Maven Overview Presentation for ASPIRE Consortium

Author: Stephane Ribas

August 2008

Foreword and Credits

I would like to thank

- *OW@INRIA team,*
- *Vincent Massol,*
- *Mergere – Maven Team Apache,*

without whom this presentation would not have been possible.

Finally, I would like to thank OW2 for hosting this webinar.

Related links:
http://www.javapolis.com/
http://maven.apache.org/

Making your builds boring...

“Building projects should be easy and standardized. You should not be spending a substantial amount of your project time on builds. Builds should just work!”

Vincent Massol

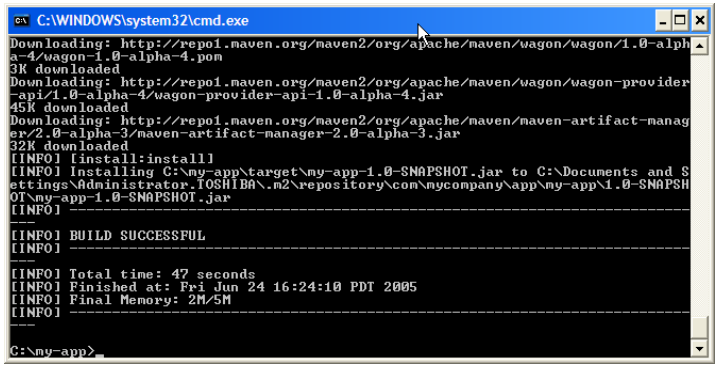
ow2
Europe Local Chapter

Agenda

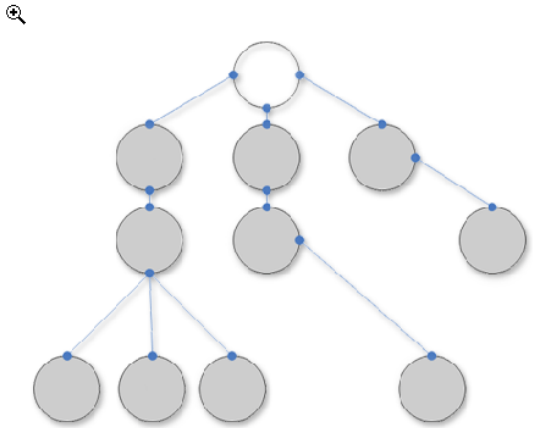
- What is Maven?
- Maven Architecture
- Build patterns
- Maven 2 plugins
- Exercises

What is Maven? (1/4)

A building tool!



A dependency management tool!



A documentation tool!

Europe Local Chapter

What is Maven? (2/4)

Maven is a process of applying patterns to a build infrastructure in order to provide a coherent view of software projects.

Objectives

- Make the development process visible or transparent
- Provide an easy way to see the health and status of a project
- Decreasing training time for new developers
- Bringing together the tools required in a uniform way
- Preventing inconsistent setups
- Providing a standard development infrastructure across projects
- Focus energy on writing applications

What is Maven? (3/4)

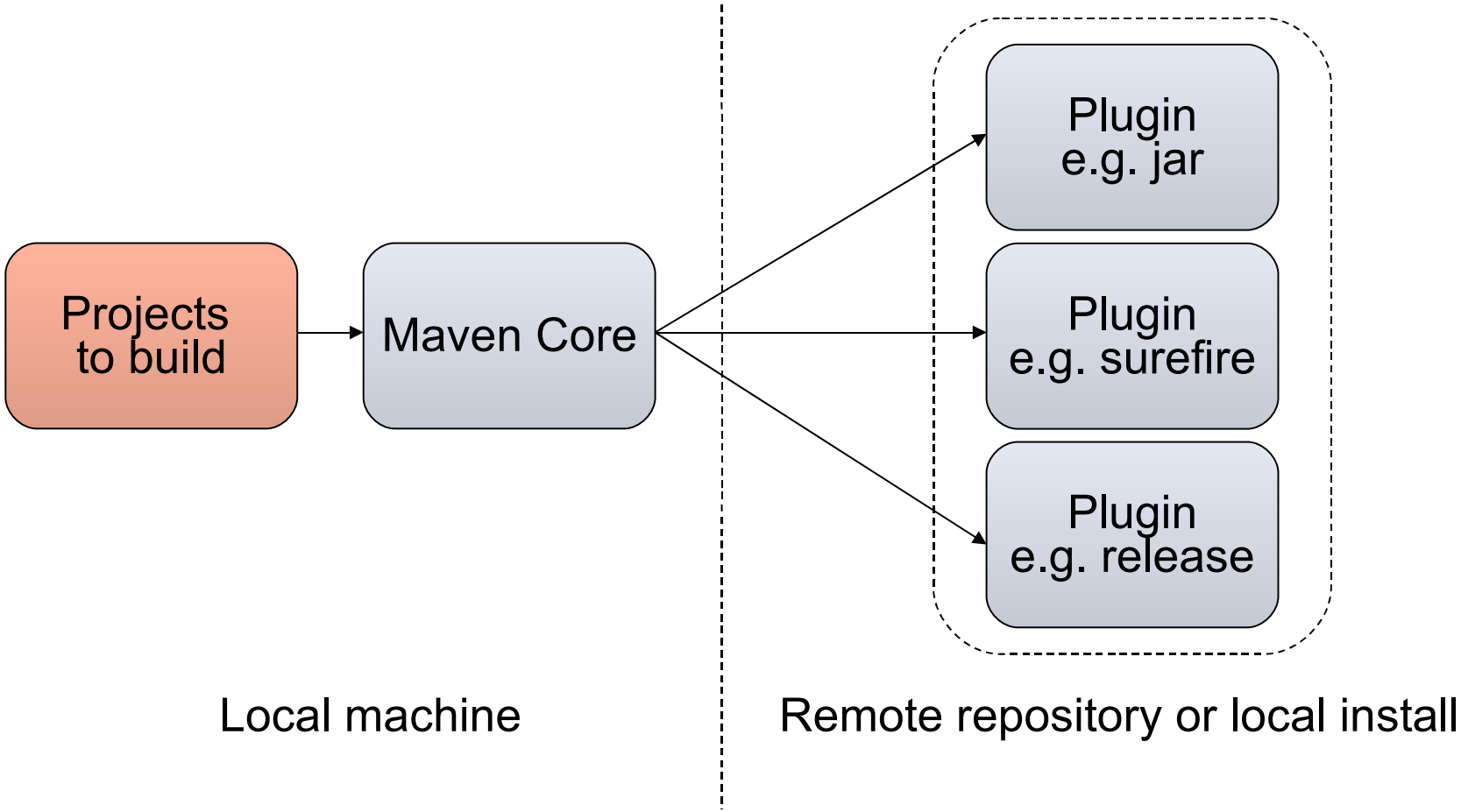
- **Is a complete rewrite from Maven 1.0/1.1**
  - Started parallel development in early 2003, well before Maven 1.0 final!
  - More consistent definition of all parts of the system
  - Architecture supports features and that the original couldn't
  - *Faster, lighter, smaller* - embeddable
  - Making it simpler to use required reworking many core concepts

