

JBOSS EAP 6



Objetivos

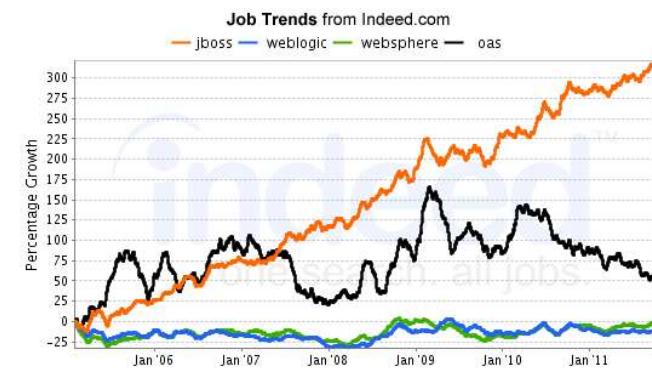
- Conocer la **arquitectura** de JBoss AS/ EAP
- Dominar el **despliegue** de aplicaciones y servicios en el servidor
- Establecer la **configuración** para las aplicaciones
- Conocer y utilizar las herramientas de **administración** y **monitorización** más usadas
- Definir e implementar una arquitectura en **clúster**
- Realizar diversas **optimizaciones** en el servidor para mejorar su **rendimiento**

- I. Introducción
- II. Instalación del servidor de aplicaciones
- III. Configuración y herramientas de administración
- IV. Despliegue de aplicaciones
- V. Seguridad de las aplicaciones
- VI. Clustering

- I. Introducción**
- II. Instalación del servidor de aplicaciones**
- III. Configuración y herramientas de administración**
- IV. Despliegue de aplicaciones**
- V. Seguridad de las aplicaciones**
- VI. Clustering**

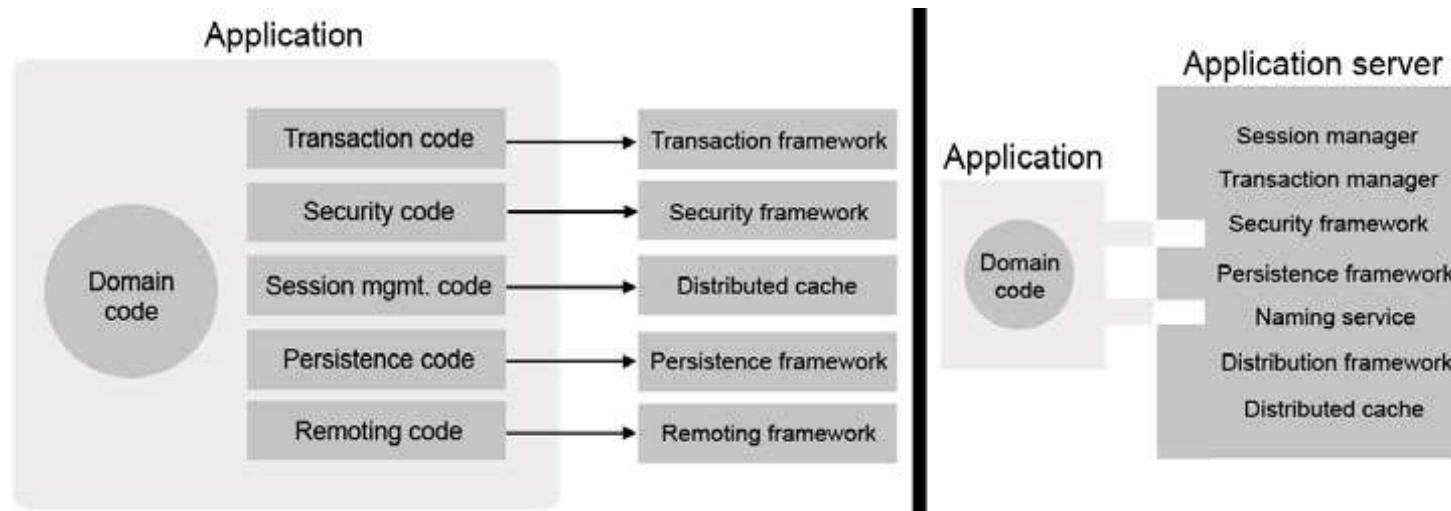
Evolución en el tiempo

- 1999 Proyecto creado por *Marc Fleury*
- 2002 JBoss AS 3
- 2004 JBoss AS 4
- 2006 JBoss AS 5 (Beta)
- **Red Hat compra JBoss en Abril del 2006**
- 2009 JBoss AS 5.1
- 2010 JBoss AS 6.0
- 2011 JBoss AS 7.0
- 2012 JBoss AS 7.1
- Enero 2013 JBoss AS 7.2
/ JBoss EAP 6.1
- **Diciembre 2013 Jboss AS 7.3
/ Jboss EAP 6.2**



¿Que es JBoss?

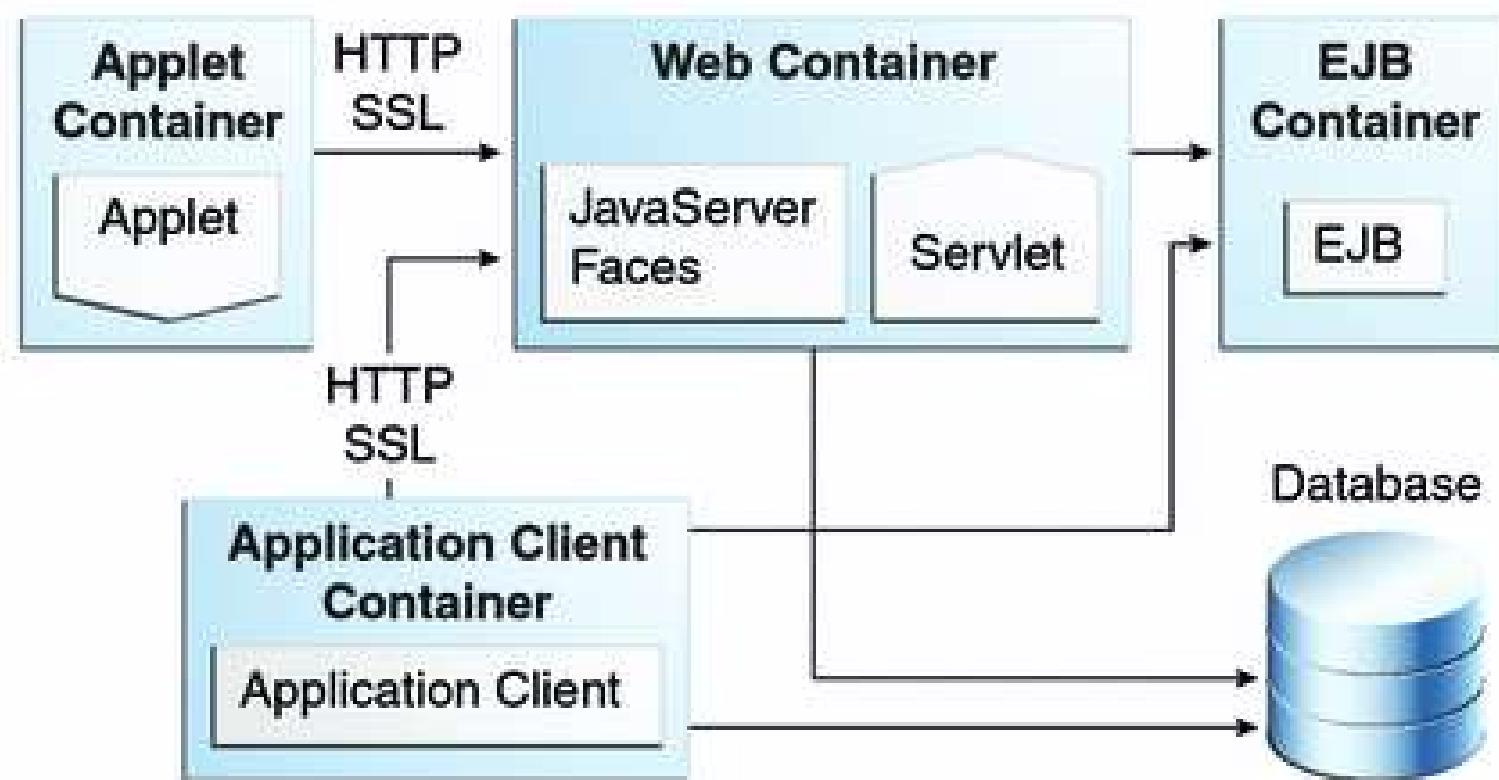
- Un Servidor de Aplicaciones J2EE (Java EE)



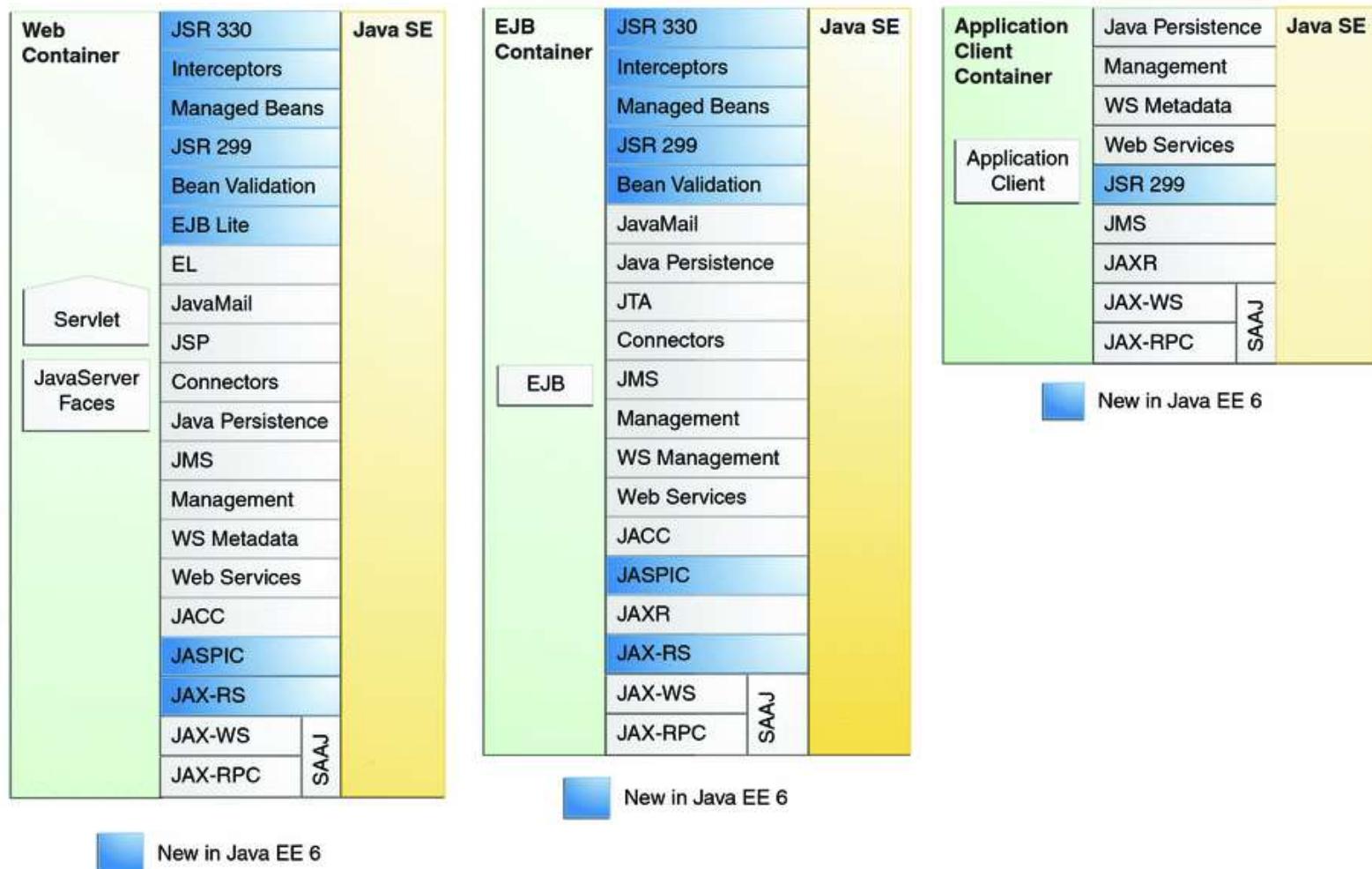
- Programar con los estándares de Java EE y usando un servidor de aplicaciones puede **reducir drásticamente la cantidad de código de integración y configuración** de otra forma se necesitaría. Además nos evita tener que desarrollar todos los servicios de las aplicaciones desde 0.

Servidores de aplicaciones JEE (Arquitectura)

- Arquitectura de un servidor JEE genérico:

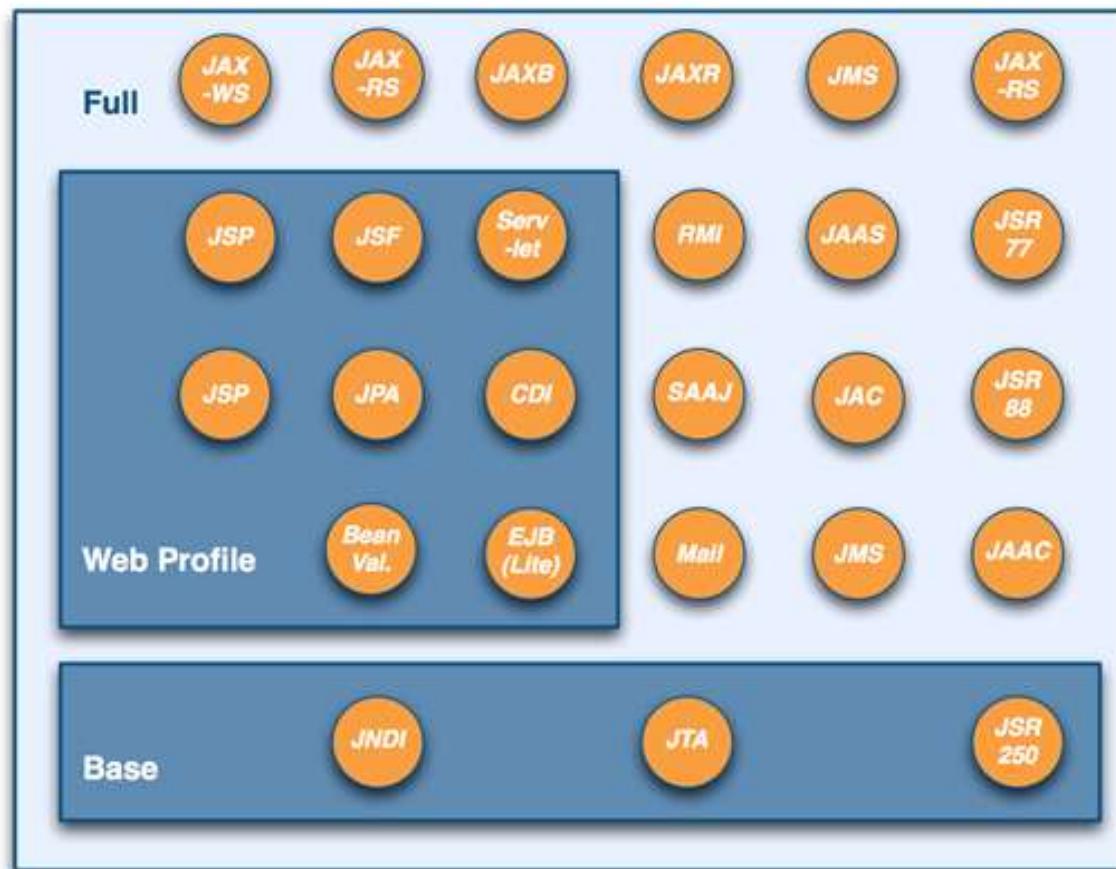


Tecnologías y API en Java EE 6



- Servidores modulares (JEE6 Profiles):

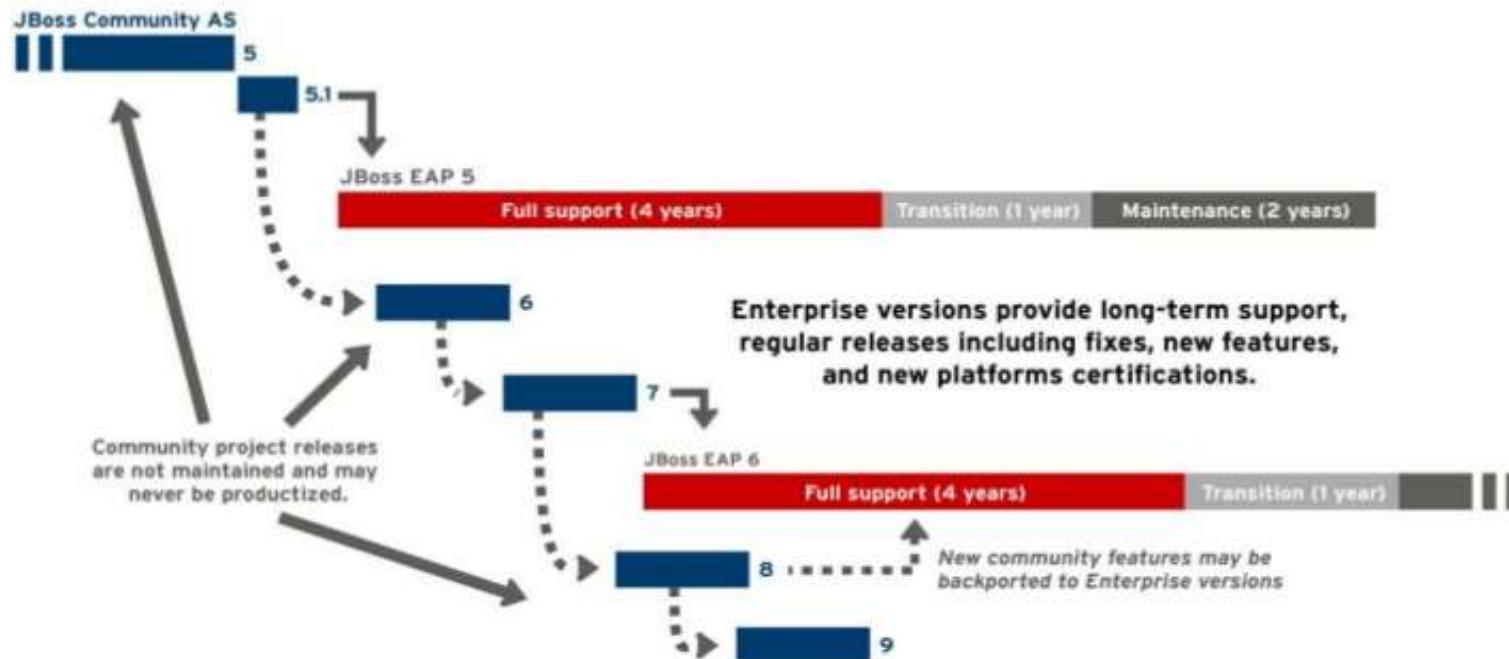
Java EE 6 Profiles



- Jboss AS 7 (Wildfly)
 - Usabilidad mejorada
 - Mayores capacidades de manejabilidad
 - Configuración simplificada
 - Alto rendimiento
 - Open source
- Jboss EAP 6 (y además)
 - Actualizaciones y parches
 - Certificado para plataformas: HW/OS/JVM/DB
 - Soporte Profesional
 - Herramientas
 - JBoss Developer Studio
 - JBoss Operations Network



JBoss AS vs JBoss EAP



While **community projects** continue to rapidly evolve,
enterprise middleware products focus on long term stability.

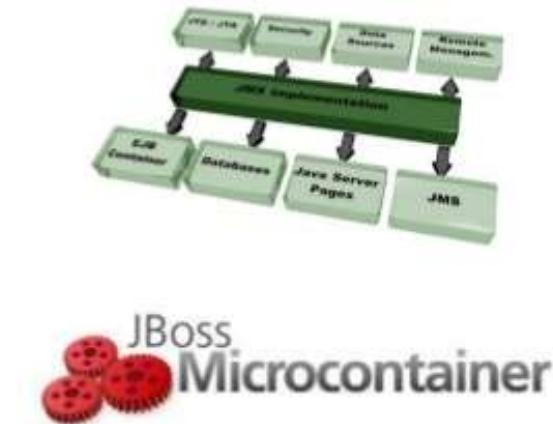
Características de la plataforma JBoss

- Las características destacadas de **JBoss** incluyen:
 - **Open Source**, Producto de licencia de código abierto sin coste adicional.
 - Cumple con los **estándares Java EE**.
 - **Fiabilidad** a nivel de empresa
 - **Incrustable**, orientado a arquitectura de servicios.
 - **Flexibilidad** consistente
 - Servicios del **middleware** para cualquier objeto de Java
 - Contrato de **Soporte profesional 24x7** ofrecido por el fabricante
 - Soporte completo para JMX

- Diferencias a lo largo del tiempo (versiones):

JBoss Kernel Taxonomy

- JBoss AS 2.x, 3.x, 4.x
 - JBoss JMX MicroKernel
- JBoss AS 5.x, 6.x
 - JBoss MC - MicroContainer
- JBoss AS7.x
 - JBoss MSC - Modular Service Container

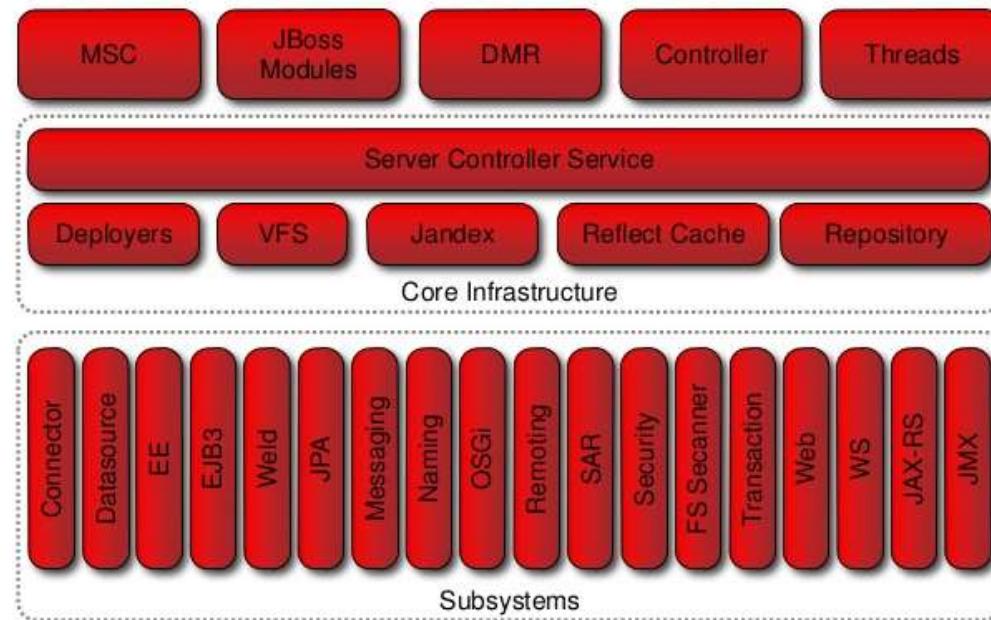


Arquitectura JBoss AS (II)

■ Jboss MSC (Modular Service Container)

- Define una **arquitectura en tres niveles** o capas:

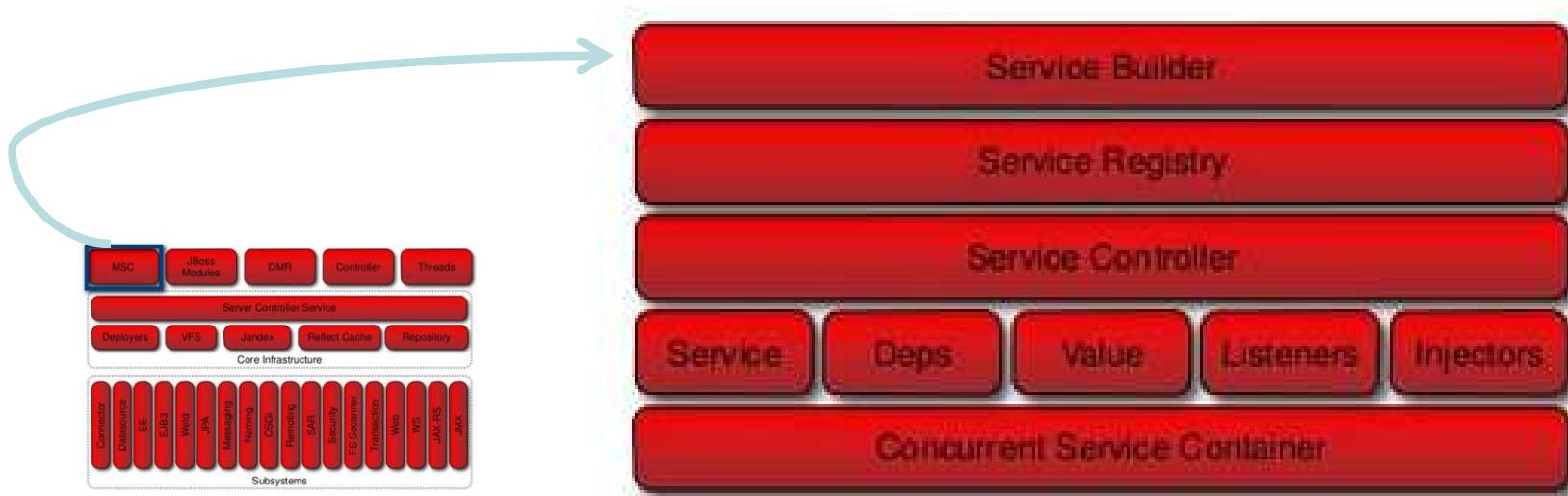
AS7 Architecture



Arquitectura JBoss AS (III)

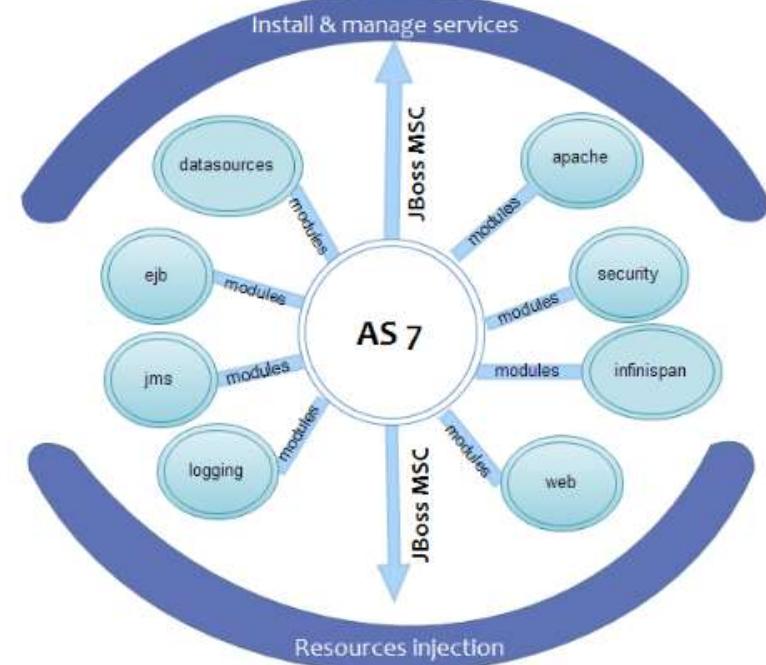
■ Jboss MSC (Modular Service Container)

- Contenedor de Pequeño y ligero
- Maquina de estados escalable y altamente concurrente
- Servicios definidos por interfaces
- Estados de los servicios: Up & Down
- Modos de arranque de los Servicios: Active, Passive, On-Demand, Lazy, Never



■ Jboss MSC (Modular Service Container)

- **Servicios**
 - En AS7 prácticamente todo es un servicio
 - Servicios pueden ser dependientes entre si
 - Todas las dependencias de un servicio se resuelven en el arranque
 - Si un dependencia va a pararse, primero MSC para todos los servicios dependientes primero
 - Inyección de dependencias en servicios
- **Ejemplos**
 - EJB'S (son 2 servicios)
 - JNDI Bindings
 - Servlets
 - Los artefactos

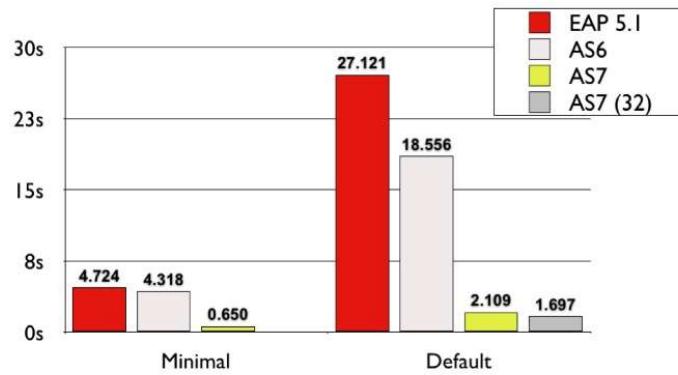


Arquitectura JBoss AS (V)

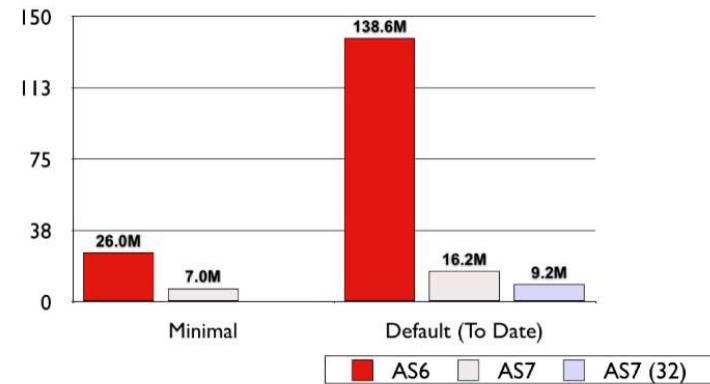
■ Jboss MSC (Modular Service Container)

- Arranque concurrente del servidor
- Despliegues concurrentes de las aplicaciones
- Índices de las anotaciones (Metadatos)
- Cacheado de los metadatos de reflection
- Carga perezosa de servicios
- Parseo de XML vía STAX
- Reescritura del servidor por completo

Boot Time Results



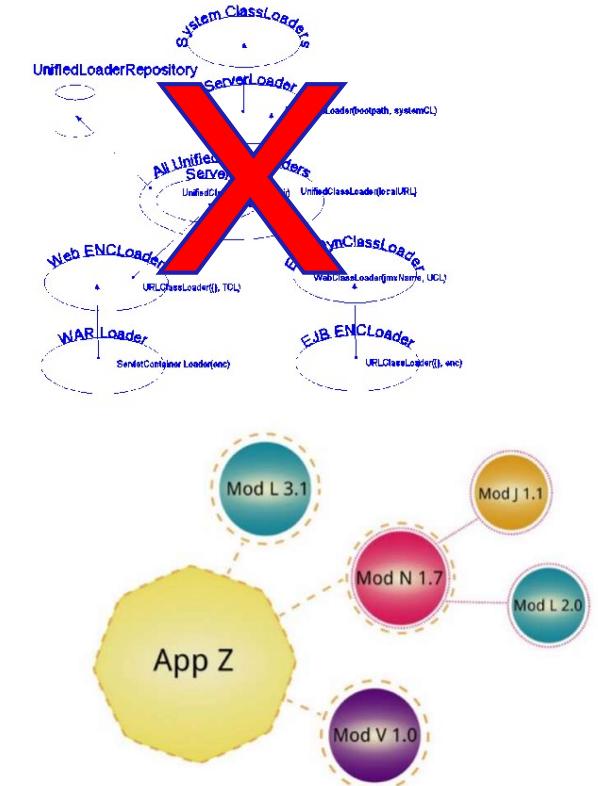
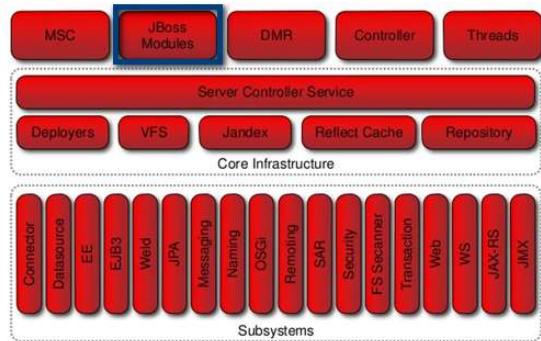
Memory Comparison



