



Maven Overview Presentation for ASPIRE Consortium

Stephane Ribas

www.ow2.org

August 2008



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

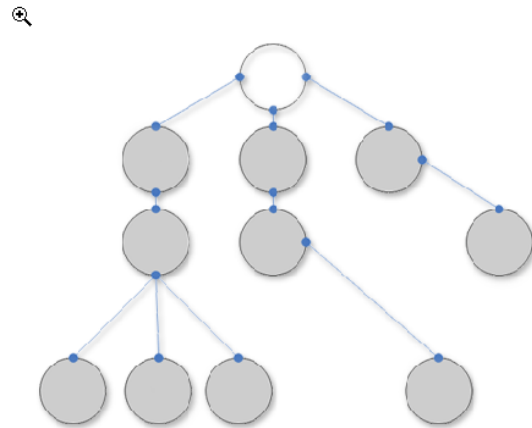
- ➔ **What is Maven?**
- ➔ **Maven Architecture**
- ➔ **Build patterns**
- ➔ **Maven 2 plugins**
- ➔ **Exercises**

What is Maven? (1/4)

A building tool!

```
C:\WINDOWS\system32\cmd.exe
Downloading: http://repo1.maven.org/maven2/org/apache/maven/wagon/wagon/1.0-alpha-4/wagon-1.0-alpha-4.pom
3K downloaded
Downloading: http://repo1.maven.org/maven2/org/apache/maven/wagon/wagon-provider-api/1.0-alpha-4/wagon-provider-api-1.0-alpha-4.jar
45K downloaded
Downloading: http://repo1.maven.org/maven2/org/apache/maven/maven-artifact-manager/2.0-alpha-3/maven-artifact-manager-2.0-alpha-3.jar
32K downloaded
[INFO] [install:install]
[INFO] Installing C:\my-app\target\my-app-1.0-SNAPSHOT.jar to C:\Documents and Settings\Administrator\TOSHIBA\m2\repository\com\mycompany\app\my-app\1.0-SNAPSHOT\my-app-1.0-SNAPSHOT.jar
[INFO] -----
[INFO] BUILD SUCCESSFUL
[INFO] -----
[INFO] Total time: 47 seconds
[INFO] Finished at: Fri Jun 24 16:24:10 PDT 2005
[INFO] Final Memory: 2M/5M
[INFO] -----
C:\my-app>
```

A dependency management tool!



A documentation tool!