What is Maven? (4/4)

Few Features Examples

- Enhanced dependency support
- Build life cycle
- Unified project file
- Enhanced plug-in support
- Multi-module project support
- Site and documentation enhancements
- Release management
- *Archetypes* - project templates
- Build Profiles

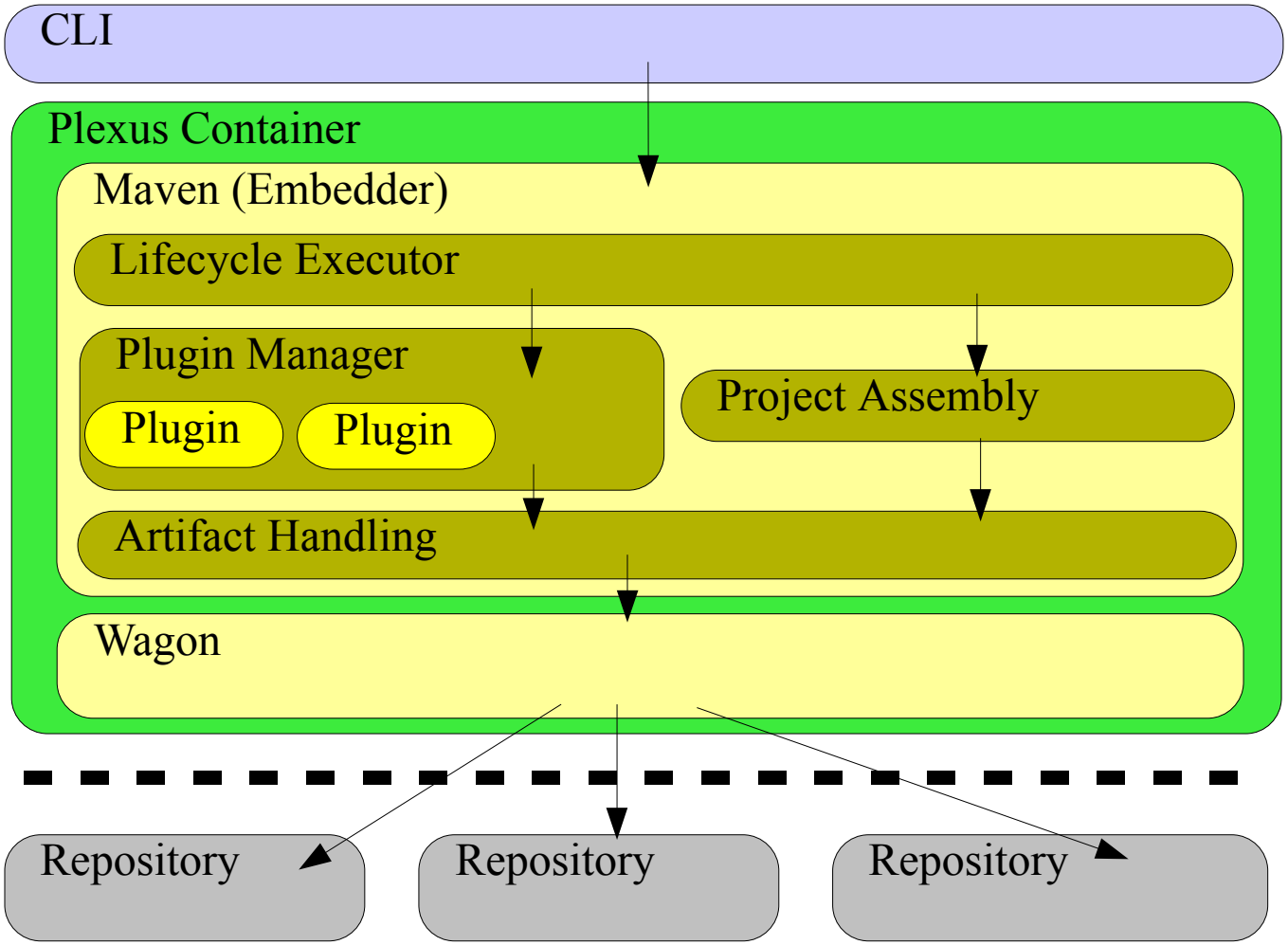
Maven Architecture



ow2 Europe Local Chapter

Maven Architecture

Want some more?



