Servidores de aplicaciones

Sesión 6:
JNDI y acceso a bases de datos con el servidor de aplicaciones

Índice

- Programación con JNDI
- WebLogic y JNDI
- Configuración de fuentes de datos
- Uso de fuentes de datos en una aplicación

JNDI

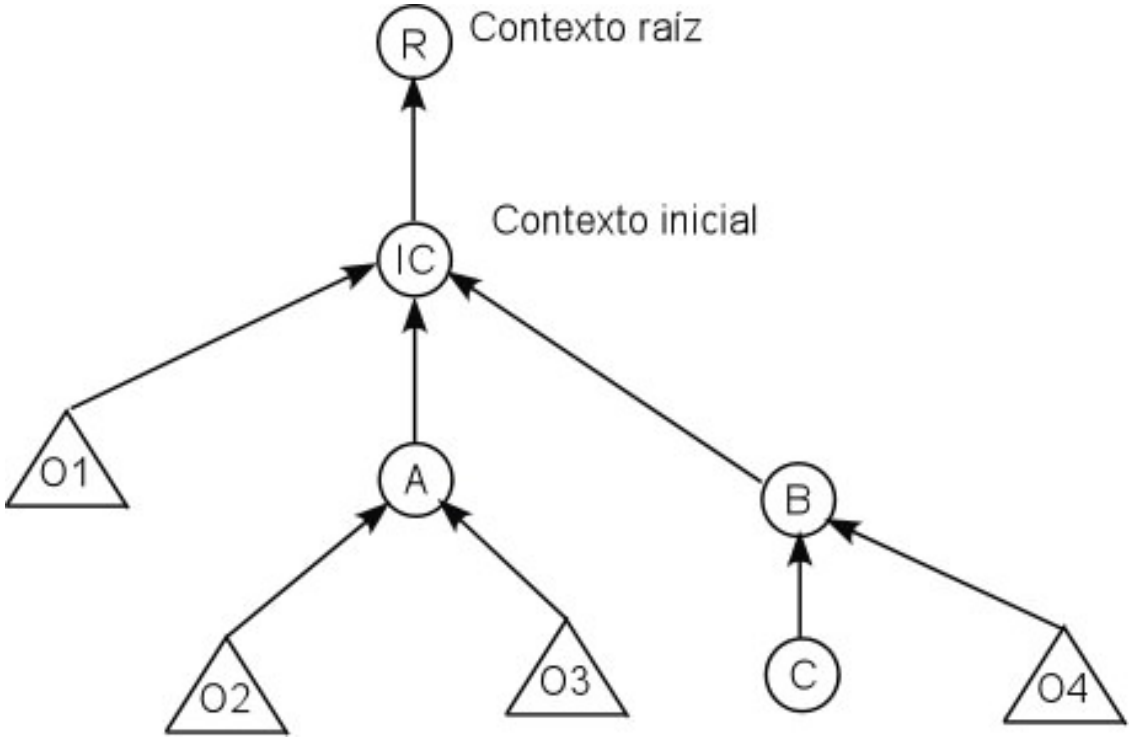
- JNDI: *Java Naming and Directory Interface*
- Permite enlazar programas Java con: sistemas de ficheros, recursos de red, objetos, etc.
- Principales funciones:
  - Servicio de nombres: asociación de nombres lógicos con recursos
  - Servicio de directorio: sistema de ficheros, de datos

Búsqueda de objetos mediante su nombre lógico

- Se mapea un objeto con un nombre lógico
- JNDI se encarga de mantener el mapeo
- Nuestro programa, si quiere obtener el objeto, llama al nombre lógico
- Ventaja: si modificamos el objeto (su nombre, por ejemplo) no tenemos que cambiar nuestro programa, sólo el enlace en JNDI

Definiciones

- Contexto: se puede entender como un directorio. Dentro de un contexto podemos tener objetos o más contextos



Programación con JNDI

- Context miContexto = null;
  Hashtable ht = new Hashtable ();
  ht.put(Context.INITIAL_CONTEXT_FACTORY,
  "weblogic.jndi.WLInitialContextFactory");
  ht.put(Context.PROVIDER_URL, "t3://localhost:7001");
  miContexto = new InitialContext (ht);

