maven

Menos mal que has venido

6 de noviembre de 2006
Escuela Técnica Superior de Ingeniería Informática
Universidad de Sevilla

PRESENTACIÓN

manuelrecena.com

recena@eii.us.es

Índice

- Objetivos
- Escenarios comunes en el desarrollo software
- Localización de posibles factores de riesgo en estos escenarios
- ¿Qué es Maven? ¿Qué nos proporciona?
- Algunos conceptos
- La principal idea en Maven: P.O.M.
- El verdadero valor: plugins

Índice

- ¿Cómo encaja Maven en estos escenarios?
- Construcción de un ejemplo sencillo y práctico: `HelloMaven`
- Conclusiones
- Referencias
- Preguntas

OBJETIVOS

OBJETIVOS

- Dar respuesta a preguntas como: ¿Qué es Maven? ¿Para qué sirve? ¿Me puede interesar? ¿Esto se usa?
- Plantear una opción más de cómo hacer las cosas
- Dar a conocer qué nos podemos encontrar en el sector público y privado

ESCENARIOS COMUNES EN EL DESARROLLO SOFTWARE

Posibles escenarios

- Empresa que desarrolla y mantiene una linea de productos software.
  - Incorporación de nuevos recursos
  - Soporte de los productos en producción
  - Configuraciones muy heterogéneas entre los entornos de producción de sus clientes
  - Desarrollos internos que se comparten entre sus productos software

Posibles escenarios

- Empresa de servicios que ofrece soluciones informáticas:
  - Desarrollan en entornos de desarrollos propios y no suelen tener acceso a los entornos en explotación
  - Varios proyectos pueden compartir desarrollos internos
  - El cliente realizará tareas de mantenimiento sobre el producto que se le desarrolla

LOCALIZACIÓN DE POSIBLES FACTORES DE RIESGO EN ESTOS ESCENARIOS

Factores de riesgo

- Definición de entornos de desarrollo, preproducción y producción
- Curva de aprendizaje de los nuevos recursos
- Dependencia de recursos concretos
- Agilidad en los procesos de distribución
- Seguimiento y control de incidencias sobre los productos
- Calidad en la documentación
- Reutilización de los desarrollos internos
- Complejidad en el mantenimiento

¿QUÉ ES MAVEN?
¿QUE NOS PROPORCIONA?

¿Qué es Maven?

- Project Management Framework
  - Propociona una estructura de soporte para ser extendido
- Herramienta para gestionar y describir proyectos software
- Concebida inicialmente para trabajar con JAVA
- ¿La evolución de Ant? ¿Un complemento para Ant?
- Licencia: Apache License 2.0

¿Recordáis los “antiguos” makefiles?