Europe Local Chapter

Build Patterns

Common project metadata format

- POM = Project Object Model = pom.xml
  - The project configuration file
- Contains metadata about the project
  - Location of directories, Developers/Contributors, Issue tracking system, Dependencies, Repositories to use, etc
- Example:

```
<project>
  <modelVersion>4.0.0</modelVersion>
  <groupId>org.codehaus.cargo</groupId>
  <artifactId>cargo-core-api-container</artifactId>
  <name>Cargo Core Container API</name>
  <version>0.7-SNAPSHOT</version>
  <packaging>jar</packaging>
  <dependencies/>
  <build/>
[…]
```

Minimal POM

Europe Local Chapter

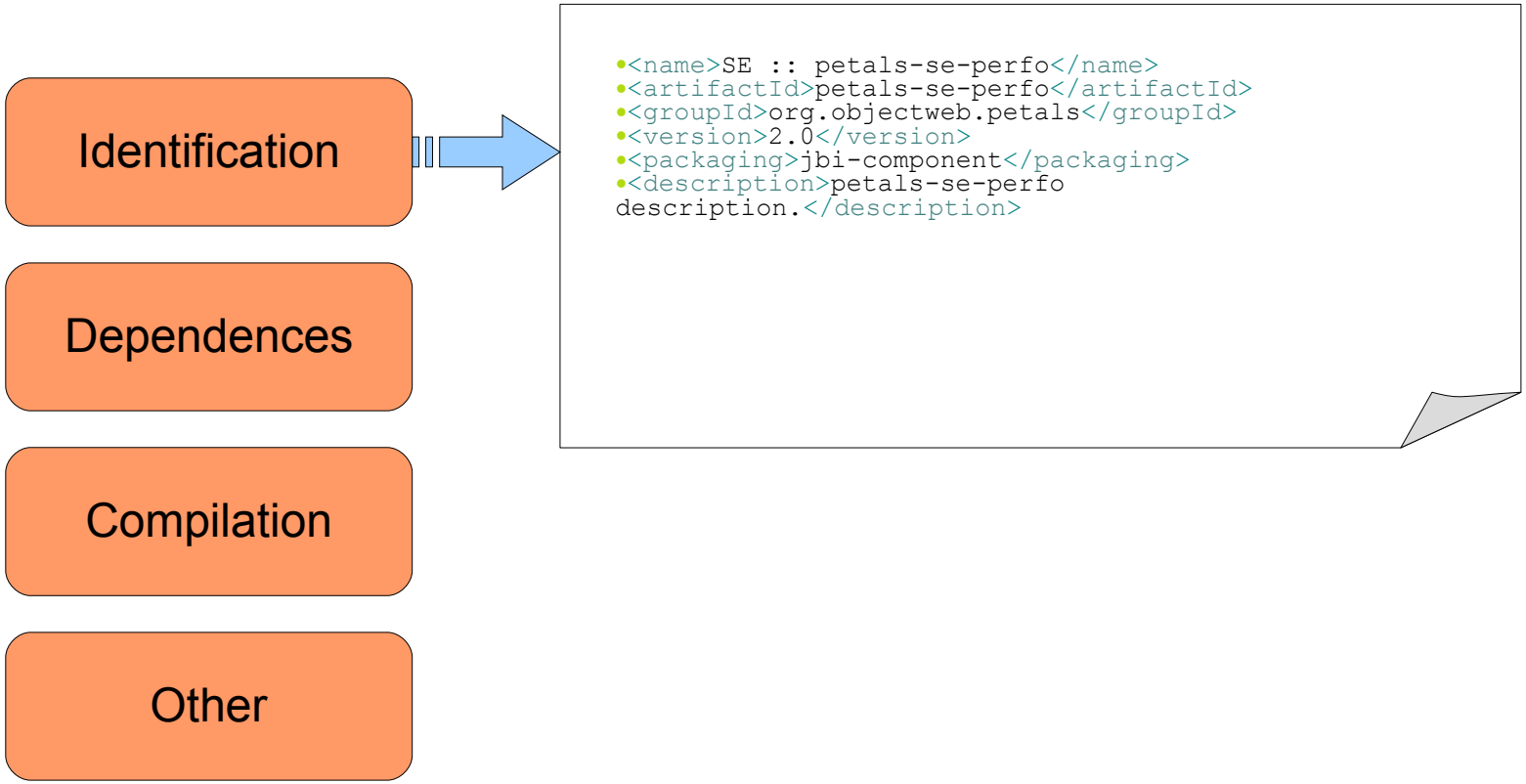
pom.xml

Structure

- Identification
- Dependences
- Compilation
- Other

pom.xml (1/4)