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

➔ Is a complete rewrite from Maven 1.0/1.1

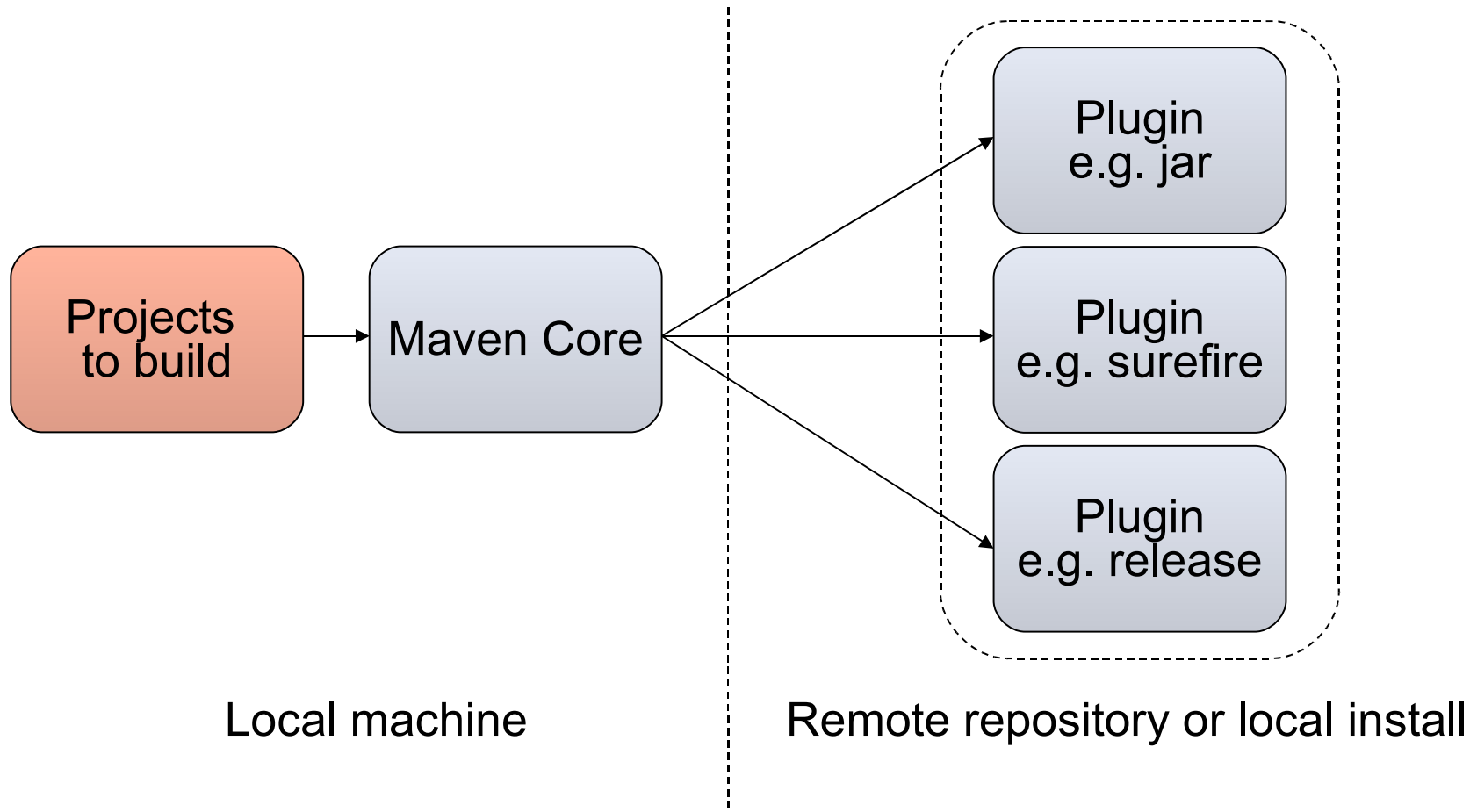
- Started parallel development in early 2003, well before Maven 1.0 final!
- More consistent definition of all parts of the system
- Architecture supports features and that the original couldn't
- *Faster, lighter, smaller* - embeddable
- Making it simpler to use required reworking many core concepts

What is Maven? (4/4)

Few Features Examples

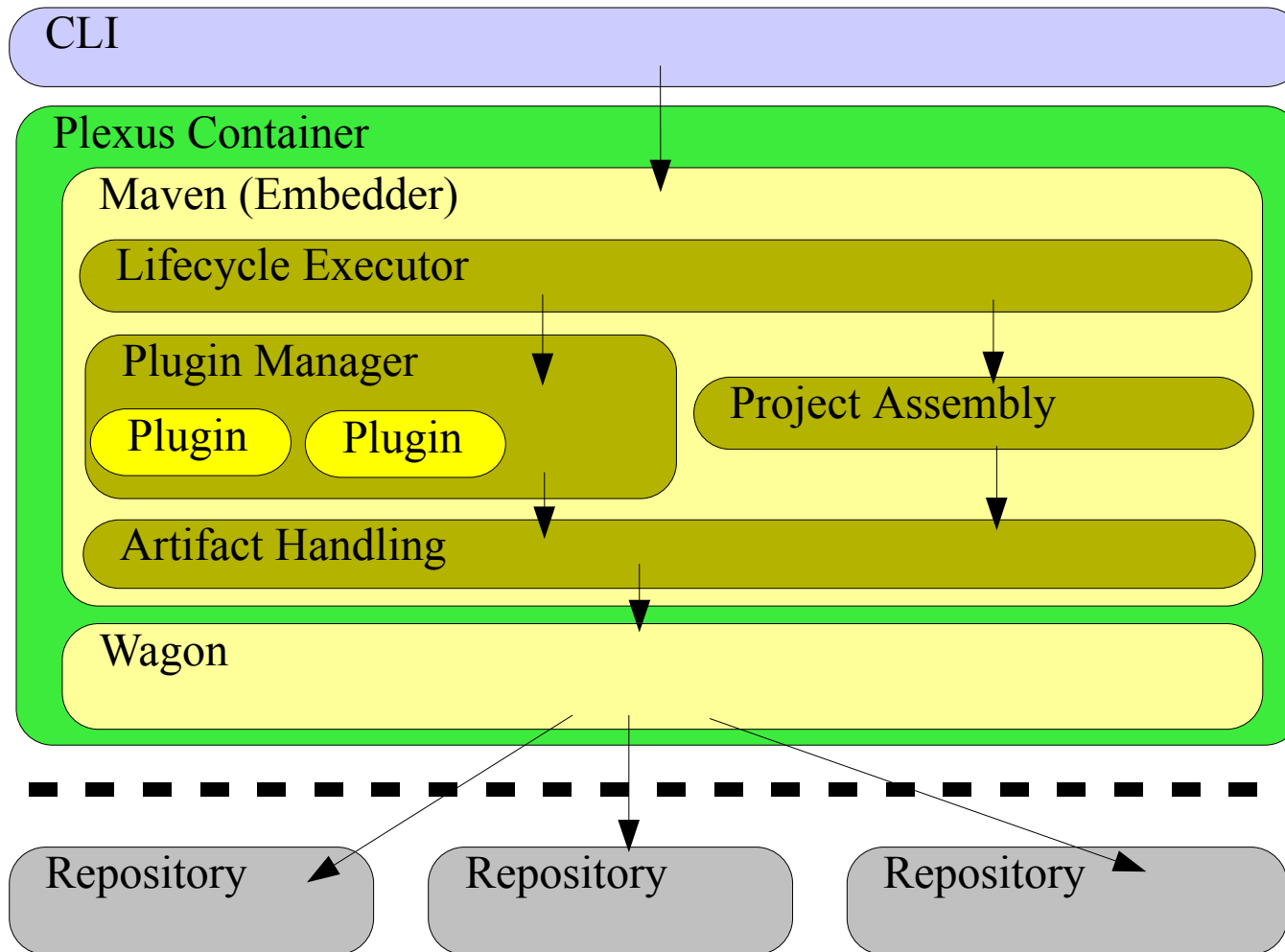
- ➔ **Enhanced dependency support**
- ➔ **Build life cycle**
- ➔ **Unified project file**
- ➔ **Enhanced plug-in support**
- ➔ **Multi-module project support**
- ➔ **Site and documentation enhancements**
- ➔ **Release management**
- ➔ ***Archetypes* - project templates**
- ➔ **Build Profiles**