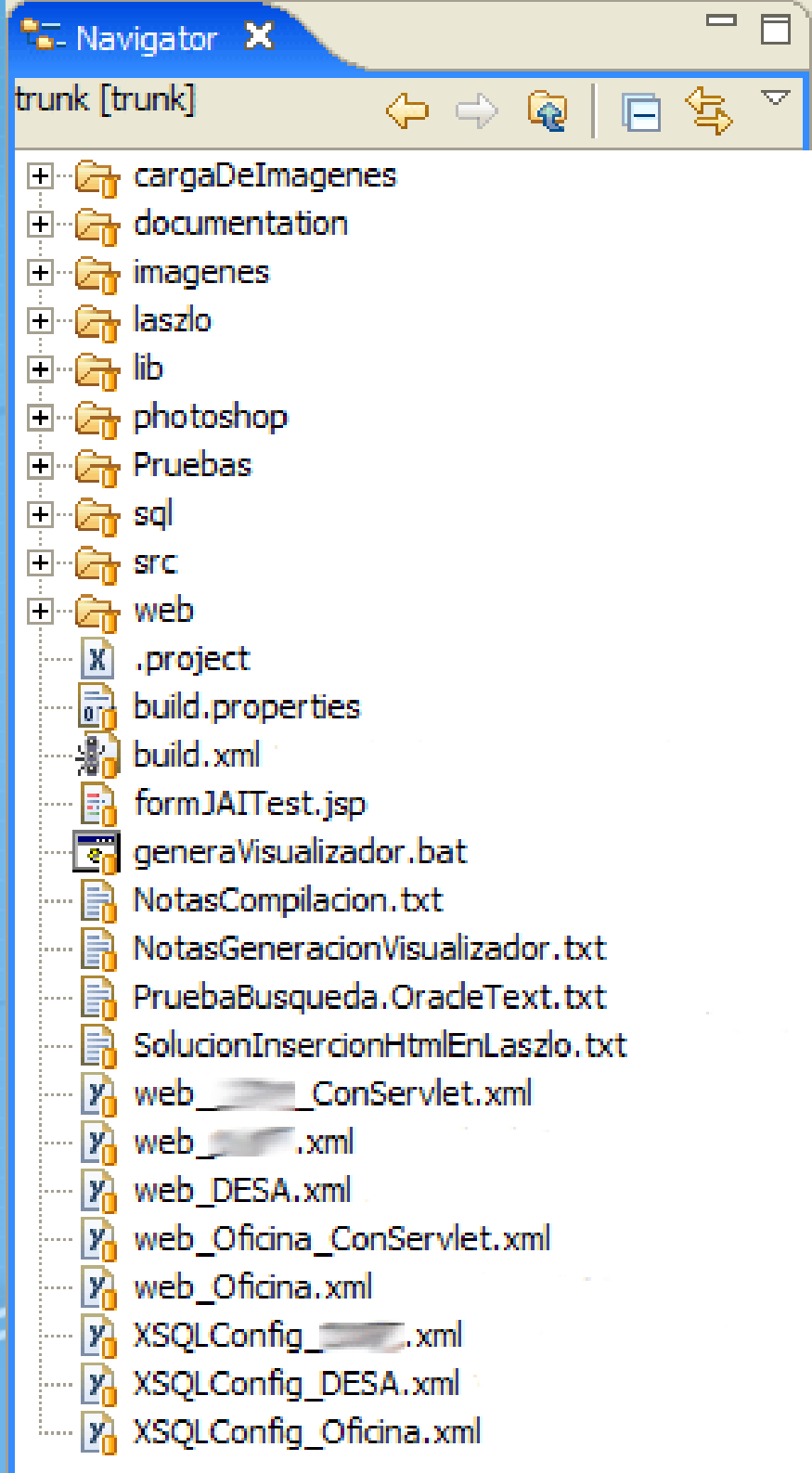
¿Qué nos proporciona?

- Un modelo estandar para gestionar y describir proyectos
- Tras la instalación, dispondremos de funcionalidades que nos facilitarán tareas a distintos niveles
  - Inicialización de proyectos, integración con IDEs, configuración de proyectos (JDKs, dependencias, ...)
- Procedimientos por defecto para la realización de las tareas base