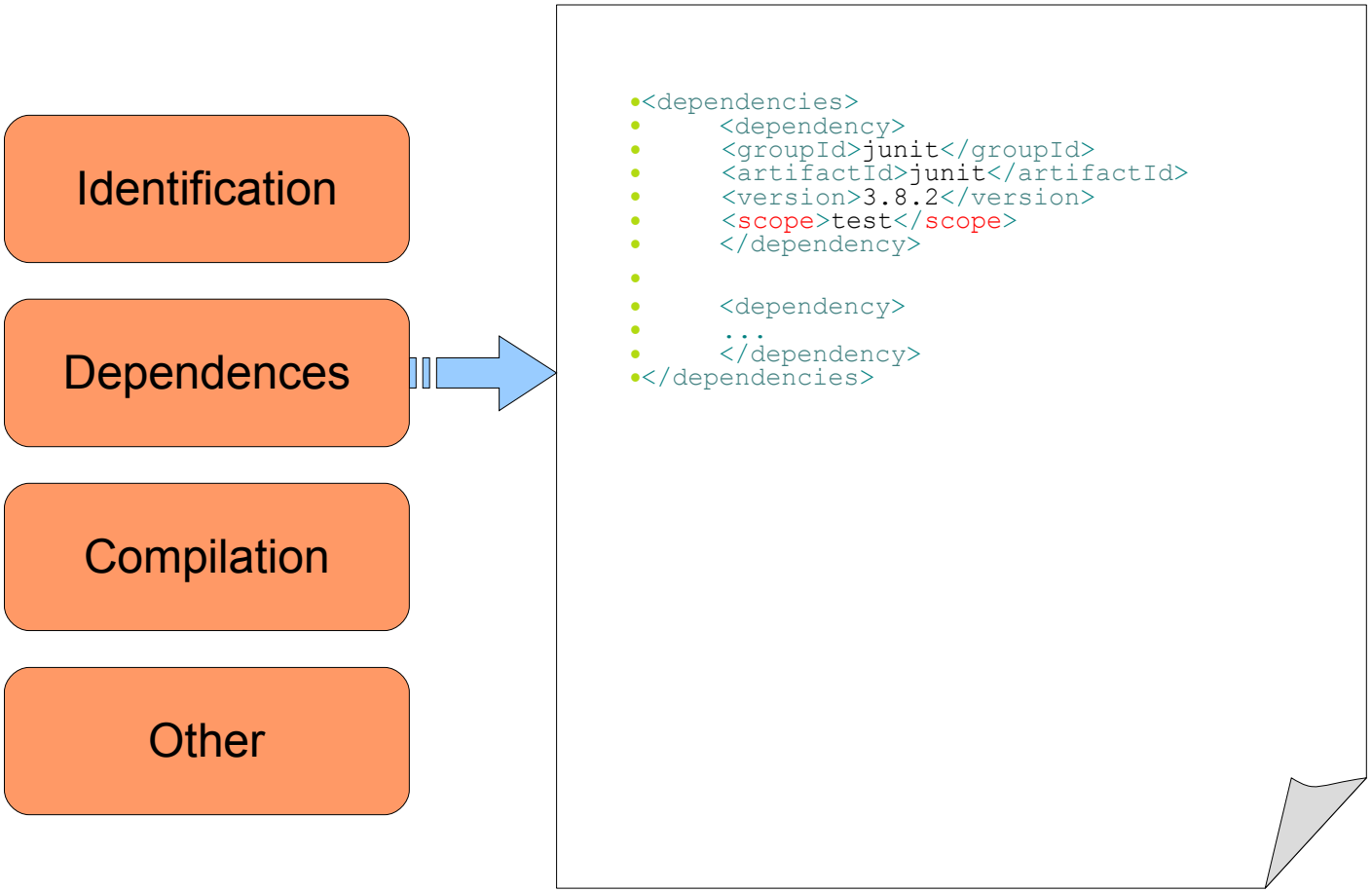
Examples



Europe Local Chapter

pom.xml (2/4)

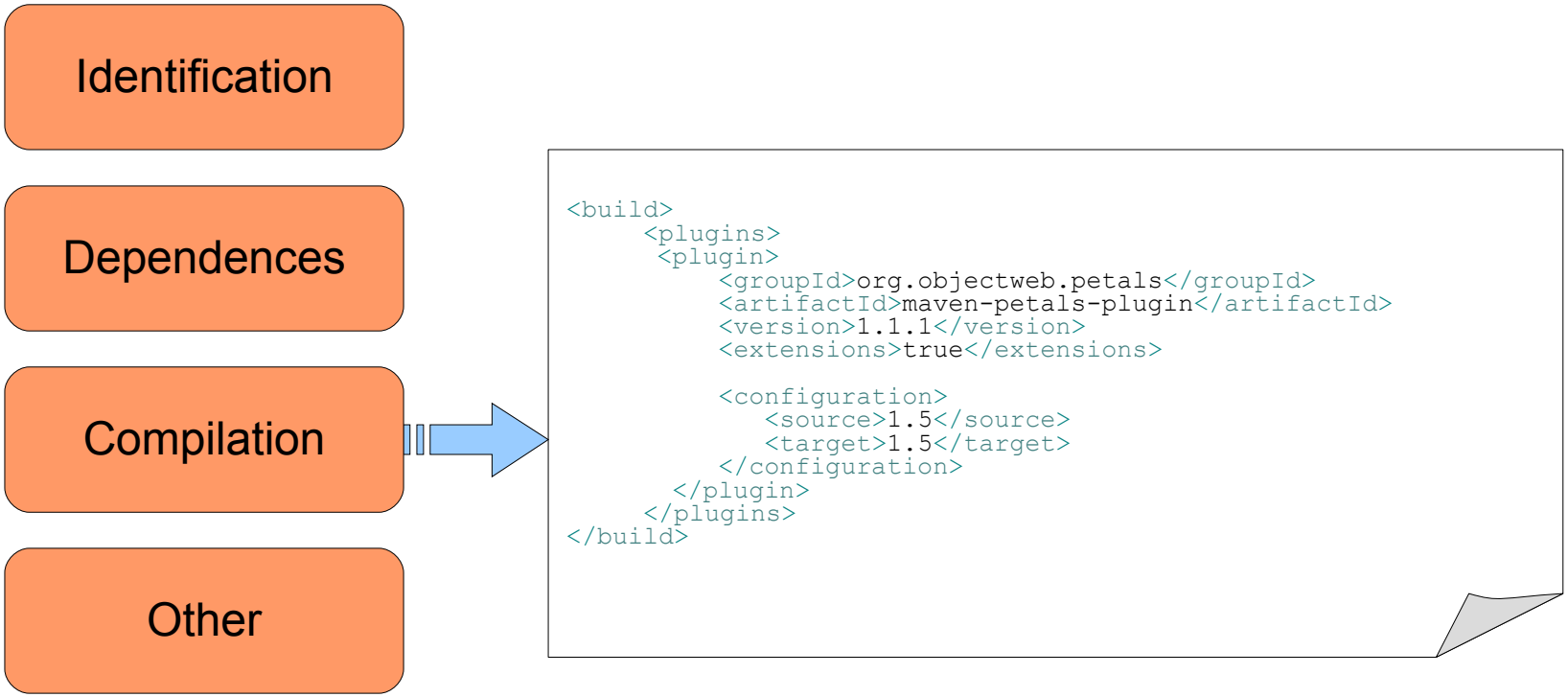
Examples



ow2 Europe Local Chapter

pom.xml (3/4)

Examples



Europe Local Chapter

pom.xml (4/4)

