



Eusko Jaurlaritzaren Informatika Elkarte
Sociedad Informática del Gobierno Vasco

Manual de tareas ant.

Fecha: 11/11/2009

Referencia:

EJIE S.A.
Mediterráneo, 14
Tel. 945 01 73 00*
Fax. 945 01 73 01
01010 Vitoria-Gasteiz
Posta-kutxatila / Apartado: 809
01080 Vitoria-Gasteiz
www.ejie.es

Este documento es propiedad de EJIE, S.A. y su contenido es confidencial. Este documento no puede ser reproducido, en su totalidad o parcialmente, ni mostrado a otros, ni utilizado para otros propósitos que los que han originado su entrega, sin el previo permiso escrito de EJIE, S.A.. En el caso de ser entregado en virtud de un contrato, su utilización estará limitada a lo expresamente autorizado en dicho contrato. EJIE, S.A. no podrá ser considerada responsable de eventuales errores u omisiones en la edición del documento.



Control de documentación

Título de documento:

Histórico de versiones

Código	Versión	Fecha	Resumen de cambios
	1.0	18/02/2008	
	1.1	23/04/2009	Cambios en tareas test, testImp y nueva tarea borrar_testing.

Cambios producidos desde la última versión

Cambios en tareas test, testImp y nueva tarea borrar_testing.

Control de difusión

Responsable: Ander Martínez

Aprobado por: Begoña Gutiérrez

Firma:

Fecha:

Distribución:

Referencias de archivo

Autor: CAC

Nombre archivo:

Localización:

Contenido

	Capítulo/sección	Página
	Introducción	1
	Sintaxis de ant	2
	Targets	3
	1.1 Targets de compilación y preparación	3
	1.2 Targets de despliegue	5
	1.3 Targets de creación y eliminación de pooles y datasources	7
	1.4 Targets de generación de ficheros jar	8
	1.5 Targets de publicación de Web Services	10
	1.6 Targets para test de la infraestructura en ejecución	13
	1.7 Targets para configuración y creación de componentes JMS	28
	1.8 Targets para habilitar / deshabilitar aplicaciones	29
	1.9 Targets para test de calidad	30
	1.10 Targets para test de implantación	32
	1.11 Targets para test de aplicaciones horizontales	40
	1.12 Targets varios	46

Introducción

El objetivo principal de este documento es informar sobre las diferentes tareas de ant que se pueden utilizar en el entorno de EJIE.

Este documento recoge parte de lo referente a las tareas ant que se encuentra en el documento "WLS810_NormasAlbergue.doc" y la especificación de las nuevas tareas ant creadas actualmente dentro de la propuesta de mejora del entorno de desarrollo planteado por el grupo de Consultoría de Areas de Conocimiento (CAC); estas nuevas tareas ant estarán en color azul.

Además, se han añadido otras tareas existentes y que no estaban reflejadas en el documento "WLS810_NormasAlbergue.doc"; estas tareas estarán en color verde.

Finalmente, hay que remarcar que no se trata de un documento definitivo y que según se vayan desarrollando las nuevas tareas se realizarán las modificaciones pertinentes.

Sintaxis de ant

Puesto que Ant necesita el fichero de configuración `build.xml`, para poder ejecutar las distintas operaciones de configuración hay que 'situarse' en el subdirectorio `java\javs` de la aplicación.

- o **Sintaxis:**

```
ant [options] [target [target2 [target3] ...]]
```

- o **Options:**

- help**, imprime este mensaje.

- projecthelp**, imprime los targets del proyecto.

- version**, imprime la versión.

- diagnostics**, imprime información que puede ser útil para detectar problemas.

- quiet** o **-q**, imprime la información mínima.

- verbose** o **-v**, imprime información adicional.

- debug**, imprime información de debugging.

- emacs**, produce información de logging sin adornos.

- logfile file** o **-l file**, escribe el resultado de la ejecución en un fichero.

- logger classname**, fija la clase que va a implementar el logging.

- listener classname**, añade una instancia de clase como un proyecto de listener.

- buildfile filename** o **-file filename** o **-f filename**, utiliza el buildfile especificado.

- find file**, busca el fichero de configuración, `build.xml` por defecto, desde el directorio actual hacia sus directorios padre, usando el primero que encuentre.

- propertyfile name**, carga todas las propiedades del fichero con la propiedad `-D` que le precede.

- inputhandler class**, se indica la clase que manejará las peticiones de entrada.

- Dproperty=value**, define el valor de una propiedad para el proyecto.

Targets

1.1 Targets de compilación y preparación

- **help**, imprime los targets del proyecto.
- **clean**, borra los dos directorios que se crean durante la compilación. Por un lado borra el directorio `dist`, en el que se genera el fichero `.ear` de la aplicación y por otro lado borra el directorio `exe`, que tiene exactamente la misma estructura que el fichero `.ear`, pero en este directorio el `ear` se encuentra en modo `exploded`.
- **compile**, esta tarea compila las clases java de la aplicación. Si no se pasa ninguna opción se compilarán todas las clases de la aplicación. Opciones disponibles:
 - **path**, indica el subdirectorio a compilar.
 - **filej**, indica el fichero a compilar. Este parámetro no debe incluir la extensión del fichero (p.ej: `aaaHelloWorld`)
 - Combinación de ambos parámetros
- **create**, llama a la tarea `appc` que realiza diferentes acciones dependiendo del tipo de módulo sobre el que actúe. En los módulos `war` comprueba que la configuración del módulo sea válida y precompila los `jsp`s. En los módulos `EJB` comprueba que la configuración del módulo sea válida y genera los `stubs` y los `skeletons`. En los módulos `ear` comprueba que la configuración sea correcta y actúa sobre cada uno de los módulos. Esta tarea es proactiva en su ejecución, por ejemplo si un `jsp` no ha sido modificado no intentará precompilarlo otra o si la interfaz `home` y `remote` de un `EJB` no han cambiado no intentará regenerar los `stubs` y los `skeletons`. Si no se pasa ninguna opción actuará sobre todos los módulos de la aplicación. Opciones disponibles:
 - **pathCreate**, indica el nombre del módulo sobre el que se ejecutará la tarea.
- **all**, compilación completa de la aplicación. No realiza un limpiado previo de las clases compiladas ni hace una compilación de las `JSP`s sino que las copia directamente a la zona `exe` de la aplicación y se compilan en ejecución.
- **compileGeremua**, tarea que compila las clases java de una aplicación para Geremua versión 2
- **allGeremua**, tarea similar al `all` pero para aplicaciones con Geremua versión 2
- **prepare**, esta tarea realiza la preparación del `ear` para un Entorno en concreto (Internet, Intranet). Con esta tarea lo que se permite es de una aplicación en concreto desplegar partes diferentes en distintos entornos de producción. La tarea prepara un fichero `.ear` para el entorno en el que se la invoque en la carpeta de distribución.

Ejemplos de los Targets de compilación y preparación

- Imprime los targets del proyecto:

```
ant -help
```

- Borrado de los directorios creados durante la compilación:

ant **clean**

- Compilar las clases comunes:

ant -Dpath=classes **compile**

- Compilar las clases de un EJB:

ant -Dpath=aaaSaludaEJB **compile**

- Compilar las clases de un War:

ant -Dpath=aaaSaludaWar **compile**

- Compilar las clases de todos los Wars:

ant -Dpath=*War **compile**

- Compilar las clases de todos los EJBs:

ant -Dpath=*EJB **compile**

- Compilar una clase concreta de las comunes:

ant -Dpath=classes -Dfilej=aaaVarios **compile**

- Compilar únicamente la interfaz remota de un EJB:

ant -Dpath=aaaSaludaEJB -Dfilej=aaaSaluda **compile**

- Compilar todos los ficheros java que se encuentran en la aplicación:

ant **compile**

- Compilar todos los ficheros java que se encuentran en la aplicación desarrollada con Geremua versión 2:

ant **compileGeremua**

- Chequear la configuración de un EJB y generar los stubs y skeletons:

ant -DpathCreate=aaaSaludaEJB **create**

- Chequear la configuración de un War y recompilar los jsps modificados:

ant -DpathCreate=aaaSaludaWar **create**

- Chequear la configuración de la aplicación completa y preparar cada uno de los módulos:

ant **create**

- Compilar una aplicación completa (compilar todos los ficheros java, chequeo de las configuraciones, precompilación de los jsps, generación de stubs y skeletons...):

ant **all**

- Compilar una aplicación completa (compilar todos los ficheros java) de manera optimizada:

ant **allGeremua**

- Preparación del ear para el entorno Internet.

ant -Dproduccion=Internet **prepare**

1.2 Targets de despliegue

Estas tareas deben ser invocadas una vez que la tarea `all` o en su defecto `compile` hayan sido ejecutadas satisfactoriamente. La única excepción a lo anteriormente comentado es la tarea `undeploy`.

- **deployExploded**: Esta tarea desplegará la aplicación en modo `exploded` y `no-stage` y su filesystem corresponderá con la carpeta `exe` de la aplicación.
- **redployExplodedAll**: Esta tarea redesplicará toda una aplicación que se encuentra desplegada con la tarea `ant deployExploded`.
- **redployExplodedModule**: Esta tarea redesplicará un módulo de una aplicación que se encuentra desplegada con la tarea `ant deployExploded`. Parámetros:
 - **pathDeploy**, indica el nombre del módulo sobre el que se ejecutará la tarea.
- **redployExplodedFile**: Esta tarea redesplicará un fichero o los todos los ficheros de una ruta relativa a un módulo de una aplicación que se encuentra desplegada con la tarea `ant deployExploded`. Parámetros:
 - **pathDeploy**, indica la ruta del fichero sobre el que se ejecutará la tarea.
- **appStruts**: Tarea utilizada para desplegar aplicaciones con Struts, primero se lanzará esta tarea y luego las utilizadas en aplicaciones normales.
- **undeploy**, `undeploy` de la aplicación completa.

Nota: Se recomienda lanzar las tareas `undeploy` + `deployExploded` cuando existan cambios en la configuración de la aplicación (p.ej: agregar un nuevo módulo `ejb`), en vez de lanzar únicamente la tarea `redployExplodedAll`.

Nota2: Cuando se vaya a realizar un despliegue en modo `exploded` y `no-stage` de una aplicación que ya se encuentra desplegada pero no en esos modos es preferible realizar previamente un `undeploy` de la aplicación.

Ejemplos de los Targets de despliegue

- Despliegue de una aplicación en modo `exploded` y `no-stage`:

```
ant deployExploded
```

- Redesplicue de toda una aplicación que se encuentra desplegada con la tarea `ant deployExploded`

```
ant redployExplodedAll
```

- Redesplicue de un módulo de una aplicación que se encuentra desplegada con la tarea `ant deployExploded`

```
ant -DpathDeploy=s73aUtilsWar redployExplodedModule
```

- Redesplicue de un fichero o todos los ficheros de una ruta relativa a un módulo de una aplicación que se encuentra desplegada con la tarea `ant deployExploded`

```
ant -DpathDeploy=s73aUtilsWar/s73aTramitacionJSP/S73aPruebaRoc.jsp
```

redeployExplodedFile

- Despliegue de una aplicación con Struts, primero se lanzará esta tarea y luego las utilizadas en aplicaciones normales.

```
ant appStruts
```

- Undeploy de la aplicación completa:

```
ant undeploy
```

1.3 Targets de creación y eliminación de pools y datasources

- **createPool**, creación de un pool de conexiones jdbc. Parámetros requeridos:
 - **poolNameBD**, corresponde con el nombre del pool generado. Formato: aaaPool.
 - **userBD**, usuario del usuario en la Base de Datos. Formato: aaa.
 - **passBD**, password del usuario en la Base de Datos.
 - **conexionBD**, nombre generado en el tnsnames. Formato: aaad.
 - **usarCompatibilidad**, si se iguala a true añade la propiedad oracle.jdbc.V8Compatible=true; si se iguala a false o no se pone la propiedad será oracle.jdbc.V8Compatible=false.
- **createDataSource**, creación de un datasource asociado a un pool de conexiones JDBC. Parámetros requeridos:
 - **dataSource**, corresponde con el nombre del DataSource generado. Formato: aaaDataSource.
 - **poolNameBD**, corresponde con el pool sobre el que se generará el DataSource. Formato: aaaPool.
 - **jndiName**, nombre JNDI asociado al DataSource. Formato: aaa.aaaDataSource.
- **deletePool**, eliminación de un pool de conexiones JDBC.
- **deleteDataSource**, eliminación de un DataSource asociado a un pool de conexiones JDBC.