  - Compilación, pruebas unitarias, empaquetado,...
- Simplifica y unifica los procesos de distribución, mantenimiento de la documentación, instalación, ...

¿Qué nos proporciona?

- Estructuras comprensibles que eviten cosas como...

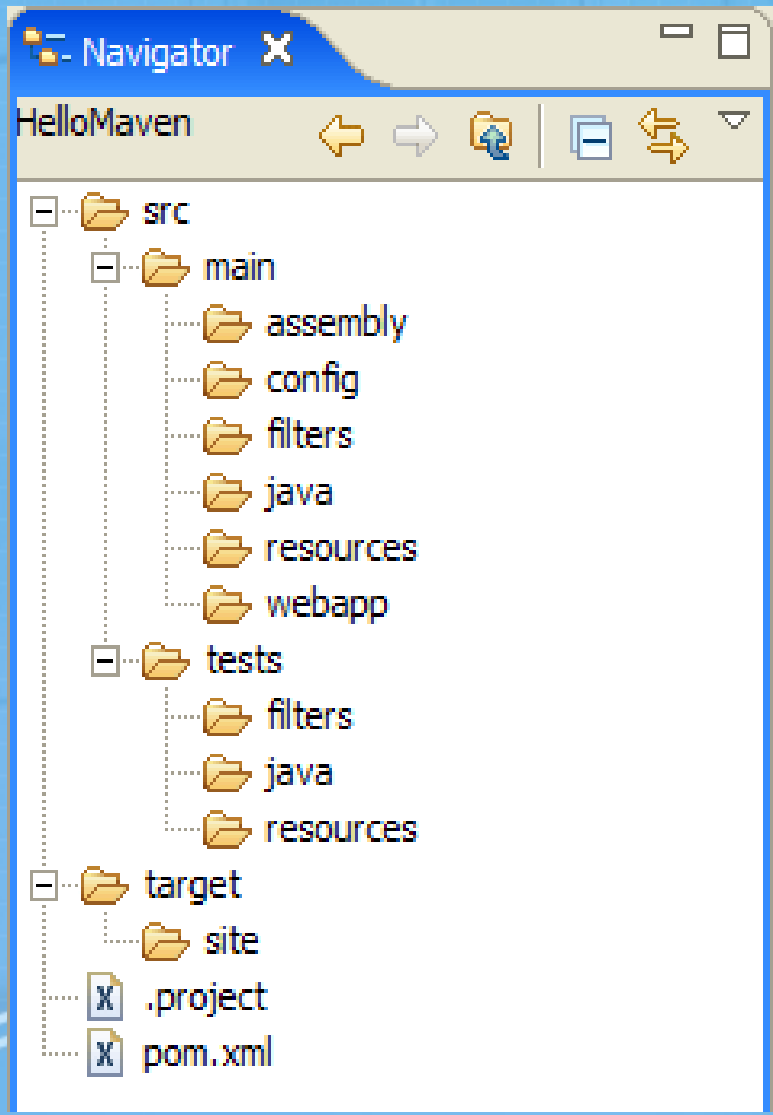
Señor Cuesta, un poquito de porfavo', que aquí no hay quien trabaje!



