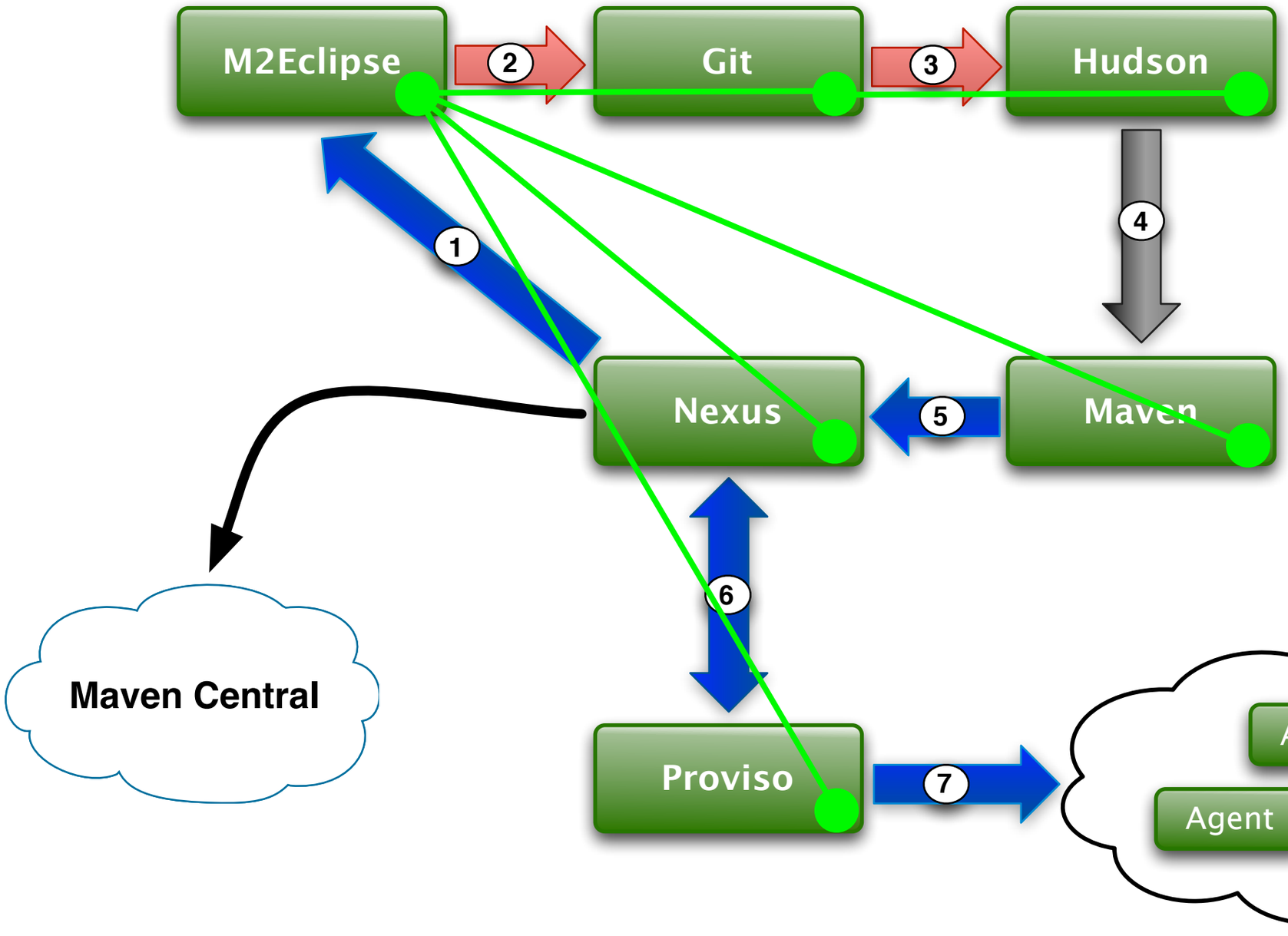
Next Generation Development Infrastructure: Maven, m2eclipse, Nexus & Hudson

The Maven-related tooling you'll be using in your infrastructure for years to come

Jason van Zyl
http://twitter.com/jvanzyl

Agenda & Session Goals

What we're going to talk about and accomplish this session



Advanced forms
of Maven like
Tycho
and
Polyglot Maven

News & Updates

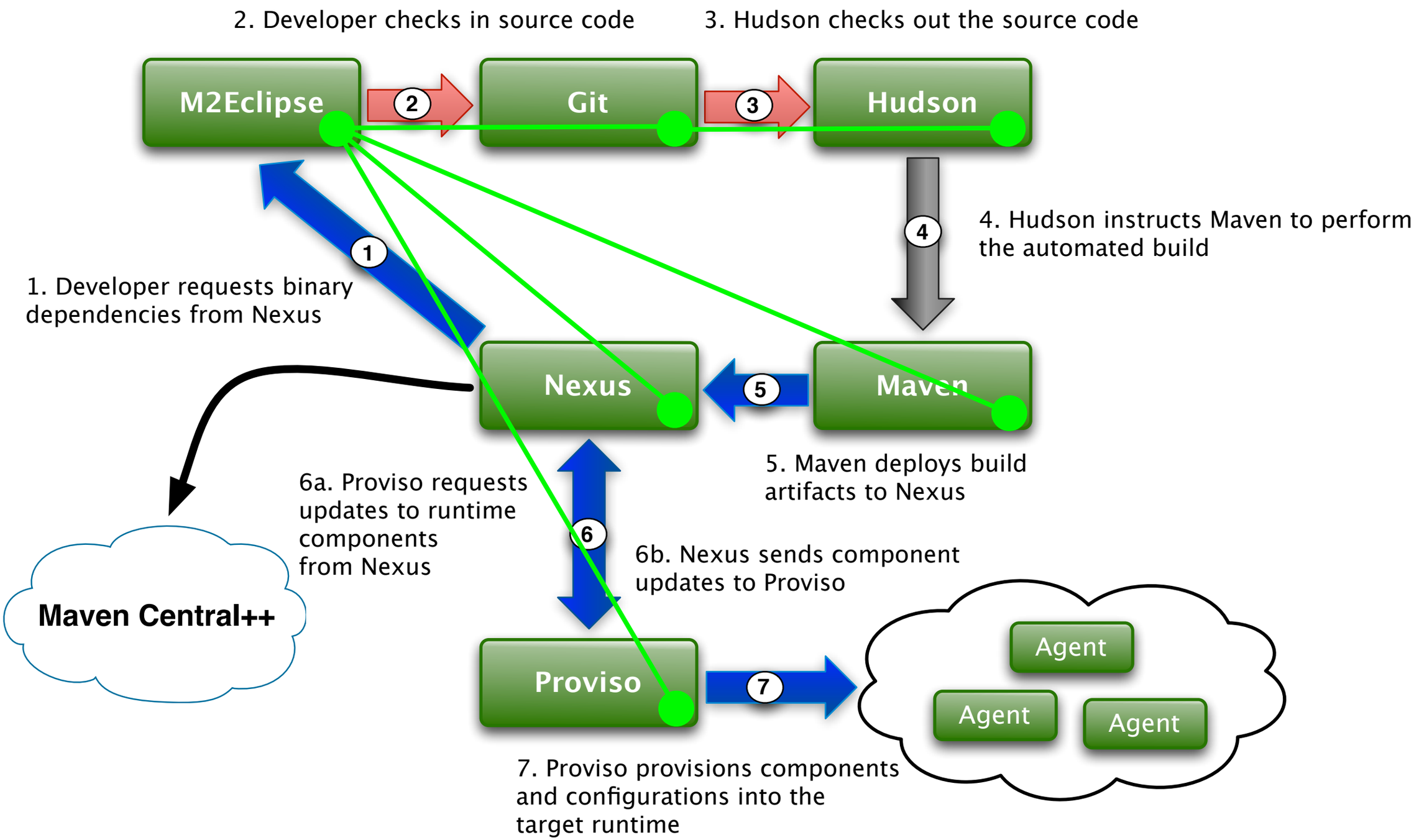
Updates on Maven, m2eclipse & Tycho

What's going on at Sonatype & in Maven land?

- Maven 3.0.3 has been released. We're working on new feature development for Maven 3.1
- Nexus 1.9.0.2 has been released
- Hudson 2.0.1 will be released this week
- m2eclipse 0.14 is in-progress and will be the basis of m2eclipse 1.0
- We are still working on bringing Tycho to the Eclipse Foundation
- We are still working on bringing Aether to the Eclipse Foundation

The ideal Maven-focused delivery infrastructure

What does that look like?