Ejemplos de los Targets de creación y eliminación de pools y datasources

- Creación de un pool de conexiones JDBC:

```
ant -DpoolNameBD=aaaPool -DuserBD=aaa -DpassBD=aaa -DconexionBD=aaad -  
DusarCompatibilidad=true createPool
```

- Creación de un datasource asociado a un pool de conexiones JDBC:

```
ant -DpoolNameBD=aaaPool -DdataSource=aaaDataSource -DjndiName=aaa.aaaDataSource  
createDataSource
```

- Eliminación de un pool de conexiones JDBC:

```
ant -DpoolNameBD=aaaPool deletePool
```

- Eliminación de un datasource asociado a un pool de conexiones JDBC:

```
ant -DdataSource=aaaDataSource deleteDataSource
```

1.4 Targets de generación de ficheros jar

- **jar**, Permite generar un fichero .jar con las clases de una aplicación siguiendo un filtro. En caso de querer acceder desde una aplicación (aaa2) a una clase de otra aplicación (aaa) será necesario usar esta tarea. La idea es que desde la aplicación aaa se genere el fichero .jar con las clases compartidas y éste sea copiado en la carpeta lib de la aplicación aaa2. Parámetros requeridos:
 - **filesJar**, filtro que define las clases a incluir.
 - **nameJar**, parámetro opcional que permite indicar el nombre de salida del fichero .jar. En caso de no indicar este parámetro el fichero de salida se llamará aaa.jar.
- **deleteJar**, Tarea que borra el jar de aplicación, es decir el jar que esta ubicado en /entorno/aplic/aaa/java/javs/aaa.jar. Permite que se le pase como parametro el jar que se quiere borrar. Por ejemplo si no queremos borrar el jar de aplicación pero queremos borrar el jar xxx.jar ubicado bajo /entorno/aplic/aaa/lib/xxx.jar pondremos ant -DPATH_JAR=/lib/xxx.jar deleteJar.
- **ejb-stubs**, Permite generar un fichero .jar con las clases cliente del EJB. En caso de querer acceder desde una aplicación (aaa2) a un EJB de otra aplicación (aaa) será necesario usar esta tarea. La idea es que desde la aplicación aaa se genere el fichero .jar con las clases cliente del EJB y éste sea copiado en la carpeta lib de la aplicación aaa2. Parámetros requeridos:
 - **ejb**, nombre del ejb del que generar el fichero .jar con el cliente.
El fichero con el cliente será generado en la carpeta /aplic/aaa/java/javs y seguirá el formato **ejb_client.jar**.
- **mix-jars**, Permite unir ficheros .jar. Parámetros requeridos:
 - **filesJar**, lista de ficheros .jar.
 - **outputJar**, nombre del fichero .jar generado.
- **ear**, Permite crear el fichero de distribución aaa.ear. Lo que hace esta tarea es crear un jar de la carpeta /aplic/aaa/exe y dejar el fichero en /aplic/aaa/dist con extensión ear. En los pasos a pruebas y producción se envía el fichero ear en vez de la carpeta /aplic/aaa/exe, que estará disponible en la carpeta /explopr/aplic/aaa/dist o /explo/aplic/aaa.

Ejemplos de los Targets de generación de ficheros jar

- Creación de un jar con las clases:

```
ant -DfilesJar=utiles/*,utilesWAR/* jar
```

- Borrado del jar prueba.jar

```
ant -DPATH_JAR=/lib/prueba.jar deleteJar
```

- Borrado del jar de aplicación (aaa.jar)

```
ant deleteJar
```

- Creación del cliente de un EJB:

```
ant -Dejb=aaaSaludaEJB ejb-stubs
```

- Creación de un jar con el contenido de dos jars existentes:

```
ant -DfilesJar=aaaClases.jar,aaaSaludaEJB_client.jar -DoutputJar=aaa.jar mix-jars
```

- Creación del ear de la aplicación:

```
ant ear
```

1.5 Targets de publicación de Web Services

En Weblogic 8 existe la posibilidad de publicar los métodos de una clase o de un EJB como un Web Service. La publicación se basa en la generación de un módulo war y un nombre de servicio asociado a una clase o a un EJB. Los nombres de los módulos war a utilizar como contenedores de Web Services deberán estar definidos en el fichero application.xml como un módulo mas y deberá existir la estructura en disco asociada. La url de acceso a los Web Services publicados estará formada por:

- nombre del sitio web
- nombre del war que contiene el Web Service
- nombre del Web Service

ej: <http://sitioweb/aaaWebService1War/aaaWebService1WS>

A ésta url habrá que realizar peticiones vía JAX-RPC, bien usando el cliente java generado dependiente de Weblogic, uno java independiente u otro cliente (p.ej: acceso desde Visual Basic).

De cara a generar un servicio web desde una clase Java o un EJB existe la posibilidad de que el proceso sea semiautomático (se genera el descriptor del servicio web *web-services.xml*) o que el fichero *web-services.xml* sea suministrado por el desarrollador.

En la Aplicación de Ejemplo existen ejemplos de cada una de las tareas anteriormente comentadas. Las tareas Ant a ejecutar se encuentran en el fichero *webServices.sh* de la carpeta */aplic/aaa/java/javs*.

- Tareas semiautomáticas: el fichero *web-services.xml* es generado por Weblogic mediante introspección por lo que la aplicación deberá estar compilada. Es necesario haber ejecutado las tareas *compile* y *create* satisfactoriamente antes de ejecutar las tareas de publicación. Tras ejecutar las tareas de publicación será necesario invocar las tareas de despliegue, *deployExploded* o *redeployExplodedAll* dependiendo de si es la primera vez, para que los Web Services estén disponibles.
 - **webServiceClaseAuto**, permite publicar una clase como un Web Service. Parámetros requeridos:
 - **war**, nombre del módulo war que contendrá el Web Service. Formato: **aaaNombreWar**.
 - **clases**, clase java a publicar como Web Service.
 - **serviceName**, nombre del servicio a generar. Formato **aaaNombreWS**.
 - **clientPackageName**, nombre del paquete sobre el que se generará el cliente java asociado.
 - **webServiceEJBAuto**, permite publicar un EJB como un Web Service. Parámetros requeridos:
 - **war**, nombre del módulo war que contendrá el Web Service. Formato: **aaaNombreWar**.
 - **ejb**, nombre del ejb a publicar como Web Service.

- **serviceName**, nombre del servicio a generar. Formato **aaaNombreWS**.
 - **clientPackageName**, nombre del paquete sobre el que se generará el cliente java asociado.
- Tareas manuales: se permite que el desarrollador aporte el fichero web-services.xml. Éste fichero deberá ser ubicado en la ruta /aplic/aaa/java/javas/aaaEAR/aaaNombreWar/WEB-INF/web-services.xml del war encargado de publicar el Web Service. Es necesario haber ejecutado las tareas compile y create satisfactoriamente antes de ejecutar las tareas de publicación. Tras ejecutar las tareas de publicación será necesario invocar las tareas de despliegue, deployExploded o redeployExplodedAll dependiendo de si es la primera vez, para que los Web Services estén disponibles.
 - **webServiceClaseConf**, permite publicar una clase como un Web Service. Parámetros requeridos:
 - **war**, nombre del módulo war que contendrá el Web Service. Formato: **aaaNombreWar**.
 - **javacomponents**, clase java a publicar como Web Service. Formato: **package.aaaNombre** sin incluir la extensión .java.
 - **javaSource**, clase java a publicar como Web Service. Formato: **package/aaaNombre.java**.
 - **serviceName**, nombre del servicio a generar. Formato **aaaNombreWS**.
 - **clientPackageName**, nombre del paquete sobre el que se generará el cliente java asociado.
 - **webServiceEJBConf**, permite publicar un EJB como un Web Service. Parámetros requeridos:
 - **war**, nombre del módulo war que contendrá el Web Service. Formato: **aaaNombreWar**.
 - **javacomponents**, clase del ejb a publicar como Web Service. Formato: **package.aaaNombre** sin incluir la extensión .java.
 - **javaSource**, clase java a publicar como Web Service. Formato: **package/aaaNombre.java**.
 - **ejbName**, nombre del ejb a publicar como Web Service.
 - **serviceName**, nombre del servicio a generar. Formato **aaaNombreWS**.
 - **clientPackageName**, nombre del paquete sobre el que se generará el cliente java asociado.
- En todos los casos se genera un cliente webService que se copia en /aplic/aaa/exe/APP-INF/lib para que pueda ser localizado en el classpath y en /aplic/aaa/exe/nombreWarWS/ para que el jar del cliente webService pueda ser descargado directamente por web (<http://maquina:puerto/aaaNombreWar/cliente.jar>). El nombre del fichero jar del cliente esta formado por el nombre del war, el nombre del servicio web y el sufijo _client. P.ej: aaaWebService1War_aaaSaludaWSClase_client.jar

- Para acceder al documento WSDL que describe el Web Service publicado la url será:
 - <http://sitioweb/aaaNombreWar/aaaNombreWS?WSDL>
- Weblogic Server 8.1 genera un cliente web para poder probar los Web Services. La url será:
 - <http://sitioweb/aaaNombreWar/aaaNombreWS>
- Otras tareas de Web Services:
 - **createWSClient**, tarea para generar un cliente de Web Service pasándole la ruta del WSDL

Ejemplos de los Targets de publicación de Web Services

- Publicación de una clase como un Web Service generando la configuración de forma automática:

```
ant -Dwar=aaaWebService1War -Dclases=utiles.aaaSaludaWS -DserviceName=aaaSaludaClase1WS -
DclientPackageName=webService1 webServiceClaseAuto
```

- Publicación de un EJB como un Web Service generando la configuración de forma automática:

```
ant -Dwar=aaaWebService2War -Dejb=aaaSaludaWSEJB -DserviceName=aaaSaludaEJB2WS -
DclientPackageName=webService2 webServiceEJBAuto
```

- Publicación de una clase como un Web Service suministrando la configuración:

```
ant -Dwar=aaaWebService3War -Djavacomponents=utiles.aaaSaludaWS -
DjavaSource=utiles/aaaSaludaWS.java -DserviceName=aaaSaludaClase3WS -
DclientPackageName=webService3 webServiceClaseConf
```

- Publicación de un EJB como un Web Service suministrando la configuración:

```
ant -Dwar=aaaWebService4War -Djavacomponents=aaaSaludaWSEJB.aaaSaludaWS -
DjavaSource=aaaSaludaWSEJB/aaaSaludaWS.java -DejbName=aaaSaludaWSEJB -
DserviceName=aaaSaludaEJB4WS -DclientPackageName=webService4 webServiceEJBConf
```

- Generación de un cliente Web Service pasándole la ruta del WSDL:

```
ant -DrutaWSDL=http://webldes55:7041/s73aUtilsWar/S73aGetSolicitud?WSDL -
Dpaquete=com.ejie.client -DnombreCliente=S73aGetSolicitud createWSClient
```

1.6 Targets para test de la infraestructura en ejecución

- **test**, tarea para recoger información útil para el desarrollador en cuanto a la situación en la que se encuentran todos los componentes que participan en una aplicación. Además esta tarea generará los correspondientes informes dependientes de los parámetros solicitados y visibles a través del dashboard. Los resultados, dependiendo de un parámetro, se pueden mostrar por pantalla o generar un informe visible a través del dashboard.

Parámetro opcional:

- **screen**: Los valores posibles son true (visualiza los resultados por pantalla) o false (genera informe que alimentará al dashboard). Si no se especifica un valor para dicho parámetro el valor por defecto es false, es decir, que alimenta al dashboard y no genera el resultado por pantalla (el dashboard se ha tenido que inicializar ejecutando la tarea `copiar_testing` y el posterior `deployExploded`).

Parámetros requeridos:

- **param**, indica el componente sobre el que se ejecutará la tarea. Los componentes posibles son los siguientes:

- **server**

El objetivo de esta tarea es presentar información relacionada con la instancia en la que está alojada una aplicación así como del estado en el que se encuentra dicha instancia y la máquina física que la alberga.