Examples

Identification

Dependences

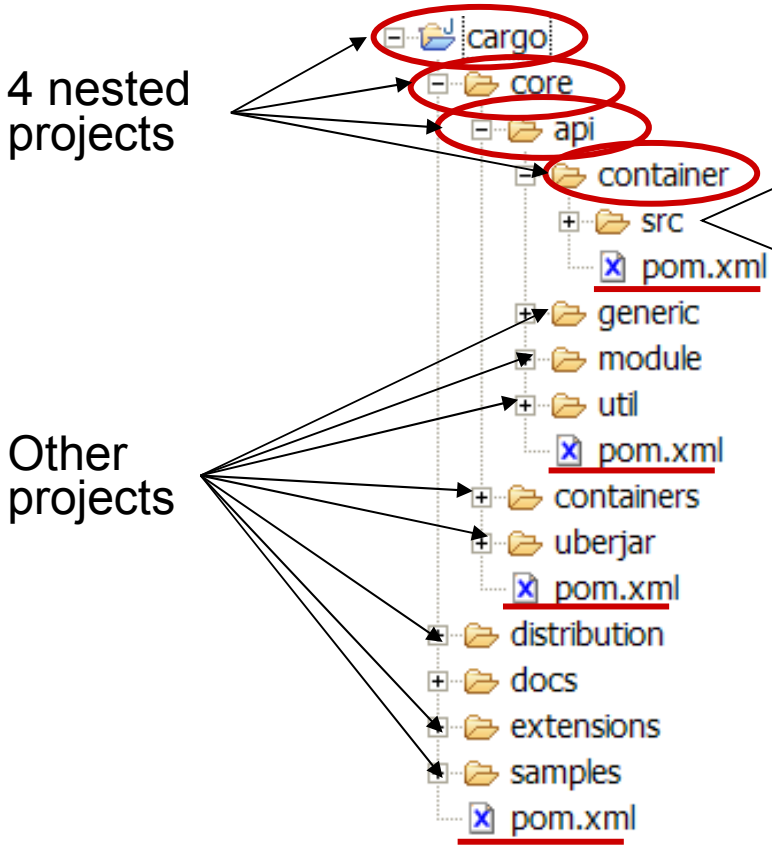
Compilation

Other

```
<repositories>
      <repository>
            <id>apache-snapshots</id>
            <name>Apache SNAPSHOT Repository</name>
            <url>
      http://people.apache.org/repo/m2-snapshot-repository/
            </url>
            <snapshots>
                  <enabled>true</enabled>
            </snapshots>
      </repository>
...
<repositories>
```

Build Patterns

Common directory organization



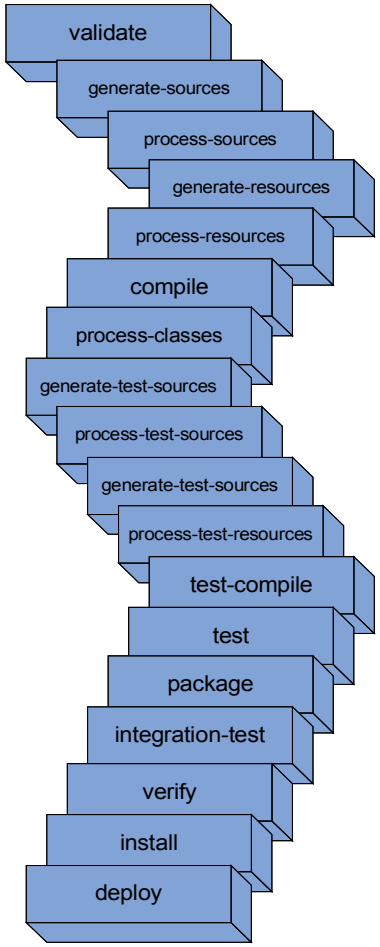
- src/
  - main/
    - java/
    - resources/
    - webapp/
    - application/
    - groovy/
  - test/
    - java/
    - resources/
    - cactus/
  - site/

OW2 Europe Local Chapter

Build Patterns

Common way to build applications (1/2)

- Builds in Maven *follow a pattern*
- Ensures developers moving between projects *do not need to learn new processes*