Overview

Maven

Maven has become the de facto build tool in the enterprise

- Maven 3.x has been in the wild for 8 months, currently at 3.0.3
- Sonatype has been working on some advanced features
  - Async HTTP Client Connector
  - Concurrent-safe local repository implementation
  - Layered local repository implementation
  - Improved and dynamic extension mechanism
  - JSR330-based plugins with a new Java5-based API

m2eclipse

m2eclipse is the standard Maven integration for Eclipse

- m2eclipse has been successfully transitioned to the Eclipse Foundation
- m2eclipse will graduate with Indigo and be shipped with the standard Eclipse distributions in June
- Extension point API is final
- Huge performance improvements
- Extension discovery mechanism: very much like the Mylyn connector discovery mechanism

Hudson

Continuous integration has become a daily part of developer life

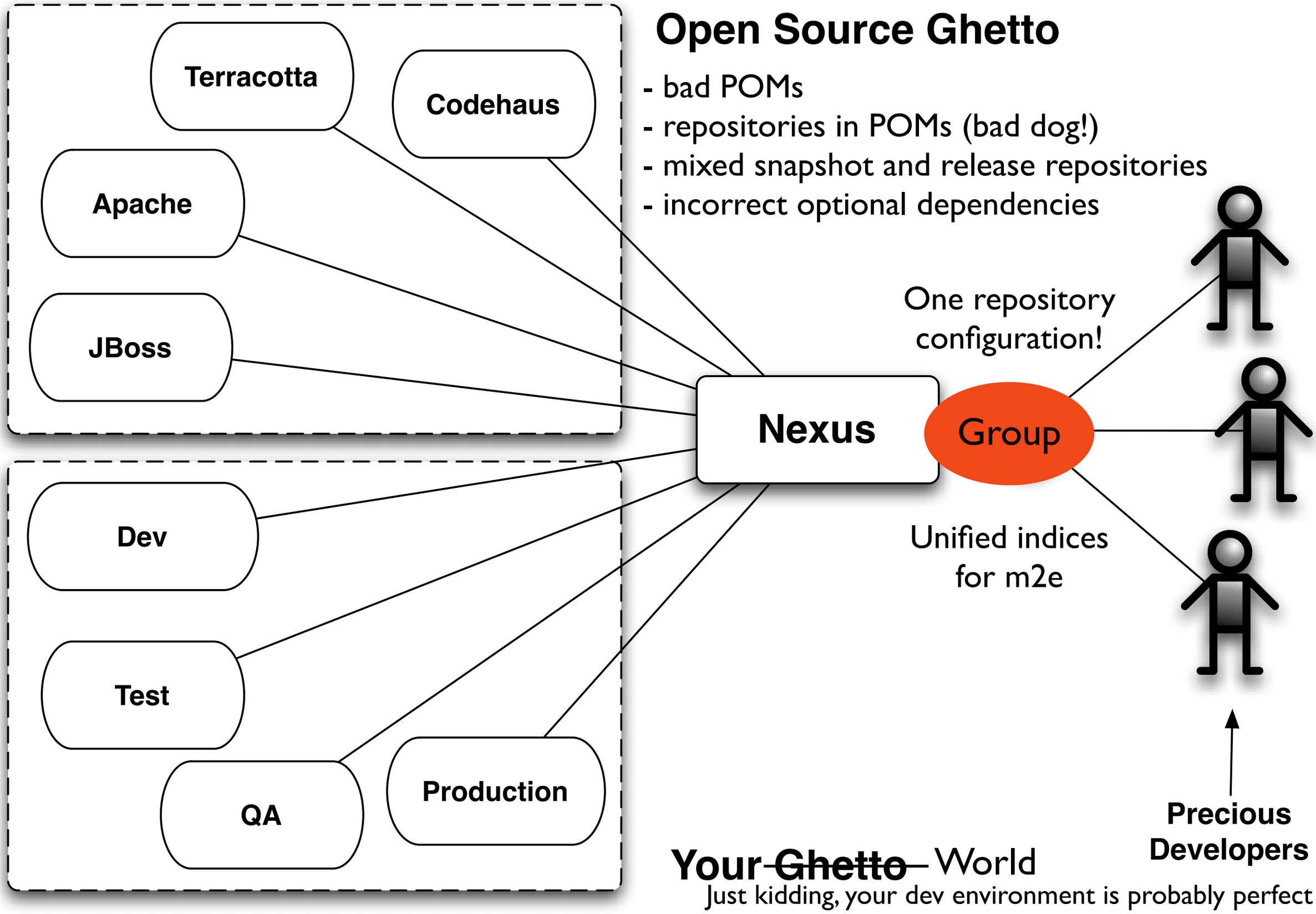
- Moving Hudson to the Eclipse Foundation
- Removal of all L/GPL dependencies
- Vast improvement of the test suite
- Automated release infrastructure
- Transition the core to using JSR330
- Use of JAXRS for webservices
- GWT UI option
- Maven 3.x integration from Sonatype
  - Uses JSR330, JAXRS, and GWT

Why Nexus?

Using an enterprise repository manager is becoming standard

- Using binary artifacts is the standard way to integrate 3rd party functionality whether that be from within your organization or from OSS communities. Groups generally consume dependencies in binary form, not source form.
- Nexus provides an analog to your SCM except that Nexus controls binaries. You can think of Nexus as binary configuration management system. It really is very similar because you're moving streams of binaries around in very much the same way you move streams of sources around.
- Any measure of reuse will come from analyzing the traffic of binary artifacts through your system.
- Any legal compliance or procurement process will be based on gating the use of binary artifacts.
- Analysis as it pertains to artifacts requires an application like Nexus. Parsing log files from your Apache instance can't provide the information you need to understand the use of artifacts in your organization.

Typical Nexus Setup



Open Source Ghetto

- bad POMs
- repositories in POMs (bad dog!)
- mixed snapshot and release repositories
- incorrect optional dependencies

Your ~~Ghetto~~ World

Just kidding, your dev environment is probably perfect

Let’s dig in!

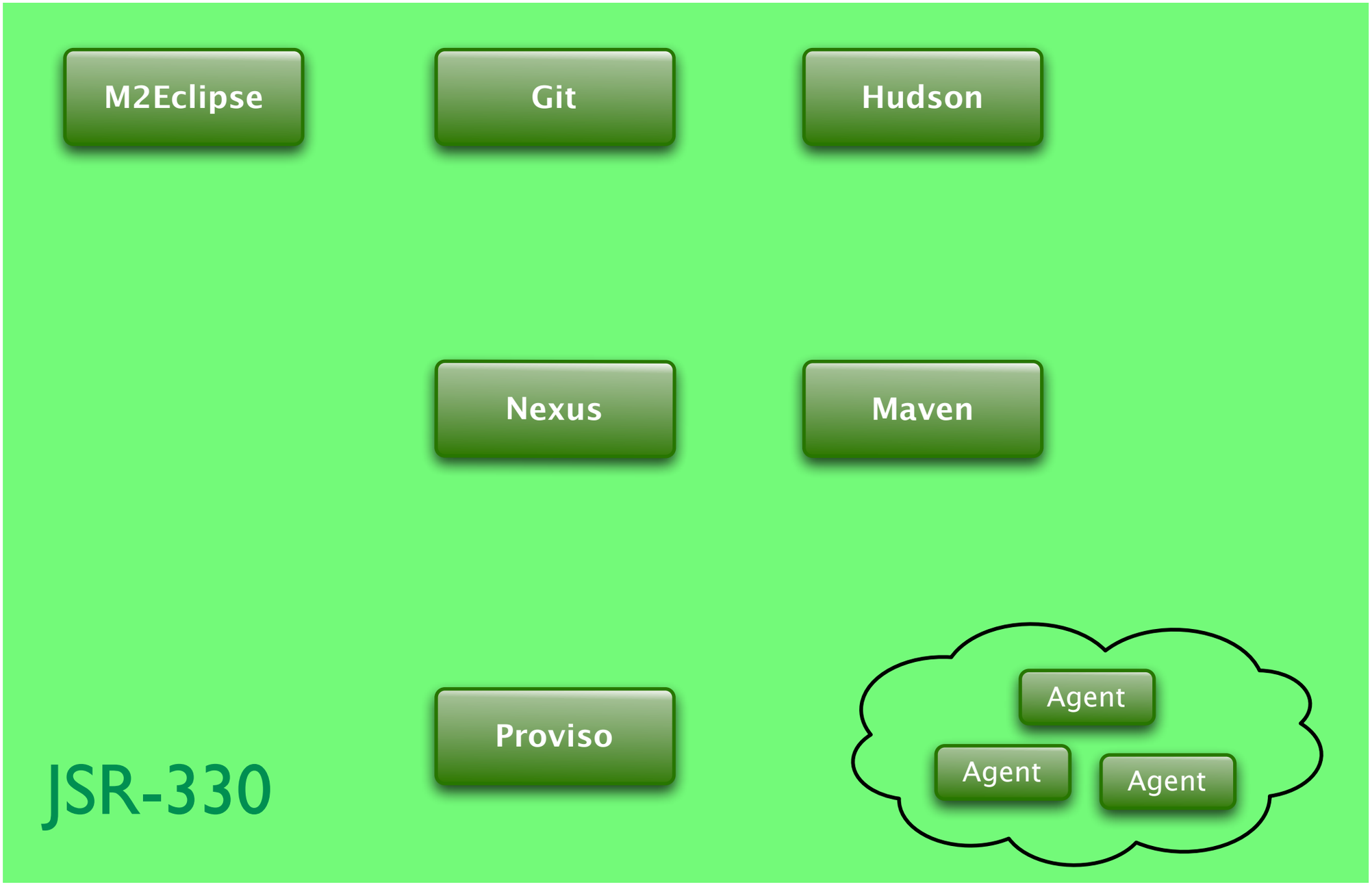
The ideal Maven-focused delivery infrastructure

What does that look like?