Arquitectura JBoss AS (VI)

■ Jboss Modules

- Classloading modular
 - Un grafo de cargadores de clases no un árbol
 - Un módulo delega en otros como iguales
 - Un módulo importa otros módulos
 - Dependencias transitivas (ocultas por defecto)
 - Diferentes versiones de un modulo pueden coexistir
 - Los módulos sólo ven lo que importan
 - Redefinición dinámica de módulos
- Carga de clases concurrente
- Resolución de dependencias O(1)
- Definiciones de módulos externas fuera del jar



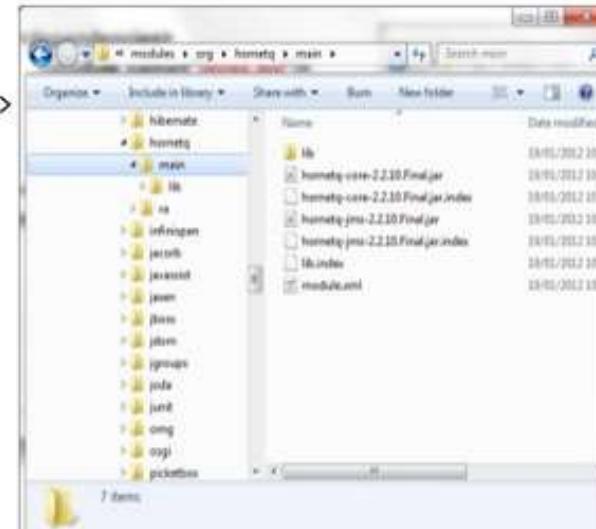
Arquitectura JBoss AS (VII)

■ Jboss Modules

modules\org\hornetq\main\module.xml

```
<module xmlns="urn:jboss:module:1.0" name="org.hornetq">
    <resources>
        <resource-root path="hornetq-core-2.2.10.Final.jar"/>
        <resource-root path="hornetq-jms-2.2.10.Final.jar"/>
    </resources>

    <dependencies>
        <module name="javax.api"/>
        <module name="javax.jms.api"/>
        <module name="javax.resource.api"/>
        <module name="org.jboss.jts"/>
        <module name="org.jboss.netty"/>
    </dependencies>
</module>
```



- ahora la ruta es **modules\system\layers\base\org\...**

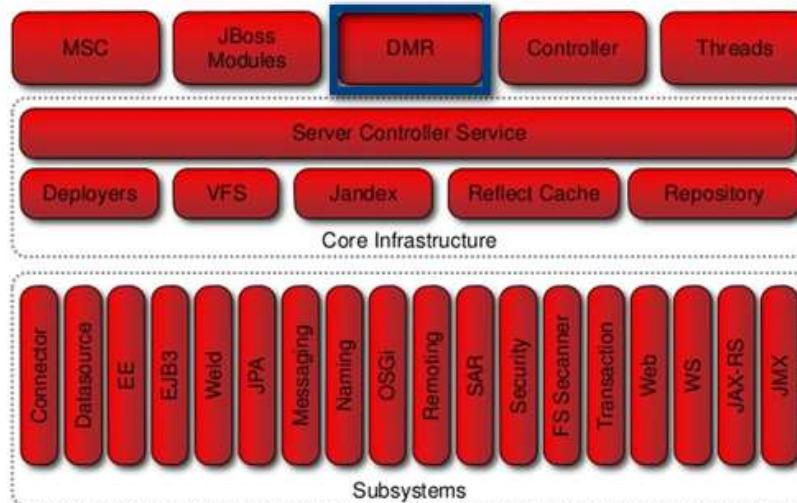
■ Jboss Modules

- **Deployments**
 - Los despliegues también son módulos (.ear, .war, .jar, etc)
 - Dependencias explícitas en META-INF/manifest.mf
 - Dependencias implícitas según la tecnología y meta información
 - En las clases vía Anotaciones
 - En el modulo vía descriptor de despliegue
- **OSGi**
 - AS7 es 100% compatible con OSGi Core 4.2
 - OSGi Bundles son JBoss Modules
 - Despliegue normal o asistido

Arquitectura JBoss AS (IX)

■ Jboss DMR (Dynamic Model Representation)

- Api central para configurar el servidor
- Centrada en el usuario no en sintaxis
- Servicios definidos por interfaces
- Convertible a/desde JSON
- También define un protocolo binario



User-focused Configuration

```
<bean name="TransactionManager" class="com.arjuna.ats.jbosstablet.jta.TransactionManagerService">
    <annotation>@org.jboss.aop.microcontainer.aspects.jmx.JMXName="jboss.service=TransactionManager",
    exposedInterface="com.arjuna.ats.jbosstablet.jta.TransactionManagerServiceMBean.class, registerDirectly=true"</annotation>
    <annotation>@org.jboss.managed.api.annotation.ManagementObject(name="TransactionManager", componentType=@org.jboss.managed.api.annotation.ManagementComponent(type = "MBean", subtype = "JTA"), targetInterface="com.arjuna.ats.jbosstablet.jta.TransactionManagerServiceMBean.class)"</annotation>
    <property name="transactionTimeout">300</property>
    <property name="objectStoreDir">${jboss.server.data.dir}/tx-object-store</property>
```



```
<xsubsystem xmlns="urn:jboss:domain:transactions:1.0">
    <recovery-environment socket-binding="txn-recovery-environment"
        status-socket-binding="txn-status-manager"/>
    <core-environment socket-binding="txn-socket-process-id"/>
</xsubsystem>
```

Management APIs

- Command Line Interface (CLI)
- Remote Java API
- HTTP/JSON API
- GWT-based console
- JMX mapping

- **Jboss DMR (Dynamic Model Representation)**

- Equivalencia entre XML y objetos de dominio

Tx configuration snippet from standalone.xml

```
...  
<subsystem  
    xmlns="urn:jboss:domain:transactions:1.1">  
    <core-environment>  
        <process-id>  
            <uuid/>  
        </process-id>  
    </core-environment>  
    <recovery-environment socket-binding="txn-recovery-environment" status-socket-binding="txn-status-manager"/>  
    <coordinator-environment default-timeout="300"/>  
</subsystem>  
...
```



...maps to Domain Model

```
...  
    "transactions" : {  
        "default-timeout" : 300,  
        "enable-statistics" : false,  
        "enable-tsm-status" : false,  
        "jts" : false,  
        "node-identifier" : "1",  
        "object-store-path" : "tx-object-store",  
        "object-store-relative-to" : "jboss.server.data.dir",  
        "path" : "var",  
        "process-id-socket-max-ports" : 10,  
        "process-id-uuid" : true,  
        "recovery-listener" : false,  
        "relative-to" : "jboss.server.data.dir",  
        "socket-binding" : "txn-recovery-environment",  
        "status-socket-binding" : "txn-status-manager"  
    },
```

Administración JBoss AS (I)

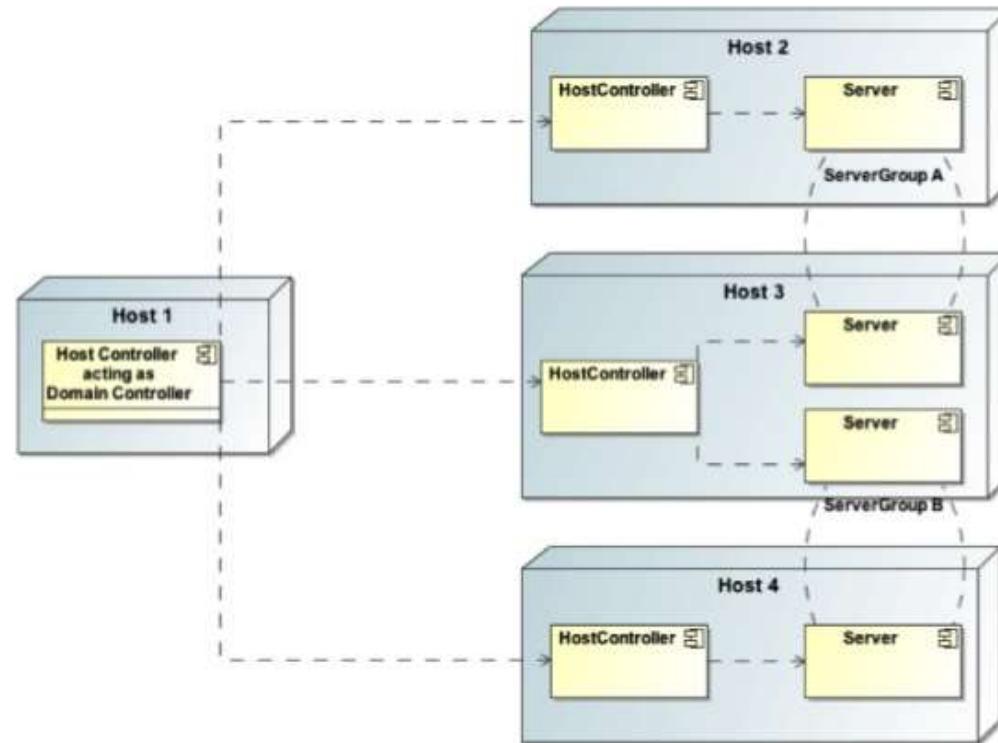
- Toda la configuración reside en un sólo fichero
 - **standalone.xml**
 - **domain.xml**
- API de administración nueva que **persiste los cambios** en la configuración
- API de administración puede gestionar todos los servidores del **dominio**
- Consola de administración amigable vía **Web**
<http://localhost:9990>
- Herramienta de línea de comandos para usar en **Scripts**
jboss-cli.sh ó **jboss-cli.bat**

Administración JBoss AS (II)

- Dos maneras de trabajar con la plataforma
 - **Standalone**
 - Servidor único sobre una única JVM (Modelo Tradicional)
 - Herramientas de administración IN-VM
 - No hay gestión del ciclo de vida del servidor (Sólo parada)
 - Bueno para entornos de desarrollo o topologías empresariales con herramientas de terceros
 - ¡El modo Standalone también permite clústers!
 - **Domain**
 - Modelo Múltiples JVM/ Múltiples Servidores
 - Administración coordinada por el proceso controlador del dominio (**Domain Controller Process**)
 - Múltiples instancias de servidor por equipo (**Host**)
 - Gestión del ciclo de vida de los servidores por el controlador (**Host Controller**)

- Elementos de un Dominio de Administración JBoss

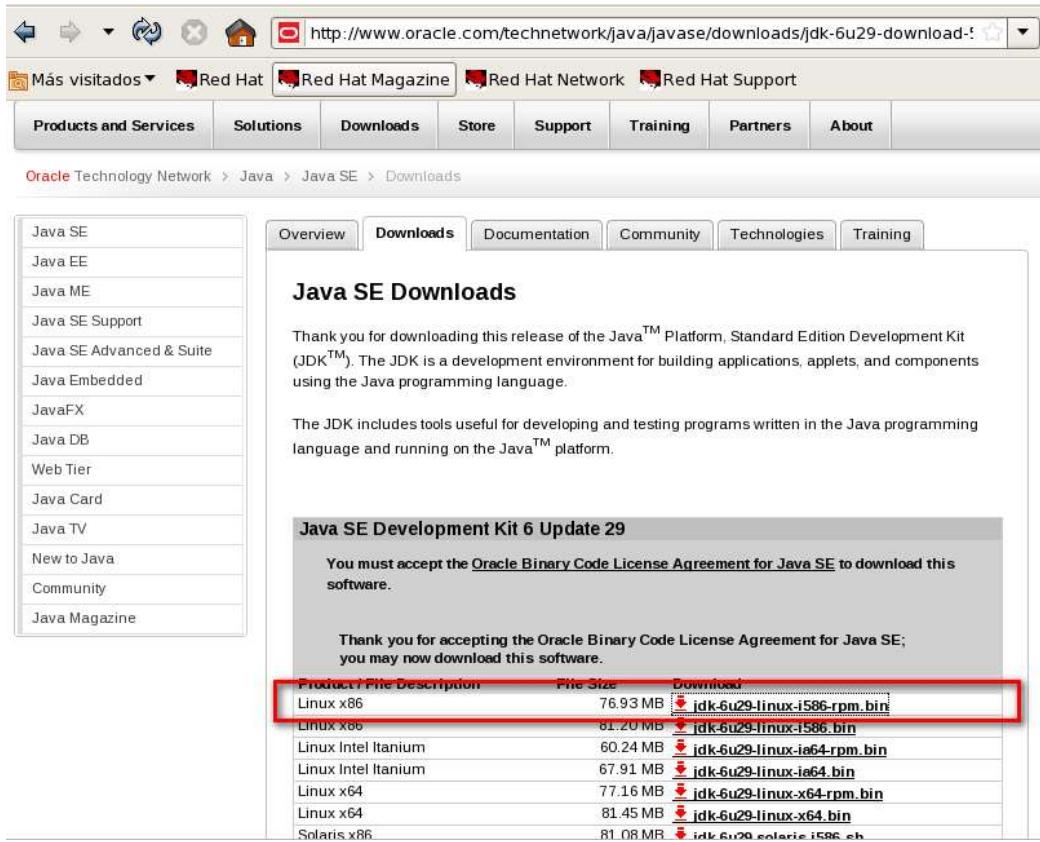
Domain Mode Physical Topology



- I. Introducción
- II. Instalación del servidor de aplicaciones**
- III. Configuración y herramientas de administración
- IV. Despliegue de aplicaciones
- V. Seguridad de las aplicaciones
- VI. Clustering
- VII. Optimización

Requisitos previos (I)

Sólo es **necesario** tener instalado previamente el **JDK** ó el **JRE** de Oracle (<http://java.sun.com>) o bien OpenJDK

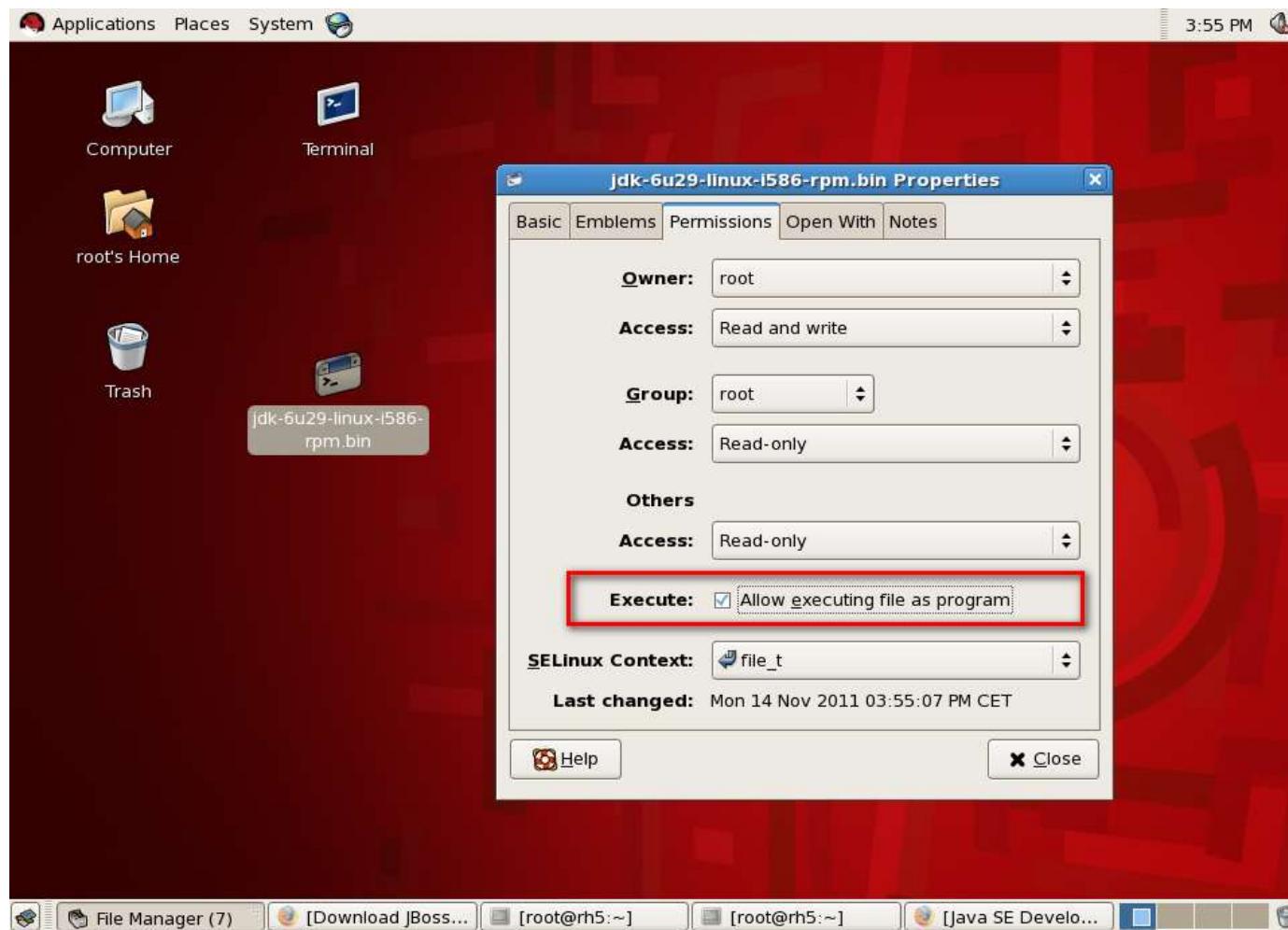


The screenshot shows the Oracle Java SE Downloads page. At the top, there's a navigation bar with links for Products and Services, Solutions, Downloads, Store, Support, Training, Partners, and About. Below that is a breadcrumb trail: Oracle Technology Network > Java > Java SE > Downloads. On the left, there's a sidebar with links for Java SE, Java EE, Java ME, Java SE Support, Java SE Advanced & Suite, Java Embedded, JavaFX, Java DB, Web Tier, Java Card, Java TV, New to Java, Community, and Java Magazine. The main content area has tabs for Overview, Downloads, Documentation, Community, Technologies, and Training. The Downloads tab is selected. A section titled "Java SE Downloads" explains the purpose of the JDK. Below it, a box for "Java SE Development Kit 6 Update 29" states that users must accept the Oracle Binary Code License Agreement for Java SE to download. A message at the bottom of this box says "Thank you for accepting the Oracle Binary Code License Agreement for Java SE; you may now download this software." A table lists download options for different platforms:

Product / File Description	File Size	Download
Linux x86	76.93 MB	jdk-6u29-linux-i586-rpm.bin
Linux x64	81.20 MB	jdk-6u29-linux-i586.bin
Linux Intel Itanium	60.24 MB	jdk-6u29-linux-ia64-rpm.bin
Linux Intel Itanium	67.91 MB	jdk-6u29-linux-ia64.bin
Linux x64	77.16 MB	jdk-6u29-linux-x64-rpm.bin
Linux x64	81.45 MB	jdk-6u29-linux-x64.bin
Solaris x86	81.08 MB	jdk-6u29_solaris-i586_eh

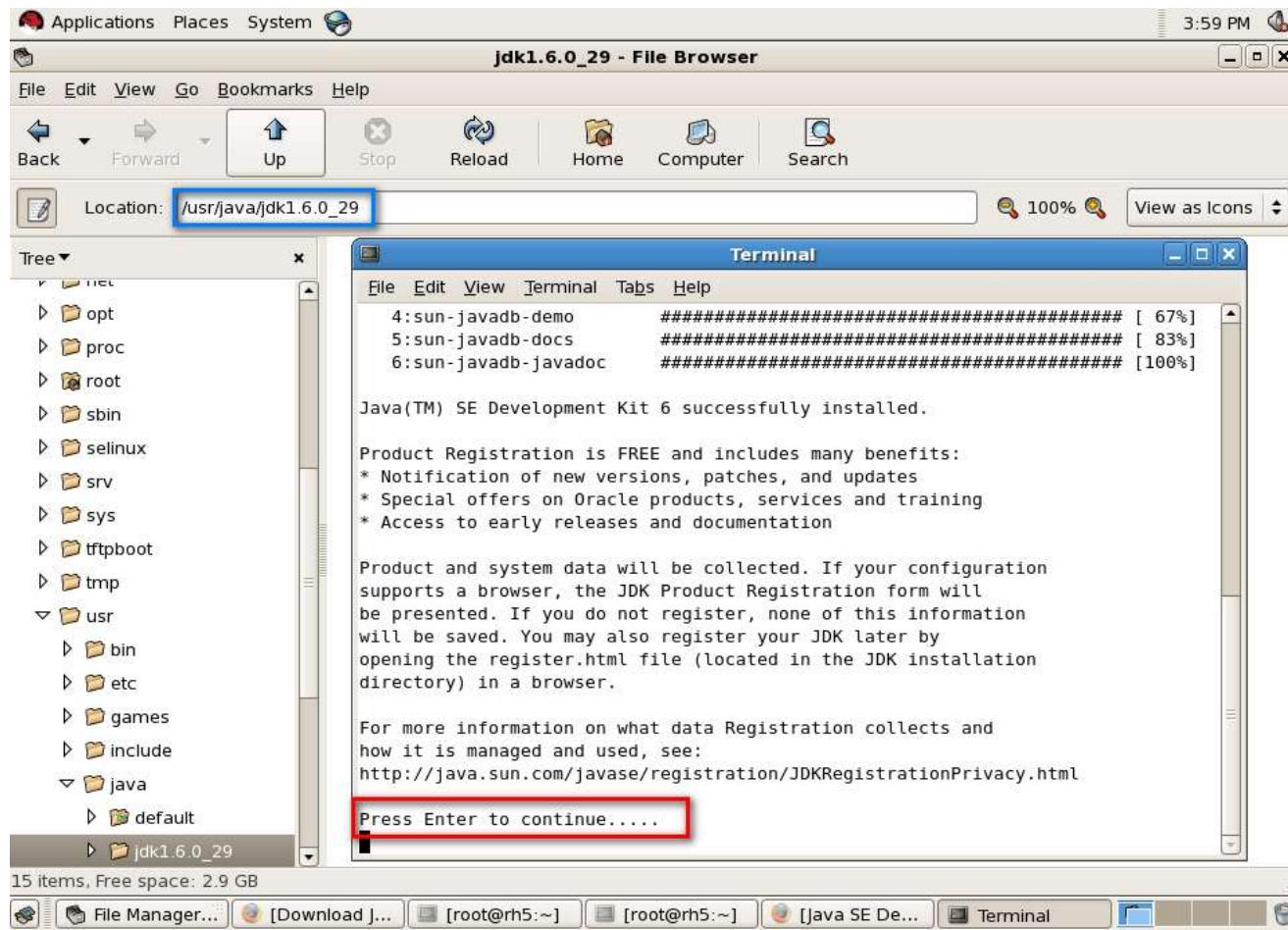
Requisitos previos (II)

Damos permiso de ejecución al archivo descargado



Requisitos previos (III)

Por defecto la instalación se realiza en el directorio **/usr/java/**



Requisitos previos (IV)

- Creamos un archivo **/etc/profile.d/java.sh**

```
#touch /etc/profile.d/java.sh  
#chmod +x /etc/profile.d/java.sh  
#vi /etc/profile.d/java.sh
```

```
#Set Env Variables for Java  
JAVA_HOME=/usr/java/jdk1.6.0_29  
export JAVA_HOME  
export PATH=$JAVA_HOME/bin:$PATH
```

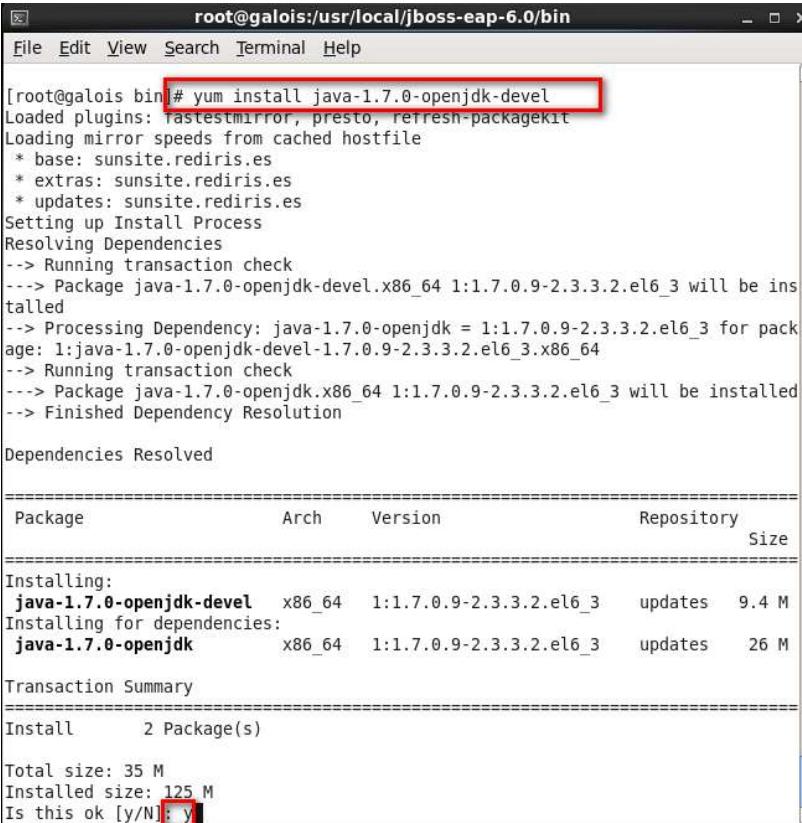
- Cerramos la sesión de shell para recargar
- Testeamos la instalación

```
# java -version
```

Requisitos previos (V)

■ Instalación del OpenJdk en RedHat (Preferible)

- La instalación de OpenJdk se puede hacer desde yum
- Se instala en **/usr/lib/jvm/java-1.7.0-openjdk.x86_64/bin**



The screenshot shows a terminal window titled "root@galois: /usr/local/jboss-eap-6.0/bin". The user is running the command `yum install java-1.7.0-openjdk-devel`. The terminal output shows the package being installed from the sunsite.rediris.es repository, along with its dependencies and transaction summary.

```
[root@galois bin]# yum install java-1.7.0-openjdk-devel
Loaded plugins: fastestmirror, presto, refresh-packagekit
Loading mirror speeds from cached hostfile
 * base: sunsite.rediris.es
 * extras: sunsite.rediris.es
 * updates: sunsite.rediris.es
Setting up Install Process
Resolving Dependencies
--> Running transaction check
--> Package java-1.7.0-openjdk-devel.x86_64 1:1.7.0.9-2.3.3.2.el6_3 will be installed
--> Processing Dependency: java-1.7.0-openjdk = 1:1.7.0.9-2.3.3.2.el6_3 for package: 1:java-1.7.0-openjdk-devel-1.7.0.9-2.3.3.2.el6_3.x86_64
--> Running transaction check
--> Package java-1.7.0-openjdk.x86_64 1:1.7.0.9-2.3.3.2.el6_3 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

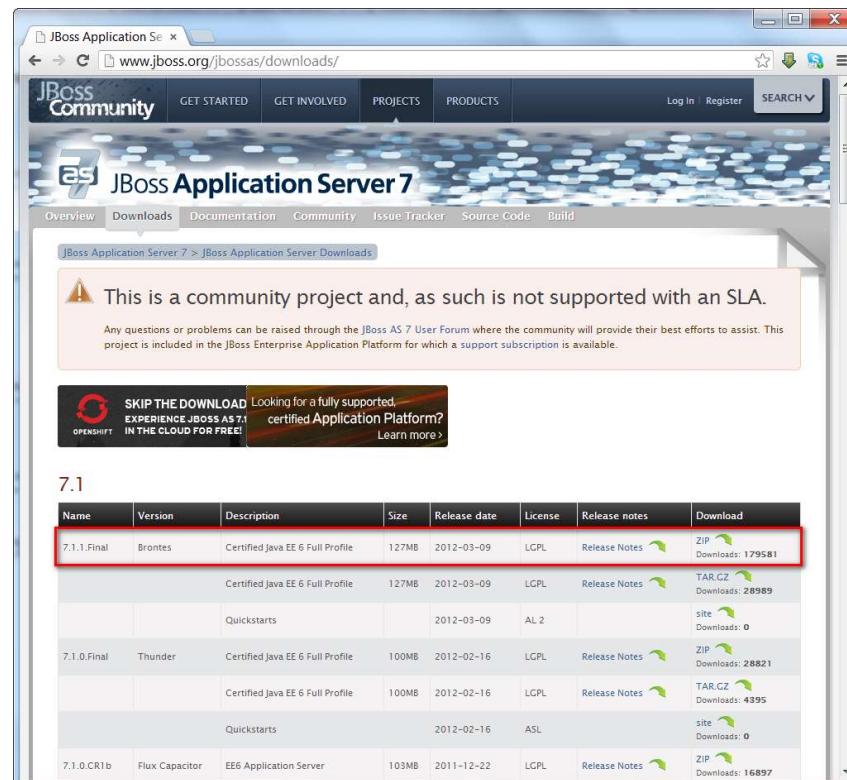
=====
Package           Arch    Version        Repository      Size
=====
Installing:
 java-1.7.0-openjdk-devel x86_64  1:1.7.0.9-2.3.3.2.el6_3  updates   9.4 M
Installing for dependencies:
 java-1.7.0-openjdk     x86_64  1:1.7.0.9-2.3.3.2.el6_3  updates   26 M

Transaction Summary
=====
Install      2 Package(s)

Total size: 35 M
Installed size: 125 M
Is this ok [y/N]: y
```

Instalación del servidor (I)

- Ir a <http://www.jboss.org/jbossas/downloads/>
- JBoss EAP puede descargarse de Red Hat Network
<https://rhn.redhat.com>



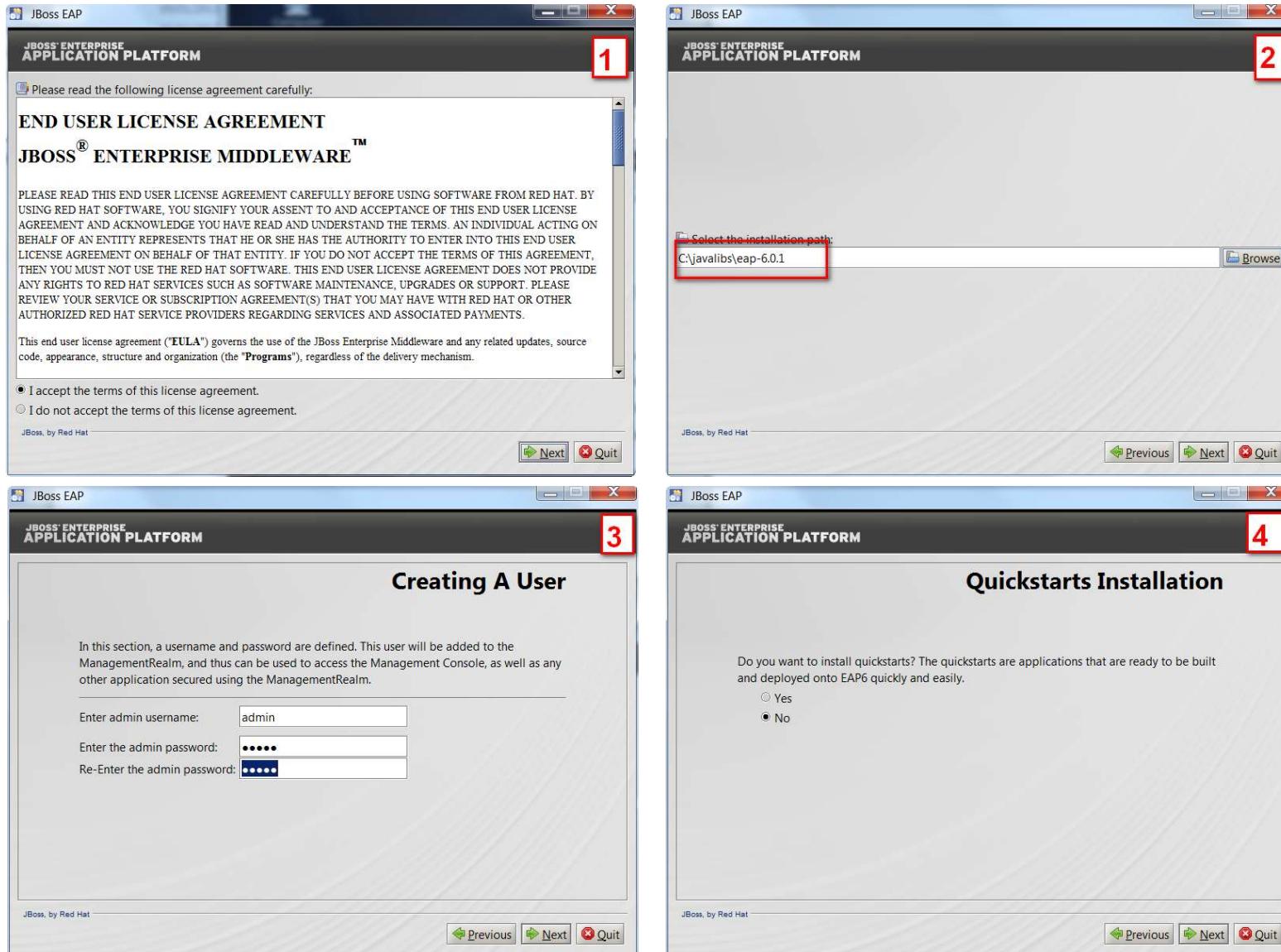
Instalación del servidor (II)