La información facilitada incluye:

- Instancia de la aplicación
- Dirección del cluster
- Máquinas activas del cluster
- Relación de máquinas que componen el cluster junto con su estado.
- Numero de procesadores
- Carga media de los procesadores (junto con su porcentaje sobre el total)
- Memoria física libre (junto con su porcentaje sobre el total)
- Memoria física usada (junto con su porcentaje sobre el total)
- Memoria física total
- Memoria libre de la máquina virtual (junto con su porcentaje sobre el total)
- Memoria usada de la máquina virtual (junto con su porcentaje sobre el total)
- Memoria total de la máquina virtual
- Versión de Java
- Numero actual de demonios java
- Numero actual de threads java (demonios y no demonios)

Un ejemplo del resultado de la ejecución de esta tarea sería:

```
[java] *****  
[java] ***** SERVIDORES *****  
[java] Instancia de la aplicacion:  apps1
```

```

[java] Direccion del cluster:  webldes55:7041,webldes56:7041
[java] Maquinas activas del cluster:  2
[java]
[java] Servidor:  inter_apps_dpto_apps111 , Estado:  RUNNING
[java] Numero de procesadores:  8
[java] Carga media de los procesadores:  0.125  ( 12.5 %)
[java] Memoria fisica libre:  5236797440  ( 31 %)
[java] Memoria fisica usada:  11572547584  ( 68 %)
[java] Memoria fisica total:  16809365504
[java] Memoria libre de la maquina virtual:  57243224  ( 27 %)
[java] Memoria usada de la maquina virtual:  155626888  ( 74 %)
[java] Memoria total de la maquina virtual:  209715200
[java] Version de Java:  1.4.2_04
[java] Numero actual de demonios Java:  99
[java] Numero actual de threads Java (demonios y no demonios):  126
[java]
[java]
[java] Servidor:  inter_apps_dpto_apps112 , Estado:  RUNNING
[java] Numero de procesadores:  8
[java] Carga media de los procesadores:  0.125  ( 12.5 %)
[java] Memoria fisica libre:  5236801536  ( 31 %)
[java] Memoria fisica usada:  11572555776  ( 68 %)
[java] Memoria fisica total:  16809365504
[java] Memoria libre de la maquina virtual:  57200344  ( 27 %)
[java] Memoria usada de la maquina virtual:  152514856  ( 72 %)
[java] Memoria total de la maquina virtual:  209715200
[java] Version de Java:  1.4.2_04
[java] Numero actual de demonios Java:  97
[java] Numero actual de threads Java (demonios y no demonios):  124

```

Sobre estos resultados se pueden sacar una serie de conclusiones e interpretaciones:

- o Si el valor del estado de la máquina no es RUNNING, esto indica que la máquina no está arrancada correctamente y por lo tanto la aplicación no funcionará.
- o Si cualquiera de las memorias, ya sea la referente a la física como a la de la máquina virtual, tiene un porcentaje de uso cercano al 100% eso significará que se está llegando al límite de capacidad de procesamiento de las peticiones y por lo tanto afectará negativamente a la aplicación.
- o El resto de los datos, como la versión de Java y el número de procesadores son datos meramente informativos sobre la máquina analizada.

- **app**

El objetivo de esta tarea es presentar información general relacionada con los módulos que conforman una aplicación así como datos de si se ha realizado alguna modificación desde el último despliegue y cuándo se ha producido ese último despliegue.

La información facilitada incluye:

- o Nombre de la aplicación
- o Si la aplicación está desplegada o no
- o Cuándo se ha realizado el último despliegue
- o Si se ha modificado algo en el último despliegue
- o Tipo de despliegue
- o Datos de los componentes de aplicación por máquina del cluster junto con su estado.
- o Datos de los componentes web-service por máquina del cluster junto con su estado.
- o Datos de los componentes EJB por máquina del cluster junto con su estado

Un ejemplo del resultado de la ejecución de esta tarea sería:

```
[java] *****
[java] ***** APLICACION *****
[java]
[java] ***** Datos generales de aplicacion
*****
[java] Nombre de la aplicacion: s73a
[java] Desplegado: true
[java] Ultimo despliegue: 1193141672176
[java] Se ha modificado algo con el ultimo despliegue: true
[java] Tipo de despliegue: EAR
[java]
[java] ***** Datos de los componentes de aplicacion
*****
[java] Modulo: s73aL3PrototipoWar Servidor: inter_apps_dpto_apps111 Estado:
Desplegado
[java] Modulo: s73aL3PrototipoWar Servidor: inter_apps_dpto_apps112 Estado:
Desplegado
[java] Modulo: s73aPasarelaWar Servidor: inter_apps_dpto_apps111 Estado:
Desplegado
[java] Modulo: s73aPasarelaWar Servidor: inter_apps_dpto_apps112 Estado:
Desplegado
[java] Modulo: s73aPruebasWar Servidor: inter_apps_dpto_apps111 Estado:
Desplegado
[java] Modulo: s73aPruebasWar Servidor: inter_apps_dpto_apps112 Estado:
Desplegado
[java]
[java] ***** Datos de los componentes web service
*****
[java] Modulo: s73aClienteTramitacionWar Servidor: inter_apps_dpto_apps111
Estado: Desplegado
```

```

[ java ] Modulo: s73aClienteTramitacionWar Servidor: inter_apps_dpto_apps112
Estado: Desplegado

[ java ] Modulo: s73aSHFWar Servidor: inter_apps_dpto_apps111 Estado:
Desplegado

[ java ] Modulo: s73aSHFWar Servidor: inter_apps_dpto_apps112 Estado:
Desplegado

[ java ] Modulo: s73aSistemasTramitacionWar Servidor: inter_apps_dpto_apps111
Estado: Desplegado

[ java ] Modulo: s73aSistemasTramitacionWar Servidor: inter_apps_dpto_apps112
Estado: Desplegado

[ java ] Modulo: s73aUtilsWar Servidor: inter_apps_dpto_apps111 Estado:
Desplegado

[ java ] Modulo: s73aUtilsWar Servidor: inter_apps_dpto_apps112 Estado:
Desplegado

[ java ]

[ java ] ***** Datos de los componentes EJB
*****

[ java ] Modulo: s73aEstudiantesSVTEJB Servidor: inter_apps_dpto_apps111
Estado: Desplegado

[ java ] Modulo: s73aEstudiantesSVTEJB Servidor: inter_apps_dpto_apps112
Estado: Desplegado

[ java ] Modulo: s73aL3PrototipoAccesoBDEJB Servidor: inter_apps_dpto_apps111
Estado: Desplegado

[ java ] Modulo: s73aL3PrototipoAccesoBDEJB Servidor: inter_apps_dpto_apps112
Estado: Desplegado

[ java ] Modulo: s73ajmsmdbEJB Servidor: inter_apps_dpto_apps111 Estado:
Desplegado

[ java ] Modulo: s73ajmsmdbEJB Servidor: inter_apps_dpto_apps112 Estado:
Desplegado

```

Sobre estos resultados se pueden sacar una serie de conclusiones e interpretaciones:

- o Si el valor del estado de un módulo para una instancia es "No desplegado", indicará que ese modulo para esa instancia no funciona y por lo tanto que es necesario su despliegue.
- o El resto de los datos que se aportan en este test como por ejemplo, si está desplegada la aplicación o no, el timestamp de su último despliegue o si se ha realizado alguna modificación desde el último despliegue, son datos meramente informativos.

- **task**

El objetivo de esta tarea es presentar información relacionada con aquellas tareas de la aplicación lanzadas en el weblogic que no han finalizado.

De esta manera se podrá comprobar si el weblogic ha procesado o no las tareas lanzadas y poder detectar así un posible mal funcionamiento de la instancia.

La información facilitada por tarea inacabada incluye:

- o Nombre de la aplicación.
- o Hora de inicio de la tarea.
- o Descripción de la tarea.
- o En qué estado se encuentra la tarea

Un ejemplo del resultado de la ejecución de esta tarea sería:

```
[java] *****  
[java] ***** TAREAS INACABADAS ASOCIADAS A LA  
APLICACION *****  
[java] * Aplicacion: s73a  
[java] * Hora Inicio: 1192613877499  
[java] * Descripcion: [Deployer:149026]Deploy application s73a on appsl.  
[java] * Estado: Deploy Running
```

Sobre estos resultados se pueden sacar una serie de conclusiones e interpretaciones:

- o Si existen tareas inacabadas cuando ha pasado un tiempo prudencial desde el lanzamiento de esa tarea es un síntoma de que el weblogic no está procesando bien esas peticiones y por lo tanto la instancia no está funcionando debidamente. Por cada una de esas tareas inacabadas se presenta información para identificarla.

- **pool**

El objetivo de esta tarea es presentar información relacionada con el pool de conexiones de la aplicación y el estado en el que se encuentra.

La información facilitada por pool desplegado en cada máquina del cluster es:

- o Nombre del pool.
- o Estado del pool
- o Número mínimo de conexiones físicas disponibles
- o Número máximo de conexiones físicas que puede contener
- o URL
- o Conexiones activas
- o Pico máximo de conexiones activas
- o Conexiones disponibles
- o Pico máximo de conexiones disponibles
- o Conexiones no disponibles
- o Pico máximo de conexiones no disponibles
- o Peticiones de conexión en espera
- o Pico máximo de peticiones de conexión en espera
- o Conexiones no cerradas
- o Numero de intentos fallidos de conexión
- o Media de tiempo necesario para establecer una conexión (milisegundos)
- o Numero de conexiones JDBC desde que el pool ha sido instanciado
- o Numero de ocasiones en las que se ha usado una sentencia cacheada para hacer una consulta
- o Numero de ocasiones en las que no se ha usado una sentencia cacheada para hacer una consulta

Un ejemplo del resultado de la ejecución de esta tarea sería:

```
[java] *****
[java] ***** POOL *****
[java] Pool: s73aPool
[java] Numero minimo de conexiones fisicas disponibles: 1
[java] Numero maximo de conexiones fisicas que puede contener: 25
[java] URL: jdbc:oracle:thin:@s73d:1524:edel
[java]
[java] Pool: inter_apps_dpto_apps111 , Estado: Running
[java] Conexiones activas: 0
[java] Pico maximo de conexiones activas: 1
[java] Conexiones disponibles: 1
[java] Pico maximo de conexiones disponibles: 1
[java] Conexiones no disponibles: 0
[java] Pico maximo de conexiones no disponibles: 0
[java] Peticiones de conexion en espera: 0
[java] Pico maximo de peticiones de conexion en espera: 0
[java] Conexiones no cerradas: 42
[java] Numero de intentos fallidos de conexion: 0
[java] Media de tiempo necesario para establecer una conexion (milisegundos):
71
[java] Numero de conexiones JDBC desde que el pool ha sido instanciado: 1
[java] Numero de ocasiones en las que se ha usado una sentencia cacheada para
hacer una consulta: 0
[java] Numero de ocasiones en las que no se ha usado una sentencia cacheada para
hacer una consulta: 2
[java]
[java] Pool: inter_apps_dpto_apps112 , Estado: Running
[java] Conexiones activas: 0
[java] Pico maximo de conexiones activas: 1
[java] Conexiones disponibles: 1
[java] Pico maximo de conexiones disponibles: 1
[java] Conexiones no disponibles: 0
[java] Pico maximo de conexiones no disponibles: 0
[java] Peticiones de conexion en espera: 0
[java] Pico maximo de peticiones de conexion en espera: 0
[java] Conexiones no cerradas: 196
[java] Numero de intentos fallidos de conexion: 0
[java] Media de tiempo necesario para establecer una conexion (milisegundos):
68
[java] Numero de conexiones JDBC desde que el pool ha sido instanciado: 1
[java] Numero de ocasiones en las que se ha usado una sentencia cacheada para
hacer una consulta: 13
```

```
[java] Numero de ocasiones en las que no se ha usado una sentencia cacheada para hacer una consulta: 3
```

Sobre estos resultados se pueden sacar una serie de conclusiones e interpretaciones:

- Si el valor de "Peticiones de conexión en espera" es mayor que cero, o bien el pico máximo de conexiones activas es igual al máximo de conexiones que el pool puede contener, eso nos indica que debido a las necesidades de nuestra aplicación es conveniente aumentar el número máximo de conexiones que el pool puede contener y el nuevo valor debería ser aproximadamente la suma del valor anterior más el "Pico máximo de peticiones de conexión en espera".
- El valor de las conexiones que no han sido cerradas desde que se ha iniciado el pool debería ser cero, ya que en caso contrario, no se están cerrando las conexiones adecuadamente en el código.
- Otros datos que se muestran son informativos como la media de tiempo en realizar una conexión, las ocasiones en las que se usan o no sentencias cacheadas, etc...

- **webapp**

El objetivo de esta tarea es presentar información detallada sobre los componentes web de aplicación que se muestran en la tarea de app.

La información facilitada por componente web de aplicación incluye:

- Nombre del módulo.
- Servidor en el que se encuentra.
- Estado.
- Número total de sesiones abiertas de este componente.
- Pico máximo de sesiones abiertas de este componente desde que se ha iniciado el servidor.