- Shared component model
- Shared transport system
- Shared repository API
- Enriched component metadata
- Enhanced IDE connectivity to the infrastructure

Shared component model

For Sonatype this means JSR-330, Guice & Sisu



Sonatype Sisu

Extensions to Guice for JSR330+

```
@Named
@Typed( SomeType.class )
class AnotherComponent
{
}
```

```
@Named( "hint" )
@EagerSingleton
class SomeComponent
{
}
```

```
@Inject
List<Component> componentList;
```

```
@Inject
Map<String, Component> componentMap;
```

```
@Inject
@Named( "${dbdriver}://${dbhost}:${dbport:-5432}/${dbname}" )
URL databaseURL;
```

Moving from Plexus to Guice & JSR-330

Making it all work with Guice

- Requirements
  - Absolutely no code changes for any Maven, Nexus, M2Eclipse component
  - Must support Plexus' classpath/resource scanning
  - Must support Plexus' dynamic component assembly based on discovered metadata
  - Must support Plexus' configuration & converter mechanism
  - When we need changes made to the runtime container, we need those changes to be timely
- Support for arbitrary lifecycles
- We need the container to be wed with OSGi -- for us the answer is Peaberry
- Component graph proxy support: for components and configuration
- Dynamic language support

Implications of using JSR-330

We bring some sanity to tooling

- Writing plugins in various ways for tools like Maven, Nexus, Hudson, Sonar & Eclipse has a great deal of mental overhead. This burden will be removed.
- The implications for development, testing and delivery are **huge**. They cannot be understated
  - Common development models: how to create JSR-330-based plugins, better component reuse, a common understand of infrastructure tooling
  - Common testing frameworks for JSR-330 e.g Sonatype's REST/UI toolkit
  - Common bridge to OSGi
  - Common provisioning models

Sisu Maven Plugin Example

Using the same component model

```
@Goal( "webxml" )
@Phase( GENERATE_RESOURCES )
@RequiresProject
@Threadsafe
public class GenerateWebXml extends SisuMavenMojo {
    @Inject Logger logger;

    @Inject
    private Component component;

    @Inject @Named( "${project}" )
    private MavenProject project;

    @Inject @Named( "${outputDirectory}" ) @Default( "${project.build.directory}" )
    private File outputDirectory;

    @Inject
    private List<WebXmlAugmenter> webXmlAugmenters;

    public void execute() throws Exception {
        component.generate( project, webXmlAugmenters, outputDirectory );
    }
}
```

Sisu Hudson Plugin Example

Using the same component model

```
@Named
@Singleton
public class RestPlugin
    extends Plugin
{
    @Inject
    private Logger logger;

    private transient List<ApiProvider> providers;

    private boolean enabled = true;

    @Inject
    public RestPlugin(final List<ApiProvider> providers) {
        assert providers != null;
        this.providers = providers;

        logger.debug("Providers:");
        for (ApiProvider provider : providers) {
            logger.debug( "    {}", provider );
        }
    }
}
```

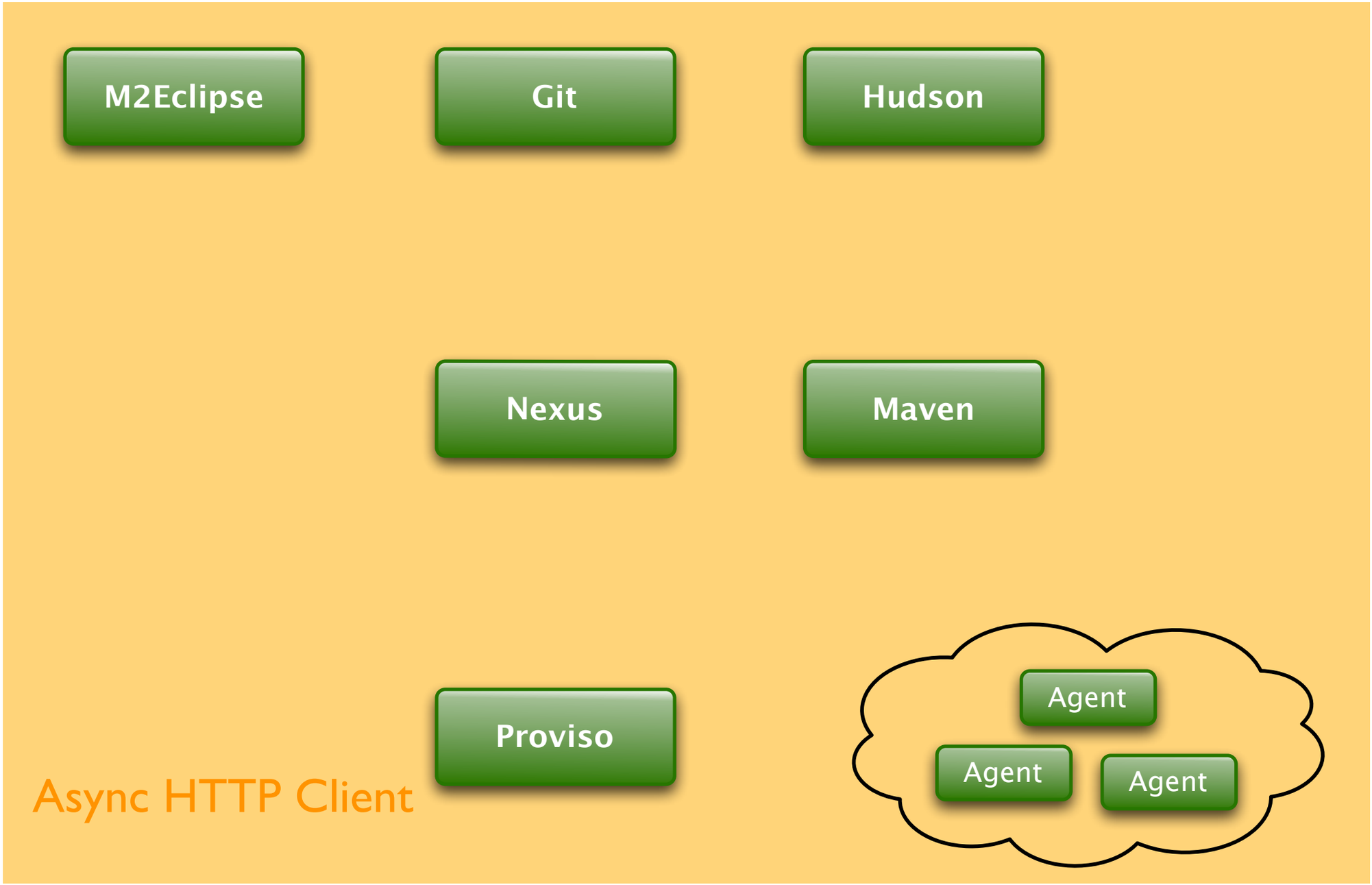
Sisu Nexus Plugin Example

Using the same component model

```
@Named
@Singleton
@Path( CapabilitiesResource.RESOURCE_URI )
@Produces( { "application/xml", "application/json" } )
@Consumes( { "application/xml", "application/json" } )
public class CapabilitiesResource
    implements Resource {
    public static final String RESOURCE_URI = "/capabilities";

    private final CapabilityConfiguration capabilitiesConfiguration;

    private final CapabilityDescriptorRegistry capabilityDescriptorRegistry;

    @Inject
    public CapabilitiesResource( CapabilityConfiguration capabilitiesConfiguration,
                                 CapabilityDescriptorRegistry capabilityDescriptorRegistry )
    {
        this.capabilitiesConfiguration = capabilitiesConfiguration;
        this.capabilityDescriptorRegistry = capabilityDescriptorRegistry;

        ...
    }
}
```