- Compile,
- Test,
- Package,
- Install,
- Deploy

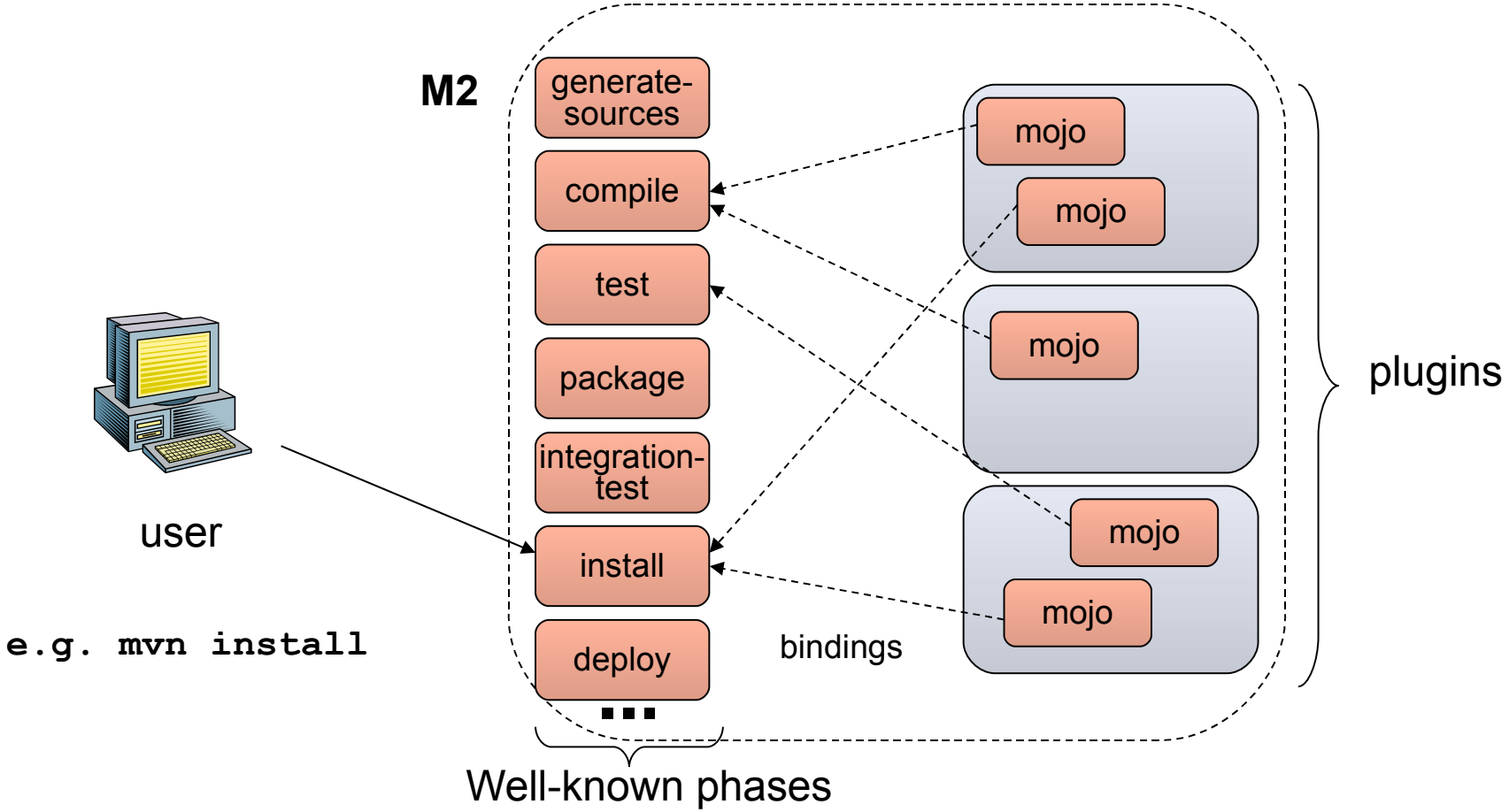
Project Life Cycle

Example of MVN Calls:
- mvn compile (compile)
- mvn install (compile, test, package, install)

ow2 Europe Local Chapter

Build Patterns

Common way to build applications (2/2)



ow2 Europe Local Chapter

Build Patterns

Environment-dependent builds (1/2)

- **Based on profiles**
  - Located in pom.xml, in profiles.xml or in settings.xml

```
<profiles>
  <profile>
    <id>tomcat5x</id>
    <activation>
      <activeByDefault>true</activeByDefault>
    </activation>
    <properties>
      <containerId>tomcat5x</containerId>
      <downloadUrl>…jakarta-tomcat-5.0.30.zip</downloadUrl>
    </properties>
  </profile>
  <profile>
    <id>orion2x</id>
    <properties>
      <containerId>orion2x</containerId>
      <downloadUrl>…orion2.0.5.zip</downloadUrl>
[…]
```

Profile that is always active

ow2 Europe Local Chapter

Build Patterns

Environment-dependent builds (2/2)

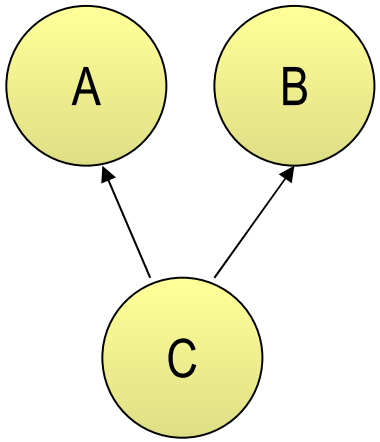
- **Different activation conditions**
  - JDK version, OS, property defined, existence of file or directory
- **Profiles can also modify plugin configurations and other POM elements**
  - Merged with the main pom.xml content
- **Profiles can be selected on the command line:**

```
mvn -P orion2x,resin3x install
```

Build Patterns

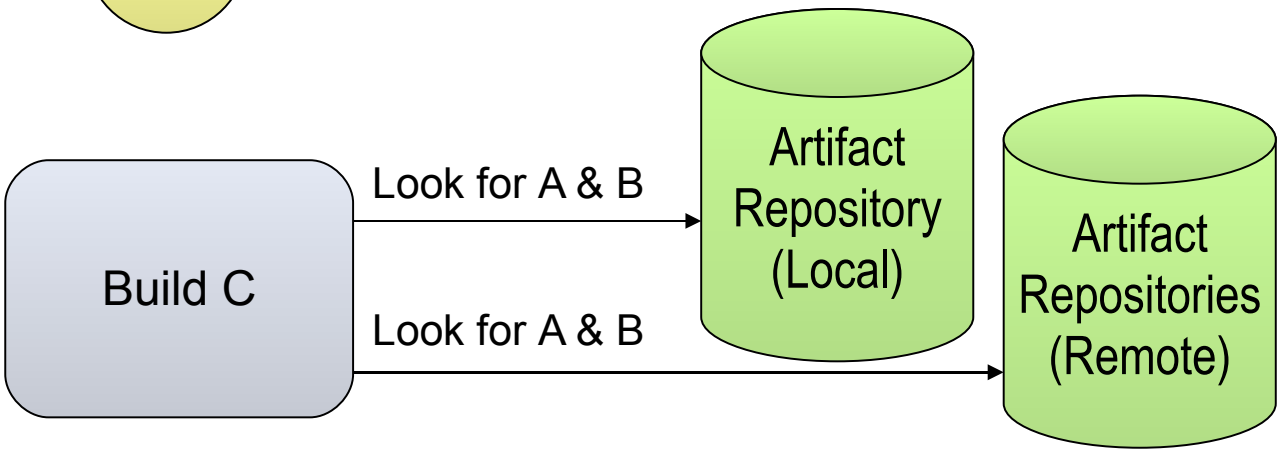
Dependency management (1/3)

- Maven uses binary dependencies




```
<dependencies>
  <dependency>
    <groupId>com.acme</groupId>
    <artifactId>B</artifactId>
    <version>[1.0,)</version>
    <scope>compile</scope>
  </dependency>
</dependencies>
```

« any version after 1.0 »



ow2 Europe Local Chapter

Build Patterns

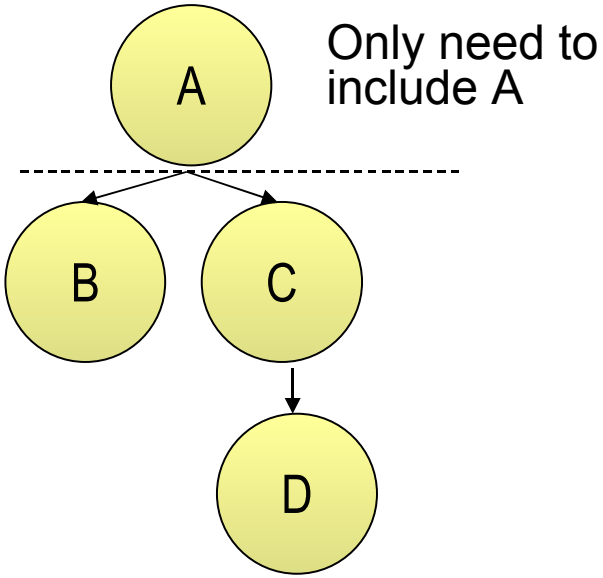
Dependency management (2/3)