Maven Architecture



Maven Architecture

Want some more?



Common project metadata format

- ➔ **POM = Project Object Model = pom.xml**
- ➔ **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

Structure

Identification

Dependencies

Compilation

Other

Examples

Identification

Dependencies

Compilation

Other

```
•<name>SE :: petals-se-perfo</name>  
•<artifactId>petals-se-perfo</artifactId>  
•<groupId>org.objectweb.petals</groupId>  
•<version>2.0</version>  
•<packaging>jbi-component</packaging>  
•<description>petals-se-perfo  
description.</description>
```

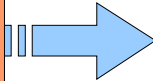
Examples

Identification

Dependences

Compilation

Other



```
•<dependencies>
•  <dependency>
•    <groupId>junit</groupId>
•    <artifactId>junit</artifactId>
•    <version>3.8.2</version>
•    <scope>test</scope>
•  </dependency>
•
•    <dependency>
•      ...
•    </dependency>
•</dependencies>
```

Examples

Identification

Dependencies

Compilation 

Other

```
<build>
  <plugins>
    <plugin>
      <groupId>org.objectweb.pets</groupId>
      <artifactId>maven-pets-plugin</artifactId>
      <version>1.1.1</version>
      <extensions>true</extensions>

      <configuration>
        <source>1.5</source>
        <target>1.5</target>
      </configuration>
    </plugin>
  </plugins>
</build>
```