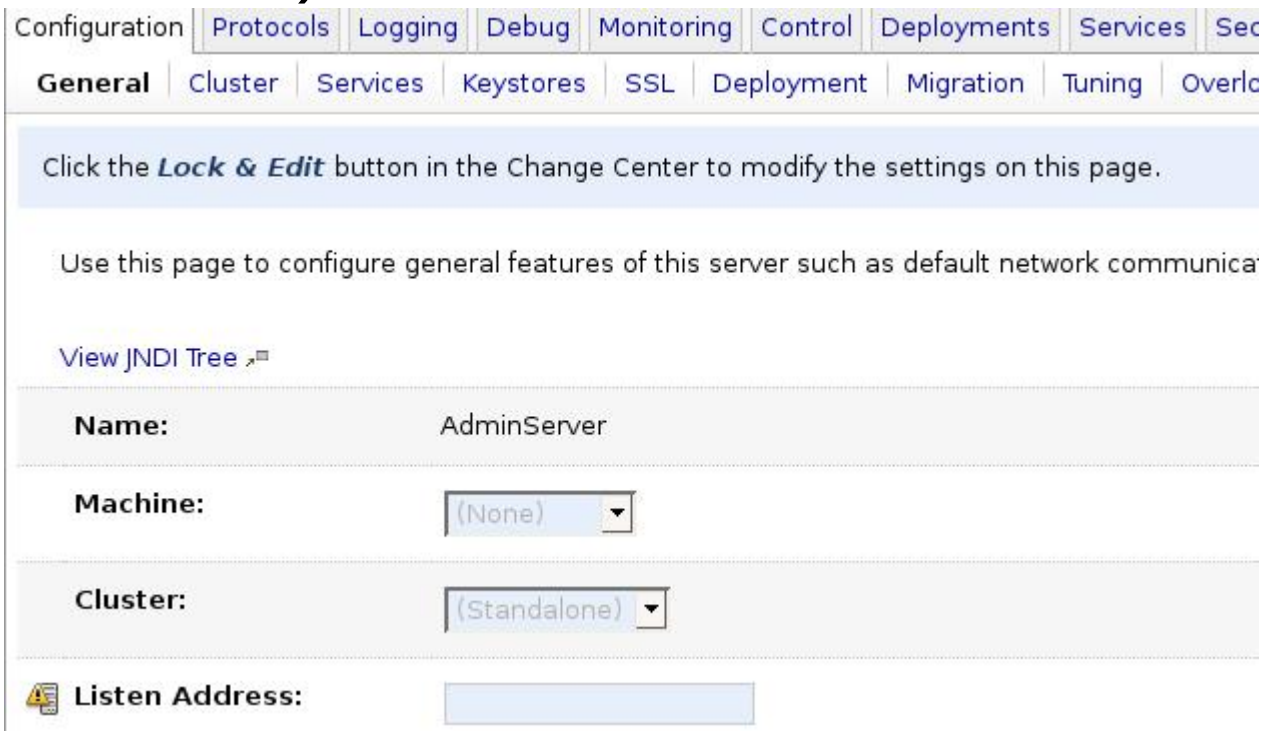
- Hay que capturar la excepción *NamingException*
- Cuando terminemos de utilizarlo llamamos al método *close*
  miContexto.close();