Un ejemplo del resultado de la ejecución de esta tarea sería:

```
[java] *****
[java] ***** Componentes web de aplicacion
*****
[java] Modulo: s73aL3PrototipoWar Servidor: inter_apps_dpto_apps111 Estado:
Desplegado
[java] Numero total de sesiones abiertas de este componente: 1
[java] Pico maximo de sesiones abiertas de este componente desde que se
ha iniciado el servidor: 1
[java]
[java] Modulo: s73aL3PrototipoWar Servidor: inter_apps_dpto_apps112 Estado:
Desplegado
[java] Numero total de sesiones abiertas de este componente: 0
[java] Pico maximo de sesiones abiertas de este componente desde que se
ha iniciado el servidor: 0
[java]
```

```

[java] Modulo: s73aPasarelaWar Servidor: inter_apps_dpto_apps111 Estado:
Desplegado
[java]          Numero total de sesiones abiertas de este componente: 0
[java]          Pico maximo de sesiones abiertas de este componente desde que se
ha iniciado el servidor: 0
[java]
[java] Modulo: s73aPasarelaWar Servidor: inter_apps_dpto_apps112 Estado:
Desplegado
[java]          Numero total de sesiones abiertas de este componente: 0
[java]          Pico maximo de sesiones abiertas de este componente desde que se
ha iniciado el servidor: 0
[java]
[java] Modulo: s73aPruebasWar Servidor: inter_apps_dpto_apps111 Estado:
Desplegado
[java]          Numero total de sesiones abiertas de este componente: 0
[java]          Pico maximo de sesiones abiertas de este componente desde que se
ha iniciado el servidor: 0
[java]
[java] Modulo: s73aPruebasWar Servidor: inter_apps_dpto_apps112 Estado:
Desplegado
[java]          Numero total de sesiones abiertas de este componente: 0
[java]          Pico maximo de sesiones abiertas de este componente desde que se
ha iniciado el servidor: 0
[java]

```

Sobre estos resultados se pueden sacar una serie de conclusiones e interpretaciones:

- o Si el estado de un módulo es "No desplegado" será un indicativo de que ese módulo no se encuentra en funcionamiento y por lo tanto será necesario desplegarlo.
- o El resto de datos son informativos como en el caso del numero de sesiones abiertas de un componente, que nos puede indicar en cuantas ocasiones se accede a un módulo concreto

- **webservice**

El objetivo de esta tarea es presentar información detallada sobre los web services desplegados en nuestra aplicación.

La información facilitada por web service incluye:

- o Nombre del web service.
- o URL del web service.
- o Número de ocasiones que se ha accedido al web service.
- o Número de ocasiones en las que se ha recibido una petición inválida.

Un ejemplo del resultado de la ejecución de esta tarea sería:

```
[java] *****
```

```
[java] ***** Web Services
*****

[java] Nombre: s73aProcessURL@inter_apps_dpto_apps111
[java] Url del WS: http://webldes55:7041/s73aSHFWar/s73aProcessURL
[java] Numero de ocasiones que se ha accedido al web service: 0
[java] Numero de ocasiones en las que se ha recibido una peticion invalida: 0
[java]
[java] Nombre: s73aProcessURL@inter_apps_dpto_apps112
[java] Url del WS: http://webldes56:7041/s73aSHFWar/s73aProcessURL
[java] Numero de ocasiones que se ha accedido al web service: 0
[java] Numero de ocasiones en las que se ha recibido una peticion invalida: 0
```

Todos los datos resultado de esta tarea son meramente informativos

- **ejb**

El objetivo de esta tarea es presentar información detallada sobre los EJBs desplegados en nuestra aplicación.

La información facilitada por EJB incluye:

- o Nombre del EJB.
- o Instancias de bean
- o Instancias de bean usadas
- o Instancias de bean disponibles
- o Veces que ha sido destruida una instancia de bean
- o Veces que ha fallado la creación de una instancia de bean
- o Threads que se han rechazado por time-out
- o Threads que están esperando a una instancia de bean
- o Pico máximo de threads que han estado esperando a una instancia de bean
- o Transacciones que han hecho commit
- o Transacciones que han hecho roll-back
- o Transacciones rechazadas por time-out
- o Beans activados
- o Beans pasivados
- o Beans que están actualmente en la caché
- o Intentos de acceso al bean desde la caché
- o Intentos de acceso al bean exitosos
- o Intentos de acceso al bean fallidos
- o Beans bloqueados
- o Intentos de obtener un bloqueo en un bean
- o Threads que esperan por un bloqueo en un bean
- o Pico máximo de threads que han esperado por un bloqueo en un bean

- o Threads rechazados por time-out esperando a un bloqueo en un bean

Un ejemplo del resultado de la ejecución de esta tarea sería:

```
[java] *****
[java] ***** EJB *****
[java] Elemento:
inter_apps_dpto_apps111_s73a_s73aEstudiantesSVTEJB_S73AEstudiantesSVT
[java]   Instancias de bean: 0
[java]   Instancias de bean usadas: 0
[java]   Instancias de bean disponibles: 0
[java]   Veces que se ha sido destruida una instancia de bean: 0
[java]   Veces que se ha fallado la creacion de una instancia de bean: 0
[java]   Threads que se han rechazado por time-out: null
[java]   Threads que esti¿n esperando a una instancia de bean: null
[java]   Pico maximo de threads que han estado esperando a una instancia de
bean: null
[java]   Transacciones que han hecho commit: 0
[java]   Transacciones que han hecho roll back: 0
[java]   Transacciones rechazadas por time out: 0
[java]   Beans activados: null
[java]   Beans pasivados: null
[java]   Beans que esti¿n actualmente en la cache: null
[java]   Intentos de acceso al bean desde la cache: null
[java]   Intentos de acceso al bean exitosos: null
[java]   Intentos de acceso al bean fallidos: null
[java]   Beans bloqueados: null
[java]   Intentos de obtener un bloqueo en un bean: null
[java]   Threads que esperan por un bloqueo en un bean: null
[java]   Pico maximo de threads que han esperado por un bloqueo en un bean:
null
[java]   Threads rechazados por time out esperando a un bloqueo en un bean:
null
[java]
[java] Elemento:
inter_apps_dpto_apps111_s73a_s73aL3PrototipoAccesoBDEJB_s73aL3PrototipoAccesoBDEJB
[java]   Instancias de bean: 0
[java]   Instancias de bean usadas: 0
[java]   Instancias de bean disponibles: 0
[java]   Veces que se ha sido destruida una instancia de bean: 0
[java]   Veces que se ha fallado la creacion de una instancia de bean: 0
[java]   Threads que se han rechazado por time-out: null
[java]   Threads que esti¿n esperando a una instancia de bean: null
[java]   Pico maximo de threads que han estado esperando a una instancia de
bean: null
```

```

[java] Transacciones que han hecho commit: 0
[java] Transacciones que han hecho roll back: 0
[java] Transacciones rechazadas por time out: 0
[java] Beans activados: null
[java] Beans pasivados: null
[java] Beans que estĩn actualmente en la cache: null
[java] Intentos de acceso al bean desde la cache: null
[java] Intentos de acceso al bean exitosos: null
[java] Intentos de acceso al bean fallidos: null
[java] Beans bloqueados: null
[java] Intentos de obtener un bloqueo en un bean: null
[java] Threads que esperan por un bloqueo en un bean: null
[java] Pico maximo de threads que han esperado por un bloqueo en un bean:
null
[java] Threads rechazados por time out esperando a un bloqueo en un bean:
null
[java]
[java] Elemento:
inter_apps_dpto_apps111_s73a_s73ajmsmdbEJB_s73ajmsmdbEJB_s73a.s73aJMSQueue@s73aJMSSer
ver_apps111
[java] Instancias de bean: null
[java] Instancias de bean usadas: null
[java] Instancias de bean disponibles: null
[java] Veces que se ha sido destruida una instancia de bean: null
[java] Veces que se ha fallado la creacion de una instancia de bean: null
[java] Threads que se han rechazado por time-out: null
[java] Threads que estĩn esperando a una instancia de bean: null
[java] Pico maximo de threads que han estado esperando a una instancia de
bean: null
[java] Transacciones que han hecho commit: null
[java] Transacciones que han hecho roll back: null
[java] Transacciones rechazadas por time out: null
[java] Beans activados: null
[java] Beans pasivados: null
[java] Beans que estĩn actualmente en la cache: null
[java] Intentos de acceso al bean desde la cache: null
[java] Intentos de acceso al bean exitosos: null
[java] Intentos de acceso al bean fallidos: null
[java] Beans bloqueados: null
[java] Intentos de obtener un bloqueo en un bean: null
[java] Threads que esperan por un bloqueo en un bean: null
[java] Pico maximo de threads que han esperado por un bloqueo en un bean:
null
[java] Threads rechazados por time out esperando a un bloqueo en un bean:
null
[java]

```

```

[java] Elemento:
inter_apps_dpto_apps112_s73a_s73aEstudiantesSVTEJB_S73AEstudiantesSVT
[java]   Instancias de bean: 0
[java]   Instancias de bean usadas: 0
[java]   Instancias de bean disponibles: 0
[java]   Veces que se ha sido destruida una instancia de bean: 0
[java]   Veces que se ha fallado la creacion de una instancia de bean: 0
[java]   Threads que se han rechazado por time-out: null
[java]   Threads que esti¿n esperando a una instancia de bean: null
[java]   Pico maximo de threads que han estado esperando a una instancia de
bean: null
[java]   Transacciones que han hecho commit: 0
[java]   Transacciones que han hecho roll back: 0
[java]   Transacciones rechazadas por time out: 0
[java]   Beans activados: null
[java]   Beans pasivados: null
[java]   Beans que esti¿n actualmente en la cache: null
[java]   Intentos de acceso al bean desde la cache: null
[java]   Intentos de acceso al bean exitosos: null
[java]   Intentos de acceso al bean fallidos: null
[java]   Beans bloqueados: null
[java]   Intentos de obtener un bloqueo en un bean: null
[java]   Threads que esperan por un bloqueo en un bean: null
[java]   Pico maximo de threads que han esperado por un bloqueo en un bean:
null
[java]   Threads rechazados por time out esperando a un bloqueo en un bean:
null
[java]
[java] Elemento:
inter_apps_dpto_apps112_s73a_s73aL3PrototipoAccesoBDEJB_s73aL3PrototipoAccesoBDEJB
[java]   Instancias de bean: 0
[java]   Instancias de bean usadas: 0
[java]   Instancias de bean disponibles: 0
[java]   Veces que se ha sido destruida una instancia de bean: 0
[java]   Veces que se ha fallado la creacion de una instancia de bean: 0
[java]   Threads que se han rechazado por time-out: null
[java]   Threads que esti¿n esperando a una instancia de bean: null
[java]   Pico maximo de threads que han estado esperando a una instancia de
bean: null
[java]   Transacciones que han hecho commit: 0
[java]   Transacciones que han hecho roll back: 0
[java]   Transacciones rechazadas por time out: 0
[java]   Beans activados: null
[java]   Beans pasivados: null
[java]   Beans que esti¿n actualmente en la cache: null

```

```

[java] Intentos de acceso al bean desde la cache: null
[java] Intentos de acceso al bean exitosos: null
[java] Intentos de acceso al bean fallidos: null
[java] Beans bloqueados: null
[java] Intentos de obtener un bloqueo en un bean: null
[java] Threads que esperan por un bloqueo en un bean: null
[java] Pico maximo de threads que han esperado por un bloqueo en un bean:
null
[java] Threads rechazados por time out esperando a un bloqueo en un bean:
null
[java]
[java] Elemento:
inter_apps_dpto_apps112_s73a_s73ajmsmdbEJB_s73ajmsmdbEJB_s73a.s73aJMSQueue@s73aJMSSer
ver_apps112
[java] Instancias de bean: null
[java] Instancias de bean usadas: null
[java] Instancias de bean disponibles: null
[java] Veces que se ha sido destruida una instancia de bean: null
[java] Veces que se ha fallado la creacion de una instancia de bean: null
[java] Threads que se han rechazado por time-out: null
[java] Threads que estig/n esperando a una instancia de bean: null
[java] Pico maximo de threads que han estado esperando a una instancia de
bean: null
[java] Transacciones que han hecho commit: null
[java] Transacciones que han hecho roll back: null
[java] Transacciones rechazadas por time out: null
[java] Beans activados: null
[java] Beans pasivados: null
[java] Beans que estig/n actualmente en la cache: null
[java] Intentos de acceso al bean desde la cache: null
[java] Intentos de acceso al bean exitosos: null
[java] Intentos de acceso al bean fallidos: null
[java] Beans bloqueados: null
[java] Intentos de obtener un bloqueo en un bean: null
[java] Threads que esperan por un bloqueo en un bean: null
[java] Pico maximo de threads que han esperado por un bloqueo en un bean:
null
[java] Threads rechazados por time out esperando a un bloqueo en un bean:
null

```

Sobre estos resultados se pueden sacar una serie de conclusiones e interpretaciones aunque hay que tener en cuenta que dependiendo del tipo y del uso que se haga de un EJB algunas de ellas vendrán informadas con el valor "null":

- o Si el resultado de "Veces que ha fallado la creación de una instancia de bean" y "Intentos de acceso al bean fallidos" es superior a cero, habría que analizar más profundamente cuál es su causa.