Examples

Identification

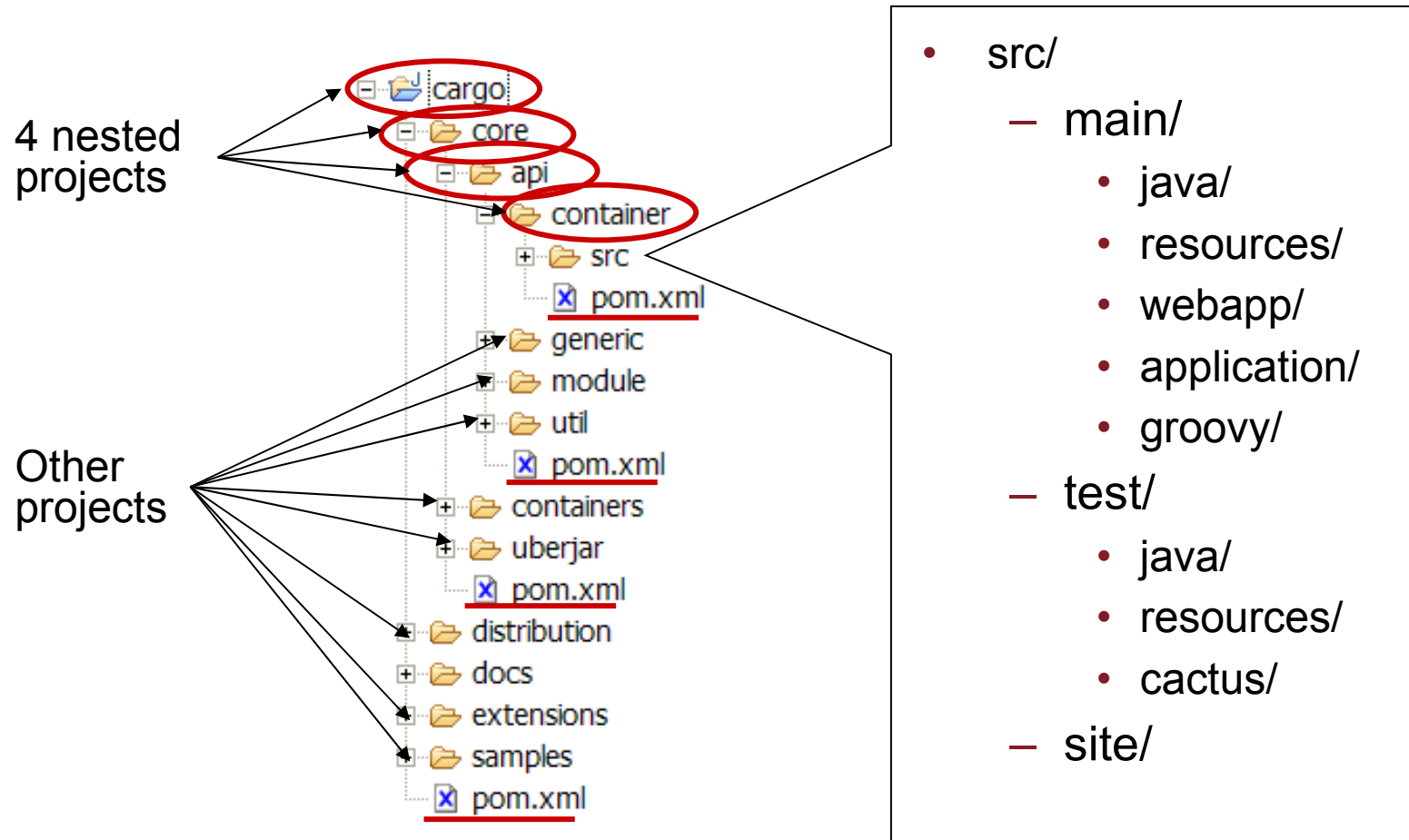
Dependencies

Compilation

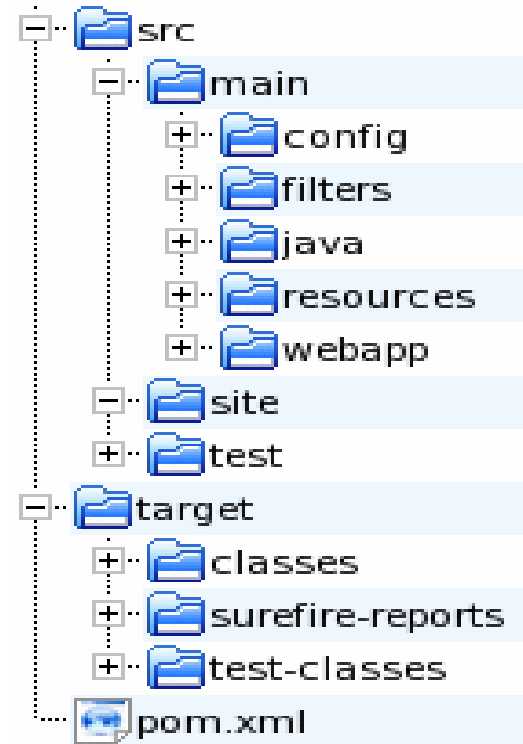
Other

```
<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>
```


Common directory organization



Common directory organization



POM.XML: The project configuration file

Common way to build applications (1/2)