ALGUNOS CONCEPTOS

Algunos conceptos

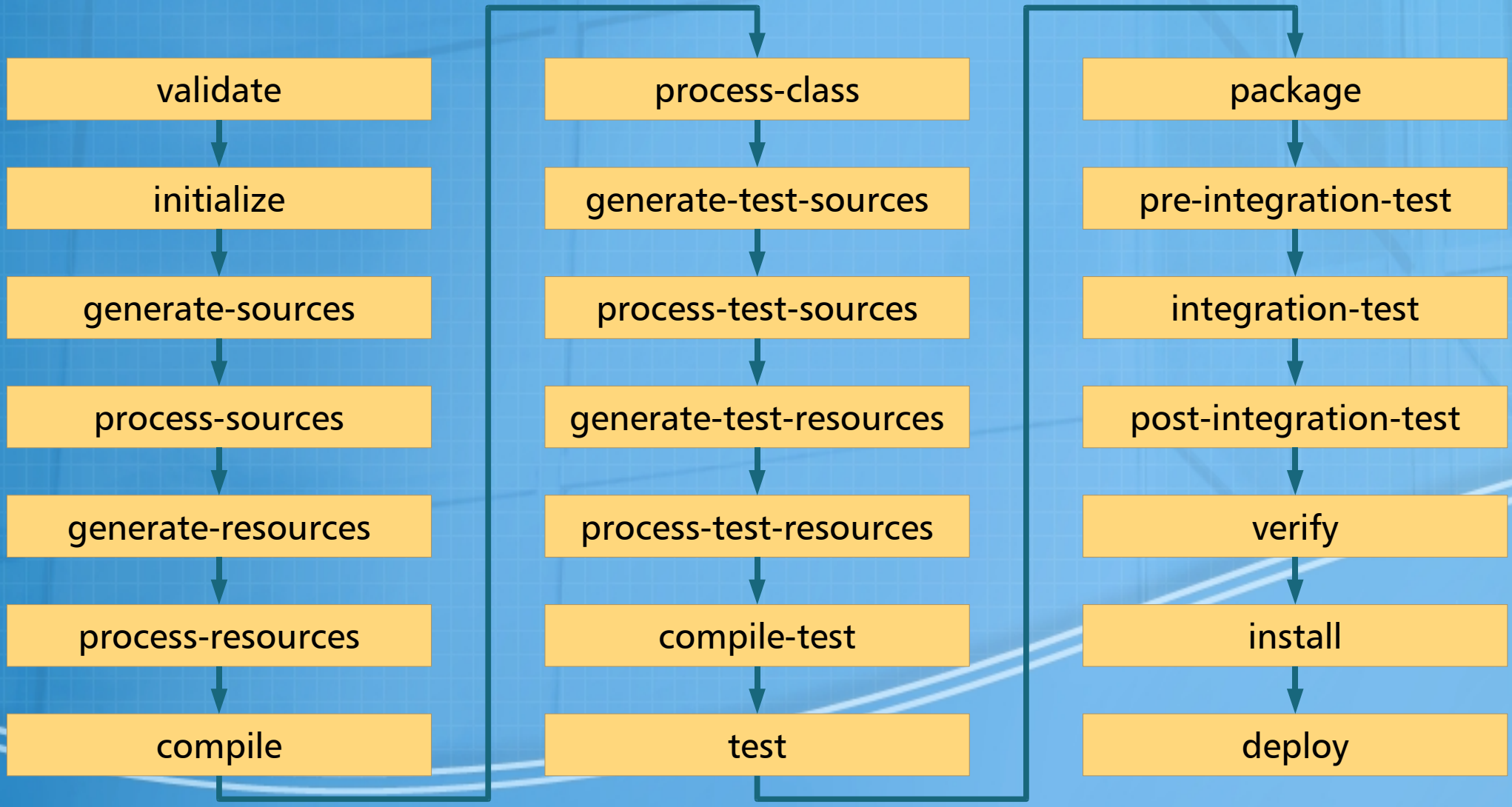
- Estructura de directorios predefinida
  - Flexibilidad
  - Estandarización
  - Fácil comprensión
  - Facilita las operaciones con los S.C.M.