- El resto de datos son informativos y sirven para poder medir el peso de los EJBs en la aplicación y tomar decisiones al respecto.

- **jms**

El objetivo de esta tarea es presentar información detallada sobre los servidores JMS asociados a nuestra aplicación.

La información facilitada por servidor JMS incluye:

- Nombre del servidor JMS.
- Estado en el que se encuentra.
- Bytes almacenados actualmente.
- Pico máximo de bytes almacenados.
- Bytes pendientes almacenados
- Bytes recibidos
- Mensajes almacenados actualmente
- Pico máximo de mensajes almacenados
- Mensajes pendientes almacenados
- Mensajes recibidos
- Sesiones instanciadas actualmente
- Pico máximo de sesiones instanciadas

Un ejemplo del resultado de la ejecución de esta tarea sería:

```
[java] *****
[java] ***** JMS *****
[java] Nombre: s73aJMSServer_apps111 , Estado: State:HEALTH_OK,ReasonCode:[
[java] Bytes almacenados actualmente: 0
[java] Pico maximo de bytes almacenados: 0
[java] Bytes pendientes almacenados: 0
[java] Bytes recibidos: 0
[java] Mensajes almacenados actualmente: 0
[java] Pico maximo de mensajes almacenados: 0
[java] Mensajes pendientes almacenados: 0
[java] Mensajes recibidos: 0
[java] Sesiones instanciadas actualmente: 0
[java] Pico maximo de sesiones instanciadas: 0
[java]
[java] Nombre: s73aJMSServer_apps112 , Estado: State:HEALTH_OK,ReasonCode:[
[java] Bytes almacenados actualmente: 0
[java] Pico maximo de bytes almacenados: 0
[java] Bytes pendientes almacenados: 0
[java] Bytes recibidos: 0
```

```
[java] Mensajes almacenados actualmente: 0
[java] Pico maximo de mensajes almacenados: 0
[java] Mensajes pendientes almacenados: 0
[java] Mensajes recibidos: 0
[java] Sesiones instanciadas actualmente: 0
[java] Pico maximo de sesiones instanciadas: 0
```

Sobre estos resultados se pueden sacar una serie de conclusiones e interpretaciones:

- Si el estado en el que se encuentra no es "HEALTH OK" entonces el servidor JMS no estará funcionando correctamente y como consecuencia se pueden estar encolando mensajes o no recibándose correctamente o algún otro tipo de error no controlado.
- Si el número de bytes-mensajes pendientes se incrementa exponencialmente eso significa que no se están enrutando bien los mensajes y se quedan en el servidor, por lo que se puede deducir que no se está dando un funcionamiento correcto del servidor JMS.
- El resto de datos que se ofrecen como resultado de la tarea ant son datos meramente informativos que ayudan a hacer un seguimiento de la aplicación.

Ejemplos de los Targets para test de la infraestructura en ejecución

- Testeo y extracción de información sobre los servidores de una aplicación por pantalla

```
ant -Dparam=server -Dscreen=true test
```

- Testeo y extracción de información sobre los módulos de una aplicación que alimenta al dashboard

```
ant -Dparam=app test
```

ó

```
ant -Dparam=app -Dscreen=false test
```

- Testeo y extracción de información sobre las tareas iniciadas y no finalizadas de una aplicación por pantalla

```
ant -Dparam=task -Dscreen=true test
```

- Testeo y extracción de información sobre el pool de conexiones de una aplicación por pantalla

```
ant -Dparam=pool -Dscreen=true test
```

- Testeo y extracción de información sobre los componentes web de aplicación presentes en la aplicación por pantalla

```
ant -Dparam=webapp -Dscreen=true test
```

- Testeo y extracción de información sobre los web services desplegados de una aplicación por pantalla

```
ant -Dparam=webservice -Dscreen=true test
```

- Testeo y extracción de información sobre los EJBs desplegados de una aplicación por pantalla

```
ant -Dparam=ejb -Dscreen=true test
```

- Testeo y extracción de información sobre los servidores JMS asociados a una aplicación por pantalla

```
ant -Dparam=jms -Dscreen=true test
```

1.7 Targets para configuración y creación de componentes JMS

- **createJMSServer**, tarea para crear un servidor JMS. La tarea internamente crea un servidor JMS por cada instancia de weblogic, en el cluster en el que se encuentra desplegada la aplicación.
- **createJMSFactory**, tarea para crear una factoría de conexiones JMS. La tarea crea un factoría JMS a través de la cual podrán crearse los diferentes Destinations, bien sean colas o topics.
- **createJMSQueue**, tarea para la generación de colas JMS. Estas colas se crearán asociadas a un servidor JMS previamente creado.
- **createJMSTopic**, tarea para la generación de topics JMS. Estos topics, al igual que las colas, se crearán asociados a un servidor JMS previamente creado.
- **createJMSDistributedQueue**, tarea para la generación de colas JMS distribuidas.
- **createJMSDistributedTopic**, tarea para la generación de topics JMS distribuidos.
- **deleteJMSQueue**, tarea para el borrado de colas JMS.
- **deleteJMSTopic**, tarea para el borrado de topics JMS.
- **deleteJMSDistributedQueue**, tarea para el borrado de colas JMS distribuidas.
- **deleteJMSDistributedTopic**, tarea para el borrado de topics JMS distribuidos.
- **addJMSQueueMember**, tarea para la asociación de colas a la cola distribuída.
- **addJMSTopicMember**, tarea para la asociación de topics al topic distribuído.

** nota: ninguna de estas tareas recibe parámetros. Internamente se generan los componentes en base a información como: código de aplicación, dominio Weblogic, cluster... y unos patrones de nomenclatura establecidos.

Existe una nota técnica que explica en detalle el uso de estos componentes:

http://elkarlan.ejie/webguneak/cac/espacio_compartido/Notas%20tecnicas/Java/J022.%20Tareas%20ant%20JMS.doc

Ejemplos de los Targets para la configuración y creación de componentes JMS

- Crear un servidor JMS

```
ant createJMSServer
```

- Crear una factoría de conexiones JMS

```
ant createJMSFactory
```

- Crear una cola JMS

ant **createJMSQueue**

- Crear un topic JMS

ant **createJMSTopic**

- Creación de colas JMS distribuidas

ant **createJMSDistributedQueue**

- Creación de topics JMS distribuidos

ant **createJMSDistributedTopic**

- Borrado de colas JMS

ant **deleteJMSQueue**

- Borrado de topic JMS

ant **deleteJMSTopic**

- Borrado de colas JMS distribuidas

ant **deleteJMSDistributedQueue**

- Borrado de topics JMS distribuidos

ant **deleteJMSDistributedTopic**

1.8 Targets para habilitar / deshabilitar aplicaciones

- **habilitar**, tarea para habilitar una aplicación. Esta tarea tomará como referencia un fichero de propiedades, el fichero `aaaApp.properties` sito en el mismo lugar que el `build.xml` (`/aplic/aaa/java/javs`) y en el que estarán indicados los datos necesarios de la aplicación como el usuario y contraseña del pool, todos aquellos datos referentes a las colas JMS si es el caso, etc....

Un ejemplo de este fichero se describe a continuación:

```
# Pool

usuarioPool=t29a

passPool=t29a

hostPool=t29d

# Colas JMS

servidorJMS=t29aJMSServer

colaJMS=t29aJMSQueue

topicJMS=t29aJMSTopic
```

- **deshabilitar**, tarea para deshabilitar una aplicación. Esta tarea tomará como referencia el mismo fichero de propiedades descrito en la tarea de habilitar.

Ejemplos de los Targets para habilitar / deshabilitar aplicaciones

- Habilitar una aplicación

ant **habilitar**

- Deshabilitar una aplicación

ant **deshabilitar**

1.9 Targets para test de calidad

- **junit**, tarea para lanzar la herramienta homologada para las pruebas unitarias JUnit. No requiere de parámetros pero si de otras consideraciones que se encuentran especificadas en el documento "Pruebas unitarias y de regresión. Manual de instalación en servidor".

Esta tarea genera los correspondientes informes de resultados y los copia a la aplicación web del dashboard (se tiene que haber creado previamente con la tarea copiar_testing y el posterior deployExploded).

- **checkstyle**, tarea para lanzar la herramienta homologada para las pruebas de calidad de código Checkstyle. No requiere de parámetros pero si de otras consideraciones como un fichero de reglas, etc... que se encuentran especificados en el documento "Pruebas unitarias y de regresión. Manual de instalación en servidor".

Esta tarea genera los correspondientes informes de resultados y los copia a la aplicación web del dashboard (se tiene que haber creado previamente con la tarea copiar_testing y el posterior deployExploded).

- **pmd**, tarea para lanzar la herramienta homologada para las pruebas de calidad de código PMD. No requiere de parámetros pero si de otras consideraciones como un fichero de reglas, etc... que se encuentran especificados en el documento "Pruebas unitarias y de regresión. Manual de instalación en servidor".

Esta tarea genera los correspondientes informes de resultados y los copia a la aplicación web del dashboard (se tiene que haber creado previamente con la tarea copiar_testing y el posterior deployExploded).

- **aus**, tarea para lanzar la herramienta homologada para identificar dependencias de API's de un producto en otro para crear un conjunto de detallados informes mostrando dónde se encuentran dichas dependencias y cómo están siendo usadas. No requiere de parámetros pero si de otras consideraciones como un fichero de reglas, etc... que se encuentran especificados en el documento "Aus. Manual de Usuario".

Esta tarea genera los correspondientes informes de resultados y los copia a la aplicación web del dashboard (se tiene que haber creado previamente con la tarea copiar_testing y el posterior deployExploded).

- **informeBD**, tarea que genera el informe de base de datos para aplicaciones en Oracle 10. Este informe nos permitirá obtener los indicadores de BBDD. Se consigue lanzando la tarea ant informeBD, un posible ejemplo sería el siguiente.

Los parámetros requeridos para ejecutar esta tarea son:

- o **instanceId**, indica el id de la instancia de BBDD de la aplicación
- o **servidor**, indica el nombre del servidor de BBDD
- o **user**, indica el nombre del usuario en la máquina de BBDD
- o **pass**, indica el password del usuario en la máquina de BBDD
- o **puerto**, indica el puerto de la instancia de BBDD

Para que salgan datos relevantes en este informe, es necesario que se ejecute después de tener movimiento en la aplicación. Los datos para el informe se recopilan cada hora, habrá que comprobar que dentro de este intervalo, se encuentran las pruebas que hemos realizado.

El informe generado estará accesible desde la aplicación web del dashboard.

- **informeJMeter**, tarea que genera el informe de JMeter. Esta tarea tiene como precondition, haber subido el archivo previamente generado a la siguiente carpeta del servidor de aplicaciones:

`/serapp/aplic/aaa/java/javs/aaaEAR/aaaTestingWar/test/informes/nombreArchivo.xml`

Los parámetros necesarios para esta tarea son:

- o **xmlJMeter**, indica el nombre del archivo en el servidor de aplicaciones.

El informe generado estará accesible desde la aplicación web del dashboard.

- **dashboard-ejie**, tarea para generar un entorno global con los resultados de todas las pruebas en servidor en formato HTML. Copia los informes de resultados correspondientes y genera el informe de dashboard y lo empaqueta en un zip y se depositarán en el directorio `/datos/aaa/lis` No requiere de parámetros (ver Dashboard-ejie. Manual rápido de usuario v1.3.doc).

Ejemplos de los Targets para test de calidad

- Testeo de pruebas unitarias - JUnit

```
ant junit
```

- Testeo de pruebas de calidad de código - Checkstyle

```
ant checkstyle
```

- Testeo de pruebas de calidad de código - PMD

```
ant pmd
```

- Testeo de dependencias de APIs - AUS

```
ant aus
```

- Generación del informe de BBDD

```
ant -DinstanceId=ede06 -Dserver=ejhp65 -Duser=aaa -Dpass=aaa -Dpuerto=1234 informeBD
```

- Generación del informe de jmeter

```
ant -DxmlJMeter=nombreArchivo.xml informeJMeter
```

- Generación de un entorno global con los resultados de todas las pruebas.

1.10 Targets para test de implantación

- **testImp**, tarea para lanzar el test de implantación que comprobará una serie de filtros. Los resultados, dependiendo de un parámetro, se pueden mostrar por pantalla o generar un informe visible a través del dashboard.

Parámetro opcional:

- screen: Los valores posibles son true (visualiza los resultados por pantalla) o false (genera informe que alimentará al dashboard). Si no se especifica un valor para dicho parámetro el valor por defecto es false, es decir que alimenta al dashboard y no genera el resultado por pantalla (el dashboard se ha tenido que inicializar ejecutando la tarea copiar_testing).