Programación con JNDI

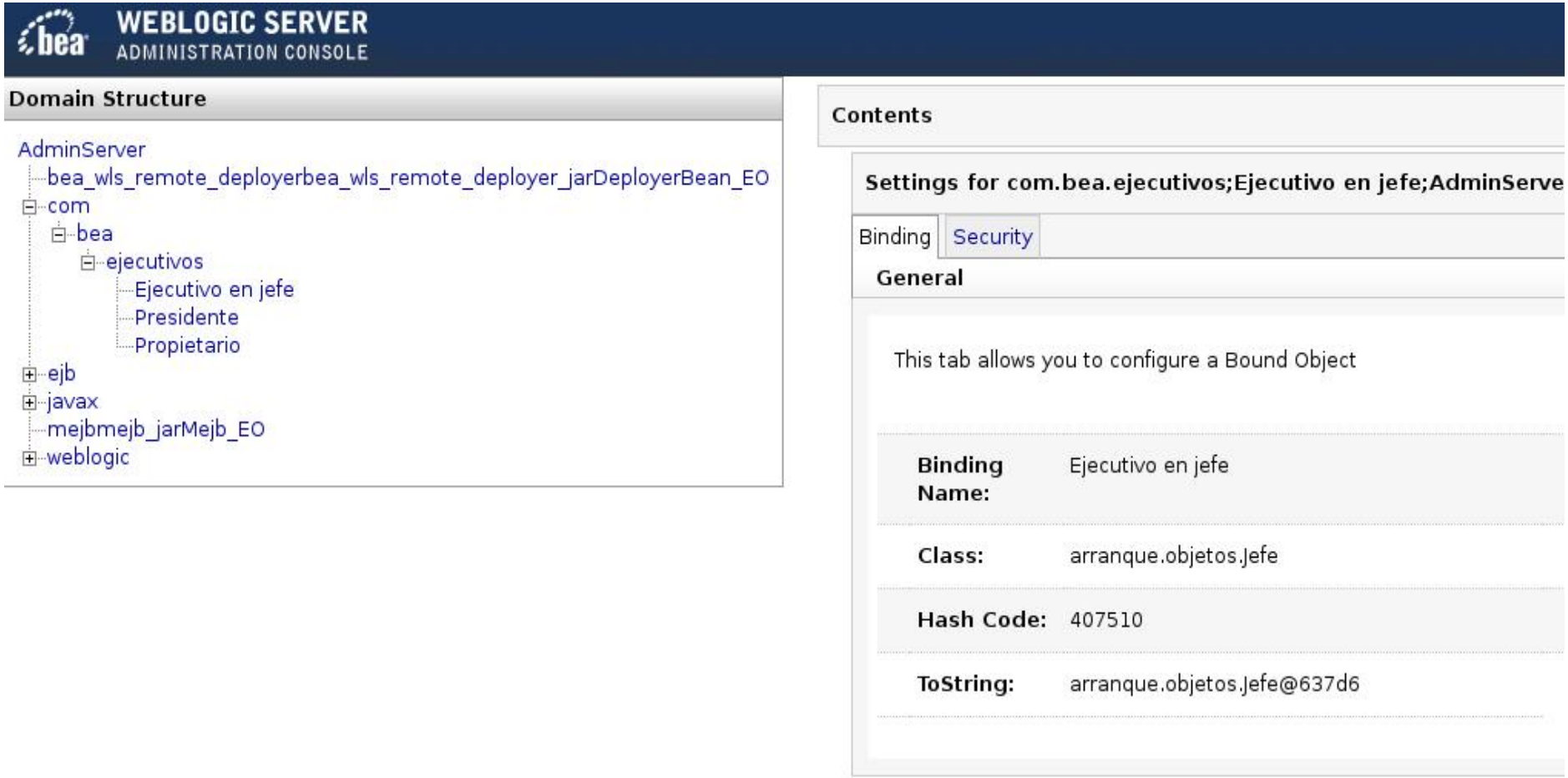
- Enlace de un objeto
  - Persona persona = new Persona();
    miContexto.bind ("objeto persona", persona);
    // miContexto.rebind ("objeto persona", persona);
- Creación de subcontexto
  - Context subcontexto =
    miContexto.createSubContext
    ("empleados");
    Persona persona = new Persona();
    subcontexto.bind ("contable", persona);
- Recuperación de objetos
  - Persona pers = (Persona) miContexto.lookup
    ("empleados/contable");

WebLogic y JNDI

- WebLogic gestiona un árbol JNDI
- Podemos ver el árbol JNDI de ciertos objetos (p.e. servidor)