- Extraemos el archivo y lo colocamos en **/usr/local** o **/home**

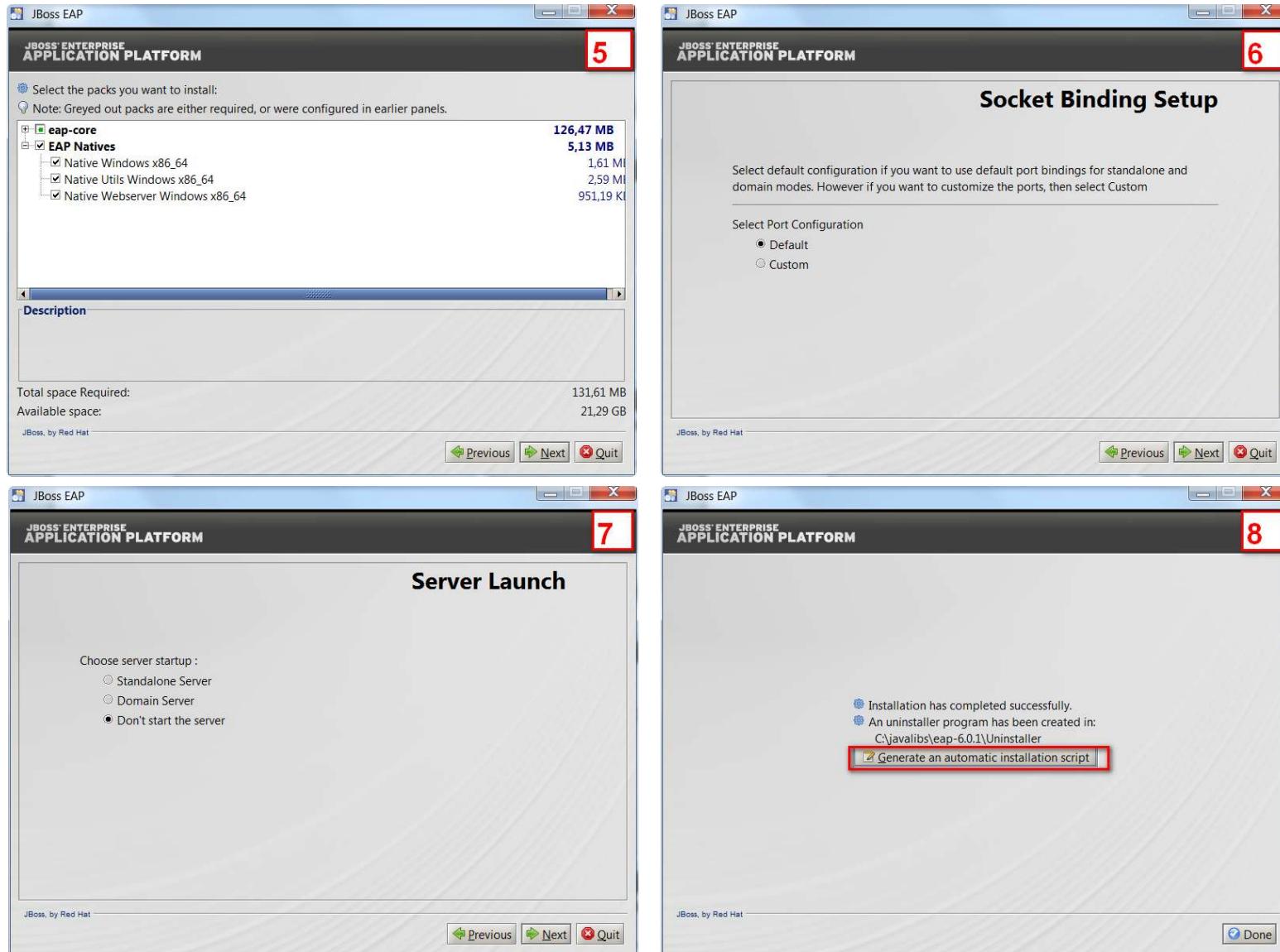
```
# cp jboss-eap-6.1.0.zip /tmp  
# cd /tmp  
# unzip jboss-eap-6.1.0.zip  
# mv jboss-eap-6.1 /usr/local
```

- También JBoss EAP dispone de un instalador (**jboss-eap-6.1.0-installer.jar**) que permite modificar la opciones más frecuentes de una instalación del servidor:
 - Puertos
 - Usuarios administrador inicial
 - Generar scripts para repetir la instalación
 - Instalación silenciosa a partir de las opciones de un fichero XML
- ```
java -jar jboss-eap-6.1.0-installer.jar instalacion-jboss.xml
```

# Instalación del servidor (III)

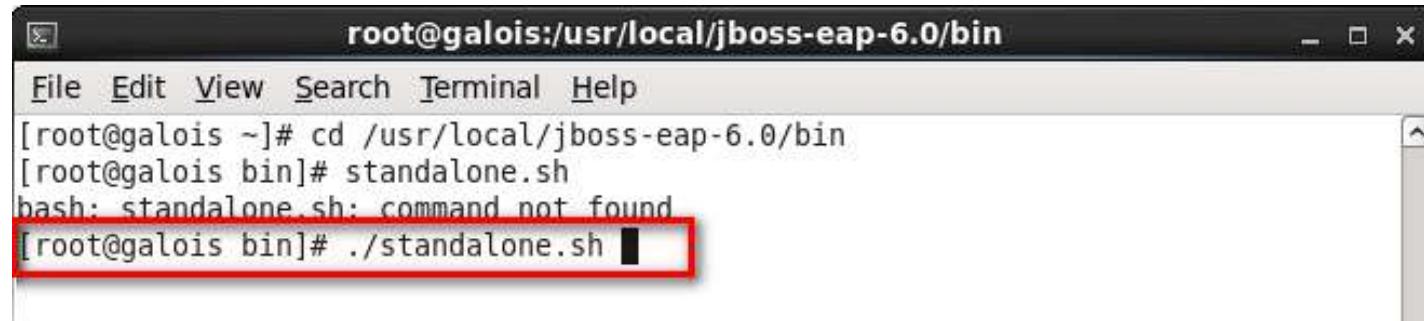


# Instalación del servidor (IV)



# Arrancando el servidor (I)

- Debemos resolver algún problema antes de lanzar el ejecutable **standalone.sh** (variable path)



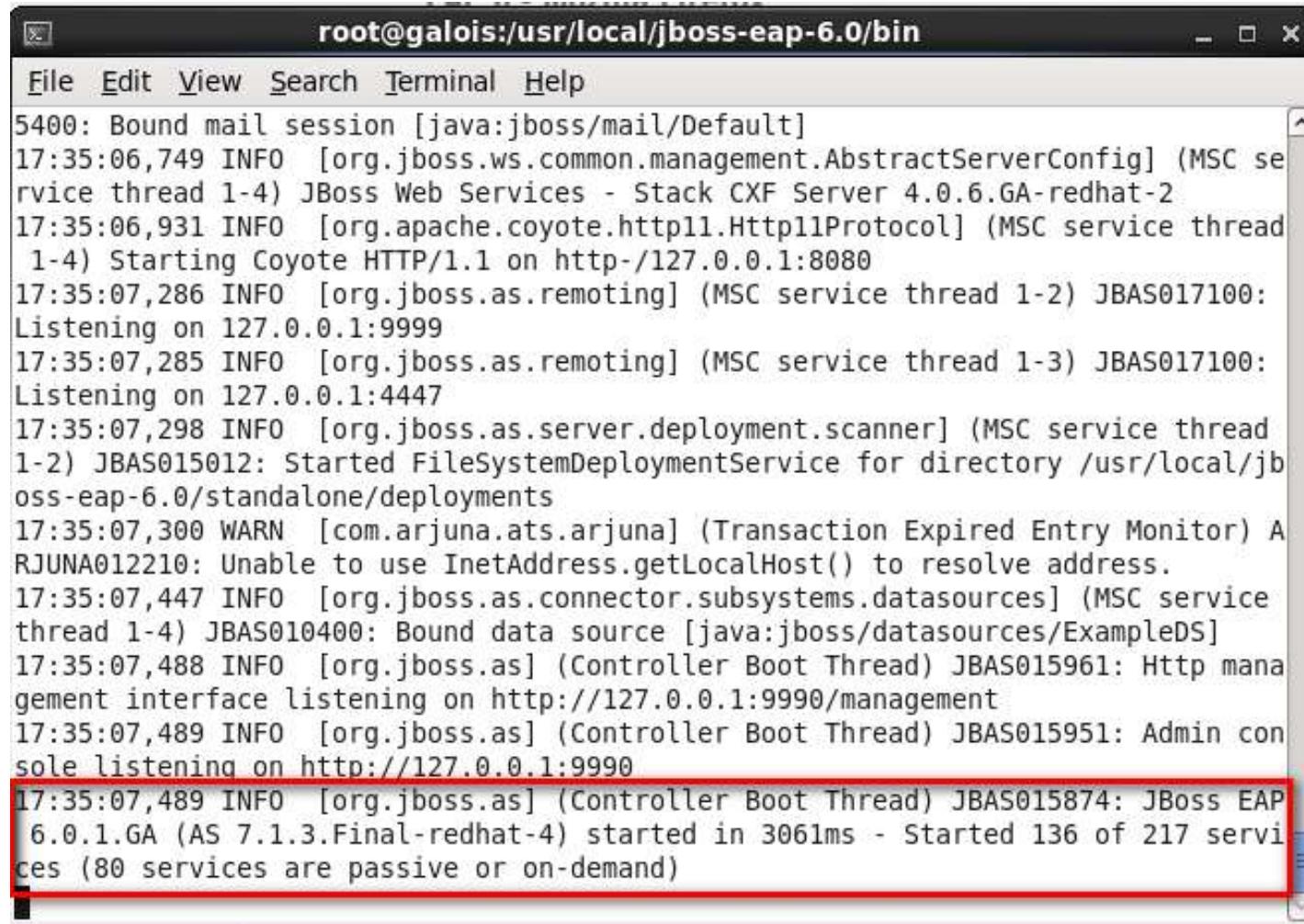
A screenshot of a terminal window titled "root@galois:/usr/local/jboss-eap-6.0/bin". The window has a standard Linux-style interface with a menu bar (File, Edit, View, Search, Terminal, Help) and a scroll bar on the right. The terminal session shows the following commands:

```
root@galois ~]# cd /usr/local/jboss-eap-6.0/bin
[root@galois bin]# standalone.sh
bash: standalone.sh: command not found
[root@galois bin]# ./standalone.sh
```

The last command, `./standalone.sh`, is highlighted with a red rectangular box.

# Arrancando el servidor (II)

- Esperamos al mensaje de servidor arrancado

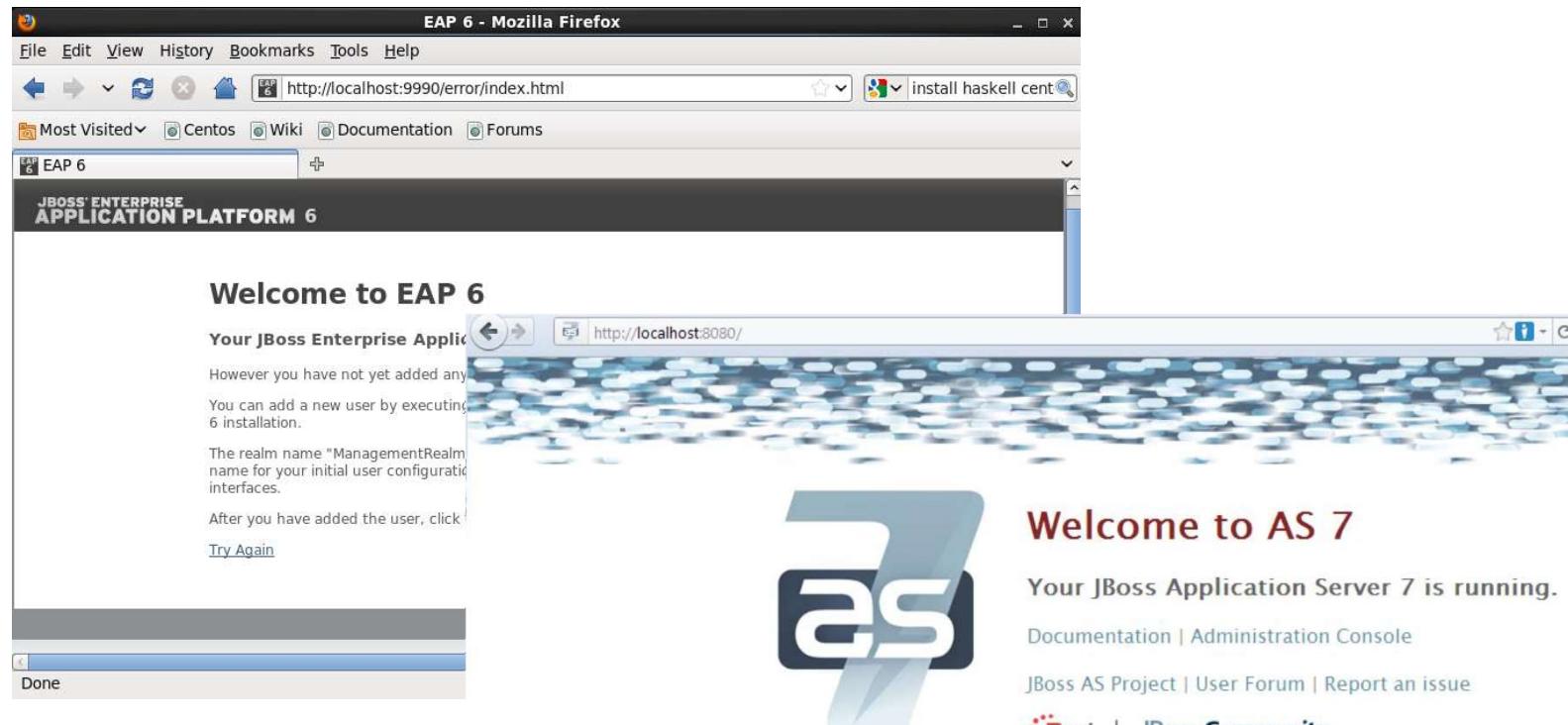


The screenshot shows a terminal window titled "root@galois:/usr/local/jboss-eap-6.0/bin". The window displays the startup logs for JBoss EAP 6.0.1.GA. The logs include various INFO messages from different components like AbstractServerConfig, Http11Protocol, and FileSystemDeploymentService. A specific message at the bottom is highlighted with a red box: "JBoss EAP 6.0.1.GA (AS 7.1.3.Final-redhat-4) started in 3061ms - Started 136 of 217 services (80 services are passive or on-demand)".

```
root@galois:/usr/local/jboss-eap-6.0/bin
File Edit View Search Terminal Help
5400: Bound mail session [java:jboss/mail/Default]
17:35:06,749 INFO [org.jboss.ws.common.management.AbstractServerConfig] (MSC service thread 1-4) JBoss Web Services - Stack CXF Server 4.0.6.GA-redhat-2
17:35:06,931 INFO [org.apache.coyote.http11.Http11Protocol] (MSC service thread 1-4) Starting Coyote HTTP/1.1 on http-/127.0.0.1:8080
17:35:07,286 INFO [org.jboss.as.remoting] (MSC service thread 1-2) JBAS017100: Listening on 127.0.0.1:9999
17:35:07,285 INFO [org.jboss.as.remoting] (MSC service thread 1-3) JBAS017100: Listening on 127.0.0.1:4447
17:35:07,298 INFO [org.jboss.as.server.deployment.scanner] (MSC service thread 1-2) JBAS015012: Started FileSystemDeploymentService for directory /usr/local/jboss-eap-6.0/standalone/deployments
17:35:07,300 WARN [com.arjuna.ats.arjuna] (Transaction Expired Entry Monitor) ARJUNA012210: Unable to use InetAddress.getLocalHost() to resolve address.
17:35:07,447 INFO [org.jboss.as.connector.subsystems.datasources] (MSC service thread 1-4) JBAS010400: Bound data source [java:jboss/datasources/ExampleDS]
17:35:07,488 INFO [org.jboss.as] (Controller Boot Thread) JBAS015961: Http management interface listening on http://127.0.0.1:9990/management
17:35:07,489 INFO [org.jboss.as] (Controller Boot Thread) JBAS015951: Admin console listening on http://127.0.0.1:9990
17:35:07,489 INFO [org.jboss.as] (Controller Boot Thread) JBAS015874: JBoss EAP 6.0.1.GA (AS 7.1.3.Final-redhat-4) started in 3061ms - Started 136 of 217 services (80 services are passive or on-demand)
```

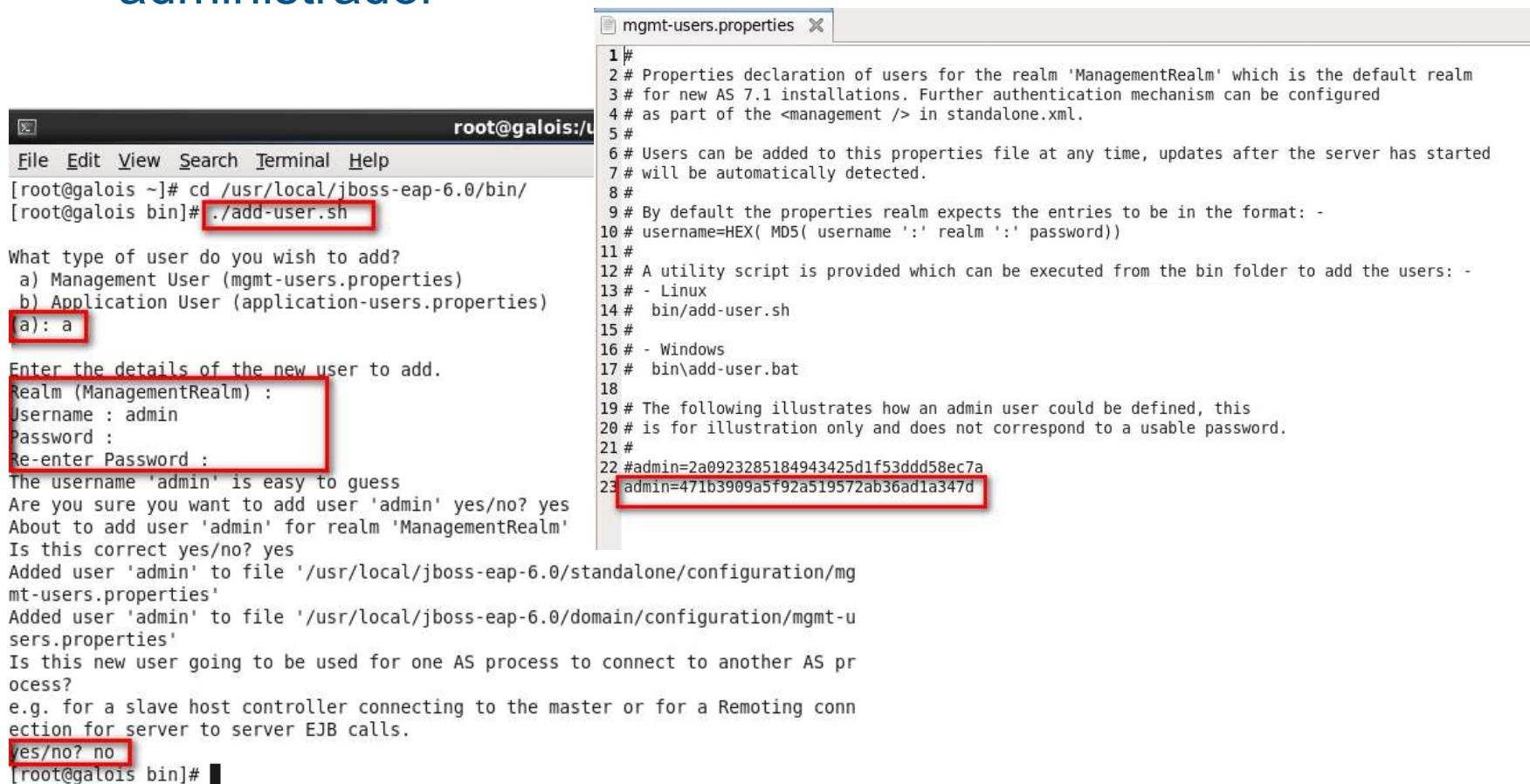
# Arrancando el servidor (III)

- Verificamos que el servidor esta ejecutándose entrando en <http://localhost:8080> o en <http://localhost:9990> (consola administrativa)



# Añadir un usuario administrador

- Ejecutamos el script **add-user** para crear un usuario administrador



The screenshot shows a terminal window titled "root@galois:/". The user runs the command `./add-user.sh`. The script prompts for the type of user (Management User), realm (ManagementRealm), and password information. It then adds the user to the properties files and asks if the user wants to be used for remoting. The terminal output is as follows:

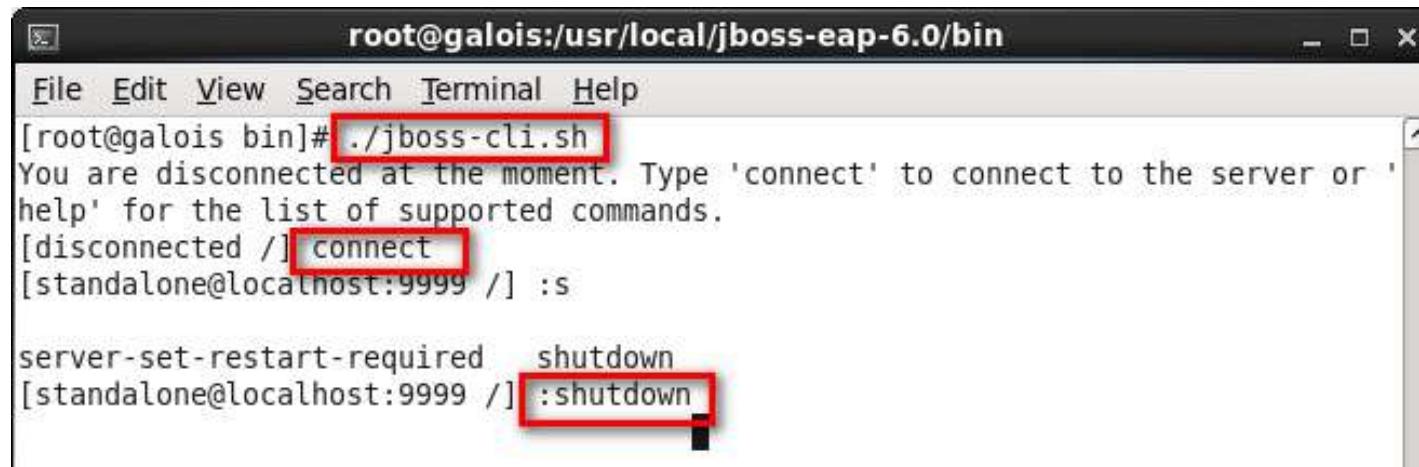
```
[root@galois ~]# cd /usr/local/jboss-eap-6.0/bin/
[root@galois bin]# ./add-user.sh
What type of user do you wish to add?
 a) Management User (mgmt-users.properties)
 b) Application User (application-users.properties)
(a): a
Enter the details of the new user to add.
Realm (ManagementRealm) :
Username : admin
Password :
Re-enter Password :
The username 'admin' is easy to guess
Are you sure you want to add user 'admin' yes/no? yes
About to add user 'admin' for realm 'ManagementRealm'
Is this correct yes/no? yes
Added user 'admin' to file '/usr/local/jboss-eap-6.0/standalone/configuration/mgmt-users.properties'
Added user 'admin' to file '/usr/local/jboss-eap-6.0/domain/configuration/mgmt-users.properties'
Is this new user going to be used for one AS process to connect to another AS process?
e.g. for a slave host controller connecting to the master or for a Remoting connection for server to server EJB calls.
yes/no? no
[root@galois bin]#
```

Simultaneously, a properties file titled "mgmt-users.properties" is shown in a separate window. The file contains the following entries, with the last two lines highlighted by a red box:

```
Properties declaration of users for the realm 'ManagementRealm' which is the default realm
for new AS 7.1 installations. Further authentication mechanism can be configured
as part of the <management /> in standalone.xml.
#
Users can be added to this properties file at any time, updates after the server has started
will be automatically detected.
#
By default the properties realm expects the entries to be in the format: -
username=HEX(MD5(username ':' realm ':' password))
#
A utility script is provided which can be executed from the bin folder to add the users: -
- Linux
bin/add-user.sh
#
- Windows
bin\add-user.bat
#
The following illustrates how an admin user could be defined, this
is for illustration only and does not correspond to a usable password.
#
#admin=2a0923285184943425d1f53ddd58ec7a
#admin=471b3909a5f92a519572ab36ad1a347d
```

# Parando el servidor (I)

- Utilizamos la interfaz de comandos **jboss-cli.sh** o **Ctrl+c** sobre la ventana de la consola del servidor



A screenshot of a terminal window titled "root@galois:/usr/local/jboss-eap-6.0/bin". The window shows the following command sequence:

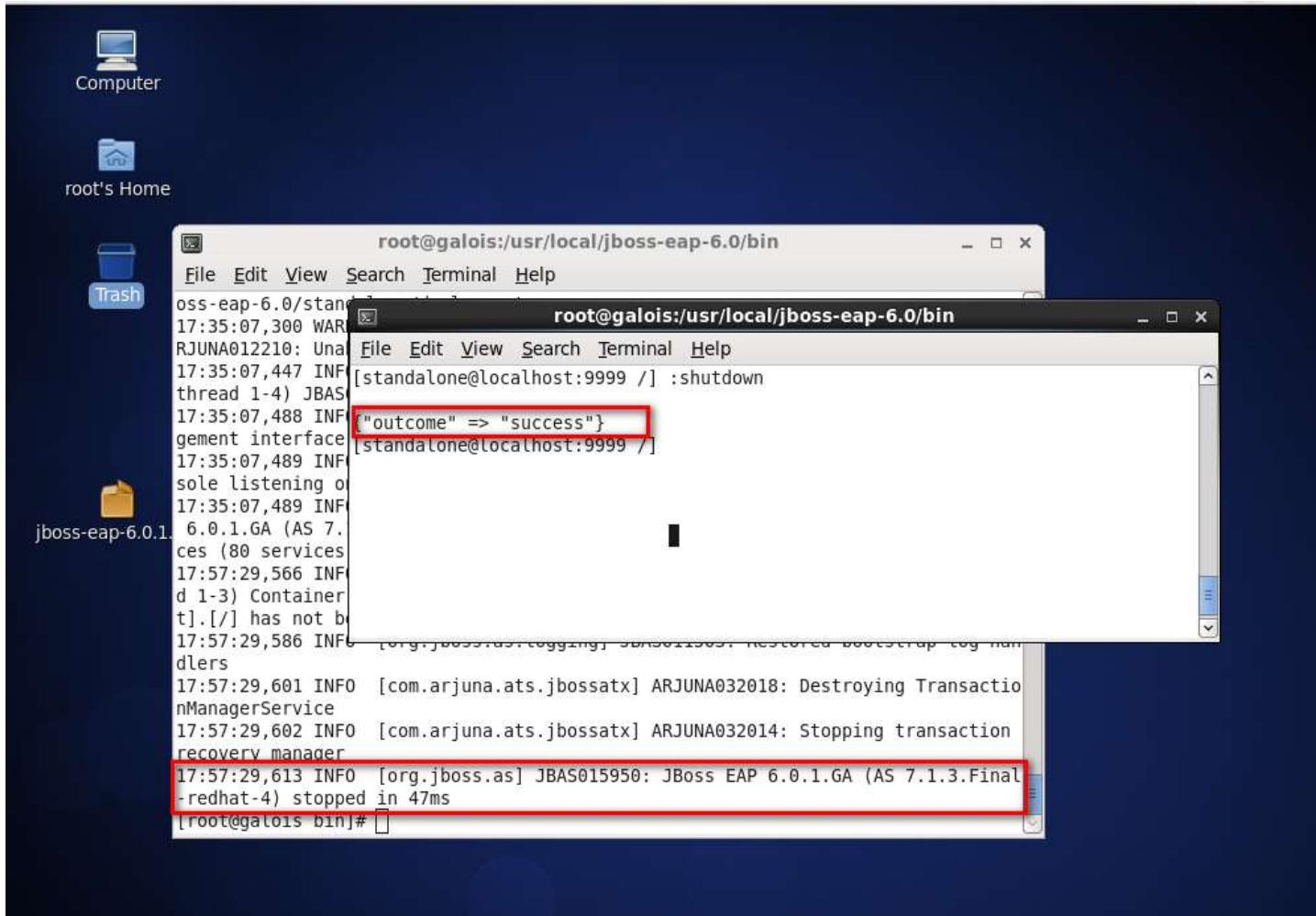
```
root@galois bin]# ./jboss-cli.sh
You are disconnected at the moment. Type 'connect' to connect to the server or 'help' for the list of supported commands.
[disconnected /] connect
[standalone@localhost:9999 /] :shutdown
```

The command `./jboss-cli.sh` is highlighted with a red box. The command `connect` and the parameter `:shutdown` in the final command are also highlighted with red boxes.

