Creació EAR en un sol pas

A qui va dirigit

Aquest Howto va dirigit a tots aquells perfils tècnics encarregats de la paquetització d'aplicacions maven en el marc del seu desplegament en el SIC.

Versió de Canigó

La creació d'artefactes empaquetats en un EAR és aplicable a qualsevol versió del Framework de Canigó. Si bé s'ha de tenir més en compte la versió de maven, que en el cas d'aquest Howto, es recomana que sigui la 2.2.1.

Introducció

Un dels passos que s'han de seguir per poder desplegar una aplicació en un servidor d'aplicacions és la construcció de l'artefacte que emmagatzemi la part dinàmica de l'aplicació. Aquest format normalment acostuma a ser un WAR o un EAR.
La generació d'un WAR és bastant directa amb Maven. No ho és tant en el cas d'un EAR. La seva generació planteja una sèrie de consideracions a tenir en compte. En aquest Howto s'explicaran aquestes consideracions així com la forma d'efectuar tot el procés de generació d'un EAR en una sola tasca de Maven.

Estructura d'un EAR

Un EAR, o Enterprise Archive, és un format d'encapsulament utilitzat per aplicacions J2EE que estiguin composades per més d'un mòdul. En el cas d'aplicacions basades en Canigó, la seva estructura és la següent:

- META-INF
  - MANIFEST.MF
  - [Descriptor].xml
  - Application.xml
- Lib
- [Module1].war
- ....
- [ModuleN].war

On;

- **MANIFEST.MF** conté informació descriptiva de l'artefacte, autors, versió i data de la compilació, versió de la JDK, eina que el va generar etc.
- **[Descriptor].xml** arxiu que conté informació relativa a com el servidor d'aplicacions tractarà la carrega de les diferents llibreries que composen els mòduls. Segons el servidor d'aplicacions pot tenir diferents noms:
  - **Jboss-app.xml** per Jboss
  - **Weblogic.xml** per Weblogic
- **Lib**, directori on es concentren les llibreries que necessiten els diferents mòduls a carregar.
- **[ModuleN]** tots i cadascun dels mòduls .war que composen l'aplicació.

Construcció d'un EAR amb Maven 2.2.1

Per poder construir un EAR amb maven en un sol pas s'han de seguir les següents recomanacions;

- Definir la següent configuració en el pom.xml de l'aplicació dintre del tag <build>:

```
<sourceDirectory>src/main/java</sourceDirectory>
resources>
        <resource>
                <directory>src/main/resources</directory>
        </resource>
</resources>
```

Figura 1. Directori de resources i codi font de l'aplicació

- Dintre de <build> i dintre del tag <plugins> definir els següents plugins:

```
<plugin>
        <artifactId>maven-war-plugin</artifactId>
        <configuration>
                <warSourceDirectory>
                        src/main/webapp
                </warSourceDirectory>
                <warName>${project.artifactId}</warName>
        </configuration>
</plugin>
```

Figura 2. Configuració del maven-war-plugin

- Aquest plugin permet definir com s'ha de construir el WAR que anirà dintre del EAR.

```
<plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-dependency-plugin</artifactId>
        <version>2.4</version>
             <executions>
                 <execution>
                       <id>copy-dependencies</id>
                       <phase>package</phase>
                       <goals>
                         <goal>copy-dependencies</goal>
                       </goals>
                       <configuration>
                         <outputDirectory>${project.build.directory}/lib</outputDirectory>
                         <overWriteReleases>false</overWriteReleases>
                         <overWriteSnapshots>false</overWriteSnapshots>
                         <overWriteIfNewer>true</overWriteIfNewer>
                       </configuration>
                  </execution>
               </executions>
</plugin>
```

Figura 3. Configuració del maven-dependency-plugin

Aquest plugin utilitza la resolució de dependències de Maven per avaluar quines són les llibreries que l'aplicació necessita i les copia al directori /lib del EAR. Aquest plugin juntament amb el fet de que les dependències de l'aplicació tingui un scope "provided" fa que no es dupliquin les llibreries que s'emmagatzemen en el EAR.

```
<plugin>
	<groupId>org.apache.maven.plugins</groupId>
	<artifactId>maven-assembly-plugin</artifactId>
	<version>2.2.2</version>
	<executions>
		<execution>
			<id>assembly-ear-app</id>
			<phase>package</phase>
			<goals>
				<goal>single</goal>
			</goals>
			<configuration>
				<descriptors>

	<descriptor>src/main/resources/assembly/ear.xml</descriptor>
				</descriptors>
				<archiverConfig>
	<appxml>${project.basedir}/src/main/resources/assembly/aplication.xml</appxml>
				</archiverConfig>
				<finalName>${project.artifactId}</finalName>
				<appendAssemblyId>false</appendAssemblyId>
			</configuration>
		</execution>
		<execution>
			<id>assembly-zip-app</id>
			<phase>package</phase>
			<goals>
				<goal>assembly</goal>
			</goals>
			<configuration>
				<descriptors>
	<descriptor>src/main/resources/assembly/static.xml</descriptor>
				</descriptors>
				<finalName>${project.artifactId}</finalName>
				<appendAssemblyId>false</appendAssemblyId>
			</configuration>
		</execution>
	</executions>
</plugin>
```

Figura 4. Configuració del maven-assembly-plugin

Aquesta configuració és una modificació de la configuració estàndard del maven-assembly-plugin que permet la construcció de l'artefacte estàtic i dinàmic de l'aplicació. La configuració que ve recomanada en el SIC serveix per construir WAR's. Aquesta permetrà la construcció d'un EAR.