Árbol mostrado



Clases de arranque y parada

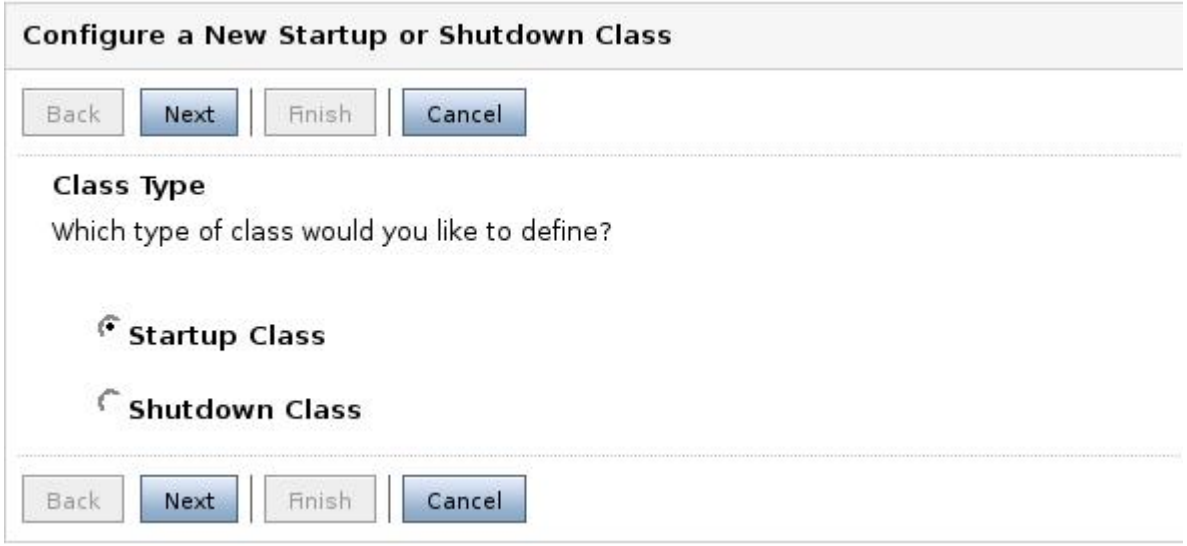
- WebLogic permite la ejecución de clases cuando un servidor arranca o para
- Puede servir para “colgar” objetos en el árbol JNDI
- Las clases deben estar disponibles en el CLASSPATH (editar los ficheros *startManagedWebLogic.cmd* y *startWebLogic.cmd*)

Despliegue de clases de arranque y parada

- Pinchamos en *Environment->Startup&stop classes*

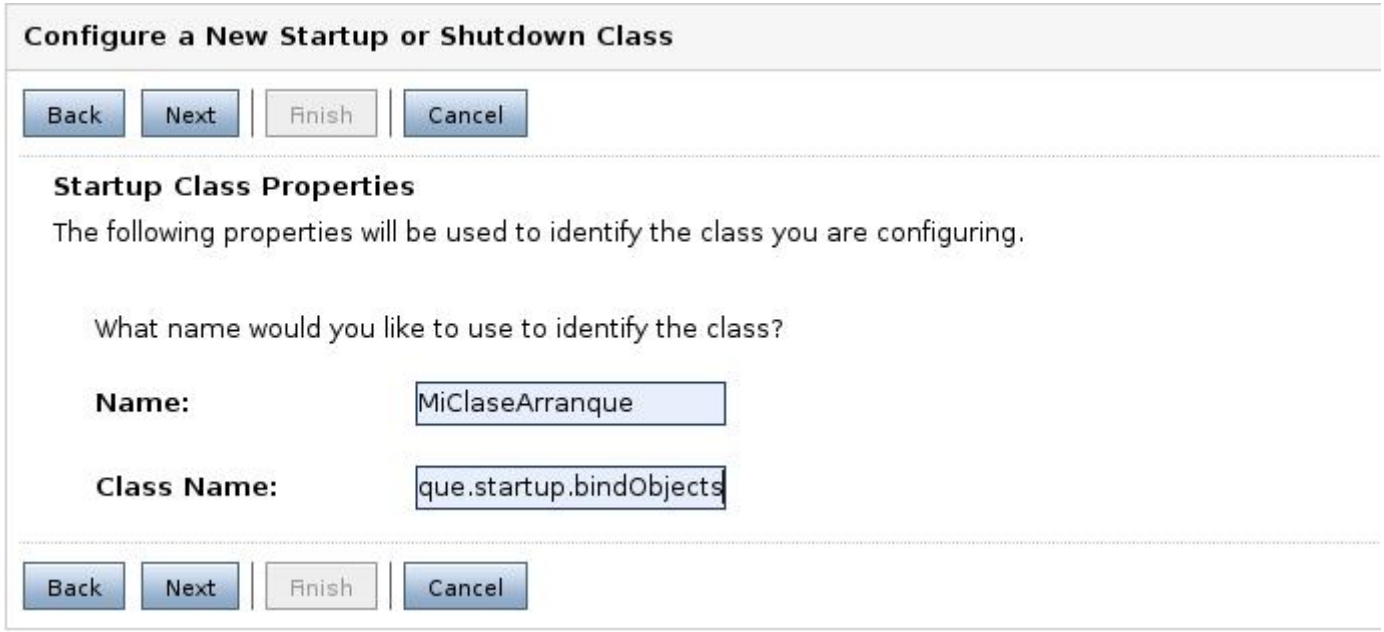
Despliegue de clases de arranque y parada