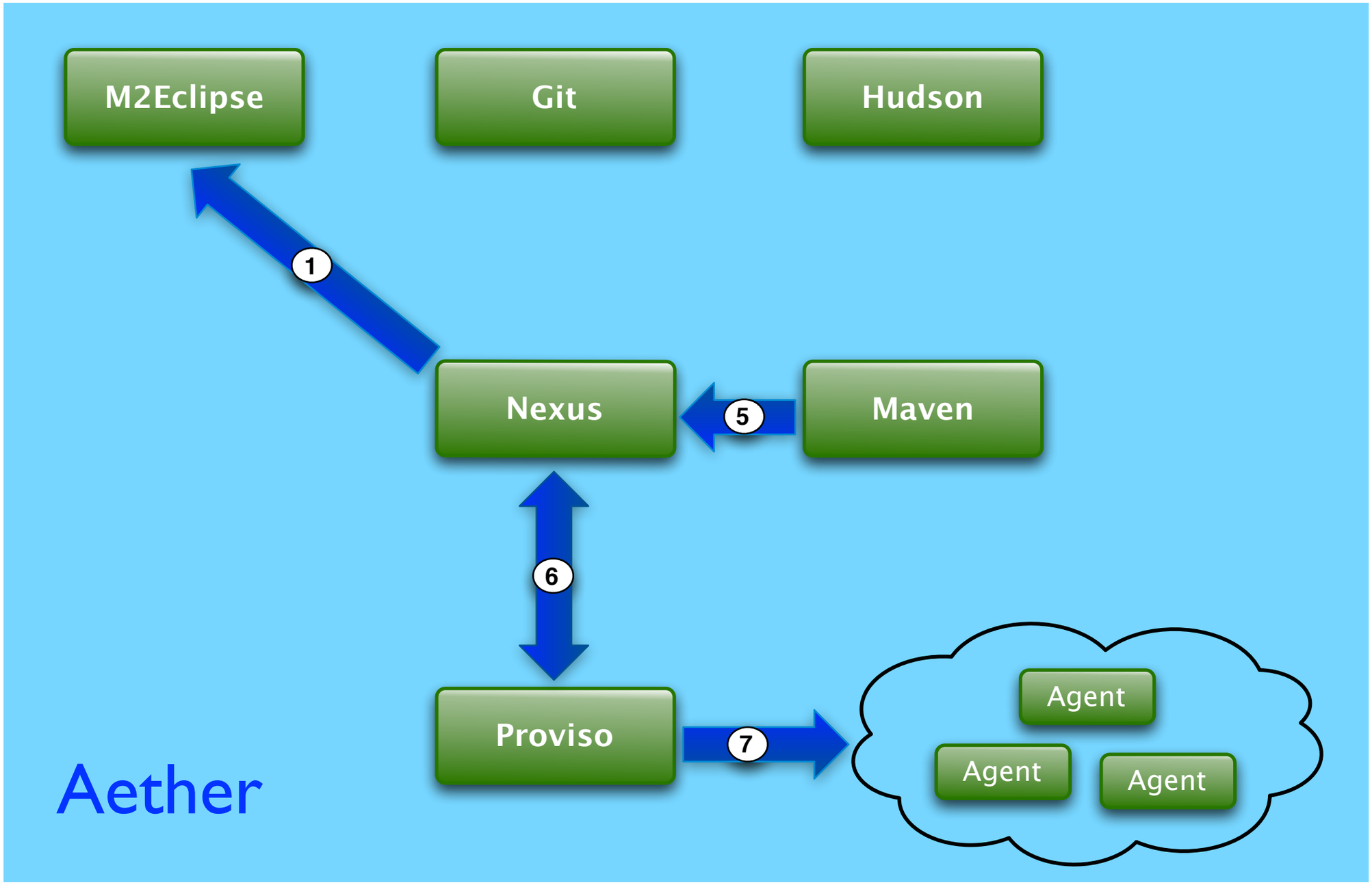
Shared transport system

For Sonatype this means the Async HTTP Client



Shared repository API

For Sonatype this means our new Aether library



Aether

Overhauled Repository Artifact Resolution API

- The artifact resolution code has always been relatively decoupled, but Aether is a completely stand-alone library and has no dependencies on Maven
- SSL support
- DAV support
- Transport
  - We’re using the Async HTTP client being developed by Jean-francois Arcand at Sonatype
- Large file support
- Resumable downloads
- Complete proxy support
- Complete NTLMv1 & v2 support
- Sonatype would like to move Aether to the Eclipse Foundation

Aether Resolution Example

Easy to embed and simply use as a library

```
public void resolve( String remoteRepository, File localRepository )
    throws DependencyCollectionException, ArtifactResolutionException
{
    Aether aether = new Aether( repoRepository, localRepository );

    AetherResult result = aether.resolve( "com.mycompany.app", "super-app", "1.0" );

    // Get the root of the resolved tree of artifacts
    //
    DependencyNode root = result.getRoot();

    // Get the list of files for the artifacts resolved
    //
    List<File> artifacts = result.getResolvedFiles();

    // Get the classpath of the artifacts resolved
    //
    String classpath = result.getResolvedClassPath();
}
```

Aether Install & Deploy Example

Easy to embed and simply use as a library

```
public void installAndDeploy( String remoteRepository, File localRepository
                              String deployRepository )
    throws InstallationException, DeploymentException
{
    Aether aether = new Aether( remoteRepository, localRepository );

    Artifact artifact =
        new DefaultArtifact( "com.mycompany", "super-core", "jar", "1.0" );
    artifact = artifact.setFile( new File( "jar-from-whatever-process.jar" ) );
    Artifact pom = new SubArtifact( artifact, null, "pom" );
    pom = pom.setFile( new File( "pom-from-whatever-process.xml" ) );

    // Install into the local repository specified
    //
    aether.install( artifact, pom );

    // Deploy to the specified deploy repository
    //
    aether.deploy( artifact, pom, deployRepository );
}
```

Enriched component metadata

What is that and how do we get it?

- First we need to clean up the way artifacts get into Maven Central
  - License information
  - Security vulnerability information
  - Compatibility information
  - Introduce new component types...

Maven Central Quality

Sonatype is working hard to clean up Maven Central

Dashboard › Repository › Home Browse ▾ Jason van Zyl ▾ Search

Home

https://docs.sonatype.org/display/Repository/Home

Added by Brian Fox, last edited by Juven Xu on Aug 26, 2010 (view change)

In this space you will find up to date documentation on the release process using Nexus at various Forges as well as requirements and tutorials for getting your builds to produce the required artifacts.

Forge Guides

Sonatype OSS Maven Repository Usage Guide
Codehaus Maven Repository Usage Guide
Uploading 3rd-party Artifacts to Maven Central
Apache Repository Usage Guide
Choosing your Coordinates
Central Repository FAQ

Tutorials

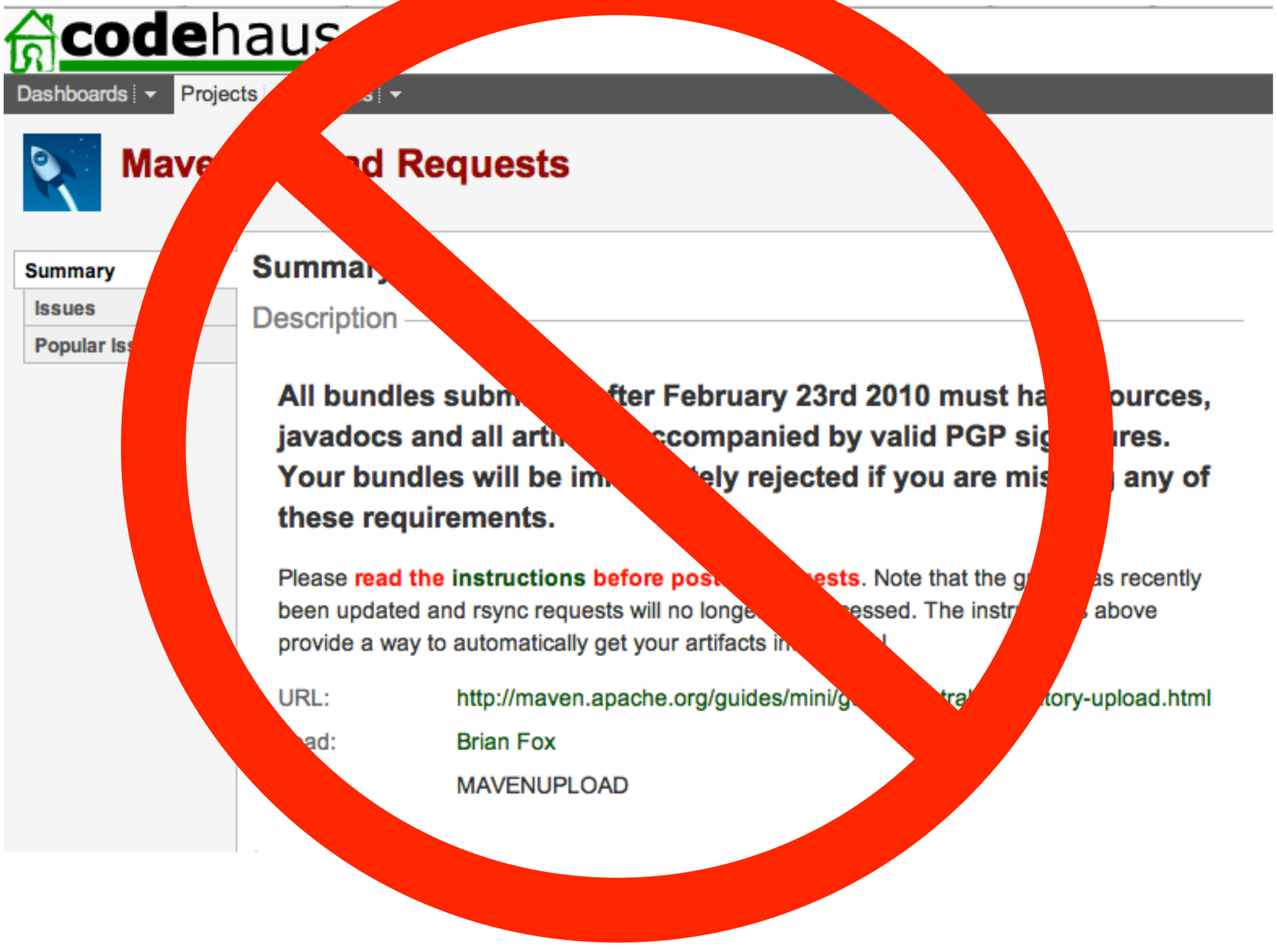
Publish Snapshots
Stage a Release
Closing a Staging Repository
Dropping a Staging Repository
Releasing a Staging Repository
How To Generate PGP Signatures With Maven

Requirements

Central Sync Requirements
Client System Prerequisites

Setting up your own Forge

How Repository.Apache.Org is configured
How Nexus.Codehaus.Org is configured
Sonatype OSS Administration Guide


codehaus
Dashboards
Projects
Summary
Issues
Description
All bundles subm ter February 23rd 2010 must ha ources,
javadocs and all arti ccompanied by valid PGP sig ures.
Your bundles will be im ely rejected if you are mis any of
these requirements.
Please read the instructions before post ests. Note that the g as recently
been updated and rsync requests will no longe essed. The instr above
provide a way to automatically get your artifacts in
URL:
http://maven.apache.org/guides/mini/g tra tory-upload.html
ad:
Brian Fox
MAVENUPLOAD

Sonatype Nexus Maven Repository Manager

http://localhost:8081/nexus/index.html#staging-upload

Google

Nexus Benson PDS Scrum MSE Backlog Sisu Guide Benson REST API Sonatype PBX Sonatype KPIs Flixster Clothes Publish

Sonatype

admin | Log Out

Sonatype Nexus™ Team Edition, Version: 1.7.1.2

Sonatype™ Servers

Nexus

Artifact Search

Advanced Search

Build Promotion

Staging Profiles
Staging Repositories
Staging Ruleset
Staging Upload

Views/Repositories

Repositories
System Feeds

Enterprise

Artifact Procurement
Maven Settings

Administration

Security

Help

Welcome | **Staging Upload**

Select Staging Upload Mode

Upload Mode: Select...