- De otra forma:  
**jboss-cli.sh --connect command=:shutdown**

# Parando el servidor (II)

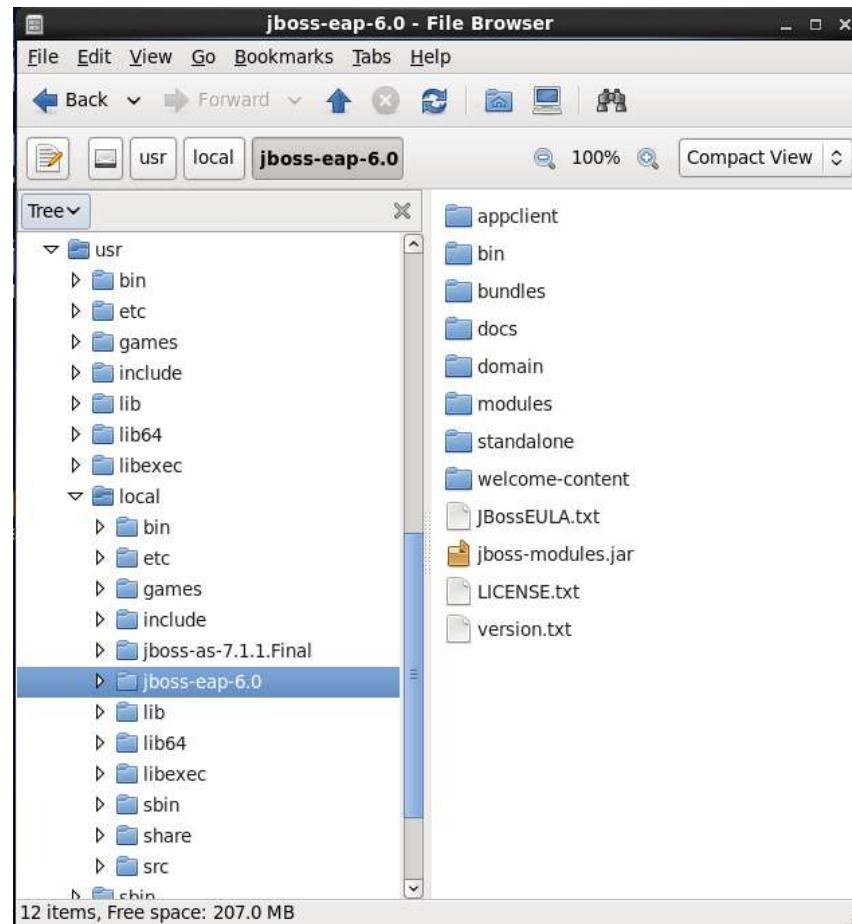
- Esperamos hasta que aparezca el mensaje de parada



```
root@galois:/usr/local/jboss-eap-6.0/bin
File Edit View Search Terminal Help
oss-eap-6.0/standalone/shutdown.sh
17:35:07,300 WARN [org.jboss.as] ARJUNA012210: Una
RJUNA012210: Una
17:35:07,447 INFO [standalone@localhost:9999 /] :shutdown
thread 1-4) JBAS
17:35:07,488 INFO ["outcome" => "success"]
gement interface
17:35:07,489 INFO [standalone@localhost:9999 /]
sole listening o
17:35:07,489 INFO
6.0.1.GA (AS 7.
ces (80 services
17:57:29,566 INF
d 1-3) Container
t].[/] has not b
17:57:29,586 INF
dlers
17:57:29,601 INFO [com.arjuna.ats.jbosstablet] ARJUNA032018: Destroying TransactionManagerService
17:57:29,602 INFO [com.arjuna.ats.jbosstablet] ARJUNA032014: Stopping transaction recovery manager
17:57:29,613 INFO [org.jboss.as] JBAS015950: JBoss EAP 6.0.1.GA (AS 7.1.3.Final-redhat-4) stopped in 47ms
[root@galois bin]#
```

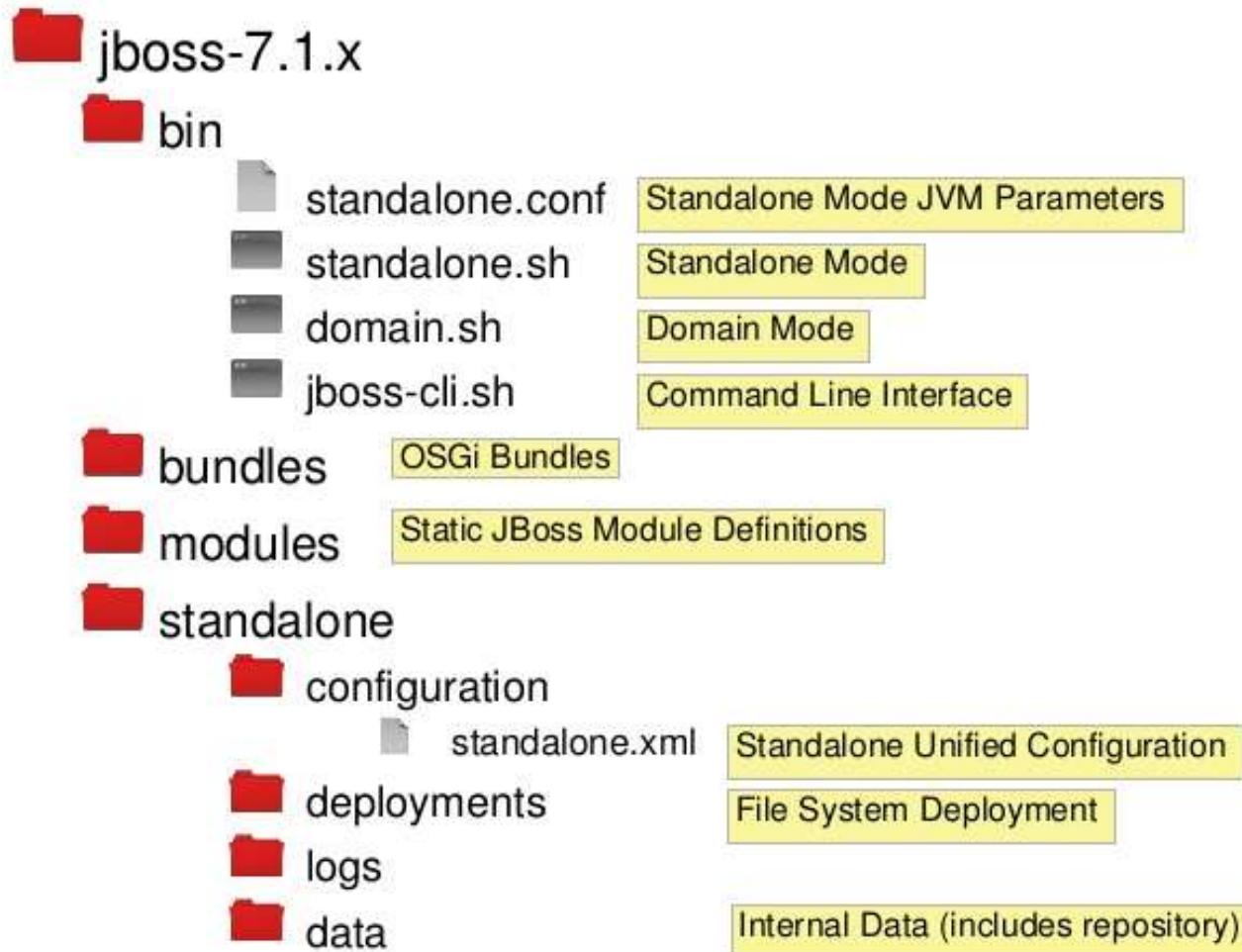
# Estructura de directorios y ficheros (I)

- Toda la configuración y uso de JBoss se realiza a partir de la carpeta **\$JBOSS\_HOME**. Dentro de esta jerarquía es importante conocer las carpetas y archivos más importantes

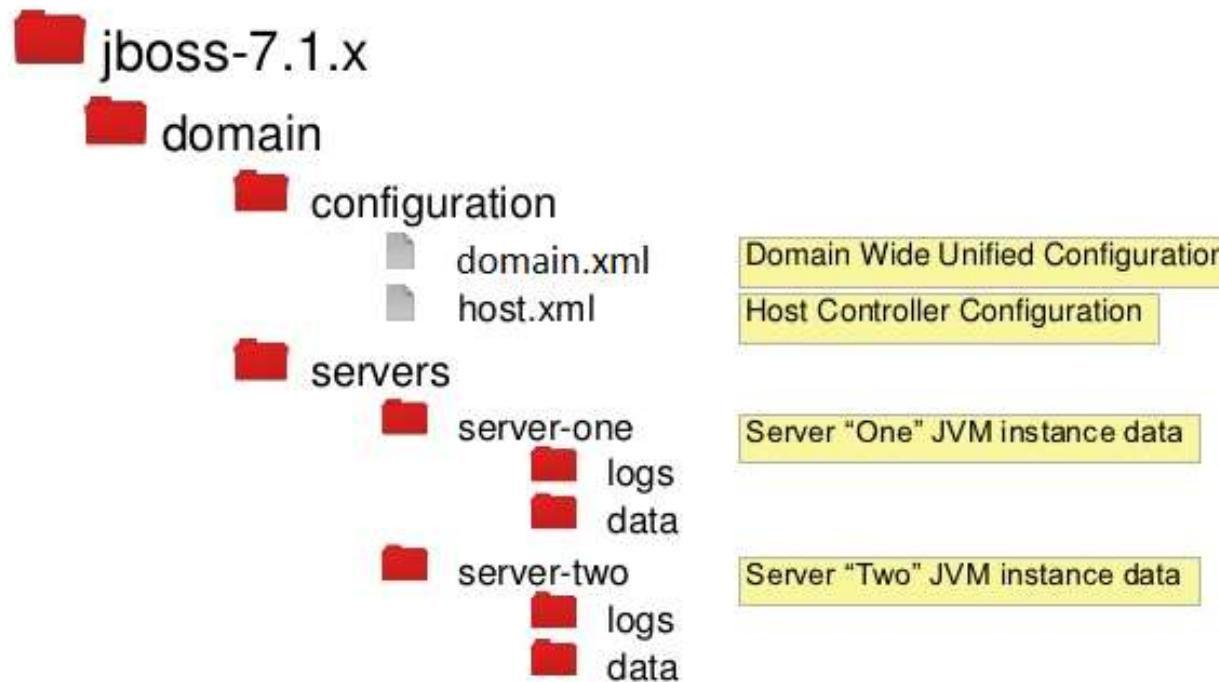


# Estructura de directorios y ficheros (II)

## Filesystem Layout



## Filesystem Layout (cont)



# Configuraciones de arranque (I)

- La configuración del servidor **standalone/configuration/standalone.xml** se puede **modificar / ampliar / reducir**, es recomendable copiar a un fichero aparte (*standalone-custom.xml*)
- El servidor ya tiene unas configuraciones de servicios preparadas
  - **standalone.xml** => configuración de los servicios por defecto
  - **standalone-ha.xml** => configuración por defecto + servicios de alta disponibilidad
  - **standalone-full.xml** => configuración de todos los servicios (JMS- HornetQ, EJB 2.x, CORBA, etc.)
  - **standalone-full-ha.xml** => configuración de todos los servicios + servicios de alta disponibilidad
- Estas configuraciones pueden reflejar los distintos **escenarios de uso** de JBoss, desarrollo, integración, preproducción, etc... con distintos servicios habilitados/deshabilitados para ellos (por ejemplo, logging)
- El propio servidor tras arrancar mantiene un historial de cambios en los archivos de configuración (versiones o snapshots) en los directorios (**También se pueden tomar snapshots desde jboss-cli**):
  - **standalone\_xml\_history**, **domain\_xml\_history** y **host\_xml\_history**

# Configuraciones de arranque (II)

- El archivo **standalone.initial.xml** contiene el original del archivo de configuración. Este archivo no se sobrescribe nunca por el servidor.
  - Si necesita restaurar la configuración inicial, no se deshaga de la instalación del servidor de aplicaciones! Basta con sustituir el archivo **standalone.xml** con **standalone\_xml\_history/standalone.initial.xml**
- El archivo de **standalone.boot.xml** contiene la configuración AS que se utilizó para el último inicio correcto del servidor. Este archivo se sobreescribe cada vez que arranquemos el servidor con éxito.
  - Para deshacer todos los cambios en la sesión actual, basta con sustituir **standalone.xml** con **standalone\_xml\_history/standalone.boot.xml**
- Por último, el **standalone.last.xml** archivo contiene la última configuración correcta, acometida en el servidor de aplicaciones.
- Cada vez que reinicie el servidor de aplicaciones, una copia de su configuración de arranque con éxito se almacena en la carpeta de **snapshot** mediante el formato **AAAAMMDD-HHMMMSstandalone.xml**.
- La carpeta **current** se utiliza como carpeta temporal para almacenar los cambios en la configuración de la sesión actual. Cada cambio en la configuración se traducirá en un archivo llamado **standalone.v [n].xml**. Donde n es el número de cambio. Cuando se reinicie el servidor de aplicaciones, estos archivos se mudaron a una carpeta con la fecha en la carpeta superior **standalone\_xml\_history**.

# Configuraciones de arranque (III)

- Para arrancar una configuración determinada hay que usar el comando:

```
$JBOSS_HOME/bin/standalone.bat --server-config=standalone-xxx.xml
```

- Para arrancar con una interfaz de red determinada (servicios públicos):

```
$JBOSS_HOME/bin/standalone.bat -b 192.168.10.1
```

- Para arrancar con una interfaz de red determinada (servicios administrador):

```
$JBOSS_HOME/bin/standalone.bat -bmanagement 192.168.10.1
```

- Otras opciones (-D):

- **-Djboss.socket.binding.port-offset=XXXX**
- **-Djboss.server.base.dir=YYYY**
- ...

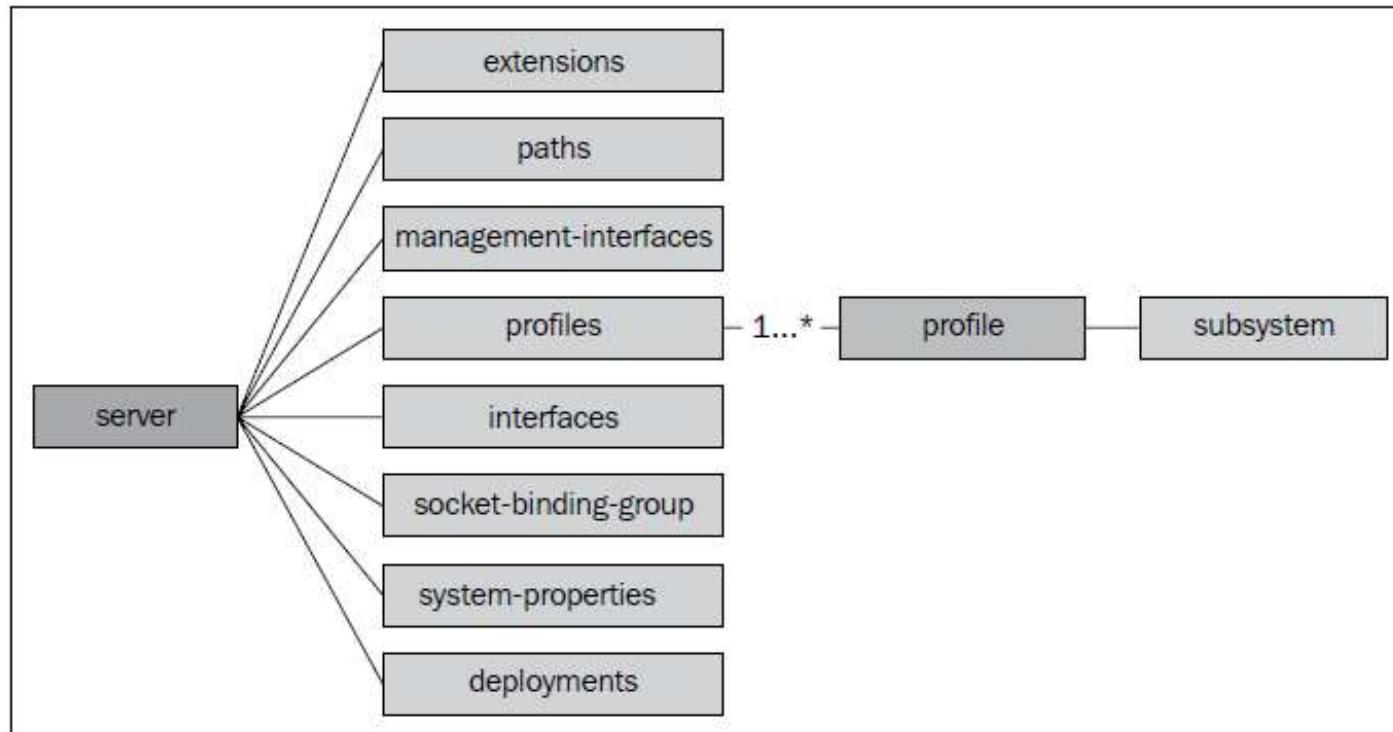
# Contenidos

---

- I. Introducción
- II. Instalación del servidor de aplicaciones
- III. Configuración y herramientas de administración**
- IV. Despliegue de aplicaciones
- V. Seguridad de las aplicaciones
- VI. Clustering
- VII. Optimización

# Estructura del fichero de configuración (I)

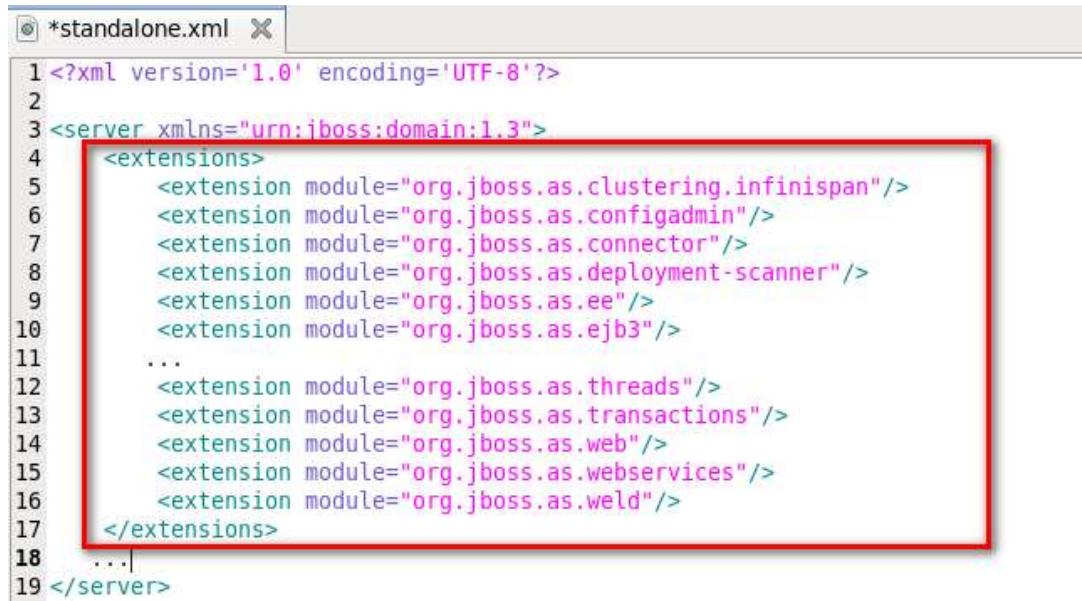
- ⌘ La configuración del servidor de aplicaciones se mantiene en un solo archivo, que actúa como una referencia principal para todas las configuraciones del servidor
- ⌘ Tanto el fichero ***standalone.xml*** y sus variantes, así como el fichero ***domain.xml*** tienen el mismo esquema xml (.xsd) para su definición:  
**\$JBoss\_HOME/docs/schema/jboss-as-config\_1\_4.xsd**



# Estructura del fichero de configuración (II)

## ⌘ Sección **Extensions**

- ❖ El servidor de aplicaciones contiene una lista de los módulos básicos, llamados **extensiones**, que son compartidos por todos sus servicios
- ❖ Las extensiones pueden ser vistas como un tipo especial de módulo que se utilizan para ampliar las funcionalidades del servidor de aplicaciones
- ❖ Al igual que los módulos estándar, se almacenan en la carpeta **JBOSS\_HOME/modules**
- ❖ Cada extensión es cargada por el cargador de clases en el momento de arranque del servidor, antes de cualquier despliegue



```
*standalone.xml X
1 <?xml version='1.0' encoding='UTF-8'?>
2
3 <server xmlns="urn:jboss:domain:1.3">
4 <extensions>
5 <extension module="org.jboss.as.clustering.infinispan"/>
6 <extension module="org.jboss.as.configadmin"/>
7 <extension module="org.jboss.as.connector"/>
8 <extension module="org.jboss.as.deployment-scanner"/>
9 <extension module="org.jboss.as.ee"/>
10 <extension module="org.jboss.as.ejb3"/>
11 ...
12 <extension module="org.jboss.as.threads"/>
13 <extension module="org.jboss.as.transactions"/>
14 <extension module="org.jboss.as.web"/>
15 <extension module="org.jboss.as.webservices"/>
16 <extension module="org.jboss.as.weld"/>
17 </extensions>
18 ...
19 </server>
```

# Estructura del fichero de configuración (III)

## ⌘ Sección Paths

↳ Debajo de las extensiones, se puede encontrar la definición de rutas (**elemento opcional**)

↳ Son nombres lógicos para las rutas del sistema de archivos

↳ Ejemplo definición

```
<paths>
 <path name="log.dir" path="mylogdir"
 relative-to="jboss.server.log.dir"/>
</paths>
```

↳ Ejemplo uso

```
<periodic-rotating-file-handler name="FILE" autoflush="true">
 <file relative-to="log.dir" path="myserver.log"/>
</periodic-rotating-file-handler>
```

↳ Los paths pueden ser absolutos ó relativos

# Estructura del fichero de configuración (IV)

## ⌘ Sección Paths

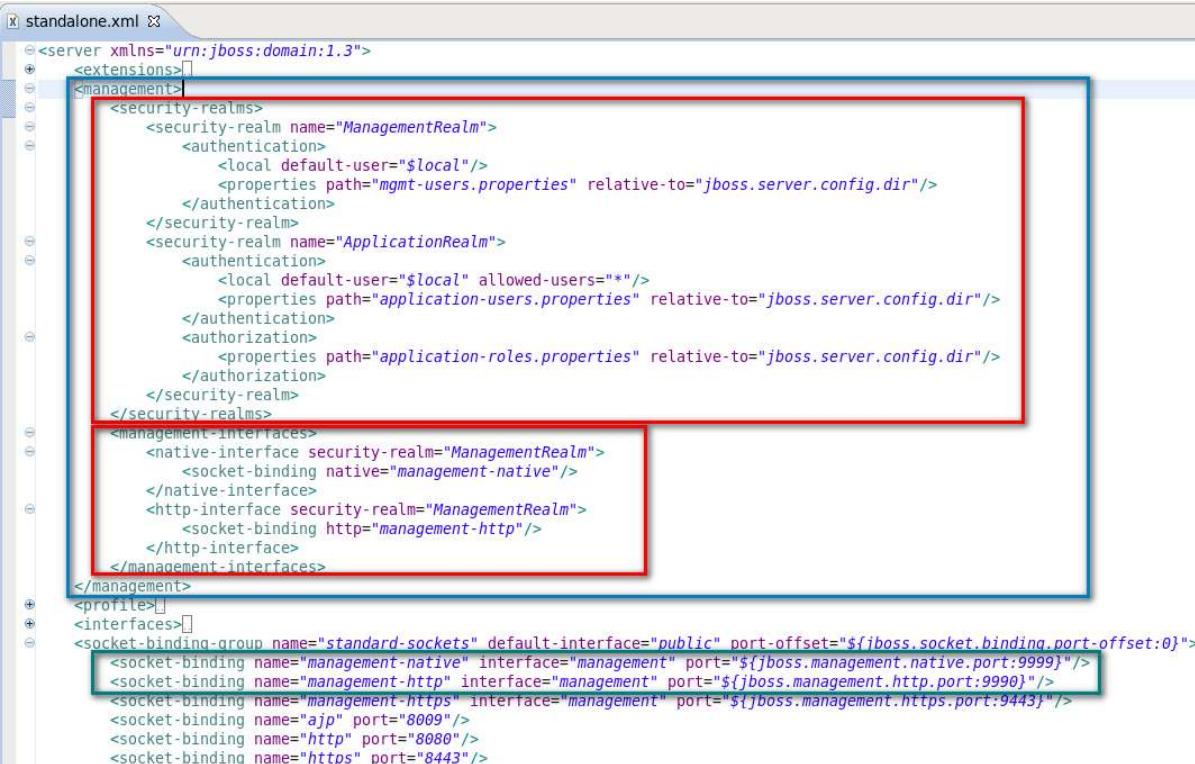
Por defecto, existen los siguientes paths de manera implícita y no pueden ser redefinidos (se usan para declarar rutas relativas):

Ruta	Significado
<b>jboss.home</b>	Directorio raíz de la instalación JBoss AS
<b>user.home</b>	Directorio del usuario
<b>user.dir</b>	Directorio de trabajo actual del usuario
<b>java.home</b>	Directorio de instalación de Java
<b>jboss.server.base.dir</b>	Directorio raíz de una instancia de servidor individual
<b>jboss.server.data.dir</b>	Directorio que el servidor utilizará para el almacenamiento persistente de datos
<b>jboss.server.log.dir</b>	Directorio que el servidor utilizará para el almacenamiento de archivo de trazas
<b>jboss.server.tmp.dir</b>	Directorio que el servidor utilizará para almacenar archivos temporales
<b>jboss.domain.servers.dir</b>	Directorio en el que un controlador de host creará el área de trabajo para instancias de servidor individuales

# Estructura del fichero de configuración (V)

## ❖ Sección Management

- ❖ Esta sección comprende la configuración de administración del servidor
- ❖ Esta dividida en dos subsecciones:
  - ❖ **Security-realms** donde se listan los diferentes reinos de seguridad
  - ❖ **Management-interfaces** donde se detallas los distintos canales usados por las herramientas de administración (web console y jboss.cli)

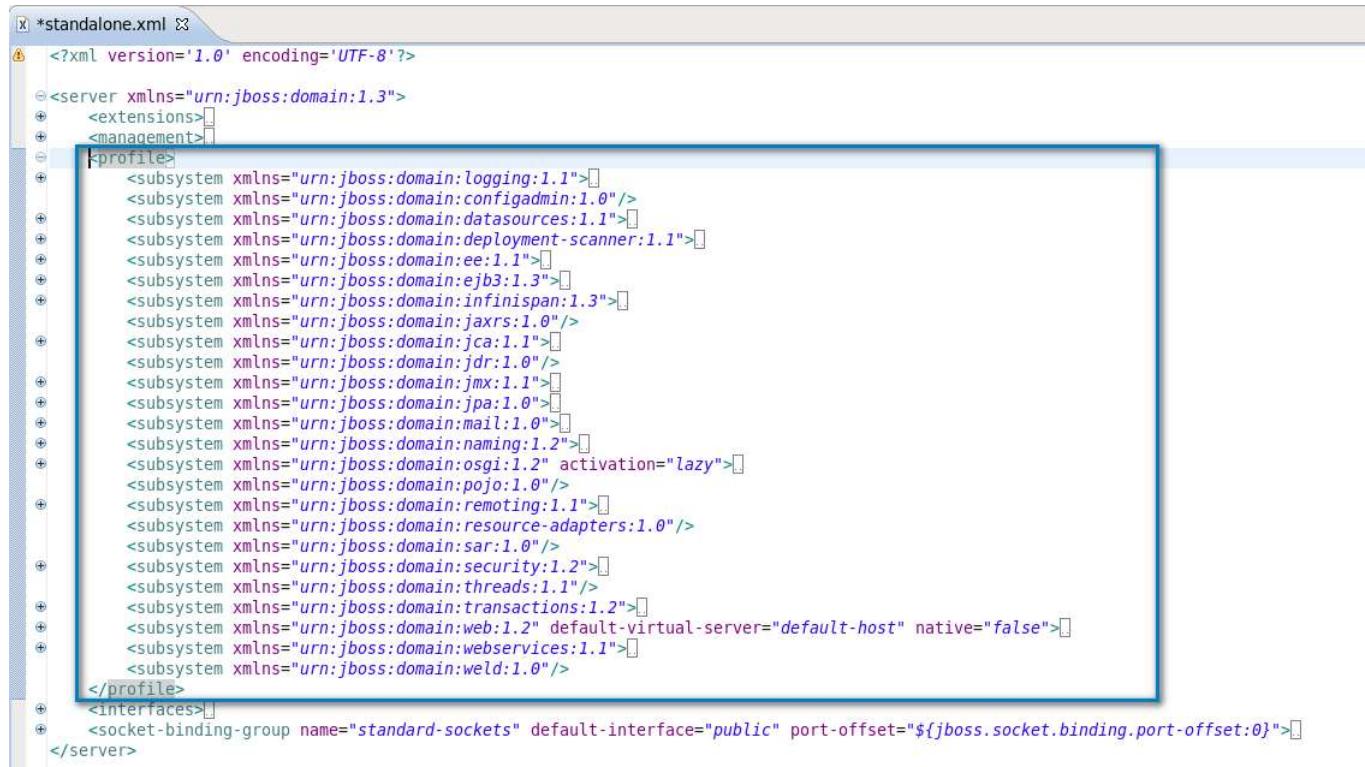


```
<server xmlns="urn:jboss:domain:1.3">
 <extensions></extensions>
 <management>
 <security-realms>
 <security-realm name="ManagementRealm">
 <authentication>
 <local default-user="$local"/>
 <properties path="mgmt-users.properties" relative-to="jboss.server.config.dir"/>
 </authentication>
 </security-realm>
 <security-realm name="ApplicationRealm">
 <authentication>
 <local default-user="$local" allowed-users="*"/>
 <properties path="application-users.properties" relative-to="jboss.server.config.dir"/>
 </authentication>
 <authorization>
 <properties path="application-roles.properties" relative-to="jboss.server.config.dir"/>
 </authorization>
 </security-realm>
 </security-realms>
 <management-interfaces>
 <native-interface security-realm="ManagementRealm">
 <socket-binding native="management-native"/>
 </native-interface>
 <http-interface security-realm="ManagementRealm">
 <socket-binding http="management-http"/>
 </http-interface>
 </management-interfaces>
 </management>
 <profile></profile>
 <interfaces></interfaces>
 <socket-binding-group name="standard-sockets" default-interface="public" port-offset="${jboss.socket.binding.port-offset:0}">
 <socket-binding name="management-native" interface="management" port="${jboss.management.native.port:9999}"/>
 <socket-binding name="management-http" interface="management" port="${jboss.management.http.port:9990}"/>
 <socket-binding name="management-https" interface="management" port="${jboss.management.https.port:9443}"/>
 <socket-binding name="ajp" port="8009"/>
 <socket-binding name="http" port="8080"/>
 <socket-binding name="https" port="8443"/>
 </socket-binding-group>
```

# Estructura del fichero de configuración (VI)

## ⌘ Sección Profile (Profiles)

- ❖ Un perfil puede ser visto como un conjunto de subsistemas
- ❖ Cada subsistema a su vez contiene un subconjunto de funcionalidades utilizadas por el servidor de aplicaciones
- ❖ Cada subsistema se configura de manera independiente de acuerdo a una gramática