Los filtros que se compueban son los siguientes:

- FILTRO 1: Chequeo de si existe el filtro de XLNets en todos los web.xml de los War de la aplicación
- FILTRO 2: Chequeo de que están declarados todos los módulos de la aplicación en el application.xml
- FILTRO 3: Se muestran todas las librerías que están puestas en la aplicación. De esta manera se podrán detectar aquellas librerías no homologadas, así como las necesarias para el correcto funcionamiento de Geremua versión 2. Estas librerías necesarias serían:
 - La librería **q702.jar**, única por aplicación, deberá encontrarse en la carpeta lib dentro de la aplicación correspondiente
 - La librería **q701.jar**, que existirá en cada uno de los módulos war de la aplicación, estará a nivel de WEB-INF/lib en cada war
- FILTRO 4: Chequeo del fichero de configuración aaaConfig.properties. Se verificará que existen las propiedades del fichero dominio, adminServerURL y manejado.
- FILTRO 5: Test para la verificación de la estructuración correcta de una aplicación con Geremua version 2. Se verificará que la estructura de proyectos de EJB debe constar de dos carpetas, una de ellas (META-INF), para los descriptors y otra (con el mismo nombre que el paquete) para los interfaces y el bean.
- FILTRO 6: Test para la verificación del uso de Operaciones de Presentación. Se verificará la existencia física del directorio de **OperacionesPresentacion** ubicado dentro de la aplicación en cada módulo war dentro de la carpeta WEB-INF. También se comprobará que cada fichero de tipo **struts-xxx.xml** que se encuentra dentro del directorio OperacionesPresentacion está registrado en el archivo **web.xml**.
- FILTRO 7: Test para la verificación del uso de Operaciones Internas. Se comprobará:
 - La existencia del fichero **configuraciónOI.xml** y que se encuentran registrados en él los nombres de los Ejes.

- La existencia del fichero de mapas correspondiente **mapasAplicacion.xml**.
- FILTRO 8: Test para la verificación de la configuración del servicio de Logs y Trazas. Se comprueba:
 - Que existe una ruta previamente creada en la ubicación **datos/aaa/log**, en la que Geremua en su arranque creará los diferentes ficheros para el volcado de logs y trazas.
 - Que en el fichero **aaa.properties** (config/aaa) existe la ruta **datos/aaa/log**, ya que esta ruta aparecerá en este fichero si se especifican ficheros de logs y trazas propios.
 - La existencia de los ficheros **configuracionLogs_aplicacion.xml** y **configuracionTraza_aplicacion.xml** en la ubicación **config/aaa**.
 - Que en el fichero **GestorArranqueConfiguracion_aplicacion.xml** se encuentran las entradas correspondientes a los componentes de Logs y Trazas. Para ello, se verificará en primer lugar la existencia de este fichero y luego la entrada **FicheroConfiguracion** que tendrá el valor de los ficheros propios de la aplicación para la gestión de Logs y Trazas (**configuracionLogs_aplicacion.xml** y **configuracionTraza_aplicacion.xml**)
- FILTRO 9: Test para la verificación de los ficheros de propiedades de la aplicación. Se comprueba la existencia de los siguientes ficheros:
 - **configuracionCache_aplicacion.xml** -> la aplicación tiene una configuración propia para la gestión de la caché .
 - **configuracionExcepciones.xml** -> la aplicación configura su propio tratamiento de excepciones.
 - **configuracioni18n.xml** -> configuración de la internacionalización de mensajes.
 - **configuracionLogs_aplicacion.xml** -> configuración del servicio de Logs a nivel de aplicación.
 - **configuracionMapeador.xml** -> configuración del fichero de Mapas de la aplicación.
 - **configuracionOI.xml** -> la aplicación dispone de Operaciones Internas para invocar a las diferentes funciones de Negocio.
 - **configuracionTablas_aplicacion.xml** -> para la configuración de catálogos de aplicación

- **configuracionTraza_aplicacion.xml** -> configuración del servicio de Trazas de la aplicación.
 - **contexto-aplicacion.xml** -> configuración del contexto de aplicación, datos compartidos entre diferentes Operaciones de Presentación de la aplicación para un mismo usuario.
 - **contextoGlobalAplicacion_t30b.xml** -> configuración del contexto global de aplicación (información de la aplicación para todos los usuarios de esa aplicación).
 - **GestorArranqueConfiguracion_aplicacion.xml** -> para indicar los componentes aplicación arrancables por parte de Geremua.
 - **mapasAplicacion.xml** -> configuración de los mapas de aplicación.
- FILTRO 10: Test de verificación del uso de componentes de Presentación. Se comprobarán los diferentes aspectos:
- ***ejieFramework-html.tld***: Comprobación de que el taglib *ejieFramework-html.tld* se encuentra en la ruta correcta para poder ser utilizado desde los diferentes wars que compongan la aplicación. Este fichero debe encontrarse en el **WEB-INF/tld** dentro de cada módulo war y es necesario para el uso de componentes como son las Listas Enlazadas, Validador Remoto, etc.
 - ***Display Table Tag***: Comprobación informativa de si se está utilizando o no la librería *displaytag-1.0-b2.jar* ubicada en el WEB-INF/lib de cada módulo war de la aplicación.
 - ***Tiles***: Confirmación de si existe o no el fichero necesario para el uso de Tiles con Geremua: **configuracionTiles.xml** (en la carpeta WEB-INF en cada módulo web).
 - ***Menus***: Verificar si se hace uso de menús comprobando si existe en el directorio WEB-INF de cada war de la aplicación el fichero **configuracionMenus.xml**.
- FILTRO 11: Test de verificación de la configuración del servicio JMS de Geremua. Se comprobarán los diferentes aspectos:
- **Q70JMSServer**: la tarea verificaría la propiedad **Name** del componente tipo *JMSServer* para comprobar que existe un servidor JMS con este nombre (Q70JMSServer). Para comprobar si el servidor JMS está activo o no, se verificaría en *JMSServerRuntime* que exista también.

- **Q70JMSFactory**: componente de tipo *JMSConnectionFactory*. Habría que verificar la propiedad **Name** en *JMSConnectionFactory* para verificar que está creada una factoría de conexiones JMS con el nombre *Q70JMSFactory*.

- **q70contextoGlobalJMSTopic**: componente de tipo *JMSTopic*. Habría que verificar estas propiedades:
 - **Name**: existe un topic con este nombre, comprobando la propiedad.
 - **Parent**: También habría que testear la propiedad *Parent*, para verificar que el topic pertenece al servidor JMS *Q70JMSServer*.

- **q70desconexionUsuarioJMSTopic**: componente de tipo *JMSTopic*. Habría que verificar las siguientes propiedades:
 - **Name**: para verificar que existe un Topic con el nombre *q70desconexionUsuarioJMSTopic Name*.
 - **Parent**: habría que verificar que el topic pertenece al servidor JMS *Q70JMSServer*.

- **Q70ForeignJMSServer**: componente de tipo *ForeignJMSServer*. Habrá que verificar las siguientes propiedades:
 - **Name**: dentro de los elementos de este componente que exista uno en el que la propiedad *Name* sea *Q70ForeignJMSServer*.
 - **ConnectionFactories**: Habrá que verificar también el atributo *ConnectionFactories* para comprobar que existe asociada al servidor JMS la factoría *Q70ForeignJMSFactory*.
 - **Destinations**: para verificar que están incluidos los dos topics necesarios: *Q70contextoGlobalForeignJMSTopic* y *Q70desconexionUsuarioForeignJMSTopic*.
 - **ConnectionURL**: para verificar que existe una URL que el servidor weblogic empleará para conectar con el proveedor del servicio JMS. Se podría mostrar como información resultante del testeo el valor de esta propiedad.

- **Q70ForeignJMSFactory**: componente de tipo *JMSForeignConnecionFactory*. Se comprobarán estas propiedades:
 - **Name**: que exista una factoría con el nombre de *Q70ForeignJMSFactory*.

- **Parent:** este atributo tendrá que contener el nombre del servidor JMS al que pertenece esta factoría (Q70ForeignJMSServer).
 - **RemoteJNDIName:** factoría de conexiones remota (en el servidor de la *ConnectionURL*).
- **q70contextoGlobalForeignJMSTopic:** componente de tipo ForeignJMSTopic. Se verificarán estas propiedades:
 - **Name:** que exista un elemento de tipo ForeignJMSTopic con este nombre (q70contextoGlobalForeignJMSTopic).
 - **Parent:** tendrá que contener el nombre del servidor JMS en el que está ubicado este topic (Q70ForeignJMSServer).
 - **RemoteJNDIName:** habrá que comprobar que la referencia JNDI del árbol en el servidor remoto (ConnectionURL) se corresponda con el valor de esta propiedad.
- **q70desconexionUsuarioForeignJMSTopic:** componente de tipo ForeignJMSTopic. Se verificarán las siguientes propiedades:
 - **Name:** que exista un elemento de tipo ForeignJMSTopic con este nombre (q70contextoGlobalForeignJMSTopic).
 - **Parent:** tendrá que contener el nombre del servidor JMS en el que está ubicado este topic (Q70ForeignJMSServer).
 - **RemoteJNDIName:** habrá que comprobar que la referencia JNDI en el árbol en el servidor remoto (ConnectionURL) se corresponda con el valor de esta propiedad (q70.q70desconexionUsuarioJMSTopic)

Un ejemplo del resultado de la ejecución por pantalla de esta tarea sería:

```
testImp:
[testImpTask] ***** FILTRO 1: Chequeo de si existe el filtro de XLNets en todos los
web.xml de los War de la aplicacion
[testImpTask] El fichero: /aplic/t29a/java/javs/t29aEAR/t29aTestingWar/WEB-
INF/web.xml NO tiene el filtro de XLNets
[testImpTask] El fichero: /aplic/t29a/java/javs/t29aEAR/t29aWar/WEB-INF/web.xml NO
tiene el filtro de XLNets
[testImpTask]
```

```
[testImpTask] ***** FILTRO 2: Chequeo de que estan declarados todos los modulos de la
aplicacion en el application.xml

[testImpTask] El fichero: /aplic/t29a/java/javas/t29aEAR/META-INF/application.xml SI
tiene el modulo t29aTestingWar

[testImpTask] El fichero: /aplic/t29a/java/javas/t29aEAR/META-INF/application.xml SI
tiene el modulo t29aWar

[testImpTask] El fichero: /aplic/t29a/java/javas/t29aEAR/META-INF/weblogic-
application.xml SI tiene el modulo t29aTestingWar

[testImpTask] El fichero: /aplic/t29a/java/javas/t29aEAR/META-INF/weblogic-
application.xml SI tiene el modulo t29aWar

[testImpTask]

[testImpTask] ***** FILTRO 3: Se muestran todas las librerias que estan puestas en la
aplicacion

[testImpTask] Libreria: /aplic/t29a/java/javas/t29aEAR/t29aTestingWar/WEB-
INF/lib/junit.jar

[testImpTask] Libreria: /aplic/t29a/lib/junit.jar

[testImpTask]

[testImpTask] ***** FILTRO 4: Chequeo del fichero de configuracion
aaaConfig.properties

[testImpTask] El fichero t29aConfig.properties SI contiene los atributos dominio,
adminServerURL y manejado

[testImpTask]

[testImpTask] ***** FILTRO 5: Test para verificacion de estructuracion correcta de
una aplicacion con Geremua version 2

[testImpTask] Para el EJB t29aPetStoreEJB NO existe un subdirectorio t29aPetStoreEJB

[testImpTask] Para el EJB t29aPetStoreEJB SI existe un subdirectorio META-INF

[testImpTask]

[testImpTask] ***** FILTRO 6: Test para verificacion de uso de Operaciones de
Presentacion

[testImpTask] NO existe la carpeta de operaciones de presentacion para el WAR
t29aTestingWar

[testImpTask] SI existe la carpeta de operaciones de presentacion para el WAR t29aWar

[testImpTask] El struts-config.xml SI existe en el web.xml del WAR t29aWar

[testImpTask] El struts-shop.xml SI existe en el web.xml del WAR t29aWar

[testImpTask]

[testImpTask] ***** FILTRO 7: Test para verificacion de uso de Operaciones Internas

[testImpTask] SI existe el fichero configuracionOI.xml

[testImpTask] El EJB t29aPetStoreEJB NO esta registrado en el fichero
configuracionOI.xml

[testImpTask] SI Existe el fichero de mapasAplicacion.xml

[testImpTask]

[testImpTask] ***** FILTRO 8: Test para verificacion de configuracion de servicio de
Logs y Trazas

[testImpTask] SI existe el fichero t29a.properties

[testImpTask] NO existe la ruta datos/t29a/log en el fichero t29a.properties

[testImpTask] NO Existe el fichero de configuracionLogs_aplicacion.xml

[testImpTask] NO Existe el fichero de configuracionTraza_aplicacion.xml
```

```

[testImpTask] NO existe la entrada configuracionLogs_aplicacion.xml correspondiente
al componente de logs en el fichero GestorArranqueConfiguracion_aplicacion.xml

[testImpTask] NO existe la entrada configuracionTraza_aplicacion.xml correspondiente
al componente de trazas en el fichero GestorArranqueConfiguracion_aplicacion.xml

[testImpTask]

[testImpTask] ***** FILTRO 9: Test para verificacion de ficheros de propiedades de la
Aplicacion

[testImpTask] NO Existe el fichero de configuracionCache_aplicacion.xml
[testImpTask] NO Existe el fichero de configuracionExcepciones_aplicacion.xml
[testImpTask] NO Existe el fichero de configuracioni18n_aplicacion.xml
[testImpTask] NO Existe el fichero de configuracionLogs_aplicacion.xml
[testImpTask] NO Existe el fichero de configuracionMapeador_aplicacion.xml
[testImpTask] NO Existe el fichero de configuracionOI_aplicacion.xml
[testImpTask] NO Existe el fichero de configuracionTablas_aplicacion.xml
[testImpTask] NO Existe el fichero de configuracionTraza_aplicacion.xml
[testImpTask] NO Existe el fichero de contexto-aplicacion_aplicacion.xml
[testImpTask] SI Existe el fichero de contextoGlobalAplicacion_t29a.xml
[testImpTask] SI Existe el fichero de GestorArranqueConfiguracion_aplicacion.xml
[testImpTask] SI Existe el fichero de mapasAplicacion.xml

[testImpTask]

[testImpTask] ***** FILTRO 10: Test de verificacion de uso de componentes de
Presentacion

[testImpTask] NO Existe el fichero ejieFramework-html.tld en el WAR t29aTestingWar
[testImpTask] SI Existe el fichero ejieFramework-html.tld en el WAR t29aWar
[testImpTask] NO Existe el fichero displaytag-1.0-b2.jar en el WAR t29aTestingWar
[testImpTask] NO Existe el fichero displaytag-1.0-b2.jar en el WAR t29aWar
[testImpTask] NO Existe el fichero configuracionTiles.xml en el WAR t29aTestingWar
[testImpTask] SI Existe el fichero configuracionTiles.xml en el WAR t29aWar
[testImpTask] NO Existe el fichero configuracionMenus.xml en el WAR t29aTestingWar
[testImpTask] NO Existe el fichero configuracionMenus.xml en el WAR t29aWar

[ java] wslsh [not connected]> @ connect

[ java] wslsh inter_apps_dpto:/> if $CONNECTED then
[ java]
[ java]
[ java] ***** FILTRO 11: Test de verificacion de configuracion de servicio JMS de
Geremua
[ java]
[ java] SI existe q70JMSServer
[ java] q70JMSServer SI esta activo
[ java]
[ java] SI existe q70JMSFactory
[ java]
[ java] SI existe q70contextoGlobalJMSTopic