Artifact(s) with a POM: GAV will be defined from a POM file.
Artifact(s) with GAV: GAV needs to be manually defined.
Artifact Bundle: A bundle file produced by Maven Repository Plugin, which should contain the POM.

Select Artifact(s) for Upload

Select Artifact(s) to Upload...

Filename:

Classifier:

Extension:

Add Artifact

Artifacts

Remove

Remove All

Description:

Upload Artifact(s) Reset

Sonatype Nexus Maven Repository Manager

http://localhost:8081/nexus/index.html#staging-upload

Nexus Benson PDS Scrum MSE Backlog Sisu Guide Benson REST API Sonatype PBX Sonatype KPIs Flixster Clothes Publish

Sonatype

admin | Log Out

Sonatype Nexus™ Team Edition, Version: 1.7.1.2

Sonatype™ Servers

Nexus

Artifact Search

Advanced Search

Build Promotion

Staging Profiles
Staging Repositories
Staging Ruleset
Staging Upload

Views/Repositories

Repositories
System Feeds

Enterprise

Artifact Procurement
Maven Settings

Administration

Security

Help

Welcome | **Staging Upload**

Select Staging Upload Mode

Upload Mode:

- Artifact(s) with a POM
- Artifact(s) with GAV
- Artifact Bundle

Artifact(s) with a a POM file.
Artifact(s) with defined
Artifact Bundle:

```
pom.xml
foo-1.0.pom.asc
foo-1.0.jar
foo-1.0.jar.asc
foo-1.0-sources.jar
foo-1.0-sources.jar.asc
foo-1.0-javadoc.jar
foo-1.0-javadoc.jar.asc
```

Select Artifact(s) for Upload

Select Artifact(s) to Upload...

Filename:

Classifier:

Extension:

Add Artifact

Artifacts

Remove

Remove All

Description:

Upload Artifact(s) Reset

Sonatype Nexus Maven Repository Manager

https://oss.sonatype.org/index.html#view-repositories;google~browsestorage

Nexus Benson PDS Scrum MSE Backlog Sisu Guide Benson REST API Sonatype PBX Sonatype KPIs Flixster Clothes Publish

Sonatype | OSS

Log In

Sonatype Nexus™ Professional Edition, Version: 1.7.2

Sonatype™ Servers

Nexus

Artifact Search

Advanced Search

Views/Repositories

Repositories

Help

Welcome

Refresh User Managed Repos

Repository

Github Repositories

Github With Staging

Google Repositories

Google With Staging

Googlecode Repositories

Googlecode With Staging

Graniteds Repositories

```
<project>
  ...
  <parent>
    <groupId>org.sonatype.oss</groupId>
    <artifactId>oss-parent</artifactId>
    <version>5</version>
  </parent>
  ...
</project>
```

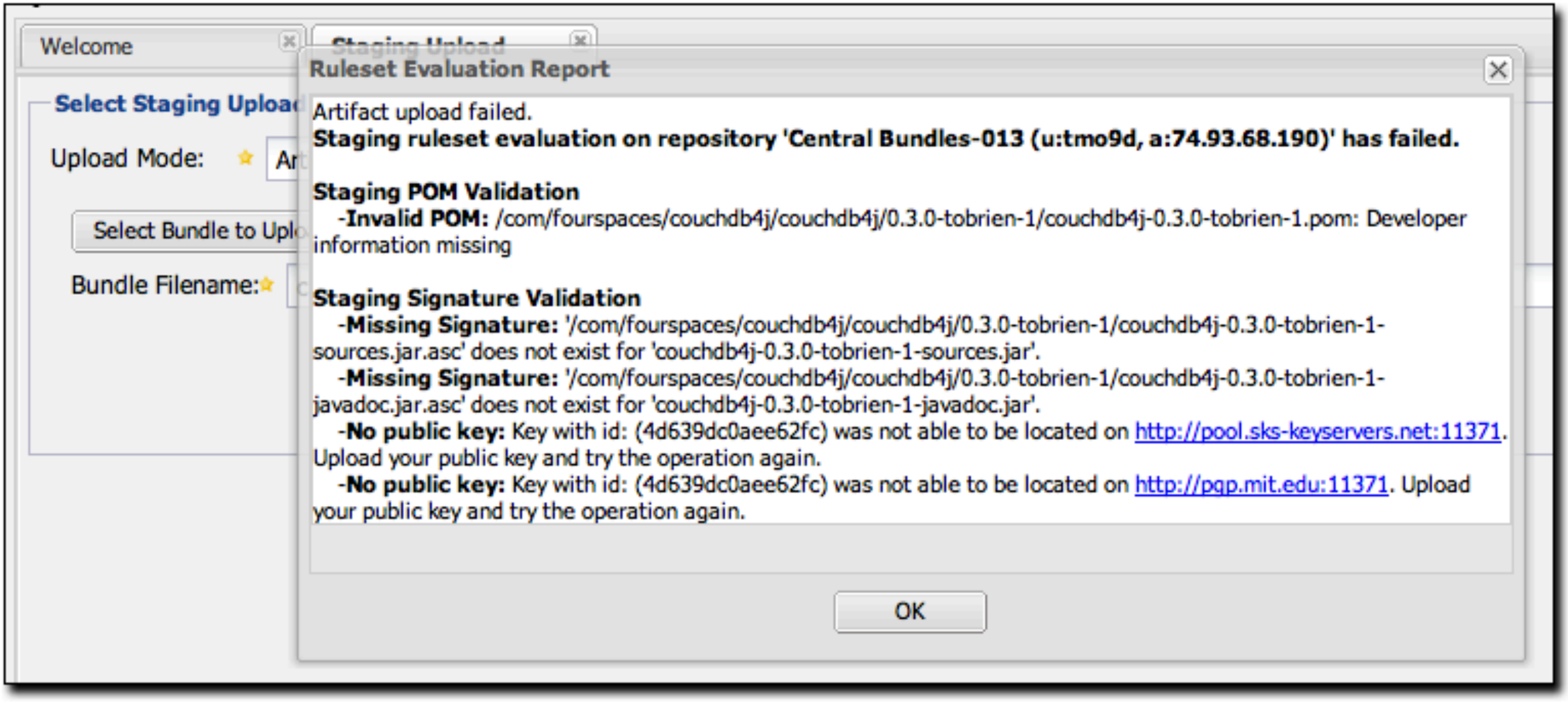
Google Repositories

Browse Storage | Browse Index

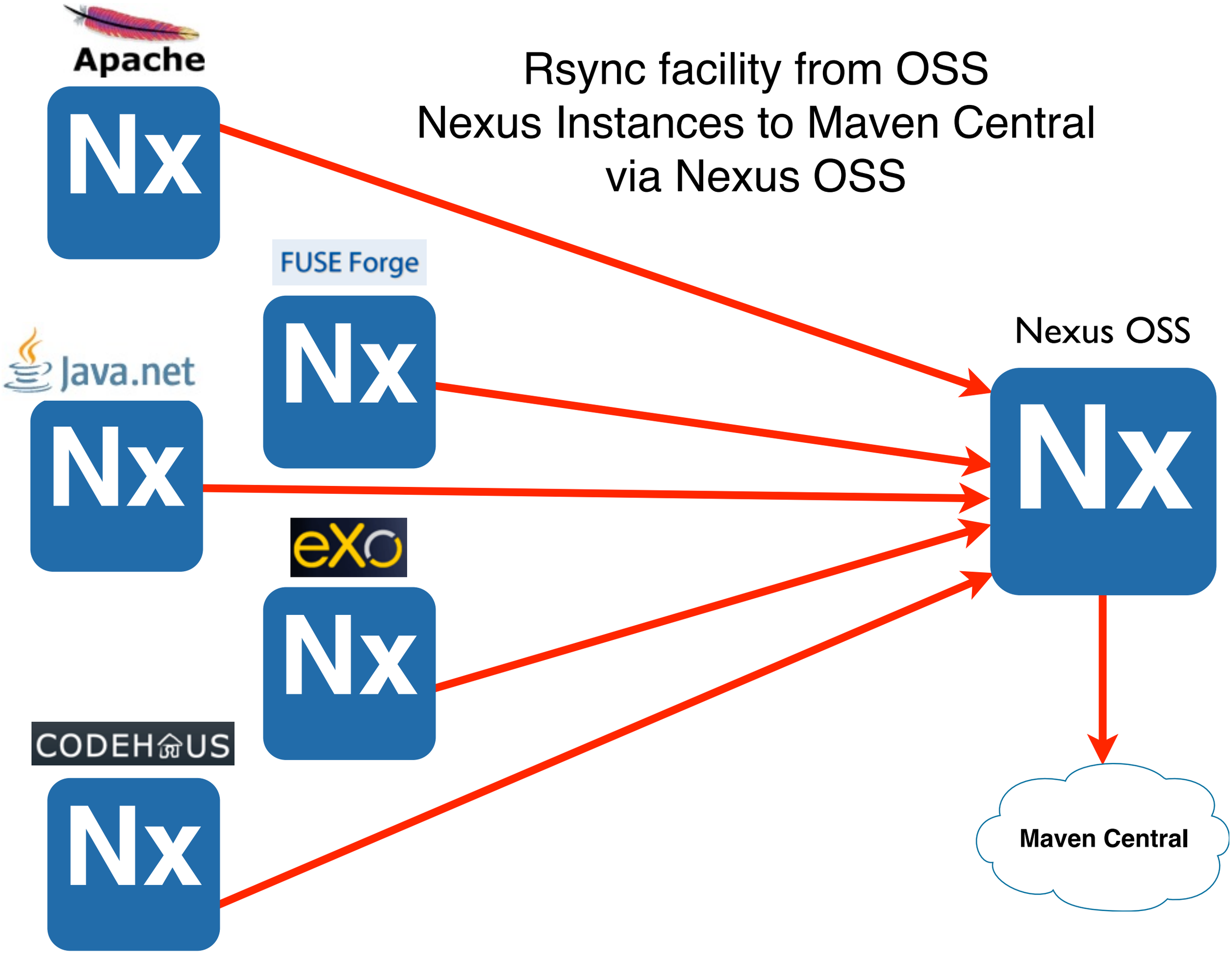
Refresh Path Lookup:

- Google Repositories
  - .index
  - .meta
  - com
    - google
      - api
      - appengine
      - classpath-explorer
      - collections
      - google
      - guava
      - gwt
      - gxp
      - inject

POM


Welcome
Staging Upload
Select Staging Upload
Upload Mode:
Select Bundle to Upl
Bundle Filename:
Ruleset Evaluation Report
Artifact upload failed.
Staging ruleset evaluation on repository 'Central Bundles-013 (u:tmo9d, a:74.93.68.190)' has failed.
Staging POM Validation
-Invalid POM: /com/fourspaces/couchdb4j/couchdb4j/0.3.0-tobrien-1/couchdb4j-0.3.0-tobrien-1.pom: Developer information missing
Staging Signature Validation
-Missing Signature: '/com/fourspaces/couchdb4j/couchdb4j/0.3.0-tobrien-1/couchdb4j-0.3.0-tobrien-1-sources.jar.asc' does not exist for 'couchdb4j-0.3.0-tobrien-1-sources.jar'.
-Missing Signature: '/com/fourspaces/couchdb4j/couchdb4j/0.3.0-tobrien-1/couchdb4j-0.3.0-tobrien-1-javadoc.jar.asc' does not exist for 'couchdb4j-0.3.0-tobrien-1-javadoc.jar'.
-No public key: Key with id: (4d639dc0aee62fc) was not able to be located on http://pool.sks-keyservers.net:11371. Upload your public key and try the operation again.
-No public key: Key with id: (4d639dc0aee62fc) was not able to be located on http://pgp.mit.edu:11371. Upload your public key and try the operation again.
OK

Rsync facility from OSS Nexus Instances to Maven Central via Nexus OSS



Enhanced IDE connectivity to the infrastructure

The IDE is the cockpit for a developer and be easy to get into

- Onboarding & updating
  - Getting developers up and running quickly & helping developers update environments and transition to new projects
- Connectivity to
  - Maven
  - SCM
  - Hudson
  - Nexus
  - Proviso

Developer Onboarding & Updating

MSE Codebase

- Eclipse Preferences
- Codebase Description/Icon
- **Source Tree** (one or more)
  - SCM Information
  - Project Information
  - Source Roots
- **MSE Lineup**
  - Eclipse Distribution
  - Eclipse Plugins/ Components