Notar que es fa referencia a dos arxius anomenats ear.xml i application.xml.

- Ear.xml

Similar al dynamic.xml recomanat pel SIC, recomanat que estigui en src\main\resources\assembly\ear.xml

```
<assembly>
  <id>ear</id>
  <formats>
    <format>ear</format>
  </formats>
  <includeBaseDirectory>false</includeBaseDirectory>
  <fileSets>
    <fileSet>
      <directory>src/main/application</directory>
      <outputDirectory>/</outputDirectory>
      <includes>
        <include>**/*</include>
      </includes>
    </fileSet>
    <fileSet>
      <directory>src/main/resources/assembly</directory>
      <outputDirectory>/META-INF</outputDirectory>
      <includes>
        <include>weblogic-application.xml</include>
      </includes>
    </fileSet>
      <fileSet>
      <directory>target</directory>
      <outputDirectory>/</outputDirectory>
      <includes>
        <include>**/*.war</include>
      </includes>
    </fileSet>
    <fileSet>
      <directory>target/lib</directory>
      <outputDirectory>/lib</outputDirectory>
      <includes>
        <include>**/*.jar</include>
      </includes>
    </fileSet>
  </fileSets>
</assembly>
```

Figura 5. Arxiu de configuració del maven-assembly-plugin per la construcció de ear's

Com es pot observar, és en aquest arxiu on es fa referència al descriptor del servidor d'aplicacions. En aquest cas, **weblogic-application.xml**. Serà en aquest arxiu on es definiran les estratègies de càrrega de les classes i llibreries de l'aplicació. La ubicació d'aquest arxiu es recomana que sigui en src\main\resources\assembly\[Servidor_aplicacions]-application.xml

- Aplication.xml

```
<?xml version='1.0' encoding='UTF-8'?>
<application xmlns="http://java.sun.com/xml/ns/j2ee"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="1.4">
  <display-name>EAR Project</display-name>
  <module>
    <web>
      <web-uri>earProject.war</web-uri>
      <context-root>earProject/AppJava</context-root>
    </web>
  </module>
</application>
```

Figura 6. Arxiu descriptor de l'aplicació

Aquest arxiu emmagatzema el descriptor de l'aplicació així com el seu context. Es recomana que estigui en src\main\resources\assembly\application.xml

- Posar totes les dependències de l'aplicació amb scope "provided".

Això es fa així per que en els diferents war's que composin l'aplicació no s'emmagatzemin les llibreries. L'objectiu és que sigui el EAR, dintre del seu directori /lib, qui emmagatzemi aquestes llibreries. Amb aquest muntatge juntament amb el descriptor del EAR (aplication.xml) es pot escollir quines llibreries es vol que es carreguin preferentment de l'EAR i quines del servidor d'aplicacions, sempre i quan estiguin duplicades.

EAR i Servidors d'aplicacions

Un dels arxius necessaris en la construcció del EAR és, com s'ha comentat en el punt anterior, l'arxiu amb el descriptor del servidor d'aplicacions.

És l'única part de la configuració que depèn del servidor d'aplicacions. A continuació s'adjunta un parell d'exemples de descriptor d'arxius;

```
<?xml version="1.0" encoding="UTF-8"?>
  <weblogic-application xmlns="http://xmlns.oracle.com/weblogic/weblogic-application"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/javaee_5.xsd
http://xmlns.oracle.com/weblogic/weblogic-application
http://xmlns.oracle.com/weblogic/weblogic-application/1.2/weblogic-application.xsd">
    <application-param>
      <param-name>webapp.encoding.default</param-name>
      <param-value>UTF-8</param-value>
    </application-param>
    <prefer-application-packages>
              <package-name>javax.jws.*</package-name>
       <package-name>org.apache.*</package-name>
       <package-name>javax.xml.parsers.*</package-name>
       <package-name>javax.xml.stream.*</package-name>
       <package-name>org.bouncycastle.*</package-name>
    </prefer-application-packages>
</weblogic-application>
```

Figura 7. Exemple de descriptor d'un servidor d'aplicacions WebLogic 10.3

```
<?xml version="1.0" encoding="UTF-8"?>
<weblogic-application xmlns="http://www.bea.com/ns/weblogic/90"
                     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <application-param>
    <param-name>webapp.encoding.default</param-name>
    <param-value>UTF-8</param-value>
  </application-param>

  <prefer-application-packages>
    <package-name>javax.wsdl.*</package-name>
    <package-name>javax.jws.*</package-name>
  </prefer-application-packages>
</weblogic-application>
```

Figura 8. Exemple de descriptor d'un servidor d'aplicacions Weblogic 9.2

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<!DOCTYPE jboss-app PUBLIC
       "-//JBoss//DTD J2EE Application 1.4//EN"
       "http://www.jboss.org/j2ee/dtd/jboss-app_4_2.dtd">
<jboss-app>
  <module-order>strict</module-order>
  <loader-
repository>cat.gencat.ctti.canigo.eforms:loader=IntegracioServeisSTD.ear<loader-
repository-config>java2ParentDelegation=true</loader-repository-config>
  </loader-repository>
</jboss-app>
```

Figura 8. Exemple de descriptor d'un servidor d'aplicacions Jboss 5.1

Per tenir més informació sobre els descriptors dels servidors d'aplicacions consultar les web oficials dels fabricants.

Finalment per qualsevol dubte o problema que pugui sorgir es pot fer una consulta al CS Canigó a la bústia oficina-tecnica.canigo.ctti@gencat.net o bé mitjançant una petició de suport a l'eina JIRA del CTTI en http://cstd.ctti.gencat.cat/jiracstd.