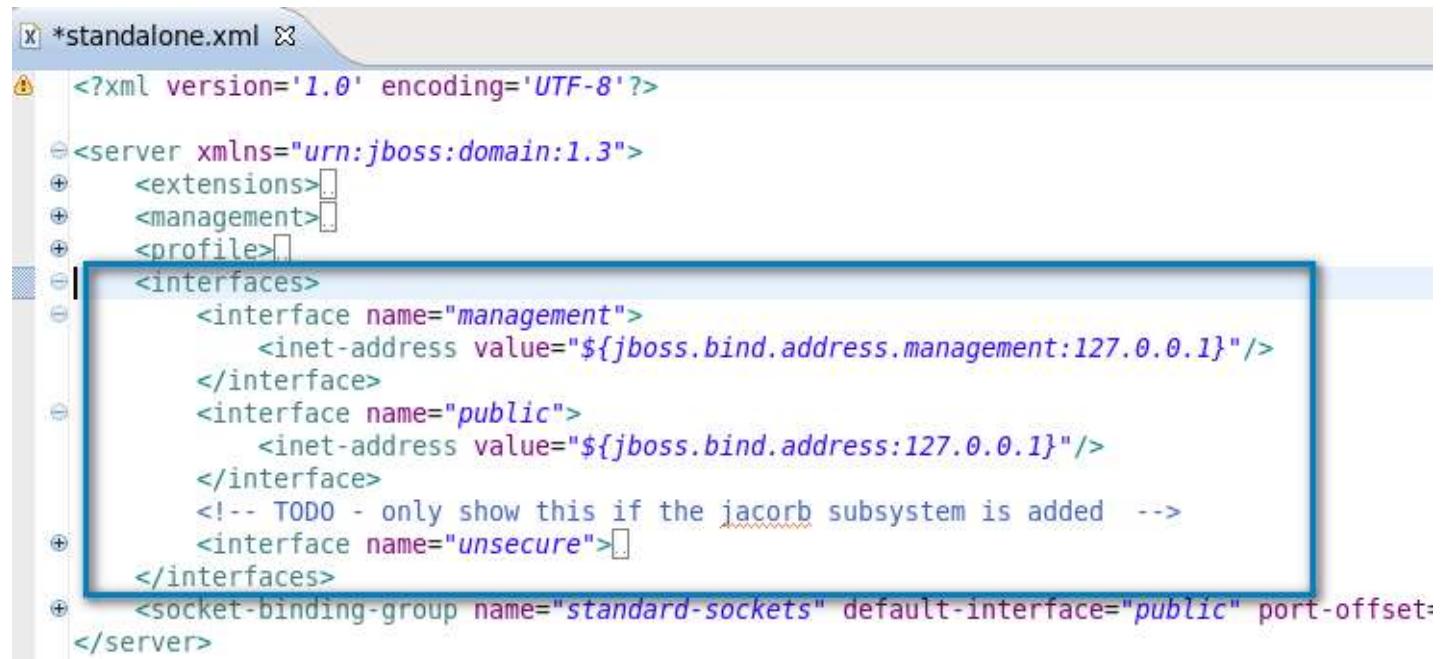
```
<?xml version='1.0' encoding='UTF-8'?>

<server xmlns="urn:jboss:domain:1.3">
 <extensions>
 <management>
 <profile>
 <subsystem xmlns="urn:jboss:domain:logging:1.1">
 <subsystem xmlns="urn:jboss:domain:configadmin:1.0"/>
 <subsystem xmlns="urn:jboss:domain:datasources:1.1">
 <subsystem xmlns="urn:jboss:domain:deployment-scanner:1.1">
 <subsystem xmlns="urn:jboss:domain:ee:1.1">
 <subsystem xmlns="urn:jboss:domain:ejb3:1.3">
 <subsystem xmlns="urn:jboss:domain:infinispan:1.3">
 <subsystem xmlns="urn:jboss:domain:jaxrs:1.0"/>
 <subsystem xmlns="urn:jboss:domain:jca:1.1">
 <subsystem xmlns="urn:jboss:domain:jdr:1.0"/>
 <subsystem xmlns="urn:jboss:domain:jmx:1.1">
 <subsystem xmlns="urn:jboss:domain:jpa:1.0">
 <subsystem xmlns="urn:jboss:domain:mail:1.0"/>
 <subsystem xmlns="urn:jboss:domain:naming:1.2"/>
 <subsystem xmlns="urn:jboss:domain:osgi:1.2" activation="lazy">
 <subsystem xmlns="urn:jboss:domain:pojo:1.0"/>
 <subsystem xmlns="urn:jboss:domain:remoting:1.1">
 <subsystem xmlns="urn:jboss:domain:resource-adapters:1.0"/>
 <subsystem xmlns="urn:jboss:domain:sar:1.0"/>
 <subsystem xmlns="urn:jboss:domain:security:1.2"/>
 <subsystem xmlns="urn:jboss:domain:threads:1.1"/>
 <subsystem xmlns="urn:jboss:domain:transactions:1.2">
 <subsystem xmlns="urn:jboss:domain:web:1.2" default-virtual-server="default-host" native="false">
 <subsystem xmlns="urn:jboss:domain:webservices:1.1"/>
 <subsystem xmlns="urn:jboss:domain:weld:1.0"/>
 </subsystem>
 </profile>
 <interfaces>
 <socket-binding-group name="standard-sockets" default-interface="public" port-offset="${jboss.socket.binding.port-offset:0}">
 </interfaces>
 </management>
 </server>
```

# Estructura del fichero de configuración (VII)

## ⌘ Sección Interfaces

- ❖ Contiene los nombres de las interfaces de red / direcciones IP o nombres de host donde el servidor de aplicaciones se puede enlazar
- ❖ De forma predeterminada, el servidor de aplicaciones standalone define dos interfaces diferentes (redes): la interfaz de gestión y la interfaz pública



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

<server xmlns="urn:jboss:domain:1.3">
 <extensions>...</extensions>
 <management>...</management>
 <profile>...</profile>
 <interfaces>
 <interface name="management">
 <inet-address value="${jboss.bind.address.management:127.0.0.1}" />
 </interface>
 <interface name="public">
 <inet-address value="${jboss.bind.address:127.0.0.1}" />
 </interface>
 <!-- TODO - only show this if the jacob subsystem is added -->
 <interface name="unsecure">...</interface>
 </interfaces>
 <socket-binding-group name="standard-sockets" default-interface="public" port-offset=...
</server>
```

# Estructura del fichero de configuración (VIII)

## ※ Sección **Socket Binding Groups**

- ❖ Un **Socket Binding** constituye una configuración con nombre de una puerta de conexión
- ❖ Sirven para configurar los puertos de red que serán abiertos para escuchar conexiones entrantes
- ❖ Usan las interfaces definidas en la sección interfaces (**public** y **management**)



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

<server xmlns="urn:jboss:domain:1.3">
 <extensions>
 <management>
 <profile>
 <interfaces>
 <socket-binding-group name="standard-sockets" default-interface="public" port-offset="${jboss.socket.binding.port-offset:0}">
 <socket-binding name="management-native" interface="management" port="${jboss.management.native.port:9999}"/>
 <socket-binding name="management-http" interface="management" port="${jboss.management.http.port:9990}"/>
 <socket-binding name="management-https" interface="management" port="${jboss.management.https.port:9443}"/>
 <socket-binding name="ajp" port="8009"/>
 <socket-binding name="http" port="8080"/>
 <socket-binding name="https" port="8443"/>
 <socket-binding name="osgi-http" interface="management" port="8090"/>
 <socket-binding name="remoting" port="4447"/>
 <socket-binding name="txn-recovery-environment" port="4712"/>
 <socket-binding name="txn-status-manager" port="4713"/>
 <outbound-socket-binding name="mail-smtp">
 <remote-destination host="localhost" port="25"/>
 </outbound-socket-binding>
 </socket-binding-group>
 </interfaces>
 </profile>
 </management>
 </extensions>
</server>
```

# Estructura del fichero de configuración (IX)

## ⌘ Sección **System Properties**

- ❖ Esta sección contiene un grupo de propiedades en todo el sistema, que se pueden añadir a al servidor de aplicaciones como parte del proceso de arranque
- ❖ Ejemplo de uso

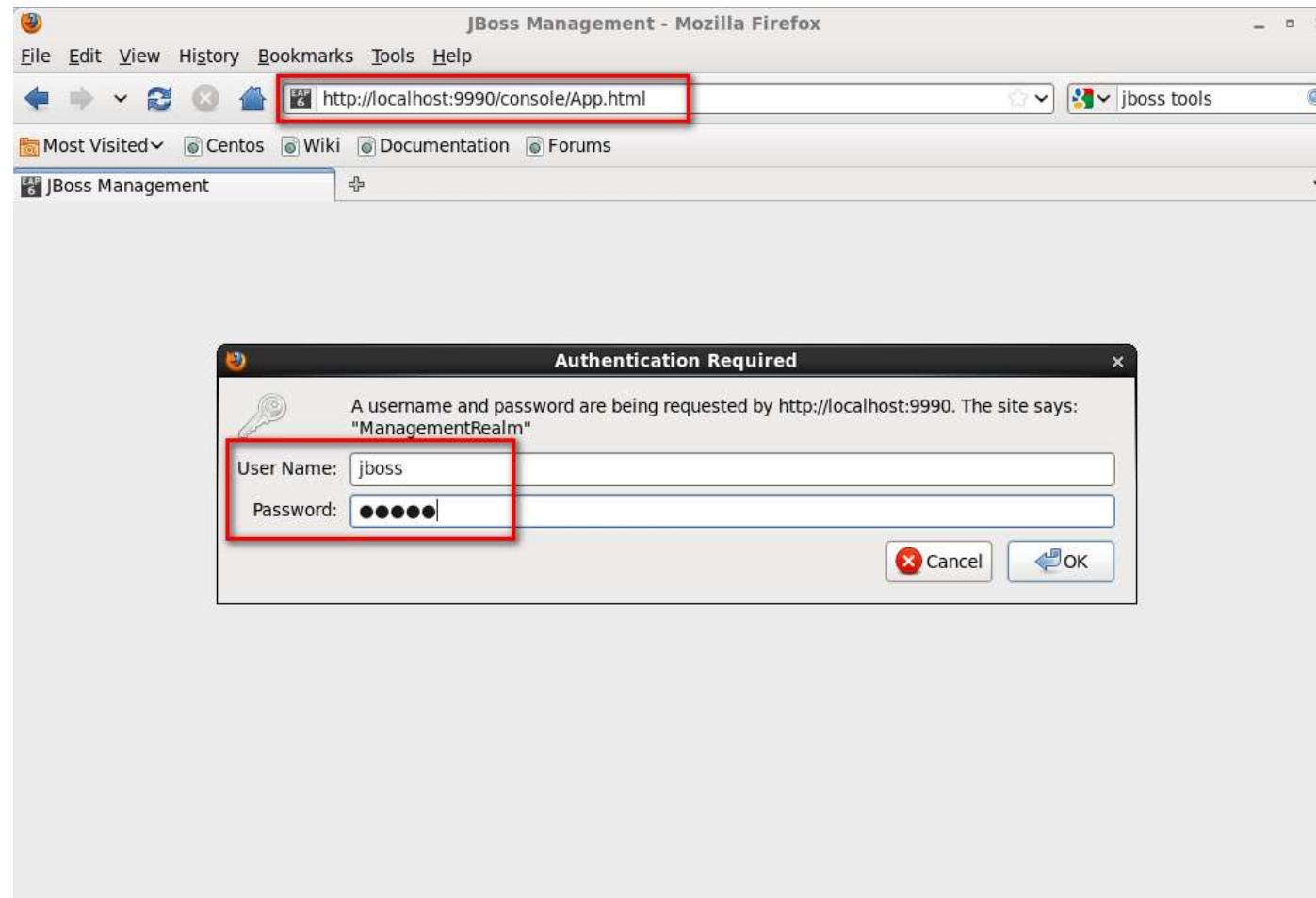
```
<system-properties>
 <property name="mipropiedad" value="mivalor"/>
</system-properties>
```

## ⌘ Sección **Deployments**

- ❖ Esta última sección del archivo de configuración contiene las aplicaciones desplegadas en al servidor de aplicaciones
- ❖ Cada vez que se despliegue una nueva aplicación o se repliegue una aplicación existente esta sección se actualiza para reflejar la pila de aplicaciones disponibles

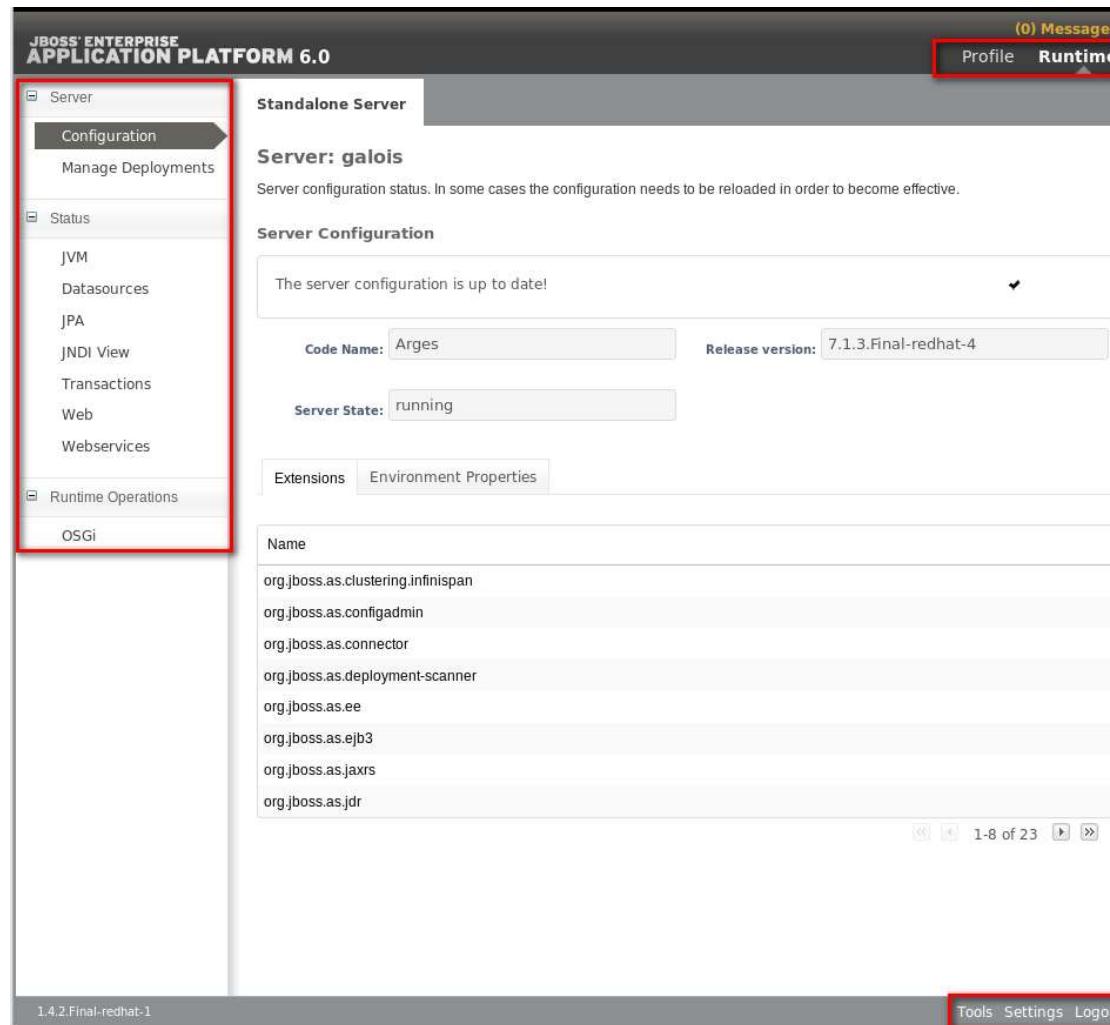
# Consola de administración web (I)

⌘ Accedemos a la consola vía <http://localhost:9990>



# Consola de administración web (II)

- ☒ La consola nos permite trabajar en modo **runtime** (Monitorizar) ó **profile** (Configurar)



The screenshot shows the JBoss Enterprise Application Platform 6.0 Web Console interface. The title bar reads "JBoss® ENTERPRISE APPLICATION PLATFORM 6.0". The top right corner shows "(0) Messages" and navigation tabs for "Profile" and "Runtime", with "Runtime" being the active tab. The left sidebar is highlighted with a red box and contains the following menu items:

- Server
  - Configuration (selected)
  - Manage Deployments
- Status
  - JVM
  - Datasources
  - JPA
  - JNDI View
  - Transactions
  - Web
  - Webservices
- Runtime Operations
  - OSGi (selected)

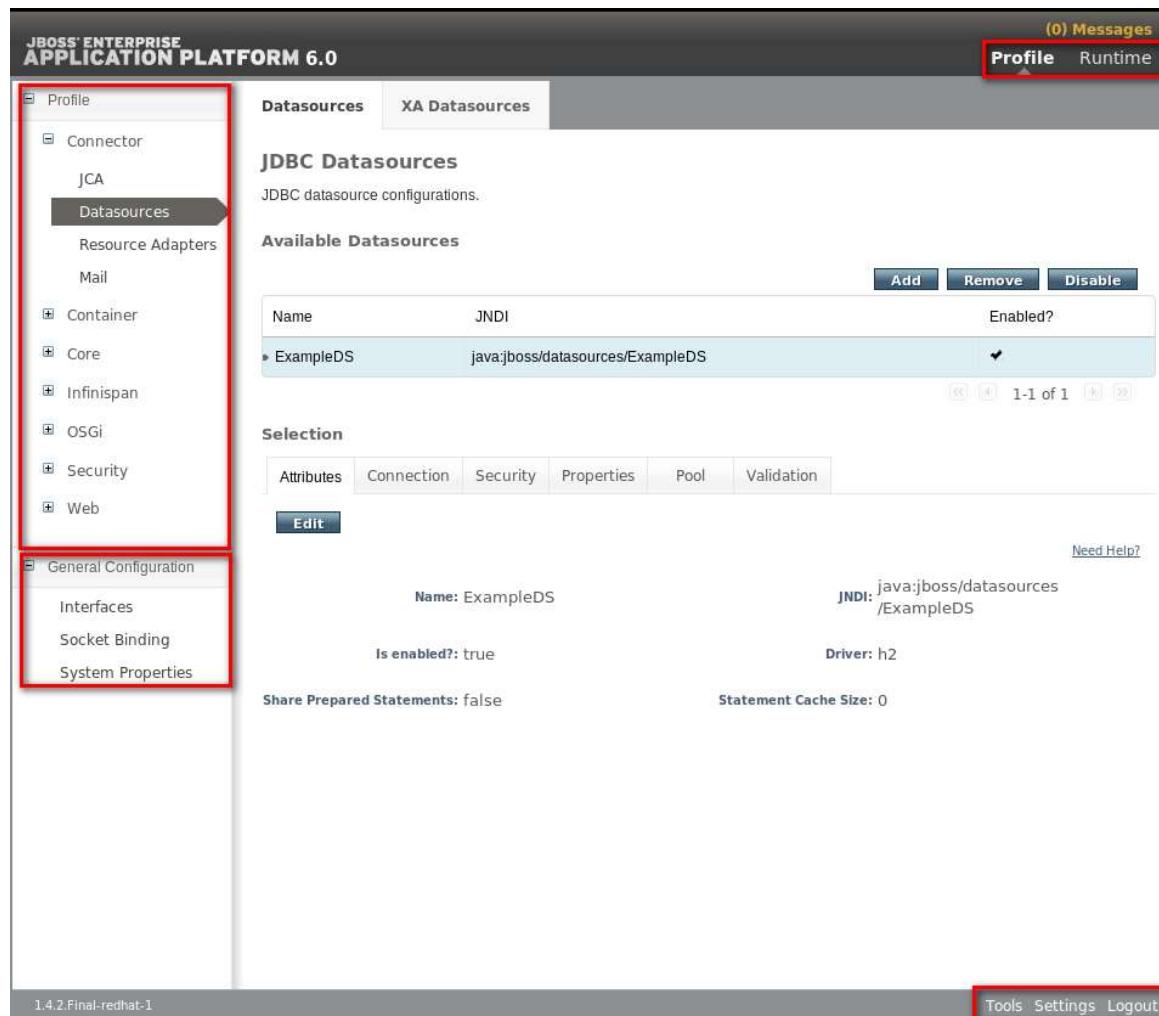
The main content area displays the following information:

- Server: galois**
- Server configuration status: "The server configuration is up to date!"
- Code Name: Arges
- Release version: 7.1.3.Final-redhat-4
- Server State: running
- Extensions tab (selected): Shows a list of OSGi extensions:
  - org.jboss.as.clustering.infinispan
  - org.jboss.as.configadmin
  - org.jboss.as.connector
  - org.jboss.as.deployment-scanner
  - org.jboss.as.ee
  - org.jboss.as.ejb3
  - org.jboss.as.jaxrs
  - org.jboss.as.jdr
- Environment Properties tab

At the bottom of the page, there are navigation icons and links: "1-8 of 23", "Tools", "Settings", and "Logout". The footer of the page also includes the text "1.4.2.Final-redhat-1".

# Consola de administración web (III)

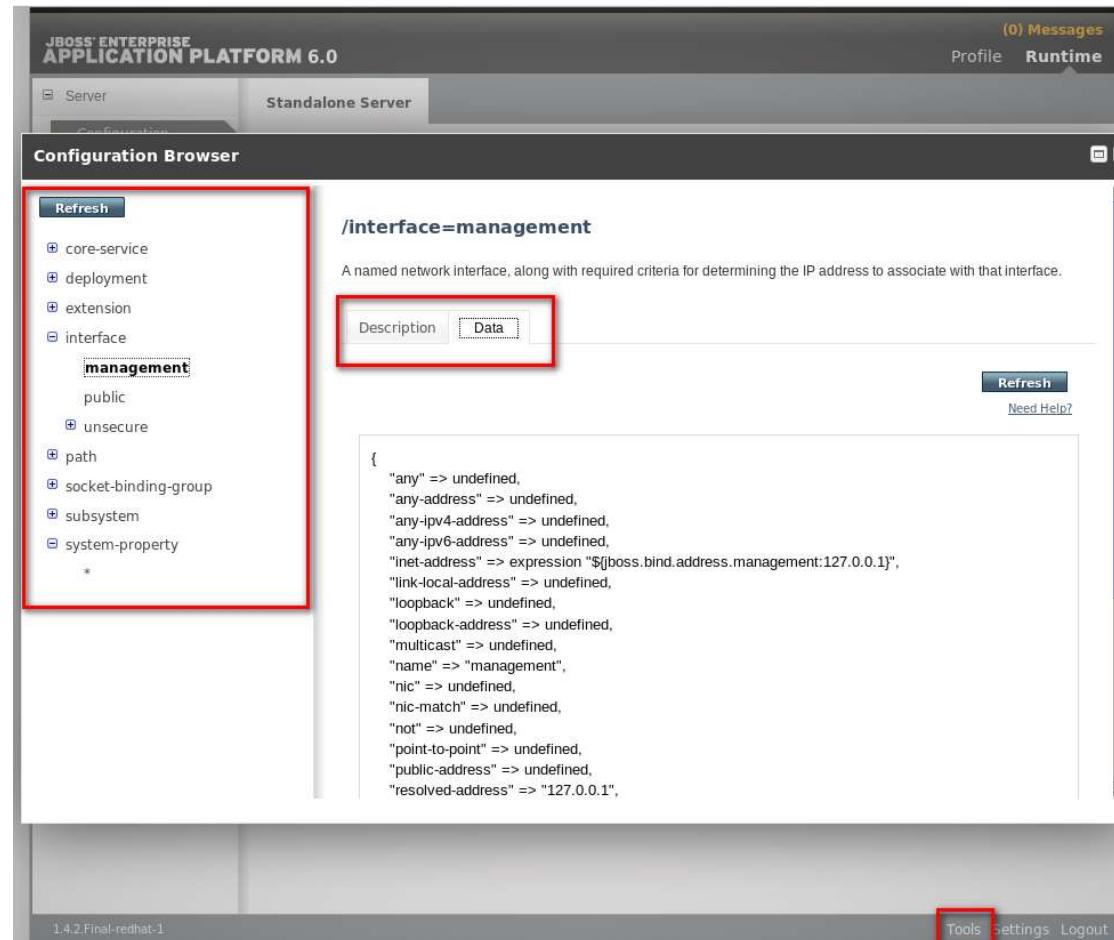
- ❖ Desde la vista **profile** podemos cambiar **casi** cualquier parte de la configuración del servidor



The screenshot shows the JBoss Enterprise Application Platform 6.0 Web Console. The top navigation bar includes tabs for 'Profile' (which is highlighted with a red box) and 'Runtime'. The left sidebar has sections for 'Profile' (with 'Datasources' selected), 'General Configuration' (with 'Interfaces', 'Socket Binding', and 'System Properties' listed), and other options like 'Connector', 'JCA', 'Resource Adapters', 'Mail', 'Container', 'Core', 'Infinispan', 'OSGi', 'Security', and 'Web'. The main content area is titled 'JDBC Datasources' and shows 'JDBC datasource configurations'. It lists an 'Available Datasources' table with one entry: 'ExampleDS' (JNDI: java:jboss/datasources/ExampleDS, Enabled: checked). Below the table are tabs for 'Attributes', 'Connection', 'Security', 'Properties', 'Pool', and 'Validation', with an 'Edit' button. At the bottom, detailed properties for 'ExampleDS' are shown: Name: ExampleDS, Is enabled?: true, Driver: h2, JNDI: java:jboss/datasources/ExampleDS, Statement Cache Size: 0, Share Prepared Statements: false. The footer includes links for 'Tools', 'Settings', and 'Logout'.

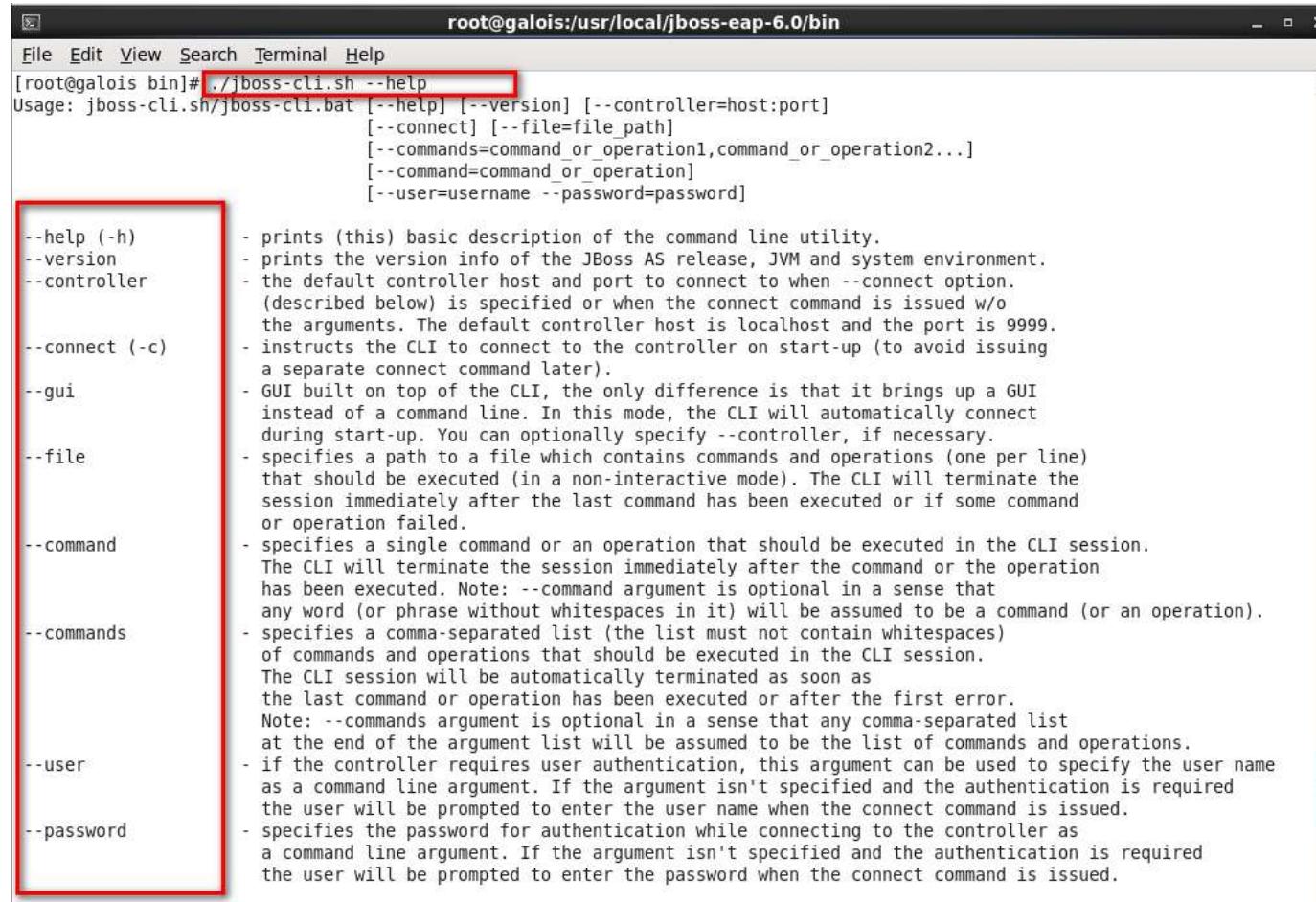
# Consola de administración web (IV)

- ⌘ También disponemos un explorador de la configuración **configuration browser** para ver los valores de todos los componentes del servidor. (Cambiado en eap-6.1 por **management model view**)



# Interfaz de comandos para administrar (I)

- ⌘ Para realizar tareas de administración periódicas ó de forma programada en el tiempo ó repetitivas tenemos la herramienta de scripting de administración **jboss-cli**



```
root@galois:/usr/local/jboss-eap-6.0/bin
File Edit View Search Terminal Help
[root@galois bin]# ./jboss-cli.sh --help
Usage: jboss-cli.sh/jboss-cli.bat [--help] [--version] [--controller=host:port]
 [--connect] [--file=file_path]
 [--commands=command_or_operation1,command_or_operation2...]
 [--command=command_or_operation]
 [--user=username --password=password]

--help (-h) - prints (this) basic description of the command line utility.
--version - prints the version info of the JBoss AS release, JVM and system environment.
--controller - the default controller host and port to connect to when --connect option.
 (described below) is specified or when the connect command is issued w/o
 the arguments. The default controller host is localhost and the port is 9999.
--connect (-c) - instructs the CLI to connect to the controller on start-up (to avoid issuing
 a separate connect command later).
--gui - GUI built on top of the CLI, the only difference is that it brings up a GUI
 instead of a command line. In this mode, the CLI will automatically connect
 during start-up. You can optionally specify --controller, if necessary.
--file - specifies a path to a file which contains commands and operations (one per line)
 that should be executed (in a non-interactive mode). The CLI will terminate the
 session immediately after the last command has been executed or if some command
 or operation failed.
--command - specifies a single command or an operation that should be executed in the CLI session.
 The CLI will terminate the session immediately after the command or the operation
 has been executed. Note: --command argument is optional in a sense that
 any word (or phrase without whitespaces in it) will be assumed to be a command (or an operation).
--commands - specifies a comma-separated list (the list must not contain whitespaces)
 of commands and operations that should be executed in the CLI session.
 The CLI session will be automatically terminated as soon as
 the last command or operation has been executed or after the first error.
 Note: --commands argument is optional in a sense that any comma-separated list
 at the end of the argument list will be assumed to be the list of commands and operations.
--user - if the controller requires user authentication, this argument can be used to specify the user name
 as a command line argument. If the argument isn't specified and the authentication is required
 the user will be prompted to enter the user name when the connect command is issued.
--password - specifies the password for authentication while connecting to the controller as
 a command line argument. If the argument isn't specified and the authentication is required
 the user will be prompted to enter the password when the connect command is issued.
```

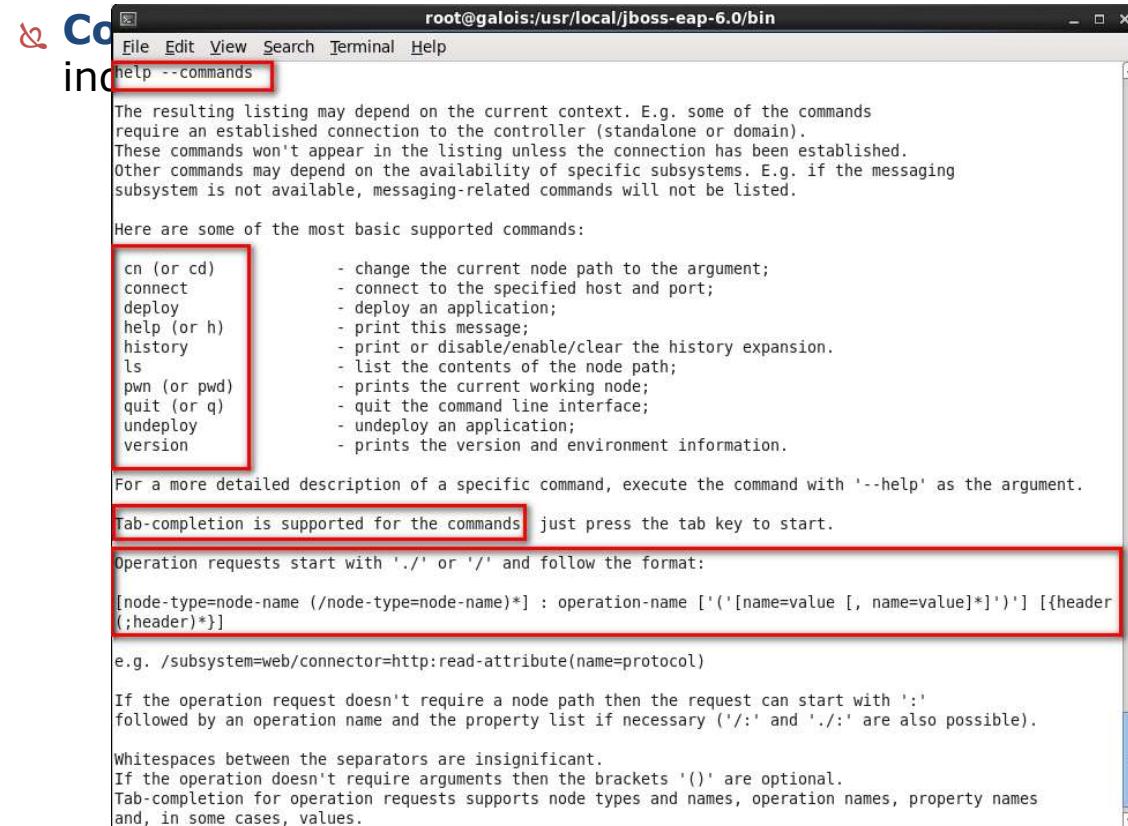
# Interfaz de comandos para administrar (II)