S.C.M: Source Code Manager

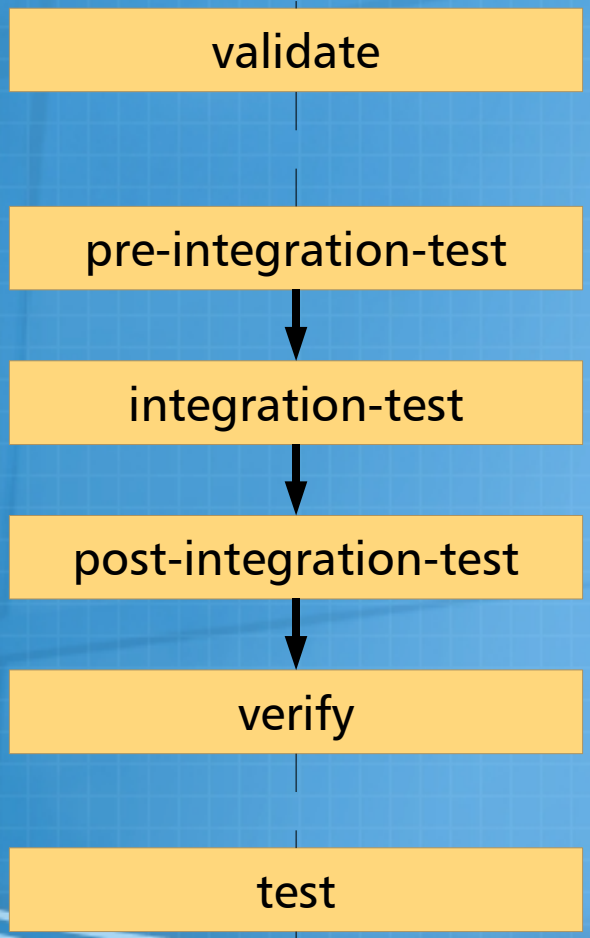
Algunos conceptos

- Ciclo de vida asociado al proceso de construcción:



Algunos conceptos

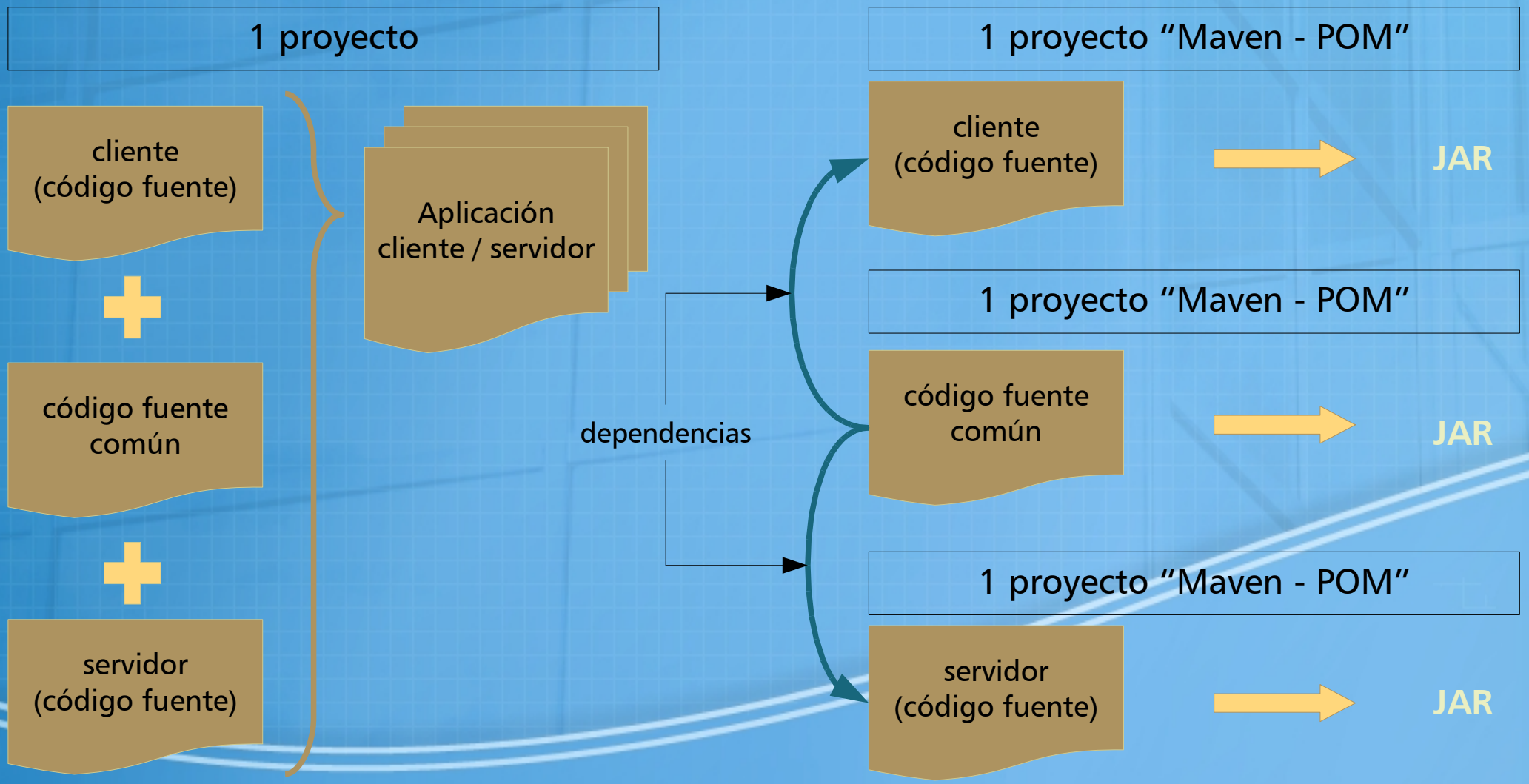
- Ciclo de vida asociado al proceso de construcción:



Estados directamente relacionados con la calidad (Quality Assurance)

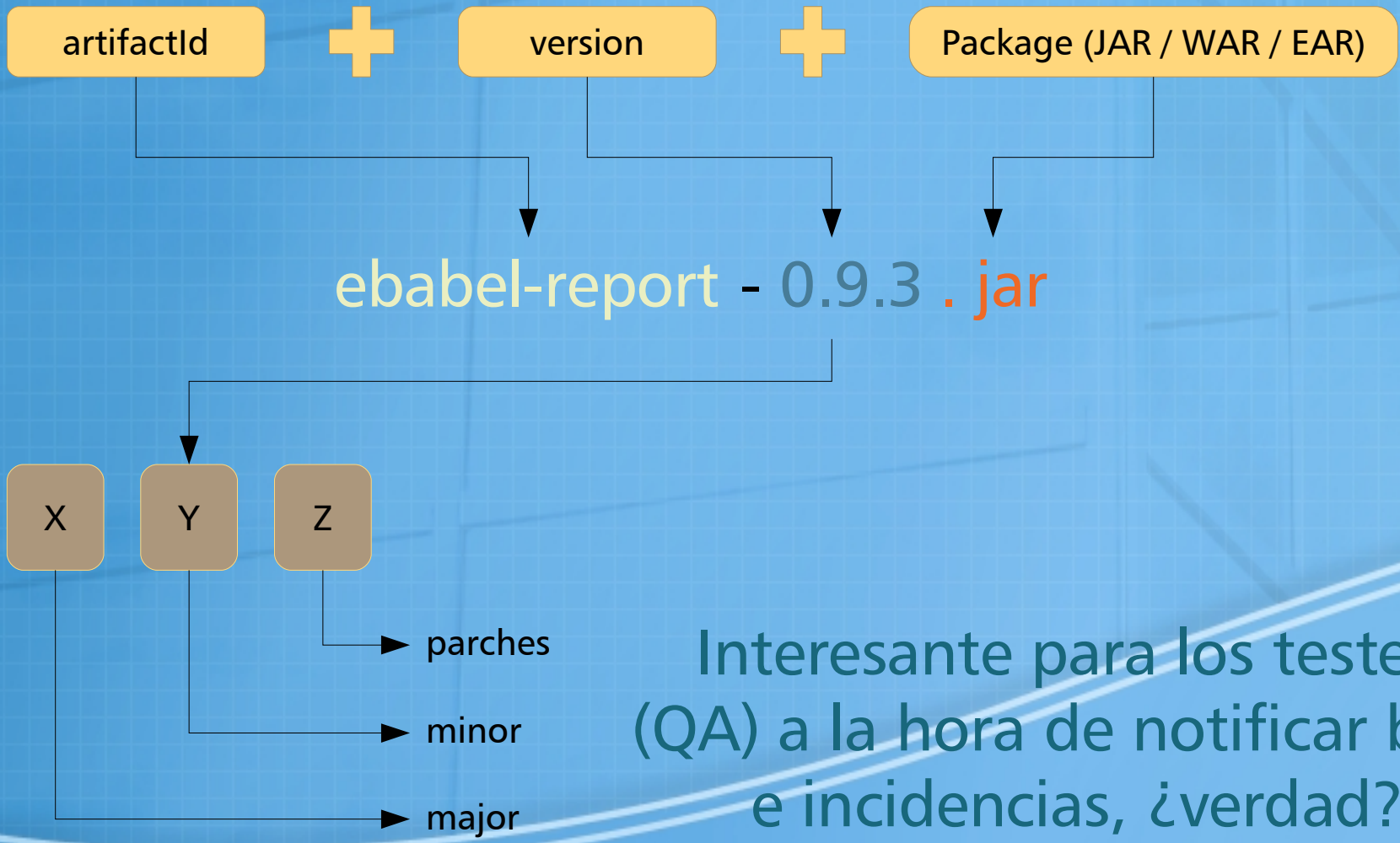
Algunos conceptos

- "One primary output per project"