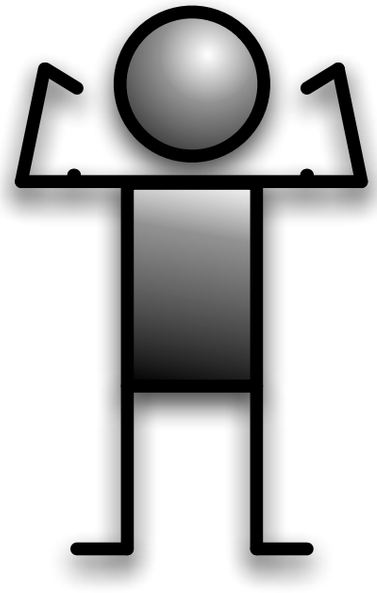
Publish Lineup

Publish Codebase

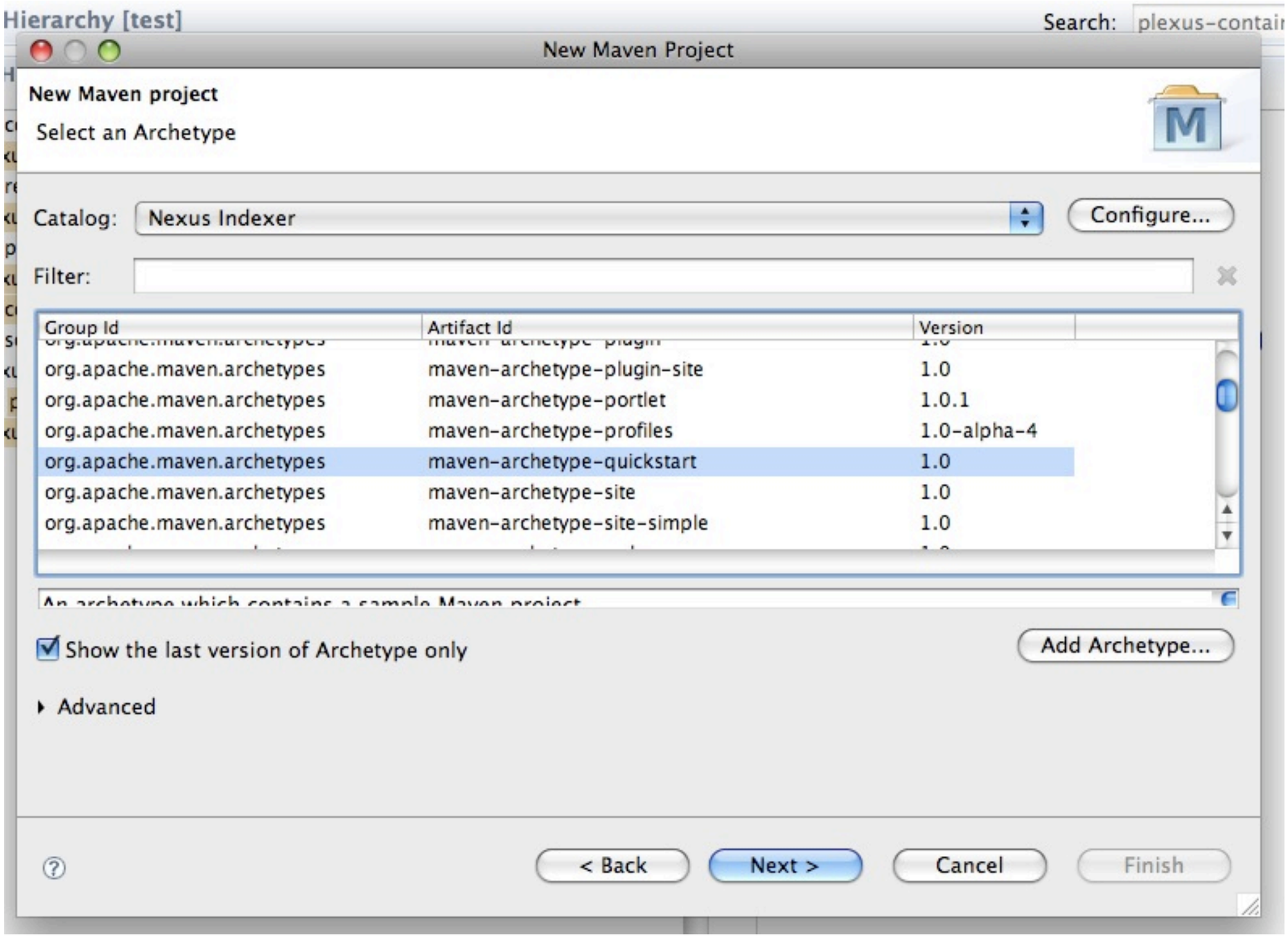
Nx

Request Codebase

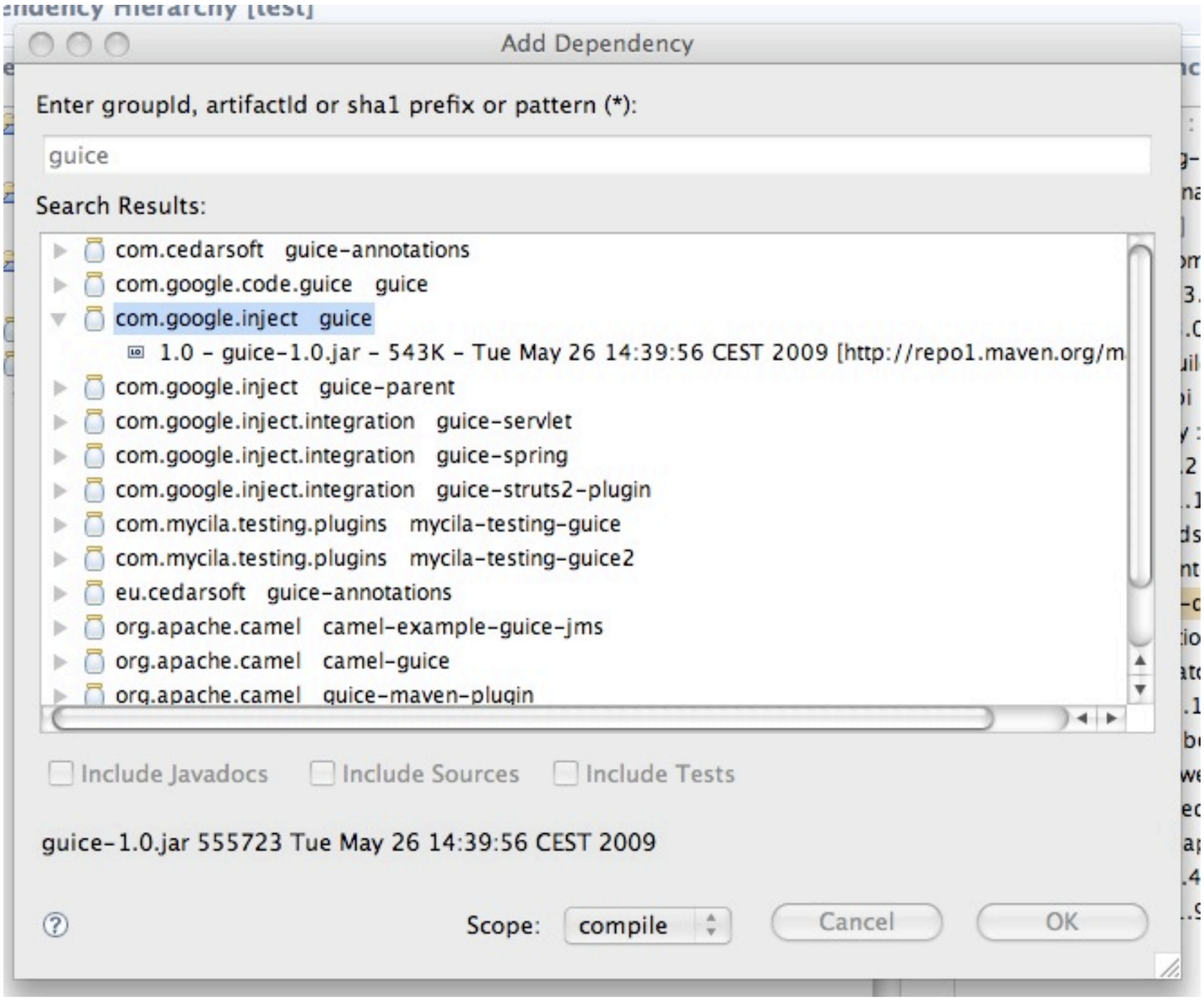
Materialize/Update



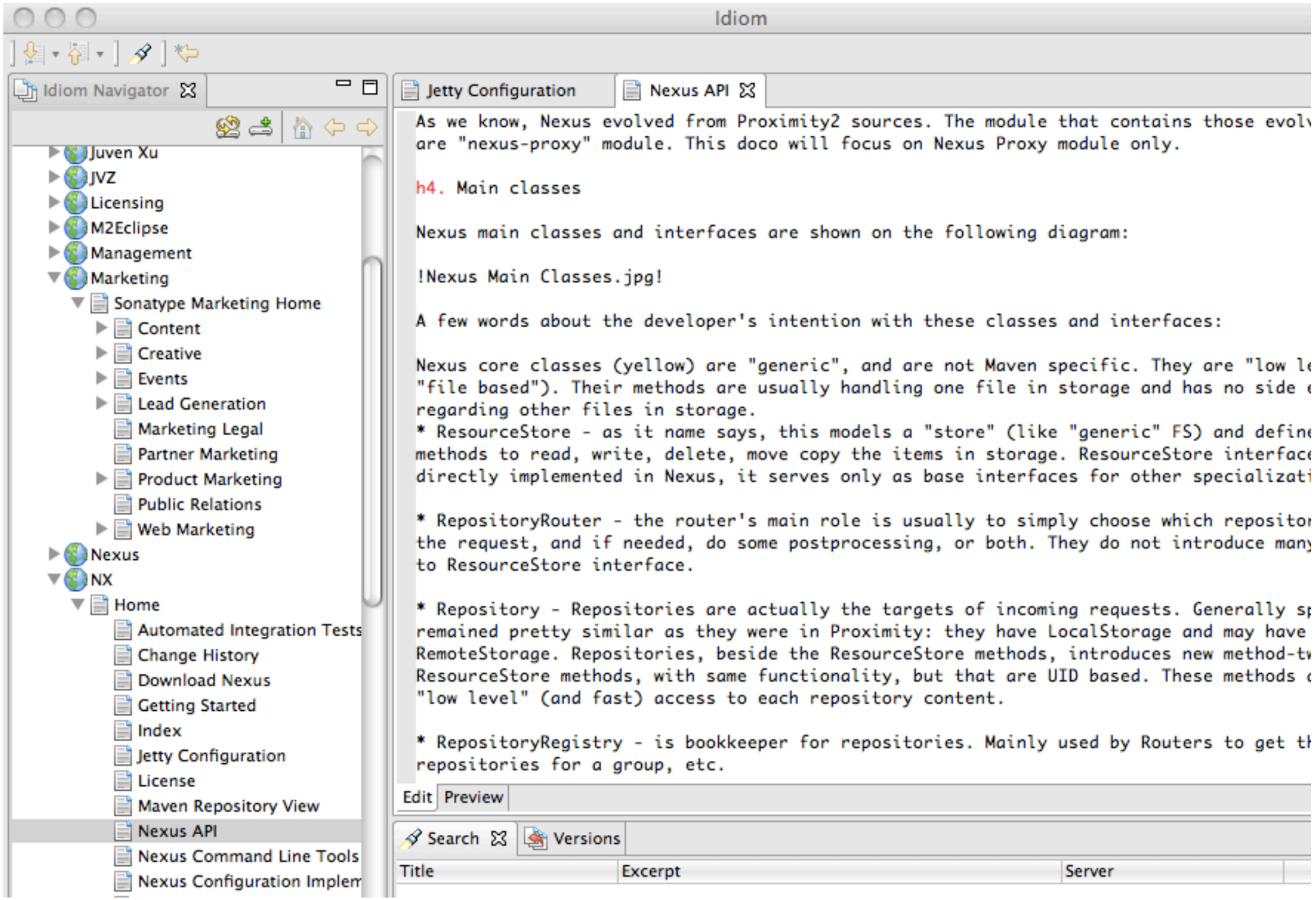
Nexus: Access to Archetypes