- Definimos el tipo de clase a desplegar



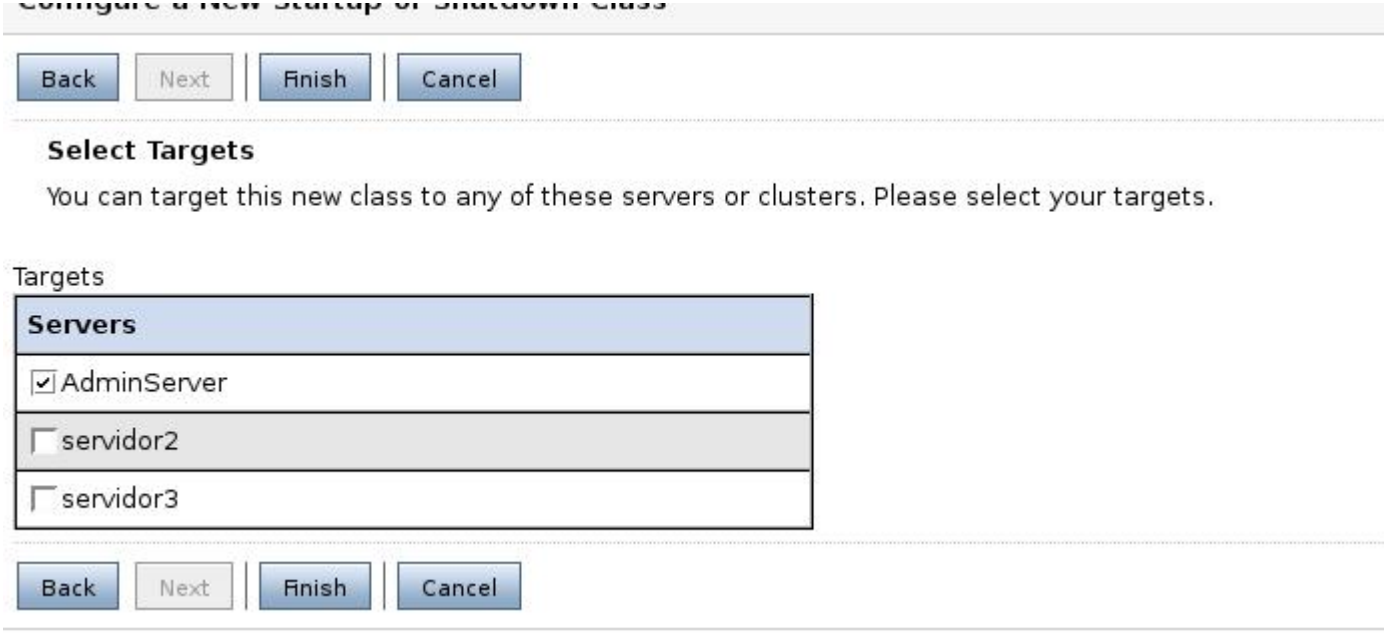
Despliegue de clases de arranque y parada

- Definimos el nombre (genérico) asignado a este despliegue y el nombre de la clase