⌘ En **jboss-cli** existen 2 tipos de mandatos:

- ❖ **Operations** Las operaciones incluyen la ruta del recurso en el que se ejecutan. Siguen el formato:

**/ruta/subruta:operación(nombre\_parametro:valor\_parametro)**

- ❖ **Commands**



```
root@galois:/usr/local/jboss-eap-6.0/bin
File Edit View Search Terminal Help
help - commands
The resulting listing may depend on the current context. E.g. some of the commands
require an established connection to the controller (standalone or domain).
These commands won't appear in the listing unless the connection has been established.
Other commands may depend on the availability of specific subsystems. E.g. if the messaging
subsystem is not available, messaging-related commands will not be listed.

Here are some of the most basic supported commands:
cn (or cd) - change the current node path to the argument;
connect - connect to the specified host and port;
deploy - deploy an application;
help (or h) - print this message;
history - print or disable/enable/clear the history expansion.
ls - list the contents of the node path;
pwn (or pwd) - prints the current working node;
quit (or q) - quit the command line interface;
undeploy - undeploy an application;
version - prints the version and environment information.

For a more detailed description of a specific command, execute the command with '--help' as the argument.

Tab-completion is supported for the commands just press the tab key to start.

Operation requests start with './' or '/' and follow the format:
[node-type=node-name (/node-type=node-name)*] : operation-name ['(['name=value [, name=value]*'])'] [{header
(;header)*}]

e.g. /subsystem=web/connector=http:read-attribute(name=protocol)

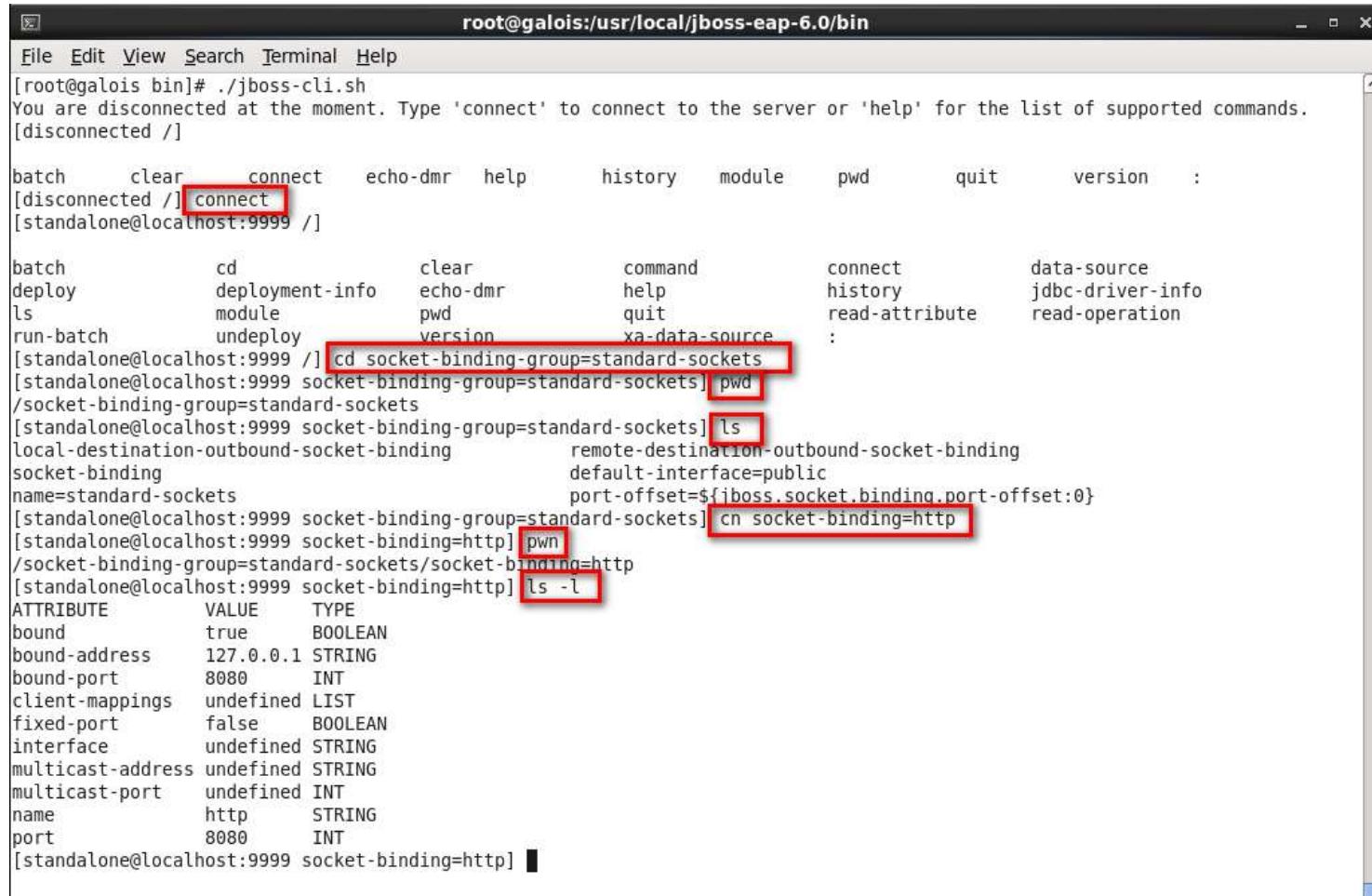
If the operation request doesn't require a node path then the request can start with ':'
followed by an operation name and the property list if necessary ('/:)' and './:' are also possible).

Whitespaces between the separators are insignificant.
If the operation doesn't require arguments then the brackets '()' are optional.
Tab-completion for operation requests supports node types and names, operation names, property names
and, in some cases, values.
```

... y ejecutan una acción  
se usan para navegar

# Interfaz de comandos para administrar (III)

## ⌘ Trabajando con **comandos**:



The screenshot shows a terminal window titled "root@galois:/usr/local/jboss-eap-6.0/bin". The user has run the command `./jboss-cli.sh`. The terminal displays the help menu for the JBoss CLI, which includes various commands like batch, clear, connect, echo-dmr, help, history, module, pwd, quit, and version. The user has connected to the standalone server at port 9999, as shown by the prompt "[standalone@localhost:9999 /]". Several commands have been highlighted with red boxes: `connect`, `cd socket-binding-group=standard-sockets`, `pwd`, `ls`, `cn socket-binding=http`, `pwn`, and `ls -l`. The output also shows the configuration details for the "http" socket binding, including its bound address (127.0.0.1), port (8080), and name (http).

```
root@galois:/usr/local/jboss-eap-6.0/bin
File Edit View Search Terminal Help
[root@galois bin]# ./jboss-cli.sh
You are disconnected at the moment. Type 'connect' to connect to the server or 'help' for the list of supported commands.
[disconnected /]

batch clear connect echo-dmr help history module pwd quit version :
[disconnected /] connect
[standalone@localhost:9999 /]

batch cd clear command connect data-source
deploy deployment-info echo-dmr help history jdbc-driver-info
ls module pwd quit read-attribute read-operation
run-batch undeploy version xa-data-source
[standalone@localhost:9999 /] cd socket-binding-group=standard-sockets
[standalone@localhost:9999 socket-binding-group=standard-sockets] pwd
/socket-binding-group=standard-sockets
[standalone@localhost:9999 socket-binding-group=standard-sockets] ls
local-destination-outbound-socket-binding remote-destination-outbound-socket-binding
socket-binding default-interface=public
name=standard-sockets port-offset=${jboss.socket.binding.port-offset:0}
[standalone@localhost:9999 socket-binding-group=standard-sockets] cn socket-binding=http
[standalone@localhost:9999 socket-binding=http] pwn
/socket-binding-group=standard-sockets/socket-binding=http
[standalone@localhost:9999 socket-binding=http] ls -l
ATTRIBUTE VALUE TYPE
bound true BOOLEAN
bound-address 127.0.0.1 STRING
bound-port 8080 INT
client-mappings undefined LIST
fixed-port false BOOLEAN
interface undefined STRING
multicast-address undefined STRING
multicast-port undefined INT
name http STRING
port 8080 INT
[standalone@localhost:9999 socket-binding=http]
```

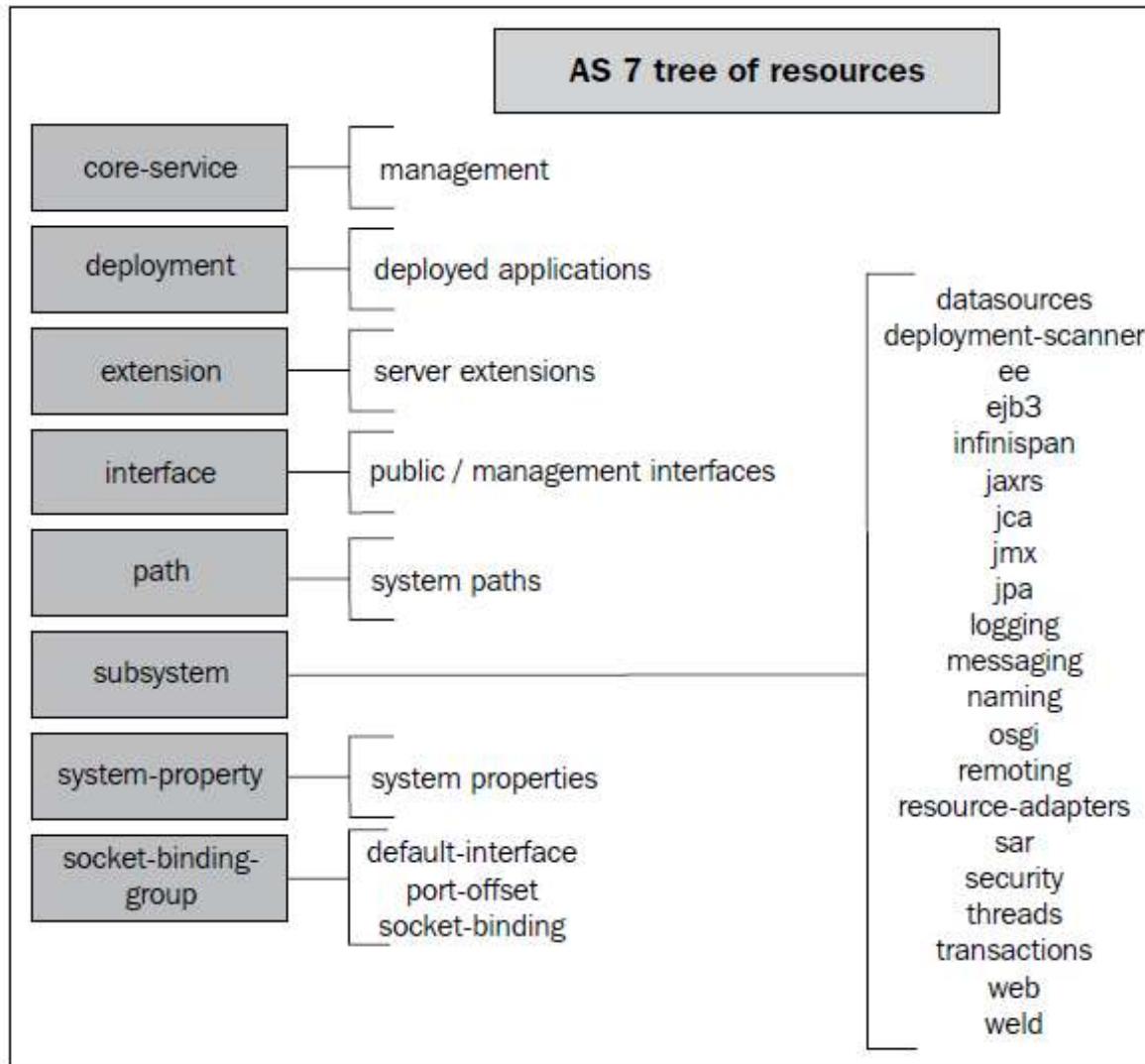
# Interfaz de comandos para administrar (IV)

## ⌘ Trabajando con **operaciones**:

```
root@galois:/usr/local/jboss-eap-6.0/bin
File Edit View Search Terminal Help
[standalone@localhost:9999 /] /socket-binding-group=standard-sockets/socket-binding=http:read-
read-attribute read-children-names read-children-resources read-children-types
read-operation-description read-operation-names read-resource read-resource-description
[standalone@localhost:9999 /] /socket-binding-group=standard-sockets/socket-binding=http:read-attribute(name=
port bound-port interface fixed-port client-mappings multicast-port
bound-address name multicast-address bound
[standalone@localhost:9999 /] /socket-binding-group=standard-sockets/socket-binding=http:read-attribute(name=port)
{
 "outcome" => "success",
 "result" => 8080
}
[standalone@localhost:9999 /] cd /socket-binding-group=standard-sockets/socket-binding=http
[standalone@localhost:9999 socket-binding=http] :read-
read-attribute read-children-names read-children-resources read-children-types
read-operation-description read-operation-names read-resource read-resource-description
[standalone@localhost:9999 socket-binding=http] :read-attribute(name=
port bound-port interface fixed-port client-mappings multicast-port
bound-address name multicast-address bound
[standalone@localhost:9999 socket-binding=http] :read-attribute(name=port)
{
 "outcome" => "success",
 "result" => 8080
}
[standalone@localhost:9999 socket-binding=http] ../socket-binding=https:read-attribute(name=port)
{
 "outcome" => "success",
 "result" => 8443
}
[standalone@localhost:9999 socket-binding=http] █
```

# Interfaz de comandos para administrar (V)

## ☒ Árbol de recursos de administración:



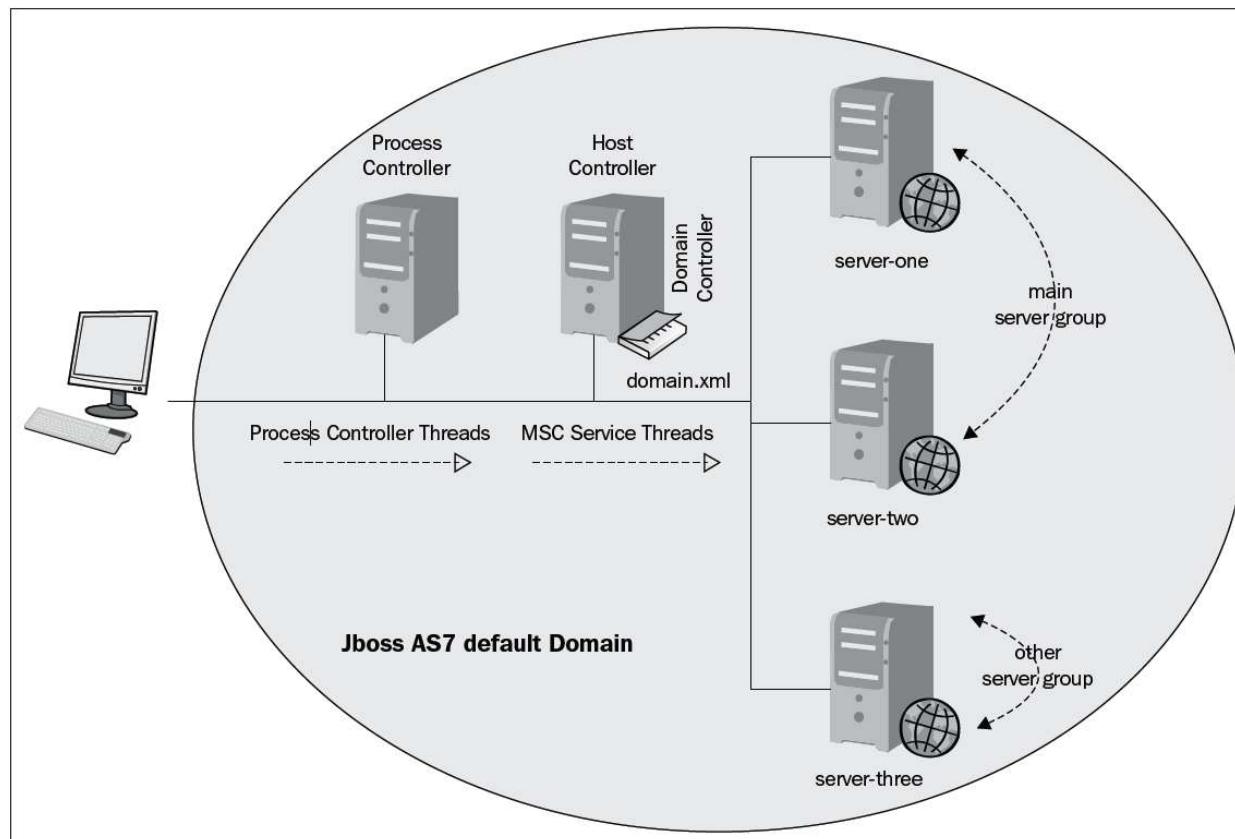
# Dominios de administración (I)

- ⌘ Un dominio es una unidad administrativa. Es un perímetro dentro del cual todos los servidores JBoss AS son administrados por un controlador de dominio (**domain controller**)
- ⌘ Desde el punto de vista de procesos, un dominio se compone de cuatro elementos:
  - ❖ **Domain controller:** es el punto de control administrativo de un dominio. Una instancia como en ejecución en modo de dominio tendrá a lo sumo un controlador de dominio. El controlador de dominio mantiene una configuración centralizada, que es compartida por las instancias de nodo pertenecientes al dominio
  - ❖ **Host controller:** Es un proceso que es responsable de coordinar con el controlador de dominio el ciclo de vida de los procesos servidor y la distribución de los despliegues, desde el controlador de dominio a las instancias del servidor
  - ❖ **Process controller:** Es un proceso muy ligero cuya función principal es lanzar los procesos de servidor y los procesos de controlador de host, y gestionar sus flujos de entrada / salida. También permite que el controlador de host sea parcheado y reiniciado sin afectar a los servidores asociados
  - ❖ **Application server nodes:** Son los procesos Java ordinarios que se asignan a instancias del servidor de aplicaciones. Cada nodo servidor, a su vez, pertenece a un **grupo de servidores** del dominio.

# Dominios de administración (II)

⌘ La configuración del dominio por defecto (**Out of the box**) es:

- ❖ Un controlador de proceso que inicia los otros procesos JVM
- ❖ Un controlador de host que actúa como controlador de dominio
- ❖ Tres nodos de servidor. Dos forman parte del grupo de servidores principal (**main server group**), y otro (en estado inactivo) es miembro del grupo otro servidor (**other server-group**)



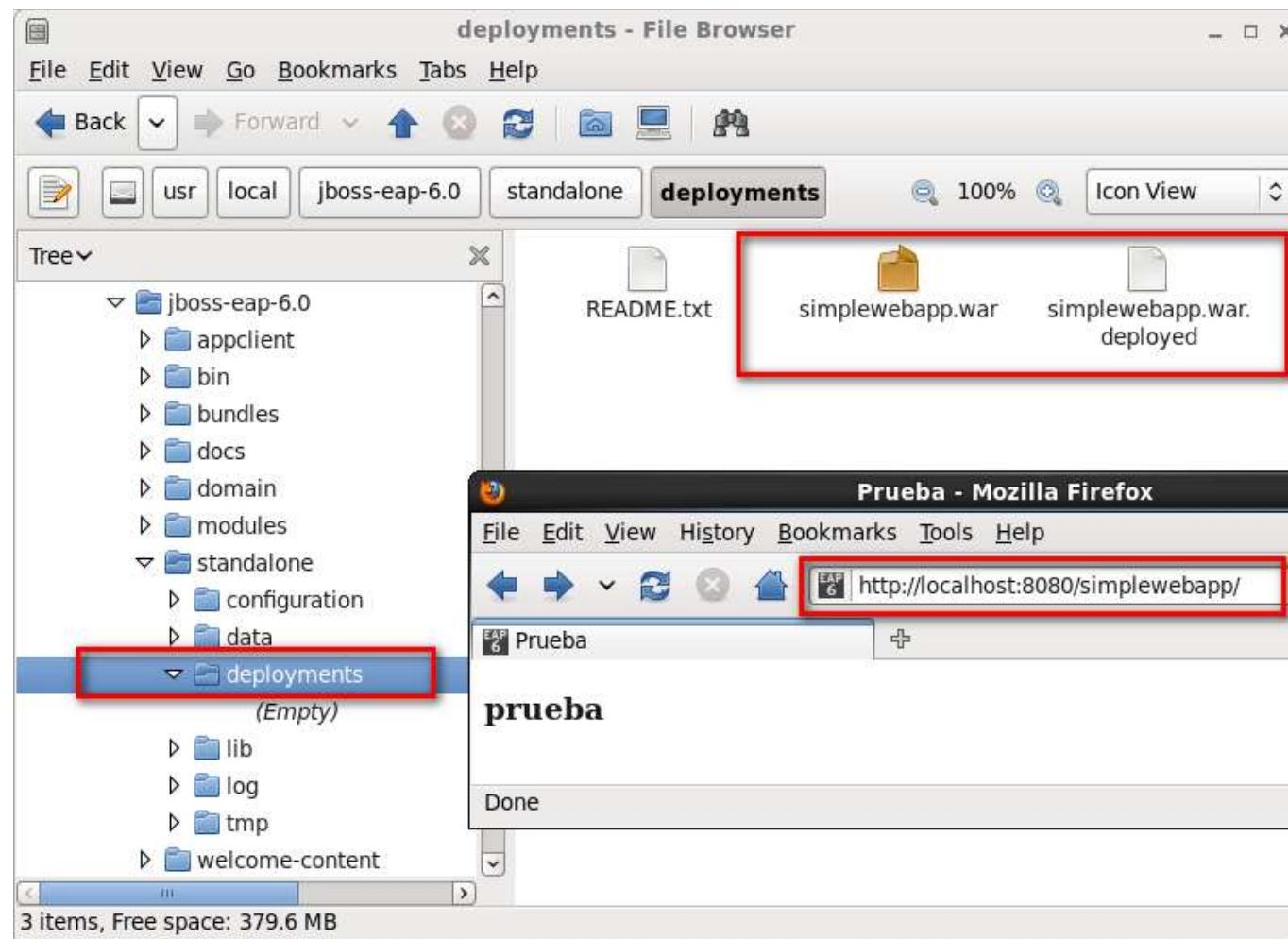
# Contenidos

---

- I. Introducción
- II. Instalación del servidor de aplicaciones
- III. Configuración y herramientas de administración
- IV. Despliegue de aplicaciones**
- V. Seguridad de las aplicaciones
- VI. Clustering
- VII. Optimización

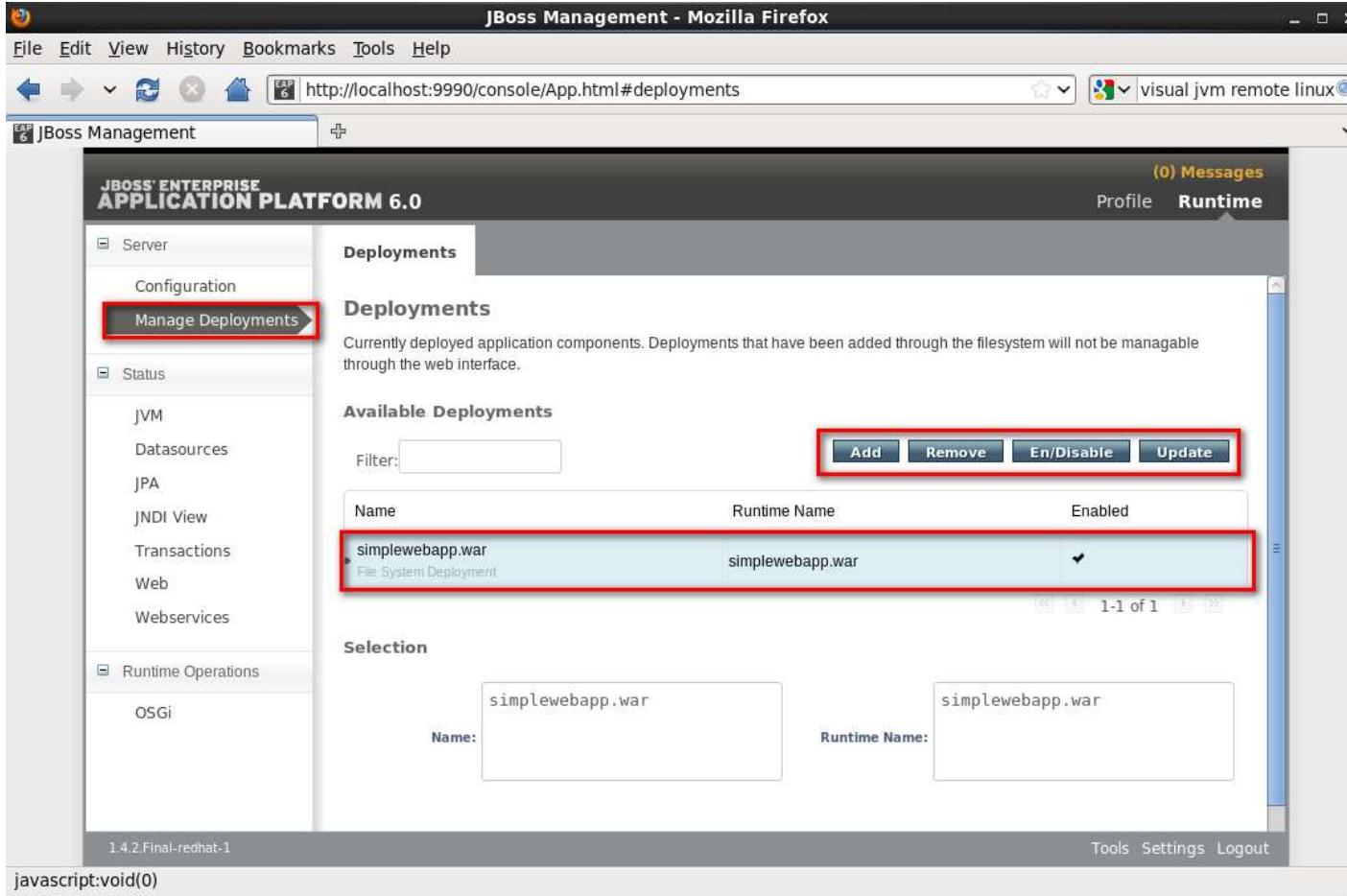
# Despliegue en caliente (I)

- Colocando el artefacto jee en la carpeta **deployments** de la configuración de servidor que estemos usando.



# Despliegue en caliente (II)

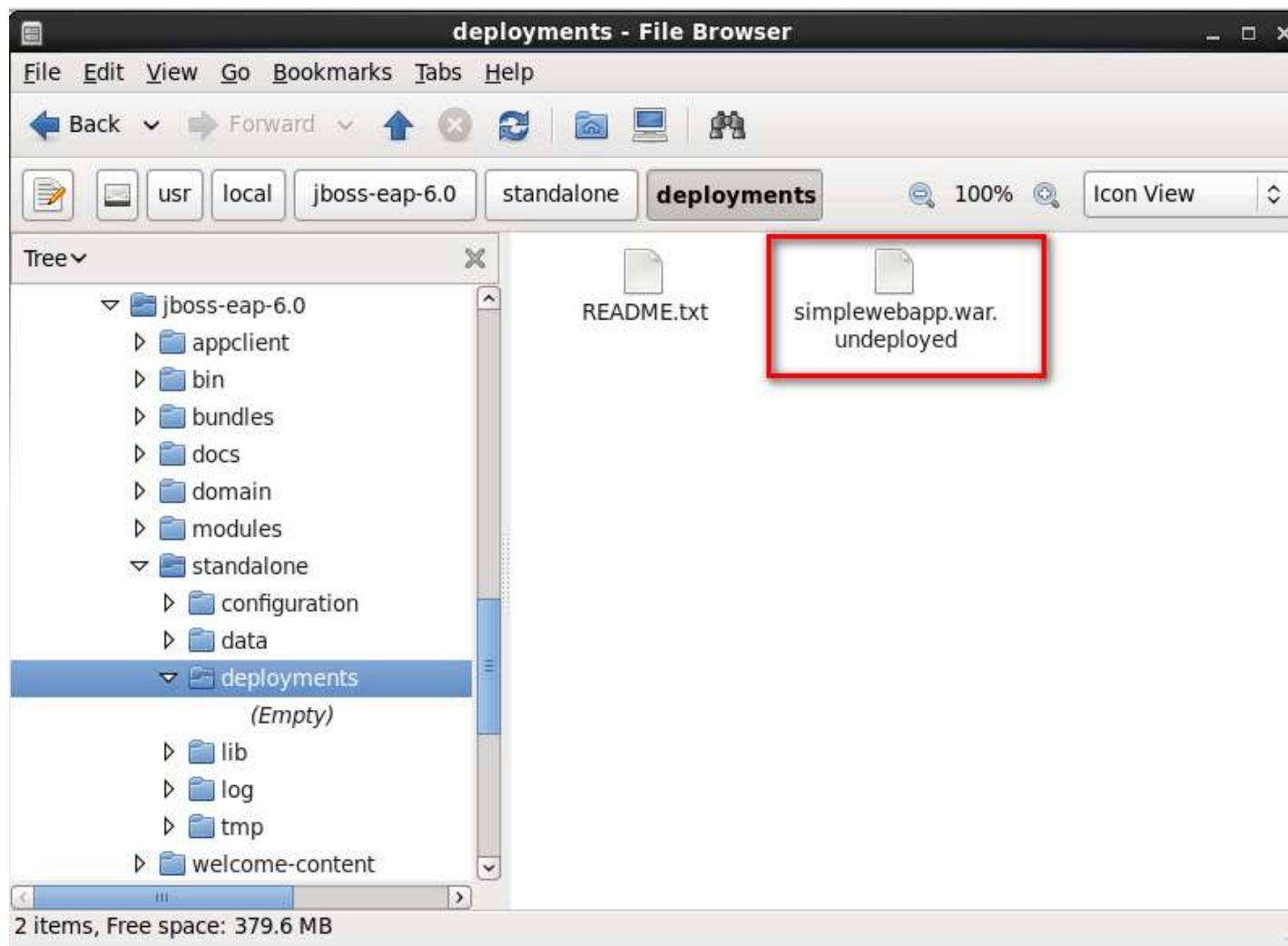
✖ La aplicación esta disponible vía consola web



The screenshot shows the JBoss Management console interface. The left sidebar has a tree view with nodes like Server, Configuration, Manage Deployments (which is highlighted with a red arrow), Status, JVM, Datasources, JPA, JNDI View, Transactions, Web, Webservices, and Runtime Operations (OSGi). The main content area is titled 'Deployments'. It displays the message: 'Currently deployed application components. Deployments that have been added through the filesystem will not be manageable through the web interface.' Below this is a section titled 'Available Deployments' with a table. The table has columns: Name, Runtime Name, and Enabled. One row is shown: 'simplewebapp.war' (with a note 'File System Deployment!'), 'simplewebapp.war', and a checked 'Enabled' checkbox. Above the table are four buttons: 'Add', 'Remove', 'En/Disable', and 'Update', all of which are highlighted with a red box. At the bottom of the table, it says '1-1 of 1'. The footer of the page includes links for Tools, Settings, and Logout.