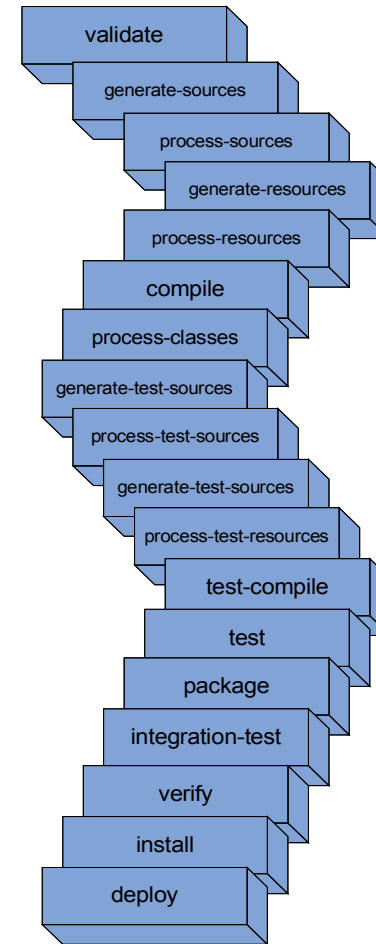
- Builds in Maven *follow a pattern*
- Ensures developers moving between projects *do not need to learn new processes*

➔ Compile,
➔ Test,
➔ Package,
➔ Install,
➔ Deploy

Project
Life
Cycle

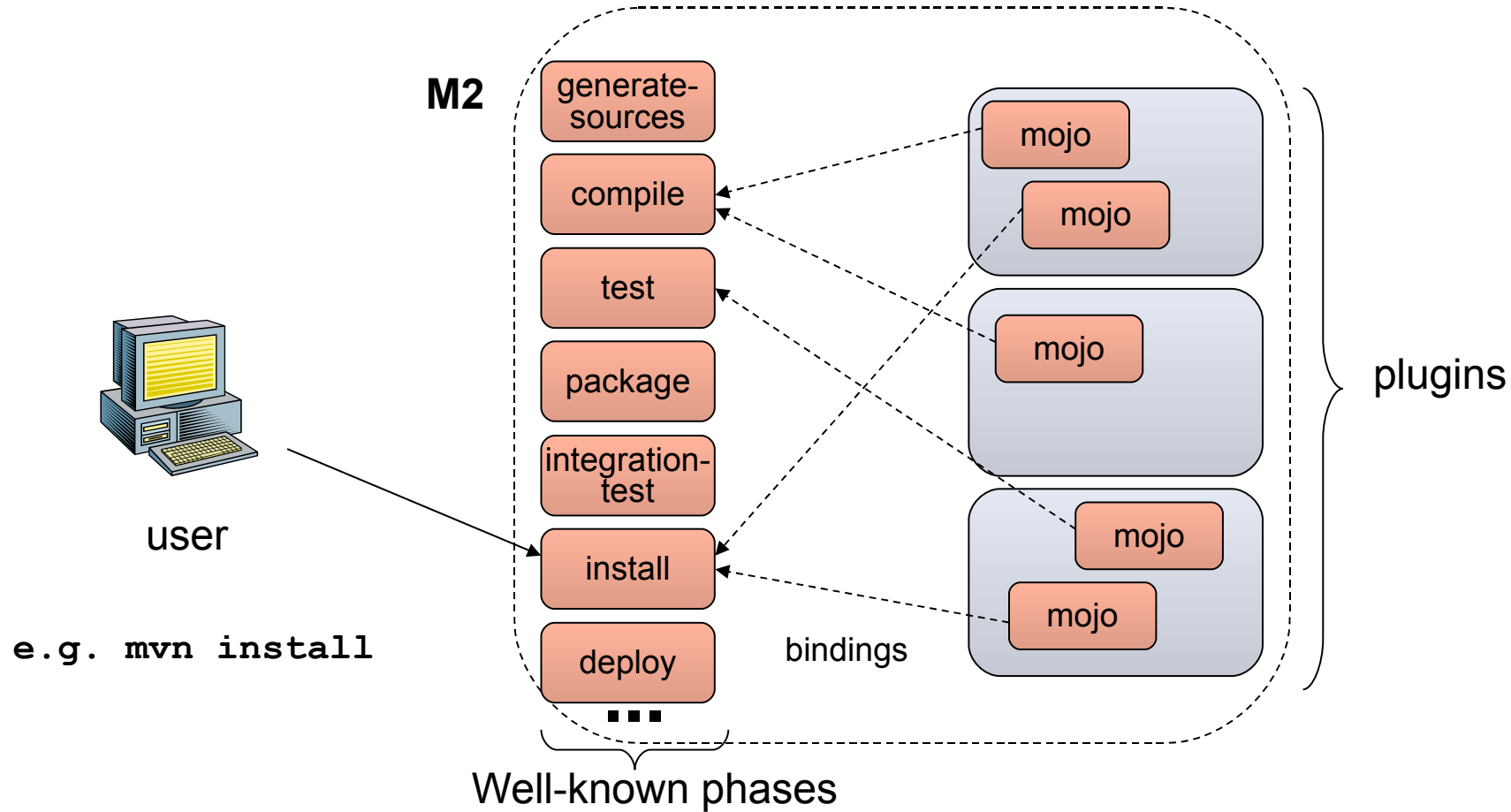
Example of MVN Calls:

- mvn compile (compile)
- mvn install (compile, test, package, install)



Build Patterns

Common way to build applications (2/2)



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>
    </properties>
  </profile>
  [...]
</profiles>
```

Profile that is always active

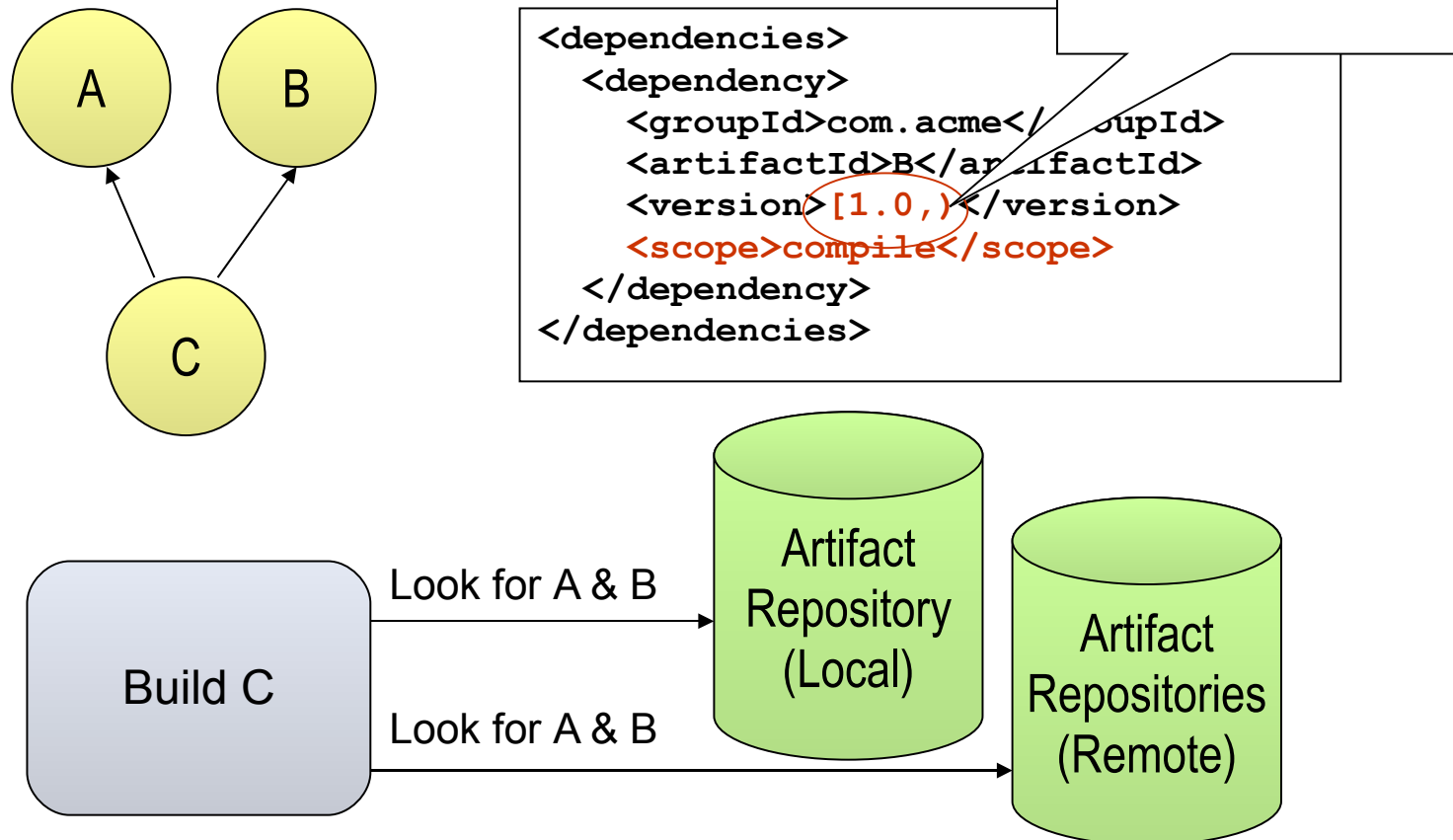
Environment-dependent builds (2/2)

- ➔ **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
```