Algunos conceptos

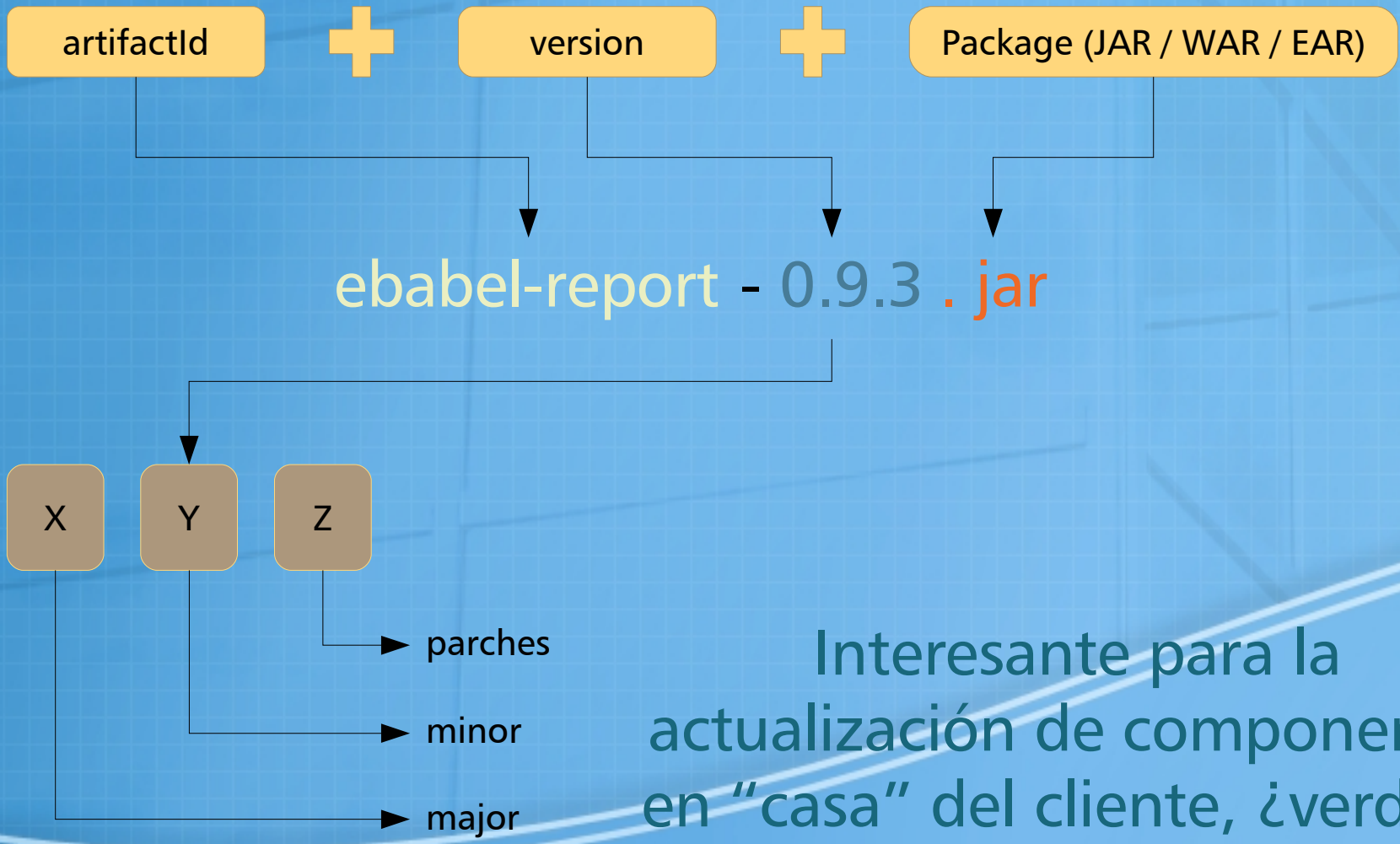
- Sistema de nomenclatura



Interesante para los testers (QA) a la hora de notificar bugs e incidencias, ¿verdad?

Algunos conceptos

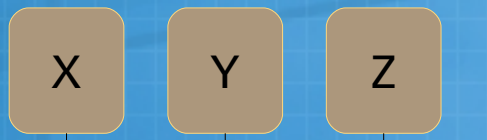
- Sistema de nomenclatura



Interesante para la actualización de componentes en “casa” del cliente, ¿verdad?

Algunos conceptos

- Sistema de nomenclatura



- Resolución de problemas
- Implementación de nuevas características, funcionalidades, mejoras.
- Puede implicar la resolución de problemas
- Una nueva release que añada funcionalidades es de la forma X.Y.0 con Y>0
- La compatibilidad con versiones hacía atrás sólo es permitida en release de la forma X.0.0 con X>0.

LA PRINCIPAL IDEA EN MAVEN: P.O.M. (Project Object Model)

P.O.M. Project Object Model

- Los proyectos en Maven están descritos mediante un archivo llamado pom.xml
- Este modelo conceptual está descrito a partir de un XMLSchema
- Describe información referente a:
  - Información general del proyecto
  - Configuraciones personalizadas para cada etepa del ciclo de vida asociado al proceso de construcción.
  - Dependencias
  - Herramientas externas:

P.O.M. Project Object Model

- Listas de correo
- Sistemas de control de versiones (S.C.M)
- Sistemas de bugtracking

– Configuración de repositorios (locales/externos)

P.O.M. Project Object Model

```
<project xmlns="http://maven.apache.org
  xsi:schemaLocation="http://maven.apac
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.manuelrecena.tutorials.H
  <artifactId>HelloMaven</artifactId>
  <packaging>jar</packaging>
  <version>1.0-SNAPSHOT</version>
  <name>Maven Quick Start Archetype</name>
  <url>http://maven.apache.org</url>
  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>3.8.1</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
</project>
```