# Despliegue en caliente (III)

✖ Borrando el archivo replegamos la aplicación.



# Despliegue en caliente (IV)

- ⌘ El servicio que escanea en busca de recursos utilizados se llama el **deployment scanner** y se configura en el archivo standalone.xml

```
<subsystem xmlns="urn:jboss:domain:deployment-scanner:1.0">
 <deployment-scanner name="default" path="deployments"
 scan-interval="5000" relative-to="jboss.server.base.dir"/>
</subsystem>
```

- ⌘ Podemos activar el despliegue en modo manual

```
<deployment-scanner scan-interval="5000"
 relative-to="jboss.server.base.dir" path="deployments"
 auto-deploy-zipped="true" auto-deploy-exploded="false/>
```

- ⌘ Existe un API de operaciones basado en ficheros/extensiones

Extensión	Creado	Significado
.dodeploy	Usuario	Crear este archivo provoca el despliegue. “Tocar” este archivo hace que se redespiece
.skipdeploy	Usuario	El despliegue automático de la aplicación está deshabilitado, mientras el archivo exista
.deployed	JBoss AS	Aplicación desplegada. Su eliminación provoca el repliegue
.undeployed	JBoss AS	Aplicación ha sido replegada. Quitarlo provoca el redespiece
.failed	JBoss AS	El despliegue de la aplicación ha fallado.
.isdeploying	JBoss AS	Despliegue de la aplicación en curso
.isundeploying	JBoss AS	Repliegue de la aplicación en curso
.pending	JBoss AS	Existe una condición que impide el despliegue (por ejemplo, copia de archivos en curso)

# Despliegue Aplicaciones: Módulo Web (I)

**Ejemplo módulo web,** Crear un Servlet que muestre la hora por pantalla.

## PASO 1: Creación de la clase del Servlet.



```
ServletFecha.java
1 package curso.jboss.servlet;
2
3+import java.io.IOException;
11
12 public class ServletFecha extends HttpServlet {
13 private static final long serialVersionUID = 1L;
14
15+ protected void doGet(HttpServletRequest request,
16 HttpServletResponse response) throws ServletException, IOException {
17 PrintWriter out = response.getWriter();
18 out.println("<html><body>fecha=>" + new Date() + "</body></html>");
19 out.close();
20 }
21
22 }
23
```

# Despliegue Aplicaciones: Módulo Web (II)

## PASO 2: Creación del descriptor de despliegue **/WEB-INF/web.xml**



```
web.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2 <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3 xmlns="http://java.sun.com/xml/ns/javaee"
4 xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
5 xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/w
6 <display-name>simplewebapp</display-name>
7 <welcome-file-list>
8 <welcome-file>index.html</welcome-file>
9 </welcome-file-list>
10 <servlet>
11 <description></description>
12 <display-name>ServletFecha</display-name>
13 <servlet-name>ServletFecha</servlet-name>
14 <servlet-class>curso.jboss.servlet.ServletFecha</servlet-class>
15 </servlet>
16 <servlet-mapping>
17 <servlet-name>ServletFecha</servlet-name>
18 <url-pattern>/ServletFecha</url-pattern>
19 </servlet-mapping>
20 <session-config>
21 <session-timeout>30</session-timeout>
22 </session-config>
23 </web-app>
```

## PASO 3: Creamos la pagina por defecto **index.html**



```
HTM index.html X
1 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
2 <html>
3 <head>
4 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
5 <title>Prueba</title>
6 </head>
7 <body>
8 <h3>prueba</h3>
9 </body>
10 </html>|
```

**PASO 4:** Crear el archivo **simplewebapp.war** con la estructura:  
**WEB-INF/classes/curso.jboss.servlet.ServletFecha.class**  
**WEB-INF/web.xml**  
**index.html**

**PASO 5:** Despliegue del archivo en el directorio **deployments**

# Despliegue Aplicaciones: Módulo Web (IV)

**PASO 6:** anulamos el despliegue del módulo

**PASO 7:** creamos el descriptor de despliegue de la aplicación enterprise **application.xml**



```
<?xml version="1.0" encoding="UTF-8"?>
<application xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns="http://java.sun.com/xml/ns/javaee"
 xmlns:application="http://java.sun.com/xml/ns/javaee/application_5.xsd"
 xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/
 <display-name>Ejemplos.1.simpleearapp.JEE5</display-name>
 <module>
 <web>
 <web-uri>Ejemplos.1.simplewebapp.JEE5.war</web-uri>
 <context-root>simplewebapp</context-root>
 </web>
 </module>
</application>
```

**PASO 8:** Crear el archivo **simpleenterpriseapp.ear**. Estructura:  
**simplewebapp.war**

**META-INF/application.xml**

**PASO 9:** Despliegue del archivo en el directorio **deployments**

# Despliegue Aplicaciones: Módulo Web (V)

**PASO 10:** anulamos el despliegue del módulo

**PASO 11:** creamos el descriptor de despliegue propietario del módulo web **jboss-web.xml**



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <jboss-web version="6.0" xmlns="http://www.jboss.com/xml/ns/javaee"
3 xmlns:javaee="http://java.sun.com/xml/ns/javaee"
4 xmlns:xml="http://www.w3.org/XML/1998/namespace"
5 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
6 xsi:schemaLocation="http://www.jboss.com/xml/ns/javaee http://www.jboss.org/schema/jbossweb/j
7 <context-root>
8 /intranet
9 </context-root>
10 </jboss-web>
11
```

**PASO 12:** Crear el archivo **simpleenterpriseapp.ear**. Estructura:

**simplewebapp.war**

META-INF/application.xml

**PASO 13:** Despliegue del archivo en el directorio **deployments**

# Despliegue Aplicaciones: Módulo Web (VI)

## **PASO 14:** listar despliegues vía **jboss-cli**:

```
[disconnected /] connect
Connected to standalone controller at localhost:9999
[localhost:9999 /] deploy
```

## **PASO 15:** anular despliegues vía **jboss-cli**:

```
[localhost:9999 /] undeploy simpleenterpriseapp.ear
'simpleenterpriseapp.ear' undeployed successfully.
```

## **PASO 16:** desplegar vía **jboss-cli** (rutas relativas o absolutas):

```
[localhost:9999 /] deploy simpleenterpriseapp.ear
'simpleenterpriseapp.ear' deployed successfully.
```

## **PASO 17:** redespliegues vía **jboss-cli**:

```
[localhost:9999 /] deploy -f simpleenterpriseapp.ear
'simpleenterpriseapp.ear' re-deployed successfully.
```

## **PASO 18:** activar/desactivar despliegues vía **jboss-cli**:

```
[localhost:9999 /] deploy simpleenterpriseapp.ear --disabled
'simpleenterpriseapp.ear' deployed successfully.
```

# Aplicaciones Web (I)

El descriptor de despliegue propietario **jboss-web.xml** tiene las siguientes secciones principales:

<b>class-loading</b>	Usado para habilitar la carga aislada de clases. Cuando queremos este comportamiento para la aplicación y no para todo el servidor.
<b>security-domain</b>	Especifica que dominio de seguridad usa la aplicación para autorización y autenticación.
<b>context-root</b>	Define la URL raíz para esta aplicación, si queremos tener una diferente a la URL implícita (el nombre del fichero war)
<b>virtual-host</b>	Especifica a que virtual host pertenece la aplicación. Debe corresponder a un virtual host definido en Tomcat (server.xml)
<b>use-session-cookies</b>	Un variable booleana que indica si la sesión se debe mantener o no vía cookies cliente.

# Aplicaciones Web (II)

<b>replication-config</b>	Define como replicar el estado de la sesión HTTP a través de un cluster
<b>resource-env-ref</b>	Mapea el Enterprise Naming Context (ENC) para un resource-env-ref definido en el web.xml file a una localización en el espacio de nombres JNDI global
<b>resource-ref</b>	Mapea el Enterprise Naming Context (ENC) para un resource-ref definido en el web.xml file a una localización en el espacio de nombres JNDI global
<b>ejb-ref</b>	Mapea el Enterprise Naming Context (ENC) para un ejb-ref definido en el web.xml file a una localización en el espacio de nombres JNDI global
<b>ejb-local-ref</b>	Mapea el Enterprise Naming Context (ENC) para un ejb-local-ref definido en el web.xml file a una localización en el espacio de nombres JNDI global
<b>servlet</b>	Usado para configuraciones específicas de los servlet en JBoss

# DataSources (I)

## 1. Instalar el **Driver JDBC** en el servidor

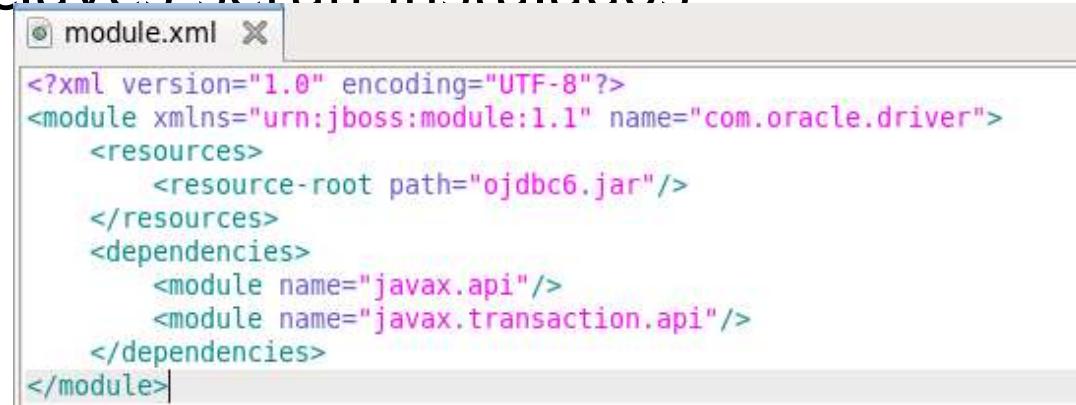
❖ En la arquitectura de servidor modular, hay varias opciones para instalar el controlador JDBC. El enfoque **recomendado** es instalarlo como módulo

❖ Instalar un nuevo módulo requiere copiar sus archivos **.jar** en la ruta adecuada y añadir el archivo **module.xml**, que declara el módulo y



**: JBOSS\_HOME  
lin**

es donde todos los componentes del módulo de claves serán instalados



```

<?xml version="1.0" encoding="UTF-8"?>
<module xmlns="urn:jboss:module:1.1" name="com.oracle.driver">
 <resources>
 <resource-root path="ojdbc6.jar"/>
 </resources>
 <dependencies>
 <module name="javax.api"/>
 <module name="javax.transaction.api"/>
 </dependencies>
</module>

```

# DataSources (II)

## 2. Crear el Pool de Conexiones (**DataSource**)

En JBoss AS 7 se pueden configurar dos tipos de fuentes de datos (**DataSources**) **Locales y Globales ó XA**

En el fichero de configuración buscamos el subsistema

```
<subsystem xmlns="urn:jboss:domain:datasources:1.0">
```

Añadimos la definición del driver

```
<driver name="oracle" module="com.oracle.driver">
 <xa-datasource-class>
 oracle.jdbc.OracleDriver
 </xa-datasource-class>
</driver>
```

- O bien via jboss-cli

```
/subsystem=datasources/jdbc-driver=oracle:add(driver-name=oracle, driver-class-name=
oracle.jdbc.OracleDriver, driver-module-name=com.oracle.driver)
```

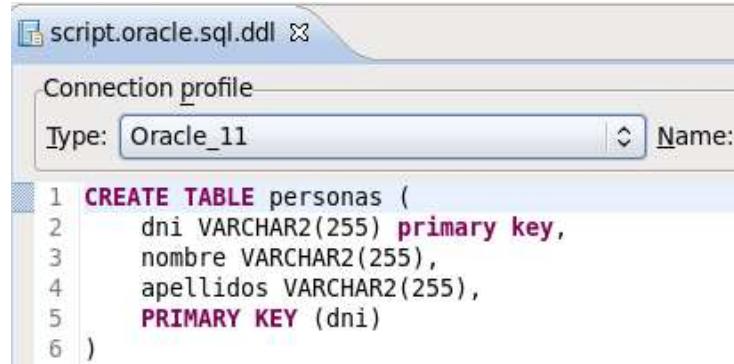
Añadimos la configuración del dataSource

```
<datasource jndi-name="OracleDS" pool-name="OracleDS" enabled="true" jta="true" use-java-
context="true" use-ccm="true">
 <connection-url>jdbc:oracle:thin:@localhost:1521:xe</connection-url>
 <driver>oracle</driver>
 <transaction-isolation>TRANSACTION_READ_COMMITTED</transaction-isolation>
 <pool>
```

# Despliegue Apps con DataSources (I)

**Ejemplo Configuración DataSources,** Crear un crear un Servlet que establezca la comunicación con un DataSource. Se utiliza una base de datos Oracle

**PASO Previo:** Es necesario la creación de una tabla llamada personas.

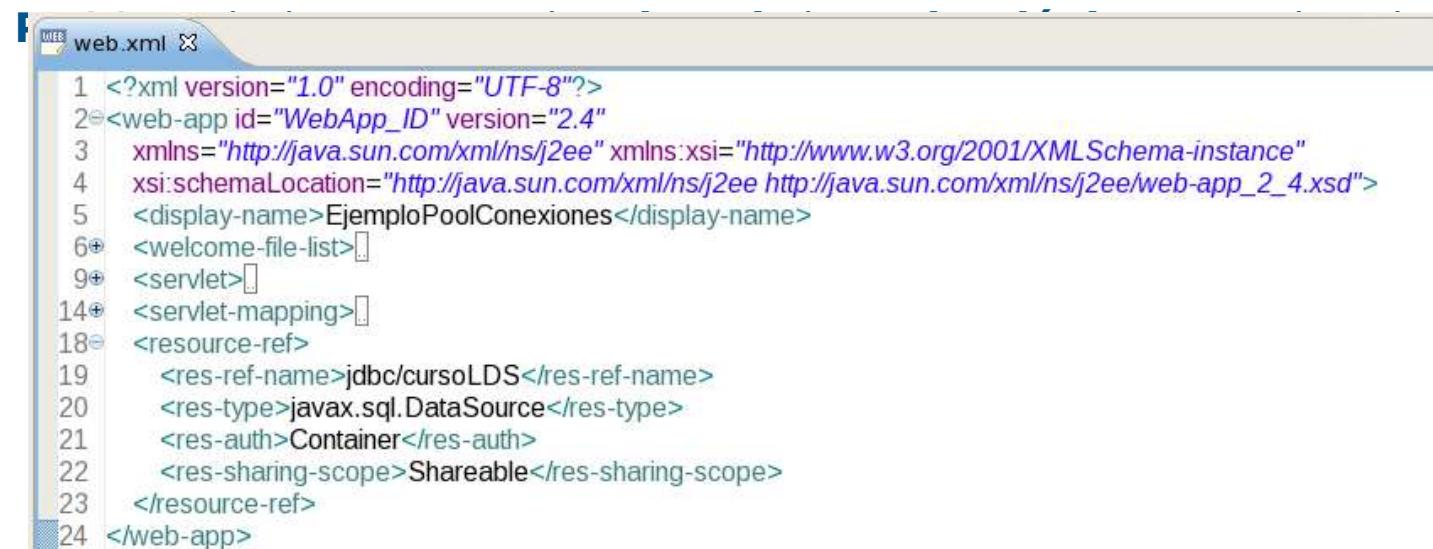


```

script.oracle.sql.ddl ✘
Connection profile
Type: Oracle_11 Name:

1 CREATE TABLE personas (
2 dni VARCHAR2(255) primary key,
3 nombre VARCHAR2(255),
4 apellidos VARCHAR2(255),
5 PRIMARY KEY (dni)
6)

```



```

web.xml ✘
<?xml version="1.0" encoding="UTF-8"?>
<web-app id="WebApp_ID" version="2.4"
 xmlns="http://java.sun.com/xml/ns/j2ee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">
 <display-name>EjemploPoolConexiones</display-name>
 <welcome-file-list></welcome-file-list>
 <servlet></servlet>
 <servlet-mapping></servlet-mapping>
 <resource-ref>
 <res-ref-name>jdbc/cursoLDS</res-ref-name>
 <res-type>javax.sql.DataSource</res-type>
 <res-auth>Container</res-auth>
 <res-sharing-scope>Shareable</res-sharing-scope>
 </resource-ref>
</web-app>

```

# Despliegue Apps con DataSources (II)

**PASO 2:** en nuestra aplicación usamos el nombre del recurso lógico obtenido vía una consulta JNDI.

```

ControlPersonas.java ✘
1 package curso.jboss.mvc.controlador;
2
3+import java.io.IOException;
4
5
6 public class ControlPersonas extends javax.servlet.http.HttpServlet implements javax.servlet.Servlet {
7
8 private static final long serialVersionUID = -115739203944836855L;
9 private Log log=LogFactory.getLog(getClass());
10 private DataSource ds;
11
12+ public void init(ServletConfig config) throws ServletException {
13+ super.init(config);
14+ try {
15+ log.info("buscando el pool de conexiones");
16+ Context ctx=new InitialContext();
17+ //ds=(DataSource)ctx.lookup("java:/ejemploDS");
18+ ds=(DataSource)ctx.lookup("java:comp/env/"+"jdbc/cursoLDS");
19+ } catch (NamingException e) {
20+ e.printStackTrace();
21+ }
22+ }
23+
24+ protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
25+ ...
26+ }
27+ }
```

**Nota.** En nuestra aplicación podemos usar el **GLOBAL JNDI Name** del datasource generado por Jboss, esta opción conduce a problemas en despliegues en plataformas heterogéneas (distintos tipos servidores) y en los distintos entornos (desarrollo, integración, producción, etc)

# Despliegue Apps con DataSources (III)

**PASO 3:** en el descriptor de despliegue **jboss-web.xml** hacemos el mapeo entre el recurso lógico y el recurso real del servidor.



```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE jboss-web PUBLIC
"-//JBoss//DTD Web Application 5.0//EN"
"http://www.jboss.org/j2ee/dtd/jboss-web_5_0.dtd">
<jboss-web>
<context-root>
 MtoPersonas2
</context-root>
<resource-ref>
 <res-ref-name>jdbc/cursoLDS</res-ref-name>
 <jndi-name>java:MySqlDS</jndi-name>
</resource-ref>
</jboss-web>
```

**Nota:** Diferentes aplicaciones pueden compartir un mismo pool de conexiones aunque en producción esta práctica se desaconseja.

**PASO 4:** Monitorizamos el dataSource desde la consola web

# Despliegue Aplicaciones: Módulo EJB 3 (I)

Los descriptores de despliegue se substituyen por **anotaciones** (@) que consisten en añadir la meta-information al código fuente.

Los descriptores XML siguen siendo posibles de utilizar y tienen **mayor precedencia** que las anotaciones.

El servidor de aplicaciones realiza las búsquedas JNDI de la aplicación JEE para los objetos gestionados (Servlets, EJB's, etc...) mediante el mecanismo de **inyección de dependencias** (@Resource, @EJB, etc..)

**JPA** (Java Persistence API) sustituye como tecnología de persistencia a los antiguos Entity Beans EJB 2.x(BMP y CMP). El proveedor de JPA en JBoss es **Hibernate 3**

**Ejemplo Configuración EJB 3.** Crear un EJB de sesión sin estado (SLSB), que utilice JPA para emular un sistema de gestión de pedidos, accesible desde un módulo web de la misma aplicación empresarial.

# Despliegue de aplicaciones: JMS (I)

## Productor Remoto JMS

```

10@ @SuppressWarnings("unchecked")
11 private static InitialContext getInitialContext() throws NamingException {
12 // Creamos un hashtable con los datos necesarios para obtener el contexto inicial
13 Hashtable env = new Hashtable();
14 env.put(Context.INITIAL_CONTEXT_FACTORY,"org.jboss.naming.remote.client.InitialContextFactory");
15 env.put("java.naming.factory.url.pkgs", "org.jboss.naming:org.jnp.interfaces");
16 env.put(Context.PROVIDER_URL, "remote://localhost:4447");
17 env.put(Context.SECURITY_PRINCIPAL, "testuser");
18 env.put(Context.SECURITY_CREDENTIALS, "testpassword");
19 return new InitialContext(env);
20 }

22@ public static void main(String args[]) {
23 try {
24 // Conseguimos de la JNDI los objetos administrados por JBoss. Son la Factoria de conexiones y la Cola
25 InitialContext contextoInicial = getInitialContext();
26 //java:jboss/exported/jms/RemoteConnectionFactory
27 QueueConnectionFactory factory = (QueueConnectionFactory) contextoInicial.lookup("jms/RemoteConnectionFactory");
28 ///java:jboss/exported/queue/MiCola
29 Queue cola = (Queue) contextoInicial.lookup("queue/MiCola");
30 // Creamos la conexion y la sesion
31 QueueConnection conexion = factory.createQueueConnection("testuser","testpassword");
32 QueueSession sesion = conexion.createQueueSession(false,Session.AUTO_ACKNOWLEDGE);
33 // Creamos una sesion de envio
34 QueueSender emisor = sesion.createSender(cola);
35 // Creamos un mensaje JMS de tipo texto
36 TextMessage mensaje = sesion.createTextMessage();
37 mensaje.setText(new Date() + " Esto es un mensaje de texto");
38 // Lo enviamos
39 emisor.send(mensaje);
40 System.out.println("Mensaje enviado: " + mensaje.getText());
41 // Cerramos la conexion
42 conexion.close();
43 } catch (Exception e) {
44 e.printStackTrace();
45 }
46 }
47}
48}

```

# Despliegue de aplicaciones: JMS (II)

## Consumidor JMS

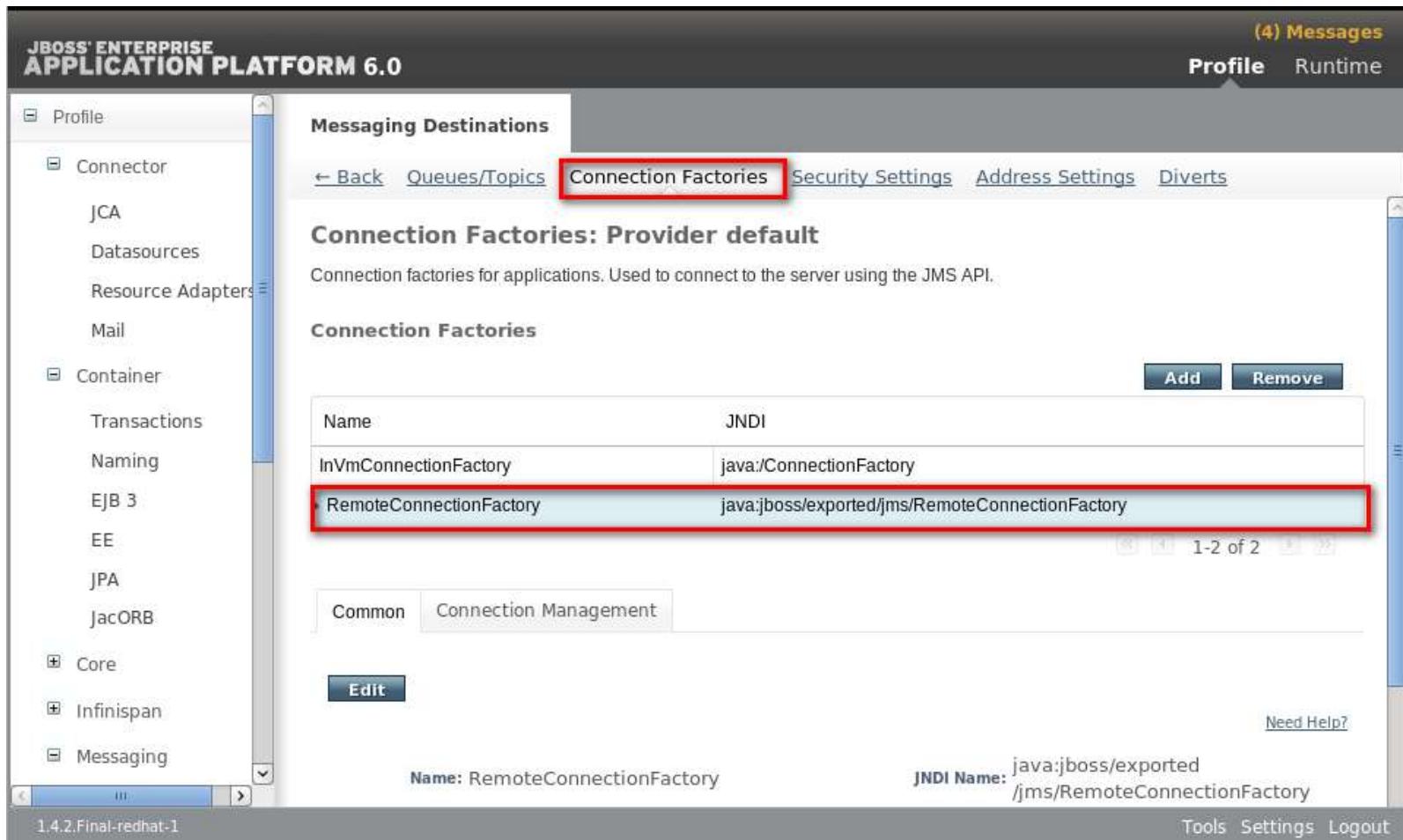
```

OyenteColaJMS.java ✘
1 package curso.jboss.jms;
2
3+import javax.ejb.ActivationConfigProperty;[]
4
5 @MessageDriven(
6 activationConfig = {
7 @ActivationConfigProperty(propertyName = "destinationType", propertyValue = "javax.jms.Queue") ,
8 @ActivationConfigProperty(propertyName="destination", propertyValue="java:jboss/exported/queue/MiCola") ,
9 @ActivationConfigProperty(propertyName = "user", propertyValue = "testuser"),
10 @ActivationConfigProperty(propertyName = "password", propertyValue = "testpassword")
11 })
12 public class OyenteColaJMS implements MessageListener {
13
14 public OyenteColaJMS() {}
15
16 △21+ public void onMessage(Message message) {
17 try {
18
19 //Pasamos el mensaje obtenido a uno de tipo texto para poder leer el contenido
20 TextMessage mensaje= (TextMessage)message;
21 System.out.println("Id del mensaje: "+mensaje.getJMSMessageID());
22 System.out.println("Destino del mensaje: "+mensaje.getJMSDestination());
23 System.out.println("Texto del mensaje: "+mensaje.getText());
24 } catch (JMSEException e) {
25
26 e.printStackTrace();
27 }
28 }
29
30 }
31
32
33
34 }
35
36 }

```

# Despliegue de aplicaciones: JMS (III)

**PASO 1:** verificamos la fábrica de conexiones en:



The screenshot shows the JBoss Enterprise Application Platform 6.0 management console. The left sidebar navigation includes Profile, Connector (JCA, Datasources, Resource Adapters), Mail, Container (Transactions, Naming, EJB 3, EE, JPA, JacORB), Core, Infinispan, and Messaging. The main content area is titled "Messaging Destinations" and "Connection Factories: Provider default". The "Connection Factories" tab is selected. A table lists two connection factories: "InVmConnectionFactory" with JNDI "java:/ConnectionFactory" and "RemoteConnectionFactory" with JNDI "java:jboss/exported/jms/RemoteConnectionFactory". The "RemoteConnectionFactory" row is highlighted with a red box. At the bottom, tabs for "Common" and "Connection Management" are visible, along with an "Edit" button and a "Need Help?" link.

Name	JNDI
InVmConnectionFactory	java:/ConnectionFactory
RemoteConnectionFactory	java:jboss/exported/jms/RemoteConnectionFactory

**Bottom Status Bar:**  
 Name: RemoteConnectionFactory      JNDI Name: java:jboss/exported/jms/RemoteConnectionFactory  
 Tools Settings Logout

# Despliegue de aplicaciones: JMS (IV)

## PASO 2: Creamos los destinos administrados

JBoss' Enterprise Application Platform 6.0

(4) Messages      Profile      Runtime

Profile    Connector    JCA    Datasources    Resource Adapters    Mail    Container    Transactions    Naming    EJB 3    EE    JPA    JacORB    Core    Infinispan    Messaging

Messaging Destinations    ← Back    Queues/Topics    Connection Factories    Security Settings    Address Settings    Diverts

JMS Endpoints: Provider default

Queue and Topic destinations.

Queues    Topics

Add    Remove

Name	JNDI
MiCola	java:jboss/exported/queue/MiCola

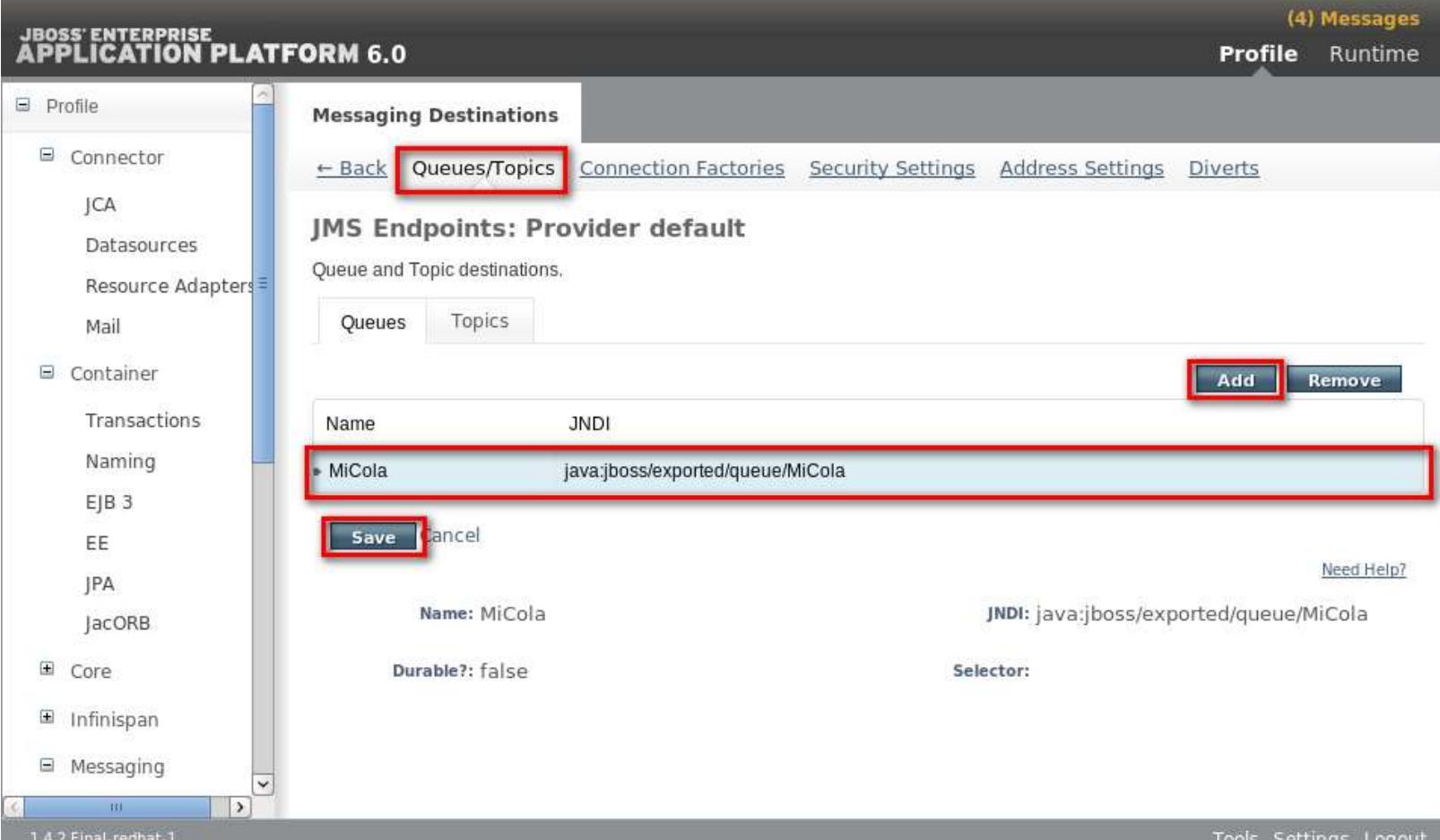
Save    Cancel

Name: MiCola    JNDI: java:jboss/exported/queue/MiCola

Durable?: false    Selector:

Tools    Settings    Logout

1.4.2.Final-redhat-1



# Despliegue de aplicaciones: JMS (IV)

## PASO 3: creamos el rol y el usuario