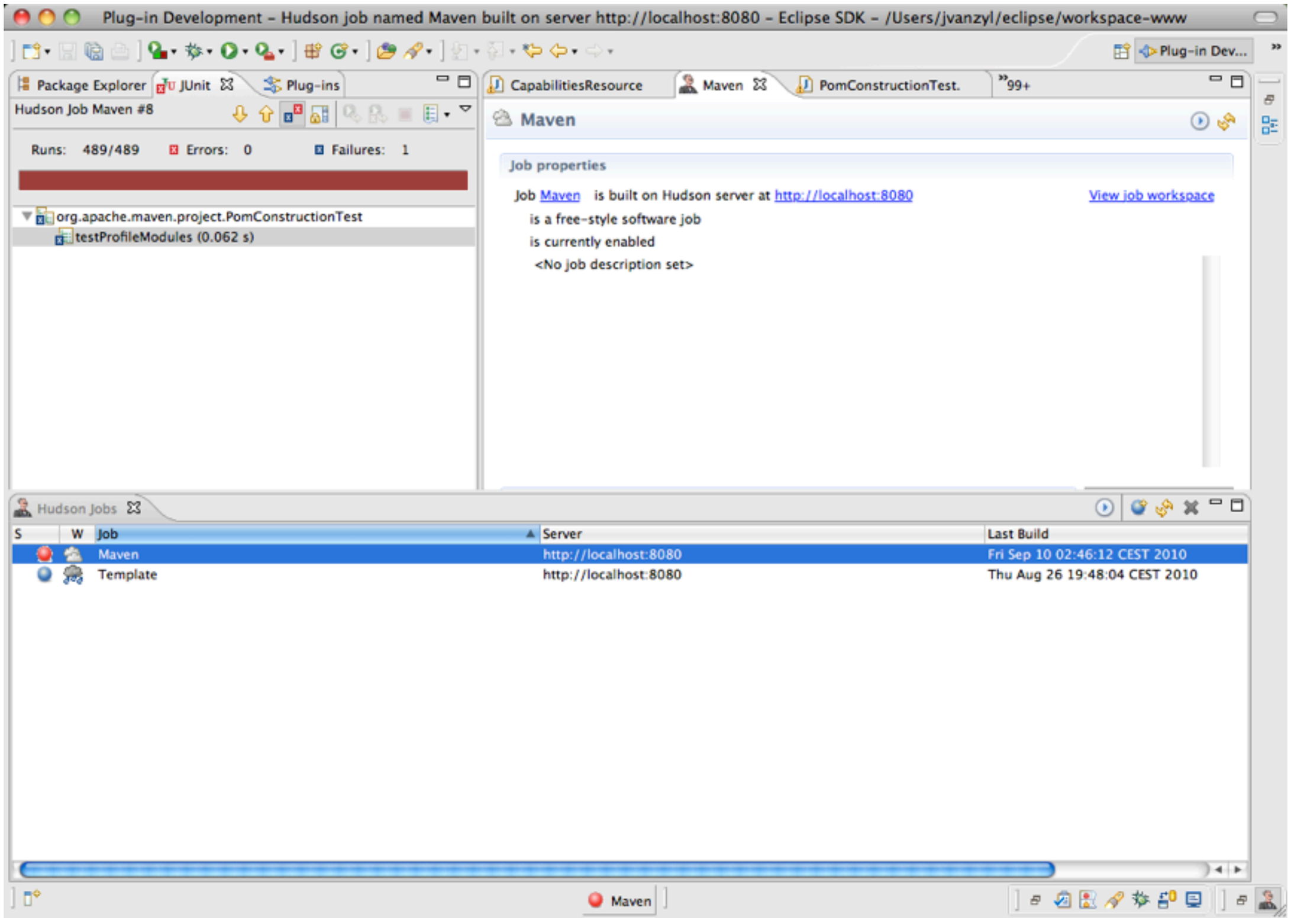
Nexus: Access to Artifacts



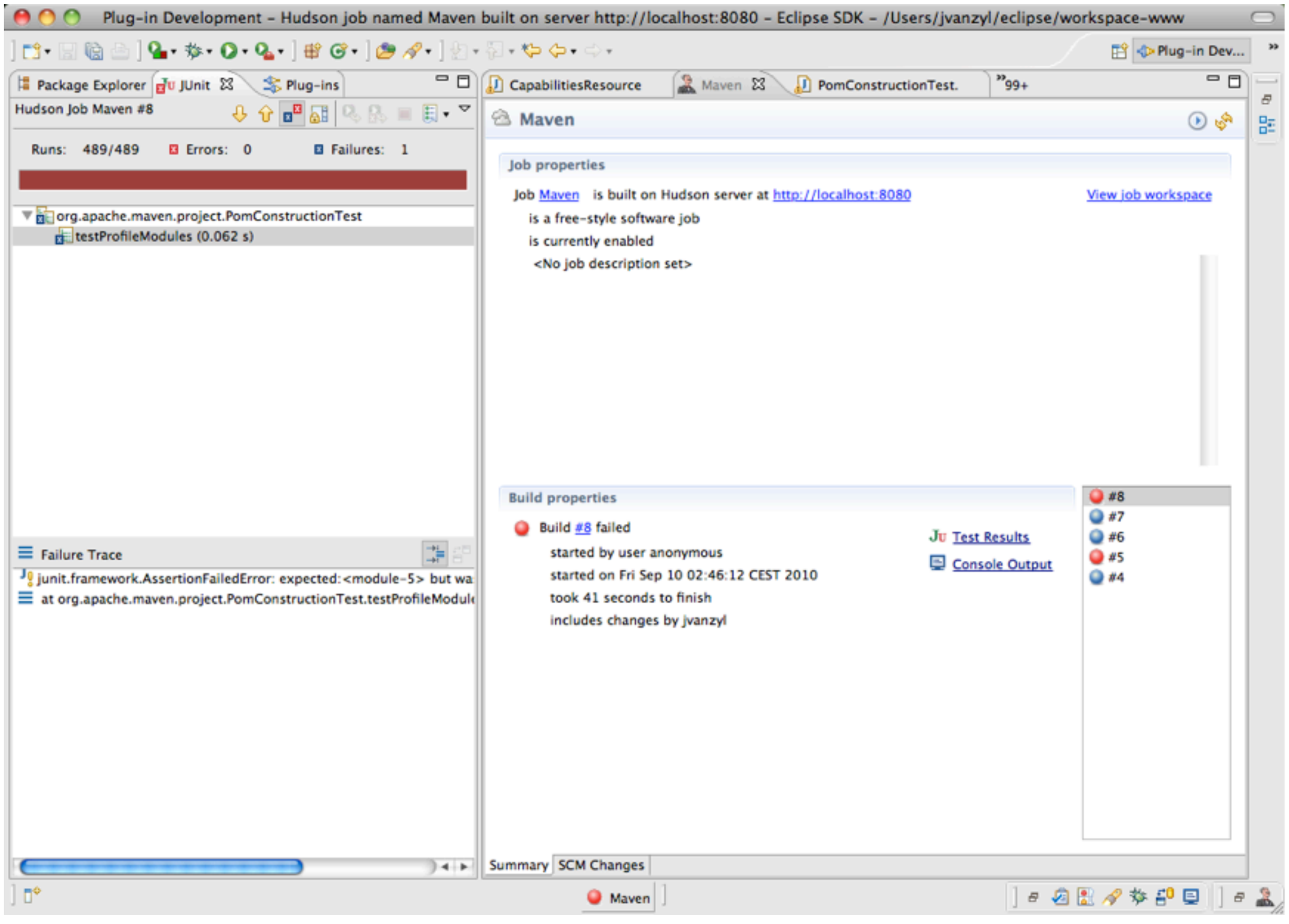
Idiom: Access Wikis



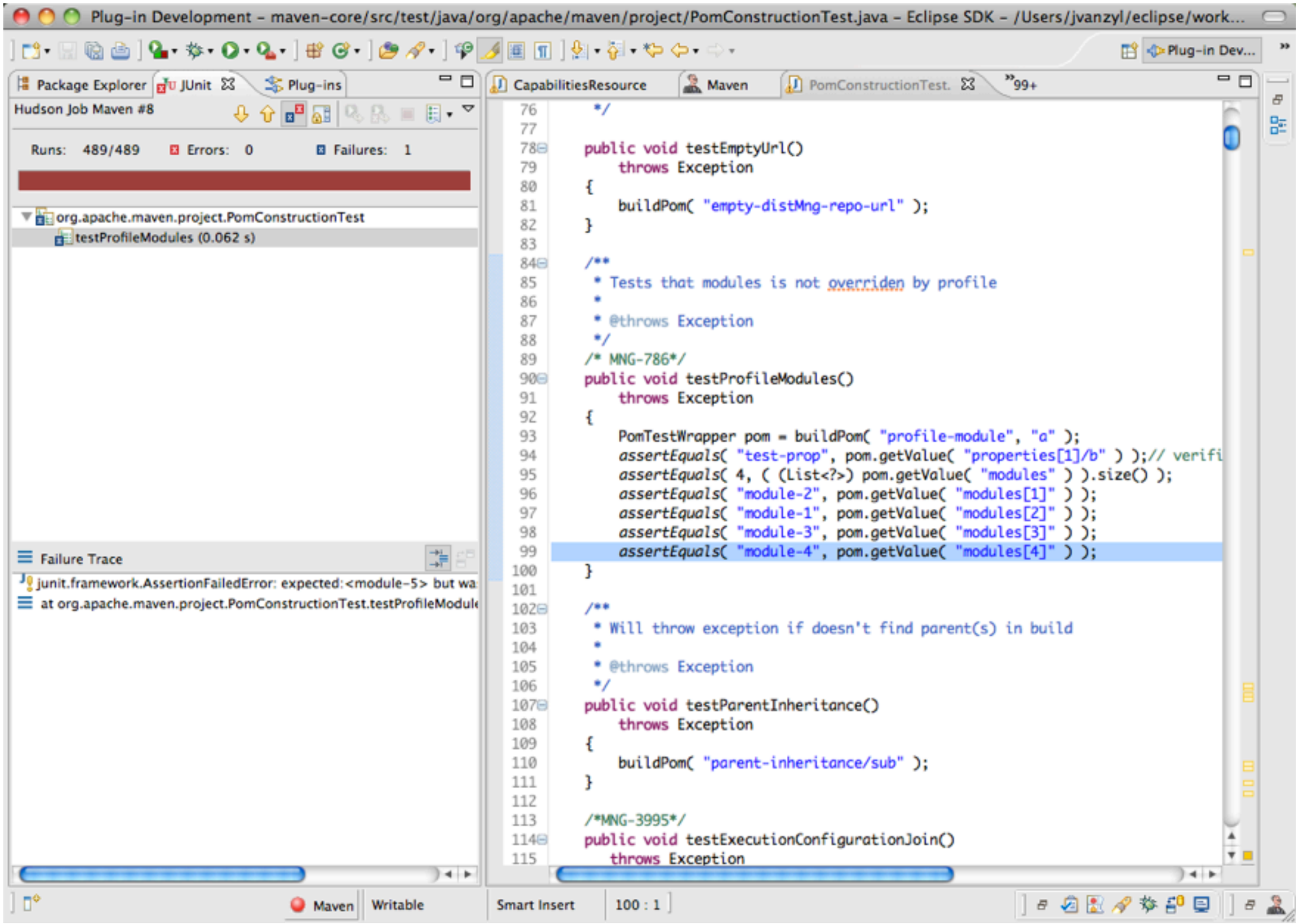
Hudson: Access to Build Jobs



Hudson: Access to Build Job Details



Hudson: Direct Navigation to Test Failures



Questions?