- Declaration will *download* it, add it to the classpaths, *bundle* it into the resulting distribution if appropriate, etc.
- Main hurdle is non-redistributable artifacts – manual installation
- *transitive* – dependencies of dependencies

Build Patterns

Dependency management (3/3)

- **Transitive dependencies**
  - Possibility to exclude some deps
  - Need good metadata
  - Ideally projects should be split
- **SNAPSHOT handling**
  - Always get latest
- **Automatic dep updates**
  - By default every day



ow2 Europe Local Chapter

Build Patterns

Transitive Dependencies

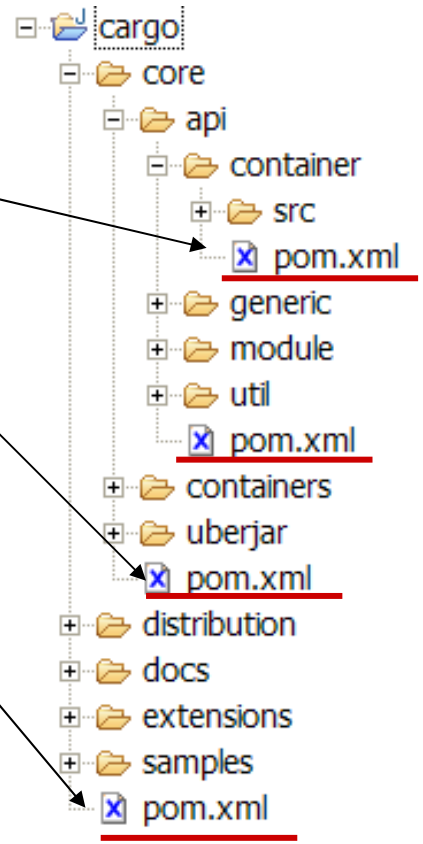
- **Always enabled in Maven 2.0**
- **Don't need to declare dependencies of dependencies yourself**
- **Frequently requested, but has more consequences than often realised...**
  - Version conflicts
  - Unwanted dependencies
  - Bad published meta data
  - Not a hard problem with good data

Build Patterns

Multi-module builds

- Integrated into Maven 2
- Run « `mvn` » at parent level
  - E.g. `mvn install` in `cargo/core/api`
  - E.g. `mvn install` in `cargo/core`
  - E.g. `mvn install` in `cargo/`
- Declare children projects in parents:

```
<modules>
  <module>core</module>
  <module>extensions</module>
  <module>samples</module>
</modules>
```

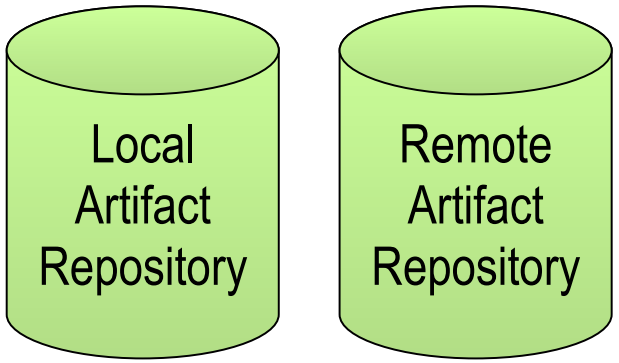


OW2 Europe Local Chapter

Build Patterns

Artifact repositories (1/2)

- **Used to store all kind of artifacts**
  - JARs, EARs, WARs, NBMs, EJBs, ZIPs, plugins, …
- **All project interactions go through the repository**
  - No more relative paths!
  - Easy to share between teams



e.g. http://ibiblio.org/maven2

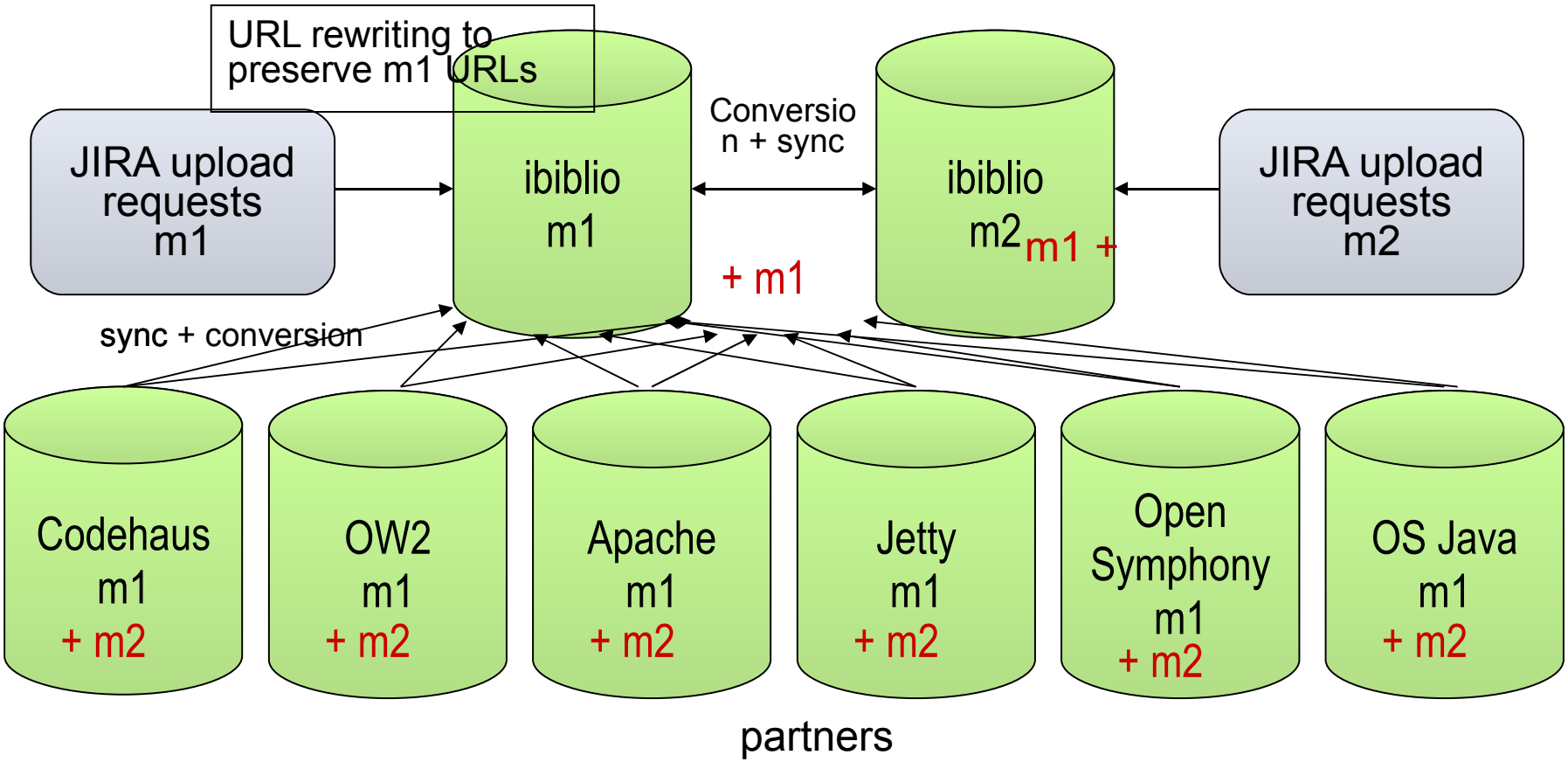
```
<repositories>
  <repository>
    <id>maven2-snapshot</id>
    <releases>
      <enabled>true</enabled>
    </releases>
    <name>Maven Central Development Repository</name>
    <url>http://snapshots.maven.codehaus.org/maven2</url>
    <layout>legacy|default</layout>
  </repository>
</repositories>
```