Dependency management (1/3)

➔ Maven uses binary dependencies



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

Dependency management (3/3)

➔ Transitive dependencies

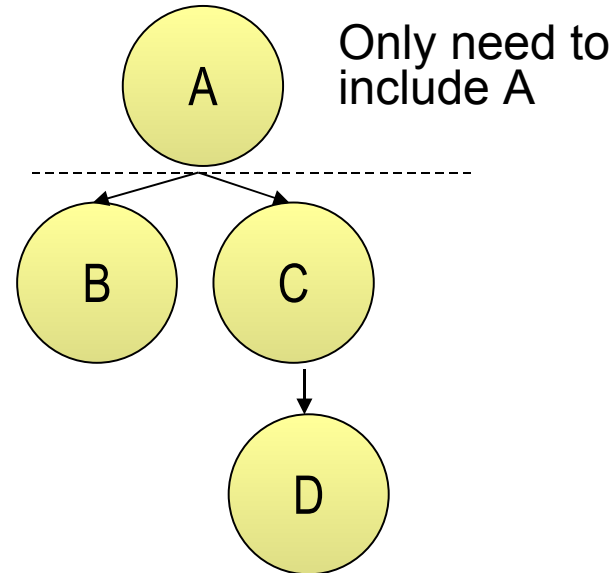
- Possibility to exclude some deps
- Need good metadata
- Ideally projects should be split

➔ SNAPSHOT handling

- Always get latest

➔ Automatic dep updates

- By default every day



Transitive Dependencies

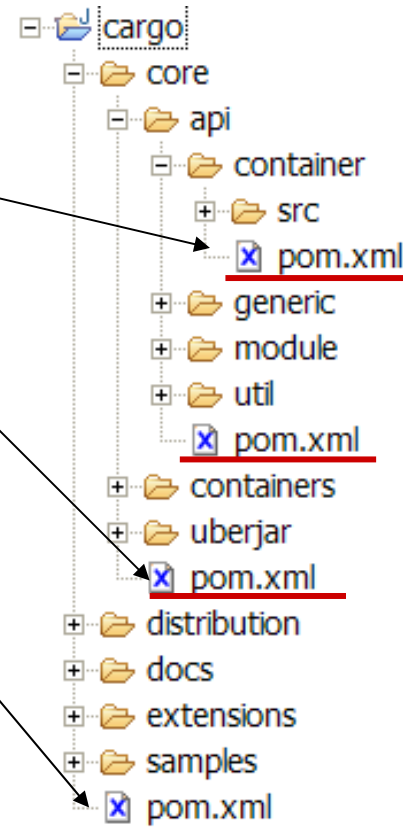
- ➔ Always enabled in Maven 2.0
- ➔ Don't need to declare dependencies of dependencies yourself
- ➔ Frequently requested, but has more consequences than often realised...
 - Version conflicts
 - Unwanted dependencies
 - Bad published meta data
 - Not a hard problem with good data

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/`

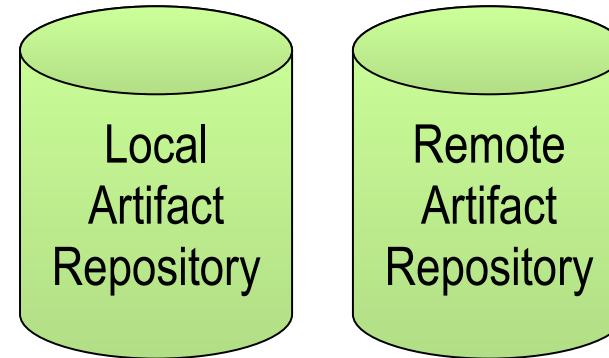
- ➔ Declare children projects in parents:

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



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

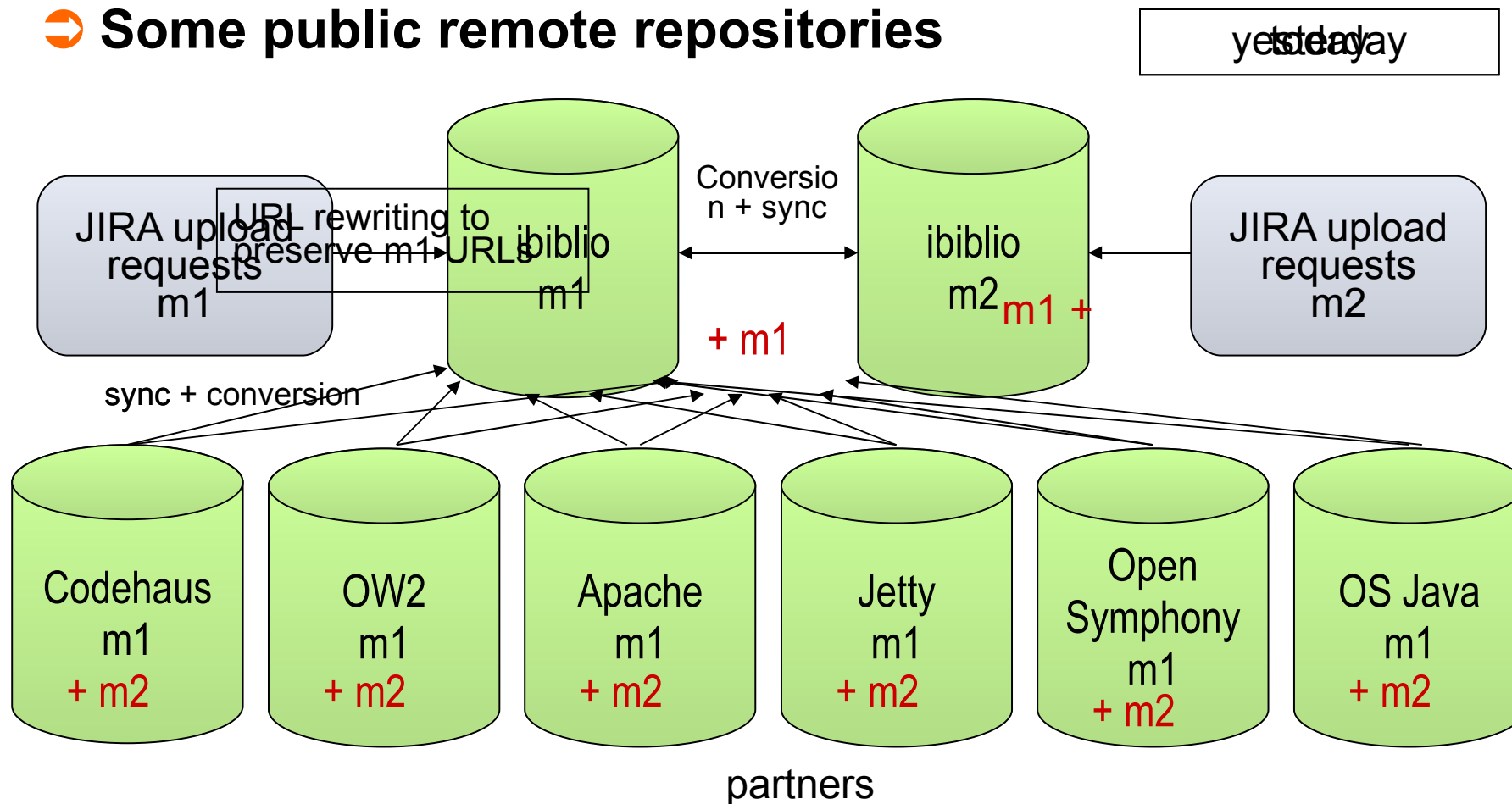


e.g. <http://ibiblio.org/maven2>

```
<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>
```

Artifact repositories (2/2)

➔ Some public remote repositories



- ➔ **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**

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

- ➔ **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).



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).

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
- Wsd12java
- 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

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

- ➔ **Exercises will be sent to you in a separate email message**

➔ Maven site and lists

- <http://maven.apache.org/>

➔ Maven Blogs

- <http://www.mavenblogs.com/>

Any Questions ?

Thanks for listening!

www.ow2.org

For more informations
Please contact
Cedric Thomas (CEO)
cedric.thomas@ow2.org