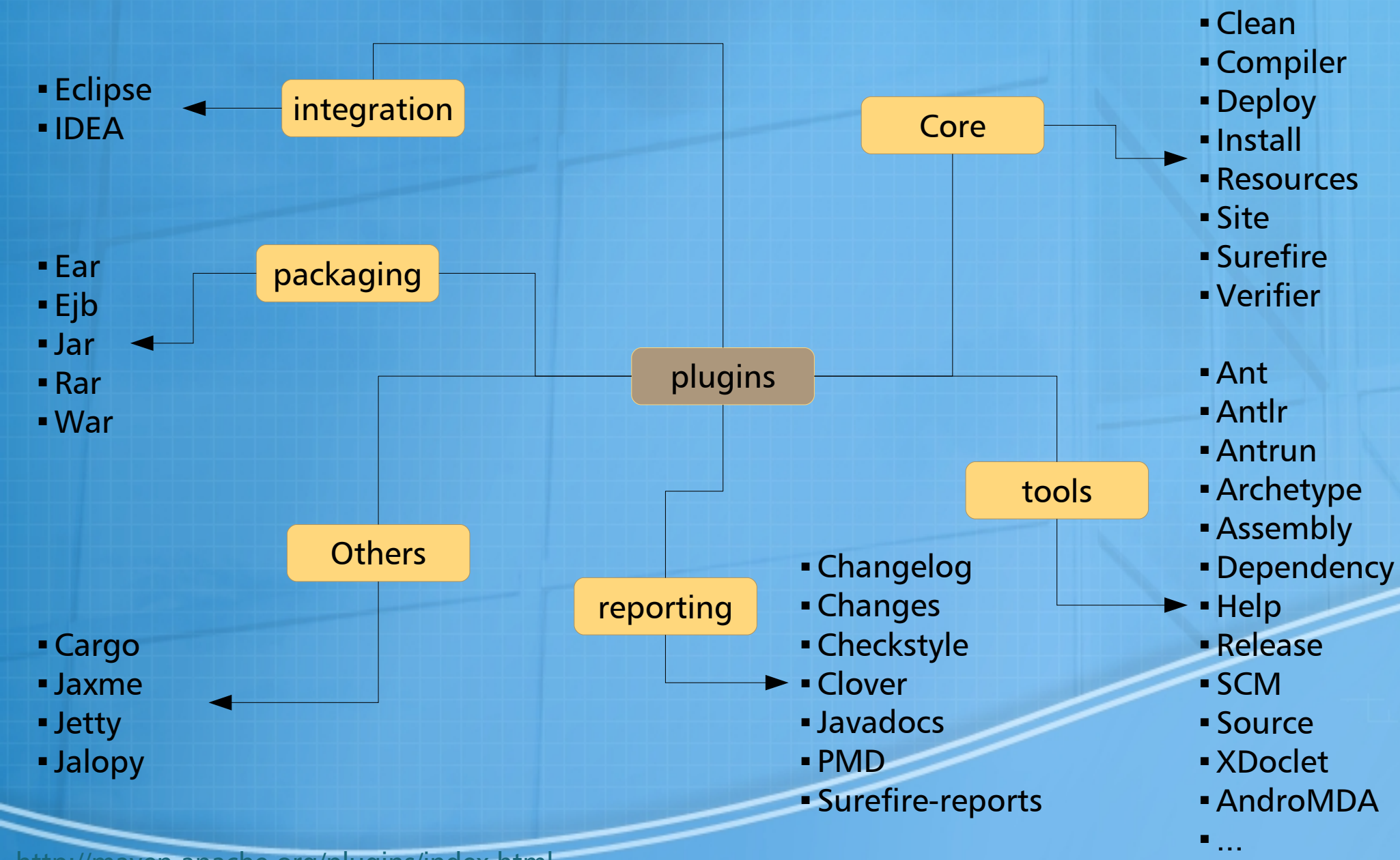
junit.pom

```
<project>
  <modelVersion>4.0.0</modelVersion>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <version>3.8.1</version>
</project>
```

mvn archetype:create -DgroupId=com.manuelrecena.tutorials.HelloMaven -DartifactId=HelloMaven

EL VERDADERO VALOR: PLUGINS

El verdadero valor: plugins



http://maven.apache.org/plugins/index.html

CÓMO ENCAJA MAVEN EN ESTOS ESCENARIOS

¿Cómo encaja Maven en estos escenarios?

- Definiendo y manteniendo proyectos de forma estandarizada
- Generando informes de los que obtener indicadores para la toma de decisiones
- Agilizando los procesos de distribución
- Centralizando la información mediante la generación de un sitio web para los proyectos
- Integrandose con sistemas de bugtracking y software de gestión de proyectos

Construcción de un ejemplo sencillo y práctico: `HelloMaven`

CONCLUSIONES

Conclusiones

- No dedicar más tiempo del necesario a definir procedimientos si herramientas como Maven ya los modelan.
- Maven como resultado de la experiencia de otros profesionales.
- Existe gran cantidad de plugins que nos permiten trabajar con framework y librerías más conocidas: hibernate, struts, Spring, ...
- La generación automatizada de código fuente, archivos de configuración, documentación, etc..., aumenta la productividad y reduce el riesgo de errores.

Conclusiones

- Existen plugins dedicados exclusivamente para asegurar la calidad del código y comprobar que se cumplen las guías de estilo definidas.

```
<!-- Se crean los archivos web.xml y struts-config.xml y se almacena en target/ebabel/WEB-INF/  -->
<webdoclet
        destdir="${project.build.directory}/${project.build.finalName}/WEB-INF"
        excludedtags="" verbose="true" force="true"
        mergeDir="src/main/merge" addedTags="">
        <!-- Se crea el archivo de configuración de struts: struts-config.xml -->
        <strutsconfigxml version="1.1"
                xmlencoding="ISO-8859-1" validateXML="true" />
        <!-- Se crea el archivo de configuración de la aplicación (deployment description): web.xml -->
        <deploymentdescriptor
                servletspec="2.4" xmlencoding="ISO-8859-1" validateXML="true" />
        <fileset
                dir="${project.build.sourceDirectory}">
                <include name="**/*.java" />
                <exclude name="**/tlds/*" />
        </fileset>
</webdoclet>
```

Conclusiones

metodología

estandarización

agilidad

productividad

usabilidad

calidad

control

REFERENCIAS

Referencias

- http://del.icio.us/recena/maven
- http://del.icio.us/tag/maven

¿PREGUNTAS?