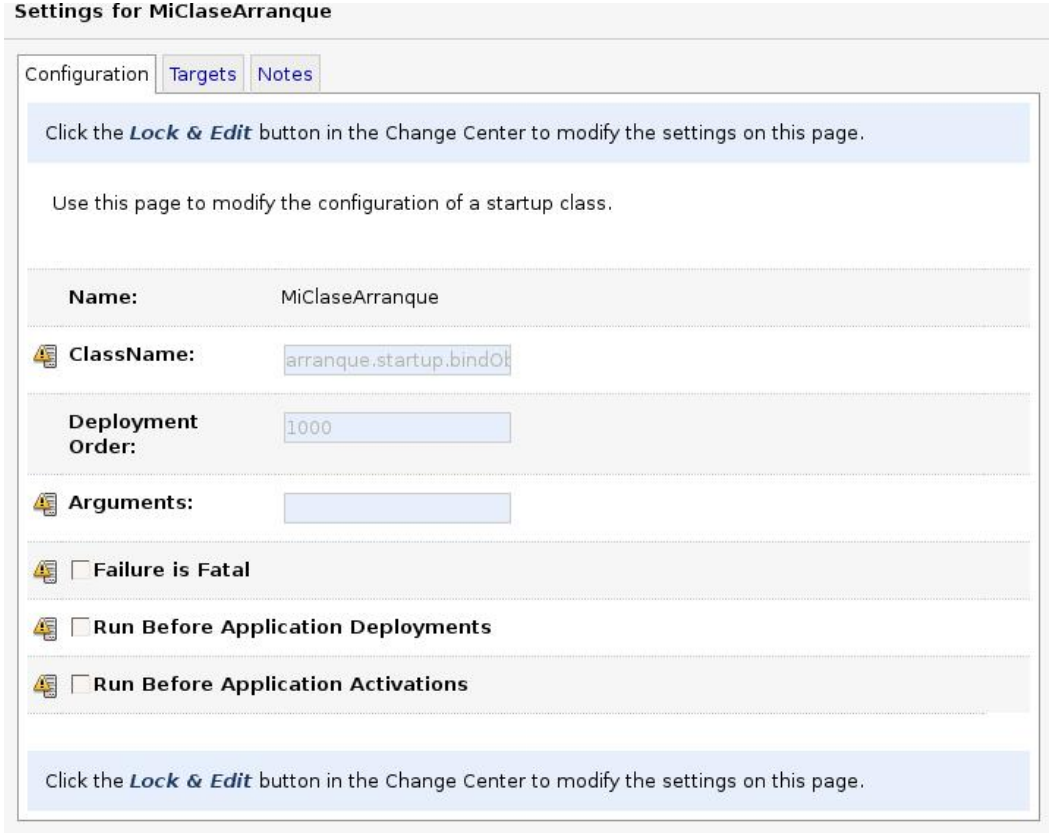
Despliegue de clases de arranque y parada