ow2 Europe Local Chapter

Build Patterns

Artifact repositories (2/2)

- Some public remote repositories



Europe Local Chapter

Build Patterns

Snapshot Handling

- Deploying to a shared repository gives a version with a *time stamp and build #*
- Don’t need to update dependency version to get updated builds
- Updates daily, on-demand, or at a particular interval
- Developers can get *access to co-workers changes earlier* without the need to update and build

Maven 2 Plugins (2/2)

- **Plugins are downloaded on demand**
  - First time they are used
- **Updates downloaded automatically**
  - Notification if newer plugin found

Site and Documentation

- **A *lot faster* than previously**
- **Accepts *several input formats***
  - Almost Plain Text (Wiki like)
  - Xdoc (Maven 1.0 compatible)
  - FAQ (Maven 1.0 compatible)
  - Docbook
- **Presently outputs XHTML, Xdoc, Docbook, Latex and RTF**
  - PDF?

Example APT Document

```
 ------
 Generating a Site
 ------
 Brett Porter
 ------
 13 May 2005
 ------

Building a Site

* Creating Content

  The first step to creating your site is to create some content. In
  Maven 2.0, the site content is separated by format, as there are several
  available.

-------------------
+- src/
   +- site/
      +- apt/
      |  +- index.apt
      +- site.xml
--------------------

  The Xdoc format is the same as
  {{{http://maven.apache.org/using/site.html} used in Maven 1.0}}.
  However, <<<navigation.xml>>> has been replaced by the site descriptor
  (see below).
```


Europe Local Chapter

Example APT Document

Apache Maven Project
http://maven.apache.org/

Maven

Last Published: Tue May 31 09:32:59 EST 2005

Apache | Maven 1.0 | Maven 2

Maven 2.0
- Introduction
- Download
- Release Notes
- General Information
- For Maven 1.0 Users
- Road Map

User's Guide
- Getting Started
- Configuration
- Dependency Mechanism
- Developing Plugins
- Developing Plugins with Marmalade
- Creating a Site

Reference
- Project Descriptor
- Settings Descriptor
- Available Plugins
- Mojo API
- Ant Tasks

Developers
- Documentation Needed

Building a Site

Creating Content

The first step to creating your site is to create some content. In Maven 2.0, the site content is separated by format, as there are several available.

```
+- src/
   +- site/
      +- apt/
      |  +- index.apt
      +- site.xml
```

The Xdoc format is the same as used in Maven 1.0. However, `navigation.xml` has been replaced by the site descriptor (see below).

ow2 Europe Local Chapter

Maven 2 Plugins (1/2)

- Antlr
- Ant
- AntRun
- AspectJ
- Assembly
- Assembly-report
- Cargo
- Castor
- Changelog
- Changes
- Commons-attributes
- Checkstyle
- Clean
- Clover
- Csharp
- Cobertura
- Compiler
- Deploy
- Ear
- Eclipse
- Ejb
- Ejb3
- Exec
- Groovy
- Help
- Hibernate2
- Idea
- Install
- Issue
- It
- Jalopy
- Jar
- Javacc
- Javadoc
- Javancss
- Jboss
- Jcoverage Jdepend
- Jdiff
- Jelly
- Jetty
- Jpox
- Jspc
- Jxr
- MAnt
- Native
- One
- Par
- Plugin
- Pmd
- Project-info-reports
- Rar
- Release
- Repository
- Resources
- Repository
- Sablecc
- Site
- Slimdog
- Source
- Surefire
- Surefire-report
- Taglist
- Tomcat
- Verifier
- Xslt
- War
- Wsdl2java
- Xdoclet
- Xmlbeans

Maven functions can be extended using plug ins

- Maven-antrun-plugin : can execute Ant scripts

Plug ins can extend also the Build Life cycle

Status:

docs.codehaus.org/display/MAVEN/Maven+Plugin+Matrix

Maven 2 Plugins (2/2)

- **Plugins are downloaded on demand**
  - First time they are used
- **Updates downloaded automatically**
  - Notification if newer plugin found

Resources

- **Maven site and lists**
  - http://maven.apache.org/
- **Maven Blogs**
  - http://www.mavenblogs.com/

Any Questions ?

Thanks for listening!

Need more information?

www.ow2.org

ow2

Europe Local Chapter