```

```

[ java ] Parent:   inter_apps_dpto:Name=q70JMSServer,Type=JMSServer
[ java ]
[ java ] SI existe q70desconexionUsuarioJMSTopic
[ java ] Parent:   inter_apps_dpto:Name=q70JMSServer,Type=JMSServer
[ java ]
[ java ] SI existe Q70ForeignJMSServer

[ java ] ConnectionFactories:
[inter_apps_dpto:ForeignJMSServer=Q70ForeignJMSServer,Name=Q70ForeignJMSFactory,Type=
ForeignJMSConnectionFactory]

[ java ] Destinations:
[inter_apps_dpto:ForeignJMSServer=Q70ForeignJMSServer,Name=q70contextoGlobalForeignJM
STopic,Type=ForeignJMSDestination,
inter_apps_dpto:ForeignJMSServer=Q70ForeignJMSServer,Name=q70desconexionUsuarioForeig
nJMSTopic,Type=ForeignJMSDestination]

[ java ] ConnectionURL:   t3://webldes57:8003
[ java ]
[ java ] SI existe Q70ForeignJMSFactory
[ java ] Parent:   inter_apps_dpto:Name=Q70ForeignJMSServer,Type=ForeignJMSServer
[ java ] RemoteJNDIName:  q70.q70JMSFactory
[ java ]
[ java ] SI existe q70contextoGlobalForeignJMSTopic
[ java ] Parent:   inter_apps_dpto:Name=Q70ForeignJMSServer,Type=ForeignJMSServer
[ java ] RemoteJNDIName:  q70.q70contextoGlobalJMSTopic
[ java ]
[ java ] SI existe q70desconexionUsuarioForeignJMSTopic
[ java ] Parent:   inter_apps_dpto:Name=Q70ForeignJMSServer,Type=ForeignJMSServer
[ java ] RemoteJNDIName:  q70.q70desconexionUsuarioJMSTopic
[ java ]

[ java ] wslsh [not connected]> quit
[ java ] bye!

```

BUILD SUCCESSFUL

Ejemplos de los Targets para test de implantación

- Lanzar el test de Implantación que alimenta al dashboard

```
ant testImp
```

```
ant testImp -Dscreen=false
```

- Lanzar el test de Implantación por pantalla

```
ant testImp -Dscreen=true
```

1.11 Targets para test de aplicaciones horizontales

- **testLote4**, Tarea para recoger información sobre el estado de los servidores que alberga las aplicaciones de lote4; si están desplegadas dichas aplicaciones y sus pools de conexiones así como el estado y datos sobre los servidores JMS y las colas asociadas.

La información presentada en esta tarea se da por cada uno de los dominios en los que están alojadas las aplicaciones de lote4; de esta manera y por cada dominio se presenta la siguiente información:

- Por cada una de las máquinas del cluster:
 - Nombre de la máquina y su estado.
 - Numero de procesadores
 - Carga media de los procesadores (junto con su porcentaje sobre el total)
 - Memoria física libre (junto con su porcentaje sobre el total)
 - Memoria física usada (junto con su porcentaje sobre el total)
 - Memoria física total
 - Memoria libre de la máquina virtual (junto con su porcentaje sobre el total)
 - Memoria usada de la máquina virtual (junto con su porcentaje sobre el total)
 - Memoria total de la máquina virtual
 - Versión de Java
 - Numero actual de demonios java
 - Numero actual de threads java (demonios y no demonios)
- Por cada una de las aplicaciones:
 - Nombre de la aplicación y si está desplegada
 - Nombre del pool y se está desplegado por cada una de las instancias.
- Por cada servidor JMS:
 - Nombre del servidor y estado
 - Conexiones actuales
 - Pico máximo de conexiones
 - Conexiones totales
- Por cada cola JMS
 - Nombre de la cola y estado
 - Bytes almacenados actualmente
 - Pico máximo de bytes almacenados
 - Bytes pendientes almacenados
 - Bytes recibidos
 - Mensajes almacenados actualmente
 - Pico máximo de mensajes almacenados
 - Mensajes pendientes almacenados
 - Mensajes recibidos

- Sesiones instanciadas actualmente
- Pico máximo de sesiones instanciadas

Un ejemplo del resultado de la ejecución de esta tarea sería:

```
[java] ***** INTER_TELETRAMITACION
*****

[java] * Maquinas activas del cluster: 2
[java] Servidor: inter_teletramitacion_apps111 , Estado: RUNNING
[java] Numero de procesadores: 8
[java] Carga media de los procesadores: 0.03453596247068028 ( 3.4535962470680284 %)
[java] Memoria fisica libre: 7993020416 ( 24 %)
[java] Memoria fisica usada: 24681611264 ( 75 %)
[java] Memoria fisica total: 32674631680
[java] Memoria libre de la maquina virtual: 428608032 ( 79 %)
[java] Memoria usada de la maquina virtual: 108262880 ( 20 %)
[java] Memoria total de la maquina virtual: 536870912
[java] Version de Java: 1.4.2_10
[java] Numero actual de demonios Java: 99
[java] Numero actual de threads Java (demonios y no demonios): 110
[java]

[java] Servidor: inter_teletramitacion_apps112 , Estado: RUNNING
[java] Numero de procesadores: 8
[java] Carga media de los procesadores: 0.034544254317692445 ( 3.4544254317692444 %)
[java] Memoria fisica libre: 7993020416 ( 24 %)
[java] Memoria fisica usada: 24681611264 ( 75 %)
[java] Memoria fisica total: 32674631680
[java] Memoria libre de la maquina virtual: 131995936 ( 24 %)
[java] Memoria usada de la maquina virtual: 404874976 ( 75 %)
[java] Memoria total de la maquina virtual: 536870912
[java] Version de Java: 1.4.2_10
[java] Numero actual de demonios Java: 100
[java] Numero actual de threads Java (demonios y no demonios): 111
[java]

[java] ***** APLICACIONES *****
[java]

[java] * Aplicacion: r02f , Desplegado: true
[java]

[java] * Aplicacion: r02k , Desplegado: true
[java]

[java] * Aplicacion: r02n , Desplegado: true
[java]
```

```

[java] ***** SERVIDORES JMS *****
[java] Servidor: inter_teletramitacion_apps111.jms , Estado:
State:HEALTH_OK,ReasonCode:[
[java] Conexiones actuales: 0
[java] Pico maximo de conexiones: 0
[java] Conexiones totales: 0
[java]
[java] Servidor: inter_teletramitacion_apps112.jms , Estado:
State:HEALTH_OK,ReasonCode:[
[java] Conexiones actuales: 0
[java] Pico maximo de conexiones: 0
[java] Conexiones totales: 0
[java]
[java]
[java] ***** COLAS JMS *****

[java] wslsh [not connected]> quit
[java] bye!

[java] wslsh [not connected]> @ connect

[java] wslsh intra_teletramitacion:/> if $CONNECTED then
[java]
[java] ***** INTRA_TELETRAMITACION
*****
[java] * Maquinas activas del cluster: 2
[java] Servidor: intra_teletramitacion_apps111 , Estado: RUNNING
[java] Numero de procesadores: 8
[java] Carga media de los procesadores: 0.2023564049226158 ( 20.23564049226158 %)
[java] Memoria fisica libre: 13916573696 ( 38 %)
[java] Memoria fisica usada: 21831237632 ( 61 %)
[java] Memoria fisica total: 35747811328
[java] Memoria libre de la maquina virtual: 281375112 ( 26 %)
[java] Memoria usada de la maquina virtual: 792366712 ( 73 %)
[java] Memoria total de la maquina virtual: 1073741824
[java] Version de Java: 1.4.2_10
[java] Numero actual de demonios Java: 100
[java] Numero actual de threads Java (demonios y no demonios): 122
[java]
[java] Servidor: intra_teletramitacion_apps112 , Estado: RUNNING
[java] Numero de procesadores: 8
[java] Carga media de los procesadores: 0.20234870465476903 ( 20.234870465476902 %)
[java] Memoria fisica libre: 13916573696 ( 38 %)
[java] Memoria fisica usada: 21831237632 ( 61 %)