```

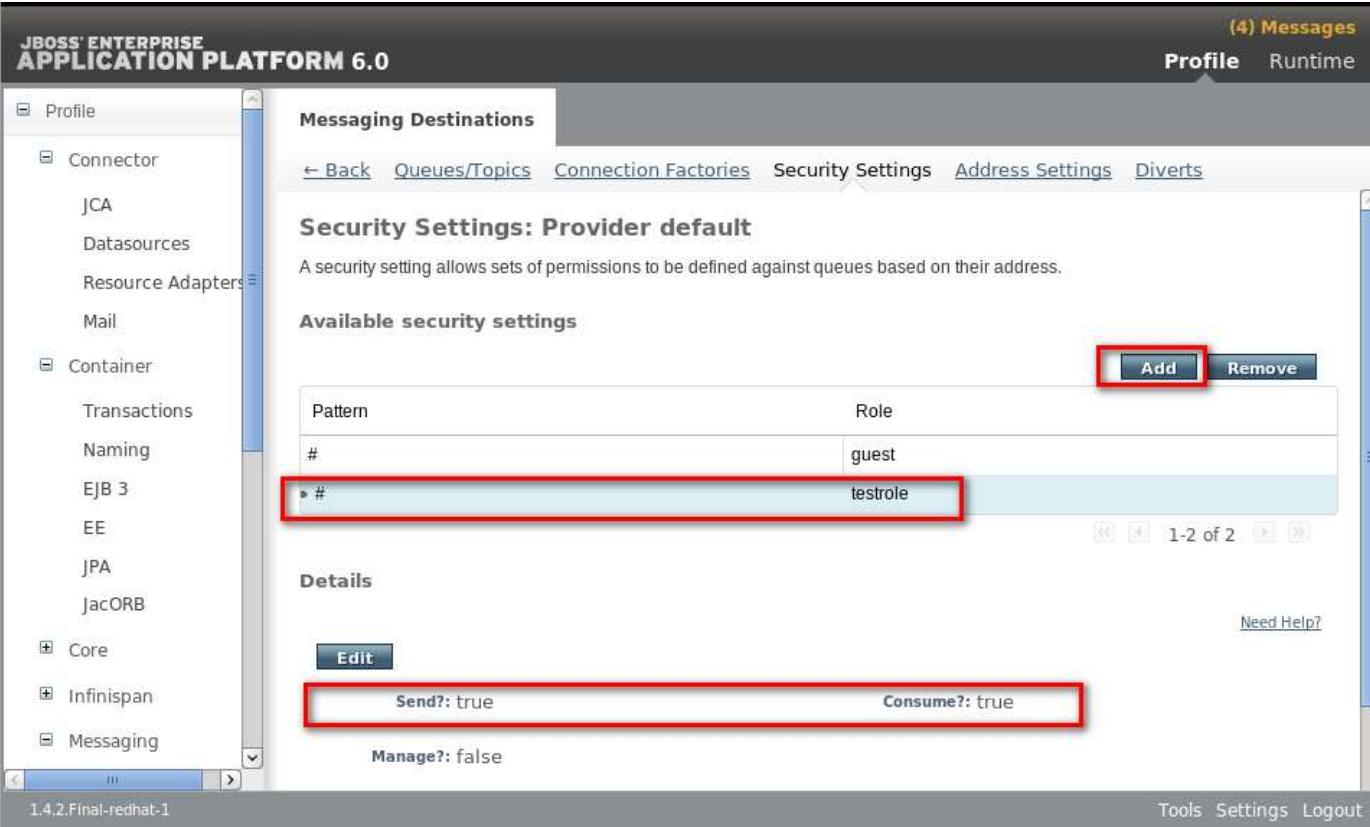
root@galois:/usr/local/jboss-eap-6.0/bin
File Edit View Search Terminal Help
[root@galois bin]# ./add-user.sh
What type of user do you wish to add?
 a) Management User (mgmt-users.properties)
 b) Application User (application-users.properties)
(a): b

Enter the details of the new user to add.
Realm (ApplicationRealm) :
Username : testuser
Password :
Re-enter Password :
What roles do you want this user to belong to? (Please enter a comma separated list, or leave blank for none)[]: testrole
About to add user 'testuser' for realm 'ApplicationRealm'
Is this correct yes/no? y
Added user 'testuser' to file '/usr/local/jboss-eap-6.0/standalone/configuration/application-users.properties'
Added user 'testuser' to file '/usr/local/jboss-eap-6.0/domain/configuration/application-users.properties'
Added user 'testuser' with roles testrole to file '/usr/local/jboss-eap-6.0/standalone/configuration/application-roles.properties'
Added user 'testuser' with roles testrole to file '/usr/local/jboss-eap-6.0/domain/configuration/application-roles.properties'
Is this new user going to be used for one AS process to connect to another AS process?
e.g. for a slave host controller connecting to the master or for a Remoting connection for server to server EJB calls.
yes/no? no
[root@galois bin]#

```

# Despliegue de aplicaciones: JMS (V)

**PASO 4:** Asociamos el rol creado con permisos sobre el sistema de mensajería



The screenshot shows the JBoss Enterprise Application Platform 6.0 administration console. The left sidebar navigation includes Profile, Connector (JCA, Datasources, Resource Adapters), Container (Transactions, Naming, EJB 3, EE, JPA, JacORB), Core, Infinispan, and Messaging. The main content area is titled "Messaging Destinations" under "Security Settings: Provider default". It displays a table of available security settings with columns "Pattern" and "Role". A row for "# guest" is present, and a new row for "# testrole" is selected and highlighted with a red box. At the top right of the table are "Add" and "Remove" buttons, with "Add" also highlighted with a red box. Below the table, the "Details" section shows "Send?: true", "Consume?: true", and "Manage?: false", with "Send?: true" and "Consume?: true" also highlighted with a red box. The top right corner of the interface shows "(4) Messages", "Profile", and "Runtime". The bottom navigation bar includes Tools, Settings, and Logout.

# Despliegue de aplicaciones: JMS (VI)

## PASO 5: Monitorizar los destinos, invocar métodos:

**JBOSS' ENTERPRISE APPLICATION PLATFORM 6.0**

(4) Messages      Profile      Runtime      Refresh

Server      Configuration      Manage Deployments

Status      JVM      Datasources      JPA

**JMS Destinations**      JNDI View      Transactions      Web      Webservices

Runtime Operations      OSGi

Queues      Topics

Queue Selection

Name	JNDI
MiCola	java:jboss/exported/queue/MiCola

1-1 of 1

Messages      Consumer

In-Flight Messages

Metric	Actual	Need Help?
Messages in Queue:	0	
In Delivery:	0	0%

Messages Processed

Metric	Actual	Need Help?
Messages Added:	2	

Tools      Settings      Logout

1.4.2.Final-redhat-1

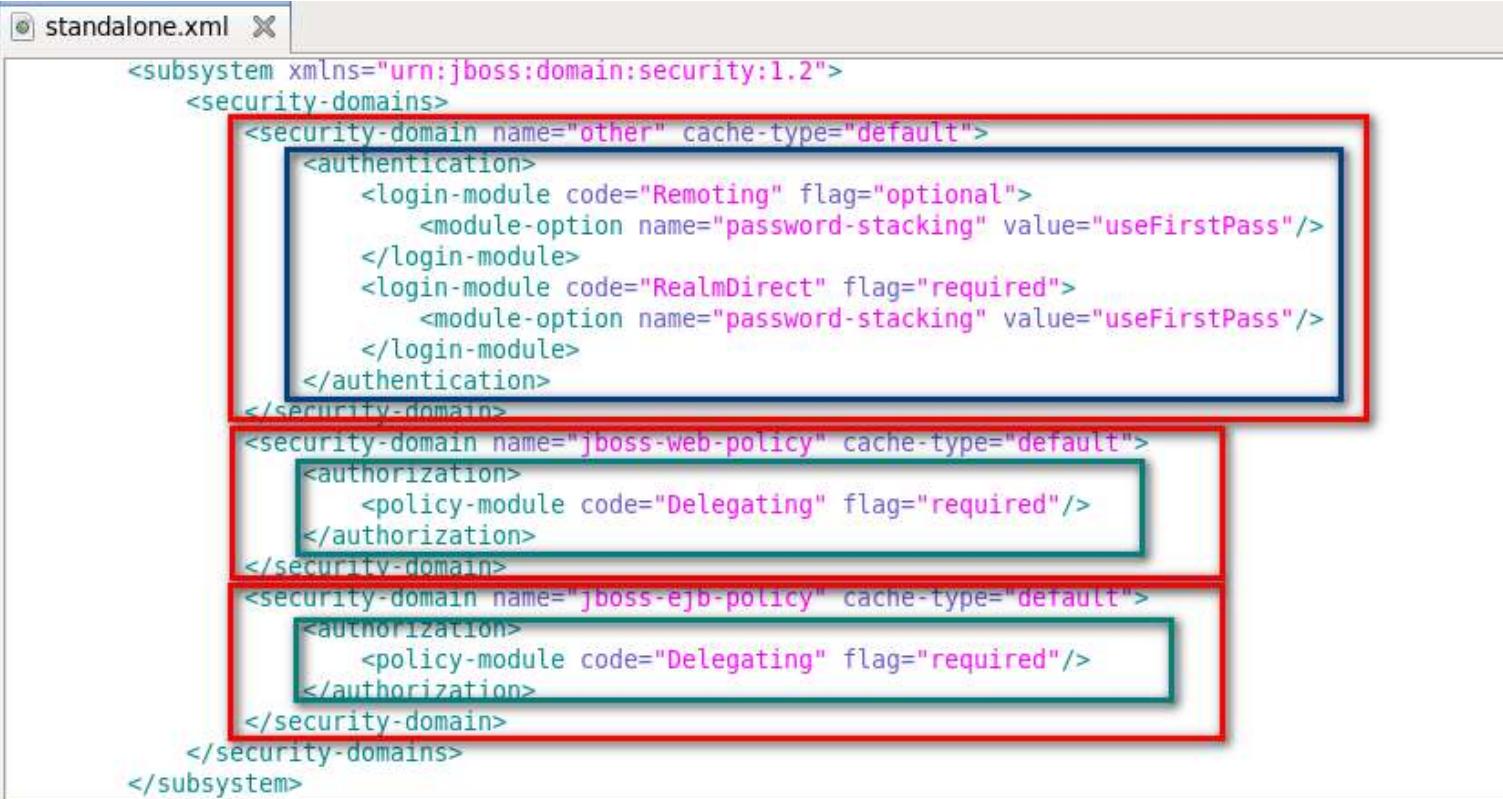
# Contenidos

---

- I. Introducción
- II. Instalación del servidor de aplicaciones
- III. Configuración y herramientas de administración
- IV. Despliegue de aplicaciones
- V. Seguridad de las aplicaciones**
- VI. Clustering
- VII. Optimización

# Modelo de seguridad JBoss (I)

- ⌘ **JBoss Security** está implementado como una extensión para el servidor de aplicaciones incluida por defecto, tanto en servidores autónomos como en servidores de dominio
- <extension module="org.jboss.as.security"/>
- ⌘ La definición de los **dominios** se realiza en el subsistema de seguridad



```
<subsystem xmlns="urn:jboss:domain:security:1.2">
 <security-domains>
 <security-domain name="other" cache-type="default">
 <authentication>
 <login-module code="Remoting" flag="optional">
 <module-option name="password-stacking" value="useFirstPass"/>
 </login-module>
 <login-module code="RealmDirect" flag="required">
 <module-option name="password-stacking" value="useFirstPass"/>
 </login-module>
 </authentication>
 </security-domain>
 <security-domain name="jboss-web-policy" cache-type="default">
 <authorization>
 <policy-module code="Delegating" flag="required"/>
 </authorization>
 </security-domain>
 <security-domain name="jboss-ejb-policy" cache-type="default">
 <authorization>
 <policy-module code="Delegating" flag="required"/>
 </authorization>
 </security-domain>
 </security-domains>
</subsystem>
```

# Modelo de seguridad JBoss (II)

- ⌘ **Ejemplo Autenticación en fichero** podemos crear el siguiente dominio:

```
<security-domain name="MiDominioSeguridad" cache-type="default">
<authentication>
<login-module code="org.jboss.security.auth.spi.UsersRolesLoginModule"
flag="required">
 <module-option name="usersProperties"
value="${jboss.server.config.dir}/ejemplo-users.properties" />
 <module-option name="rolesProperties"
value="${jboss.server.config.dir}/ejemplo-roles.properties" />
 </module-option>
 <module-option name="unauthenticatedIdentity">
anonymous
</module-option>
</login-module>
</application-policy>
</policy>
```

# Modelo de seguridad JBoss (III)

⌘ Autenticación BD un esquema posible para la base de datos:

```
CREATE TABLE Users(username VARCHAR(64) PRIMARY KEY, passwd VARCHAR(64));
CREATE TABLE UserRoles(username VARCHAR(64), userRoles VARCHAR(32));
insert into `Users` (`username`, `passwd`) values('ramon', 'ramon')
insert into `Users` (`username`, `passwd`) values('rosa', 'rosa')
insert into `UserRoles` (`username`, `userRoles`) values('ramon', 'cliente')
insert into `UserRoles` (`username`, `userRoles`) values('rosa', 'personal')
```

⌘ Podemos establecer la configuración en **login-config.xml**:

```
<security-domain name="dbdomain" cache-type="default">
 <authentication>
 <login-module code="Database" flag="required">
 <module-option name="dsJndiName"
 value="java:/MysqlDS"/>
 <module-option name="principalsQuery"
 value="select passwd from Users where username=?"/>
 <module-option name="rolesQuery"
 value="select userRoles, 'Roles' from UserRoles where username=?"/>
 </login-module>
 </authentication>
</security-domain>
```

# Modelo de seguridad JBoss (IV)

⌘ Autenticación LDAP un esquema posible para el directorio:  
Podemos establecer la configuración en **login-config.xml**:

```
<security-domain name="ldapdomain" cache-type="default">
 <authentication>
 <login-module code="LdapExtended" flag="required">
 <module-option name="java.naming.factory.initial" value="com.sun.jndi.ldap.LdapCtxFactory"/>
 <module-option name="java.naming.provider.url" value="ldap://localhost:10389"/>
 <module-option name="java.naming.security.authentication" value="simple"/>
 <module-option name="bindDN" value="uid=admin,ou=system"/>
 <module-option name="bindCredential" value="secret"/>
 <module-option name="baseCtxDN" value="ou=People,dc=example,dc=com"/>
 <module-option name="baseFilter" value="(uid={0})"/>
 <module-option name="rolesCtxDN" value="ou=Roles,dc=example,dc=com"/>
 <module-option name="roleFilter" value="(member={1})"/>
 <module-option name="roleAttributeID" value="cn"/>
 <module-option name="searchScope" value="ONELEVEL_SCOPE"/>
 <module-option name="allowEmptyPasswords" value="true"/>
 </login-module>
 </authentication>
</security-domain>
```

# Modelo de seguridad JBoss (V)

## ⌘Encriptación de contraseñas en DataSources:

- Dada la configuración en **standalone.xml**:

```
<datasource jndi-name="java:/MySqlDS" pool-name="MySqlDS_Pool">
 <connection-url>jdbc:mysql://localhost:3306/test</connection-url>

 <security>
 <security-domain>encrypted-ds</security-domain>
 </security>
</datasource>
```

El usuario inicial era de la forma

```
<security>
 <user-name>root</user-name>
 <password>PasswordXYZ</password>
</security>
```

Ejecutar los siguientes mandatos

```
$ export JBOSS_HOME=/opt/jboss-eap-6.2
```

```
$ export
CLASSPATH=${JBOSS_HOME}/modules/system/layers/base/org/picketbox/main/picketbox-
4.0.19.SP2-redhat-
1.jar:${JBOSS_HOME}/modules/system/layers/base/org/jboss/logging/main/jboss-logging-
3.1.2.GA-redhat-1.jar:$CLASSPATH
```

```
$ java org.picketbox.datasource.security.SecureIdentityLoginModule PasswordXYZ
```

Encoded password: -5bbc51443039e029747687c1d9ec6a8d.

# Modelo de seguridad JBoss (VI)

## ⌘Encriptación de contraseñas en DataSources (cont):

- Crear un nuevo dominio de seguridad en **standalone.xml**:

```
<security-domain name="encrypted-ds" cache-type="default">
 <authentication>
 <login-module
 code="org.picketbox.datasource.security.SecureIdentityLoginModule"
 flag="required">
 <module-option name="username" value="root"/>
 <module-option name="password" value="-
5bbc51443039e029747687c1d9ec6a8d"/>
 <module-option name="managedConnectionFactoryName"
 value="jboss.jca:service=LocalTxCM,name=MySqlDS_Pool"/>
 </login-module>
 </authentication>
 </security-domain>
```

- Probamos el pool desde Jboss-cli

```
[standalone@localhost:9999 /] /subsystem=datalources/data-
source=MySqlDS_Pool:test-connection-in-pool
```

# Activar Keystore y configurar SSL

## ※ Obtención de un certificado

- ❖ Paso 1: **Generar el certificado.** Utilizaremos el keytool de Sun que viene con el JDK.

`keytool -genkey -alias tomcat -keyalg RSA`

El alias es un nombre identificativo del certificado. A continuación aparecen una serie de preguntas. Si estamos instalando el certificado en la máquina local a la pregunta de ¿Cuál es su nombre y apellidos? Deberemos contestar localhost. En caso de ser un dominio público servidor.midominio.com

- ❖ Paso 2: **Configurar el conector SSL** del JBoss.

En **standalone.xml** buscamos:

```
<connector name="http" protocol="HTTP/1.1" scheme="http" socket-binding="http"/>
```

Debajo agregamos

```
•<connector name="https" protocol="HTTP/1.1" scheme="https" socket-
binding="https" enabled="true">
 <ssl key-alias="miclave" password="secreta" certificate-key-
 file="${jboss.server.config.dir}/almacen.jks" cipher-suite="ALL" protocol="TLS"/>
</connector>
```

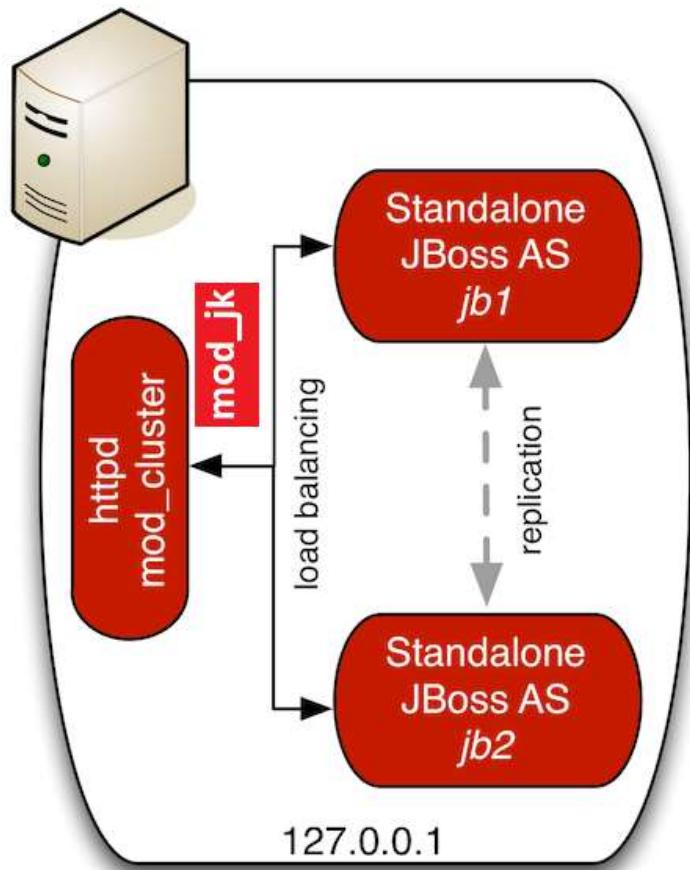
# Contenidos

---

- I. Introducción
- II. Instalación del servidor de aplicaciones
- III. Configuración y herramientas de administración
- IV. Despliegue de aplicaciones
- V. Seguridad de las aplicaciones
- VI. Clustering**

# Creación del Clúster en modo Standalone (I)

⌘ Vamos a crear un clúster en dominio con la siguiente **topología**

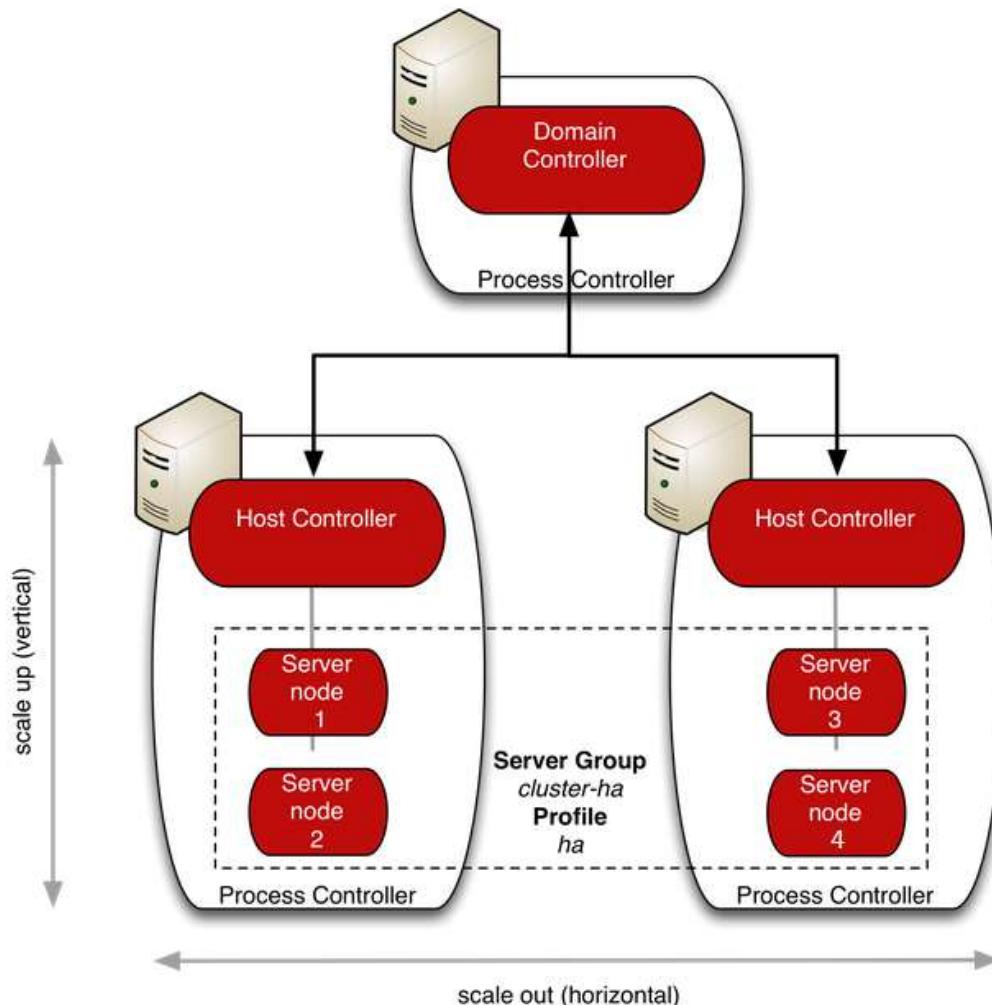


# Creación del Clúster en modo Standalone (II)

- ⌘ Arrancamos dos instancias **standalone** de 2 carpetas diferentes:  
`./standalone.sh -Djboss.server.base.dir=/opt/jboss-eap-6.2/node1  
-Djboss.node.name=node1 -c standalone-ha.xml`
  
- `./standalone.sh -Djboss.server.base.dir=/opt/jboss-eap-6.2/node2  
-Djboss.node.name=node2 -c standalone-ha.xml  
-Djboss.socket.binding.port-offset=100`
  
- ⌘ En el fichero **standalone-ha.xml** de cada una de las instancias para mantener la afinidad web (stickiness) hay que agregar el atributo **instance-id** (antiguo jvmRoute)
  - ⌘ <subsystem xmlns="urn:jboss:domain:web:1.5"  
default-virtual-server="default-host" native="false"  
**instance-id**="\${jboss.node.name}">
  
- ⌘ Creamos el archivo **uriworkermap.properties** mediante el cual asociamos los **Workers** a cada uno de los **context-root** de nuestras aplicaciones y sus recursos:  
`# Simple worker configuration file  
# /cluster=loadbalancer  
# /cluster/*=loadbalancer  
/cluster|/*=loadbalancer`

# Creación del Clúster en dominio (I)

⌘ Vamos a crear un clúster en dominio con la siguiente **topología**



# Creación del Clúster en dominio (II)

- ⌘ **Configurar el controlador del dominio**
- ⌘ Primeramente creamos el usuario administrador del dominio

```
c:\javalibs\jboss-eap-6.0\bin>add-user.bat
¿Qué tipo de usuario desea agregar?
 a) Usuario de administración (mgmt-users.properties)
 b) Usuario de la aplicación (application-users.properties)
(a):

Introduzca los detalles del nuevo usuario a agregar.
Dominio (ManagementRealm) :
Nombre del usuario : admindomain
Contraseña :
Reintroduzca la contraseña :
A punto de agregar el usuario 'admindomain' para el dominio 'ManagementRealm'
Is this correct yes/no? yes
Agregar el usuario 'admindomain' al archivo 'C:\javalibs\jboss-eap-6.0\standalone\configuration\mgmt-users.properties'
Agregar el usuario 'admindomain' al archivo 'C:\javalibs\jboss-eap-6.0\domain\configuration\mgmt-users.properties'
¿Este nuevo usuario se va a utilizar para que un proceso AS se conecte a otro proceso AS?
 por ejemplo: para que un controlador host de esclavos se conecte al maestro o para una conexión remota para llamadas EJB de servidor a servidor.
¿si/no? yes
Para representar el usuario agregue lo siguiente a la definición del servidor-identidades <secret value="amJvc3M=" />
```

- ⌘ Para el **controlador de dominio** usamos el fichero de configuración preconfigurado **host-master.xml** que incluye la configuración necesaria para el host, por ejemplo el enlace de la interfaz física de red de gestión y la configuración JVM y también el fichero domain.xml que viene con unos perfiles por defecto  
`./domain.sh --host-config host-master.xml -b 0.0.0.0 -bmanagement 0.0.0.0`

# Creación del Clúster en dominio (III)

- ⌘ **Configurar el controlador del dominio (cont.)**
- ⌘ Definir un grupo de servidores llamado **cluster-ha** para los servidores que pertenezcan a este grupo. La configuración por defecto incluye ya dos grupos de servidores. Vamos a reemplazarlos en la configuración por la configuración del siguiente grupo para nuestro dominio

```
<domain xmlns="urn:jboss:domain:1.3">
 ...
 <server-groups>
 <server-group name="cluster-ha" profile="ha">
 <jvm name="default"/>
 <socket-binding-group ref="ha-sockets" />
 </server-group>
 </server-groups>
</domain>
```

- ⌘ Cada grupo de servidores necesita un nombre único y una referencia a un perfil de la configuración del dominio.

# Creación del Clúster en dominio (IV)

## ⌘ Configurar los nodos

- ⌘ Lo primero es escoger un nombre único para cada host en nuestro dominio para evitar conflictos de nombres. De lo contrario, el valor predeterminado es el nombre de host del servidor.

```
<host name="host1" xmlns="urn:jboss:domain:1.3">
...
</host>
```

- ⌘ La siguiente configuración consiste en especificar donde se encuentra el controlador de dominio. Para que el controlador de host puede registrarse con el controlador de dominio de sí mismo. La etiqueta **remote** debe incluir el nombre de usuario y el dominio de seguridad del controlador de dominio

```
<host name="host1" xmlns="urn:jboss:domain:1.3">
...
 <domain-controller>
 <remote host="${jboss.domain.master.address}"
 port="${jboss.domain.master.port:9999}"
 username="admindomain"
 security-realm="ManagementRealm"/>
 </domain-controller>
...
</host>
```

# Creación del Clúster en dominio (V)

## ⌘ Configurar los nodos (cont.)

- ⌘ Otra cosa es dar a configurar es el valor secreto. Se necesita para la autenticación con el nombre de usuario. El valor secreto fue la última salida del mandato add-user.sh ejecutado previamente. Este valor es solamente la contraseña codificada en Base64 del usuario del dominio

```
<host name="host1" xmlns="urn:jboss:domain:1.3">
 <management>
 <security-realms>
 <security-realm name="ManagementRealm">
 <server-identities>
 <!-- Replace this with either a base64 password of
 your own, or use a vault with a vault expression -->
 <secret value="amJvc3M=" />
 </server-identities>
 ...
 </security-realm>
 ...
 </security-realms>
 ...
 </management>
 ...
</host>
```

# Creación del Clúster en dominio (VI)

## ⌘ Configurar los nodos (cont.)

- ⌘ Esta es la configuración necesaria para un host. Después de desplegar esta configuración en todos los equipos podemos iniciar el controlador de host en cada máquina con el siguiente comando:  
\$./domain.sh --host-config=host-slave.xml  
-Djboss.domain.master.address=192.168.0.1  
-b 0.0.0.0 -bmanagement 0.0.0.0
- ⌘ El último paso es configurar los servidores dentro del archivo **host-slave.xml** en todas las máquinas. Vamos a reemplazar la configuración existente por la configuración de nuestro servidor en el host perteneciente al grupo de servidores **cluster-ha**  
`<host name="host1" xmlns="urn:jboss:domain:1.3">  
...  
<servers>  
  <server name="server1" group="cluster-ha" auto-start="false"/>  
  <server name="server2" group="cluster-ha" auto-start="false">  
    <!-- server-two avoids port conflicts by incrementing the ports in  
        the default socket-group declared in the server-group -->  
    <socket-bindings port-offset="150"/>  
  </server>  
</servers>  
</host>`

## ⌘ Configuración de la replicación de sesión (HTTP Session State Replication)

- ❖ La configuración únicamente de la sticky session puede no ser suficiente en un sistema en alta disponibilidad. Si el nodo donde se dirigen las peticiones se cae toda la sesión no se podrá balancear a otro nodo y se perderá información y funcionalidad. Para ello es necesario configurar la replicación de sesiones.
- ❖ Para habilitar el clustering (replicación de sesiones) en una aplicación web hay que modificar el **web.xml**:



```
WEB web.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2 <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3 xmlns="http://java.sun.com/xml/ns/javaee" xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
4 xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
5 id="WebApp_ID" version="2.5">
6 <display-name>Uz13.EjemplosReplicacionSesionesWeb</display-name>
7 <distributable />
8 <welcome-file-list>
16 </web-app>
```

# Configuración de aplicaciones Web sobre Cluster (II)

⌘ Además el **jboss-web.xml** podemos configurar:

```
<jboss-web>
 <replication-config>
 <replication-trigger>
 SET_AND_NON_PRIMITIVE_GET
 </replication-trigger>
 <replication-granularity>
 SESSION
 </replication-granularity>
 <replication-field-batch-mode>
 true
 </replication-field-batch-mode>
 </replication-config>
</jboss-web>
```

# Jboss EAP 6

---

Muchas Gracias