```

```
[java] Memoria fisica total: 35747811328
[java] Memoria libre de la maquina virtual: 346995336 ( 32 %)
[java] Memoria usada de la maquina virtual: 726746488 ( 67 %)
[java] Memoria total de la maquina virtual: 1073741824
[java] Version de Java: 1.4.2_10
[java] Numero actual de demonios Java: 100
[java] Numero actual de threads Java (demonios y no demonios): 122
[java]
[java] ***** APLICACIONES *****
[java]
[java] * Aplicacion: r02d , Desplegado: true
[java] Nombre del pool: r02dPool
[java] Pool: intra_teletramitacion_apps111 , Estado: Running
[java] Pool: intra_teletramitacion_apps112 , Estado: Running
[java]
[java] * Aplicacion: r02e , Desplegado: true
[java] Nombre del pool: r02ePool
[java] Pool: intra_teletramitacion_apps112 , Estado: Running
[java] Pool: intra_teletramitacion_apps111 , Estado: Running
[java]
[java] * Aplicacion: r02f , Desplegado: true
[java]
[java] * Aplicacion: r02g , Desplegado: true
[java] Nombre del pool: r02gPool
[java] Pool: intra_teletramitacion_apps112 , Estado: Running
[java] Pool: intra_teletramitacion_apps111 , Estado: Running
[java]
[java] * Aplicacion: r02h , Desplegado: true
[java] Nombre del pool: r02hPool
[java] Pool: intra_teletramitacion_apps111 , Estado: Running
[java] Pool: intra_teletramitacion_apps112 , Estado: Running
[java]
[java] * Aplicacion: r02i , Desplegado: true
[java] Nombre del pool: r02iPool
[java] Pool: intra_teletramitacion_apps111 , Estado: Running
[java] Pool: intra_teletramitacion_apps112 , Estado: Running
[java]
[java] * Aplicacion: r02j , Desplegado: true
[java] Nombre del pool: r02jPool
[java] Pool: intra_teletramitacion_apps112 , Estado: Running
[java] Pool: intra_teletramitacion_apps111 , Estado: Running
[java]
[java] * Aplicacion: r02l , Desplegado: true
```

```
[java] Nombre del pool: r02lPool
[java] Pool: intra_teletramitacion_apps111 , Estado: Running
[java] Pool: intra_teletramitacion_apps112 , Estado: Running
[java]
[java] * Aplicacion: r02m , Desplegado: true
[java] Nombre del pool: r02mPool
[java] Pool: intra_teletramitacion_apps112 , Estado: Running
[java] Pool: intra_teletramitacion_apps111 , Estado: Running
[java]
[java] * Aplicacion: r02o , Desplegado: true
[java] Nombre del pool: r02oPool
[java] Pool: intra_teletramitacion_apps111 , Estado: Running
[java] Pool: intra_teletramitacion_apps112 , Estado: Running
[java]
[java] * Aplicacion: r02p , Desplegado: true
[java] Nombre del pool: r02pPool
[java] Pool: intra_teletramitacion_apps111 , Estado: Running
[java] Pool: intra_teletramitacion_apps112 , Estado: Running
[java]
[java] * Aplicacion: r02q , Desplegado: true
[java]
[java] * Aplicacion: r02r , Desplegado: true
[java] Nombre del pool: r02rPool
[java] Pool: intra_teletramitacion_apps112 , Estado: Running
[java] Pool: intra_teletramitacion_apps111 , Estado: Running
[java]
[java] * Aplicacion: r02s , Desplegado: true
[java] Nombre del pool: r02sPool
[java] Pool: intra_teletramitacion_apps111 , Estado: Running
[java] Pool: intra_teletramitacion_apps112 , Estado: Running
[java]
[java] * Aplicacion: r02t , Desplegado: true
[java] Nombre del pool: r02tPool
[java] Pool: intra_teletramitacion_apps111 , Estado: Running
[java] Pool: intra_teletramitacion_apps112 , Estado: Running
[java]
[java] * Aplicacion: r02u , Desplegado: true
[java]
[java] * Aplicacion: r02v , Desplegado: true
[java]
[java] * Aplicacion: r02w , Desplegado: true
[java] Nombre del pool: r02wPool
[java] Pool: intra_teletramitacion_apps111 , Estado: Running
```

```
[java] Pool: intra_teletramitacion_apps112 , Estado: Running
[java]
[java] * Aplicacion: r02y , Desplegado: true
[java]
[java] ***** SERVIDORES JMS *****
[java] Servidor: intra_teletramitacion_apps111.jms , Estado:
State:HEALTH_OK,ReasonCode:[
[java] Conexiones actuales: 11
[java] Pico maximo de conexiones: 16
[java] Conexiones totales: 11268
[java]
[java] Servidor: intra_teletramitacion_apps112.jms , Estado:
State:HEALTH_OK,ReasonCode:[
[java] Conexiones actuales: 13
[java] Pico maximo de conexiones: 14
[java] Conexiones totales: 12946
[java]
[java]
[java] ***** COLAS JMS *****
[java] Nombre: r02JMSServer_apps111 , Estado: State:HEALTH_OK,ReasonCode:[
[java] Bytes almacenados actualmente: 1610
[java] Pico maximo de bytes almacenados: 2305537
[java] Bytes pendientes almacenados: 0
[java] Bytes recibidos: 15287536
[java] Mensajes almacenados actualmente: 7
[java] Pico maximo de mensajes almacenados: 9
[java] Mensajes pendientes almacenados: 0
[java] Mensajes recibidos: 671
[java] Sesiones instanciadas actualmente: 0
[java] Pico maximo de sesiones instanciadas: 0
[java]
[java] Nombre: r02JMSServer_apps112 , Estado: State:HEALTH_OK,ReasonCode:[
[java] Bytes almacenados actualmente: 690
[java] Pico maximo de bytes almacenados: 57112
[java] Bytes pendientes almacenados: 0
[java] Bytes recibidos: 3054919
[java] Mensajes almacenados actualmente: 3
[java] Pico maximo de mensajes almacenados: 5
[java] Mensajes pendientes almacenados: 0
[java] Mensajes recibidos: 218
[java] Sesiones instanciadas actualmente: 0
[java] Pico maximo de sesiones instanciadas: 0
[java]
```

```
[java] wslsh [not connected]> quit
[java] bye!
```

BUILD SUCCESSFUL

Total time: 19 seconds

La interpretación de estos resultados sería la misma que la detallada anteriormente en los tests de infraestructura en ejecución en lo referente a las máquinas, las aplicaciones y las colas jms.

1.12 Targets varios

- **copiarLibApp**, Permite gestionar dependencias con terceras aplicaciones en los traspasos a pruebas y producción. Esta tarea lo que hace es copiar la librería /(explopr o explo)/aplic/bbb/java/javs/bbb.jar a la carpeta lib de la aplicación que se está traspasando. Parámetros requeridos:
 - **dependeApp**, código de aplicación del que depende la aplicación que está siendo traspasada.
- **console**, Muestra información de la aplicación: estado de la instancia, módulos de la aplicación desplegados, pool y dataSource.
- **ficherosexplo**, Tarea para traspasar uno o varios ficheros a la carpeta explopr. Parámetros requeridos:
 - **files**, lista de ficheros a traspasar.
- **copiarFicheros**, Tarea para copiar los ficheros de una extensión determinada de una ruta origen a una ruta destino.

- **rutaOrigen**, Ruta origen de los ficheros a copiar
- **rutaDestino**, Ruta destino de los ficheros a copiar
- **extensión**, Extensión de los ficheros a copiar
- **cvcs**, Tarea que extrae la aplicación correspondiente del repositorio CVS.
- **svn**, Tarea que extrae la aplicación correspondiente del repositorio Subversion. Esta tarea posee diferentes parámetros que se deberán de informar dependiendo de las necesidades de la descarga:
 - **DrepoSVN** -> Nombre del Repositorio.
 - **DuserSVN** -> Código usuario responsable aplicación.
 - **DpassSVN** -> Password del usuario.
 - **DtagSVN** -> Nombre de versión final (tag) a desplegar.
 - **DtypeSVN**-> Tipo de obtención (Habitualmente el origen es myeclipse).
 - **DcarpetaSVN** -> indicador de que es lo que se desea descargar.
 - **all** -> Descarga todo el repositorio.
 - **c** -> Descarga la carpeta código.
 - **s** -> Descarga la carpeta scripts.
 - **e** -> Descarga la carpeta estático.
 - **l** -> Descarga la carpeta librerías.
 - **d** -> Descarga la carpeta datos.
 - **f** -> Descarga la carpeta config.
- **maven**, Tarea que permite ejecutar tareas maven
 - **tarea**, Tarea de maven a lanzar
- **build-xmlbean**, Tarea que permite crear un jar con los objetos xmlbeans a partir de un esquema; los esquemas deberán estar situados en la carpeta /aplic/aaa/html/xmlbean_schemas de la propia aplicación y el jar resultante se depositará dentro de la carpeta /aplic/aaa/java/javs
- **copiar_build** (ver Dashboard-ejie. Manual rápido de usuario v1.3.doc), Tarea que copia el build.xml con las tareas actualizadas a la aplicación desde donde se lanza. Si se quiere realizar una primera copia del build.xml (debido a que no existe el target copiar_build en el build.xml de la aplicación) se podrá ejecutar el script copiar_build.sh que se encuentra en la carpeta /aplic de la siguiente manera:

```
./copiar_build.sh cod_aplic entorno
```

cod_aplic: código de la aplicación a donde se quiere copiar.

entorno: directorio correspondiente al entorno en el que se encuentra la aplicación.

- Desarrollo: /
- Softbase: /softbase_ejie/

Ejemplo:

```
./copiar_build.sh s73a /
```

El script realizará una copia del build.xml antiguo renombrándolo como build.xml.ant. Hay que tener en cuenta que si ya existía un build.xml.ant anterior, éste será sobrescrito.

- **copiar_testing** (ver Dashboard-ejie. Manual rápido de usuario v1.3.doc), Tarea que copia el aaaTestingWar genérico para el lanzamiento de pruebas a la aplicación desde la que se lance cambiándole el aaa por el código de aplicación y que modifica el application.xml de la aplicación añadiéndole el modulo copiado (el aaaTestingWar). Se controlará si ya existe el módulo en la aplicación y en el fichero application.xml. Se realizará a su vez una copia del application.xml antiguo renombrándolo como application.xml.ant.

Nota importante: Tras lanzar esta tarea ant, es necesario lanzar la tarea ant deployExploded. Con esto se evitan problemas al lanzar la tarea ant dashboard-ejie.

- **borrar_testing** (ver Dashboard-ejie. Manual rápido de usuario v1.3.doc). Esta tarea deja la aplicación en el estado inicial, antes de crear toda la estructura de carpetas y ficheros del aaaTestingWar. Realiza una copia de seguridad del fichero application.xml de la aplicación, que exista en ese momento, con nombre application.xml.ant.borrado y que se deja en la misma carpeta que el original.

Nota importante: Tras lanzar esta tarea ant, es necesario lanzar la tarea ant undeploy y posteriormente la tarea ant deployExploded, para evitar problemas en el despliegue de la aplicación

- **createPoolXA**, Tarea que crea un pool de conexión a BBDD de tipo XA. Recibe diferentes parámetros (true/false) que particularizan la configuración del driver. Pensando en una configuración estandar como punto de partida para todos los pooles XA creados, se establecen valores por defecto para estos atributos. Las necesidades y condicionantes de cada aplicación (tipo de transaccionalidad, implementación...) determinarán si es necesario activar alguna propiedad más y en ese caso se tendrán que pasar explícitamente los valores que correspondan como parámetros a la tarea ant.

Estos son los parámetros:

- **needTxCtxOnClose** (valor por defecto: true)
- **keepXaConnTillTxComplete** (valor por defecto: true)
- **newXaConnForCommit** (valor por defecto: true)
- **supportsLocalTransaction** (valor por defecto: true)
- **xaEndOnlyOnce** (valor por defecto: false)
- **keepLogicalConnOpenOnRelease** (valor por defecto: false)
- **xaSetTransactionTimeout** (valor por defecto: false)
- **xaTransactionTimeout** (valor por defecto: false)
- **usarCompatibilidad** (valor por defecto: false – propiedad oracle.jdbc.V8Compatible)

- Ejemplo tarea ant createPoolXA:

```
ant createPoolXA -DpoolNameBD=aaaPool -DuserBD=aaa -DpassBD=aaa -DconexionBD=aaad
```

(parámetros opcionales:)

```
-DneedTxCtxOnClose=true/false  
-DkeepXaConnTillTxComplete=true/false
```

- DnewXaConnForCommit=true/false
- DxaEndOnlyOnce=true/false
- DkeepLogicalConnOpenOnRelease=true/false
- DsupportsLocalTransaction=true/false
- DxaSetTransactionTimeout=true/false
- DxaTransactionTimeout=valor timeout(merico)

Ejemplos de los Targets Varios

- Paso a pruebas o producción de una aplicación aaa que hace uso de un API generado por la aplicación bbb:

```
ant -DdependeApp=bbb copiarLibApp
```

Nota: esta tarea está pensada para ser incluida en el fichero compile.txt durante los traspasos antes de lanzar ninguna tarea de compilación (ant all). En Desarrollo no podrá ser usada ya que se estaría intentado crear una dependencia con una librería no estable.

- acceder a la información de la consola en modo texto:

```
ant console
```

- Traspaso de dos ficheros a la carpeta explor:

```
ant -Dfiles=aaaSaludaWar/presentacionJSP/aaaSaluda.jsp,aaaSaludaWar/WEB-INF/web.xml ficherosexplor
```

- Copia de ficheros .class de una ruta origen a una ruta destino:

```
ant -DrutaOrigen=exe/APP-INF/classes/com/ejie/s73a/jws -DrutaDestino=exe/APP-INF/classes/com/ejie/s73a/dao -Dextension=class copiarFicheros
```

- Extracción de una aplicación del repositorio de CVS:

```
ant cvs
```

- Extracción de todo el contenido de un proyecto del repositorio de Subversion:

```
ant -DrepoSVN=t71a -DuserSVN=t71a -DpassSVN=t71a -DtagSVN=t71a_0_1 -DtypeSVN=myeclipse -DcarpetaSVN=all svn
```

- Ejecución de una tarea maven

```
ant -Dtarea=ejie-dist maven
```

- Generar un jar con los objetos xmlbeans a partir de sus esquemas.

```
ant build-xmlbean
```

- Copiado del build.xml:

```
ant copiar_build
```

- Copiado del aaaTestingWar y añadir el módulo en el application.xml de la aplicación:

```
ant copiar_testing
```