- Indicamos los servidores donde se desplegará

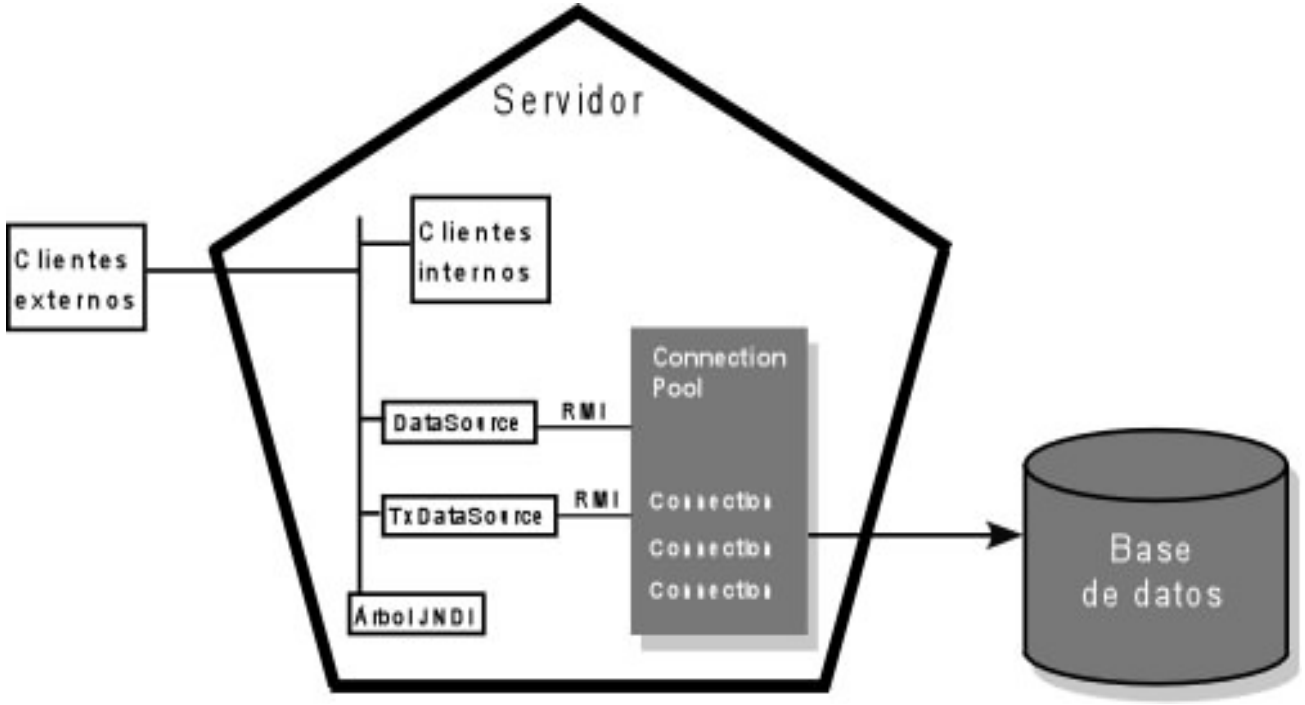


Despliegue de clases de arranque y parada

- Activamos los cambios
- Podemos configurar varias opciones más



JDBC y WebLogic

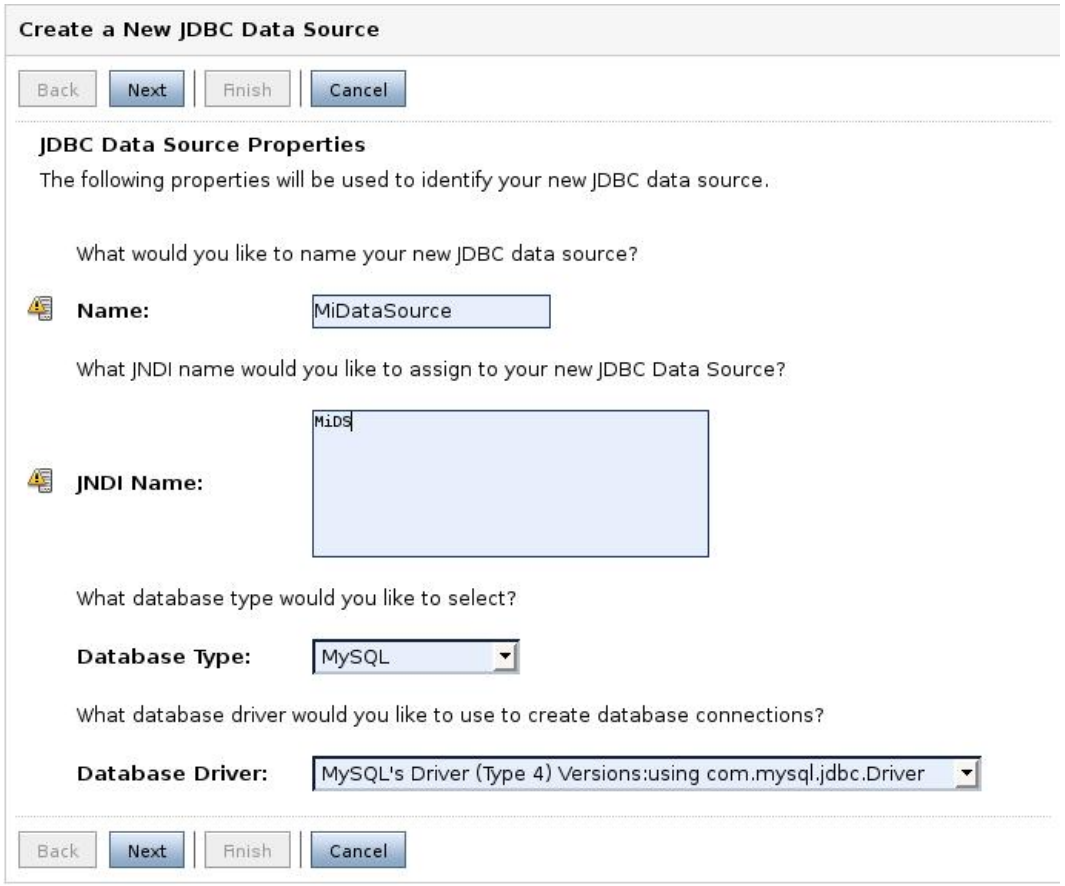


Cambio de versión

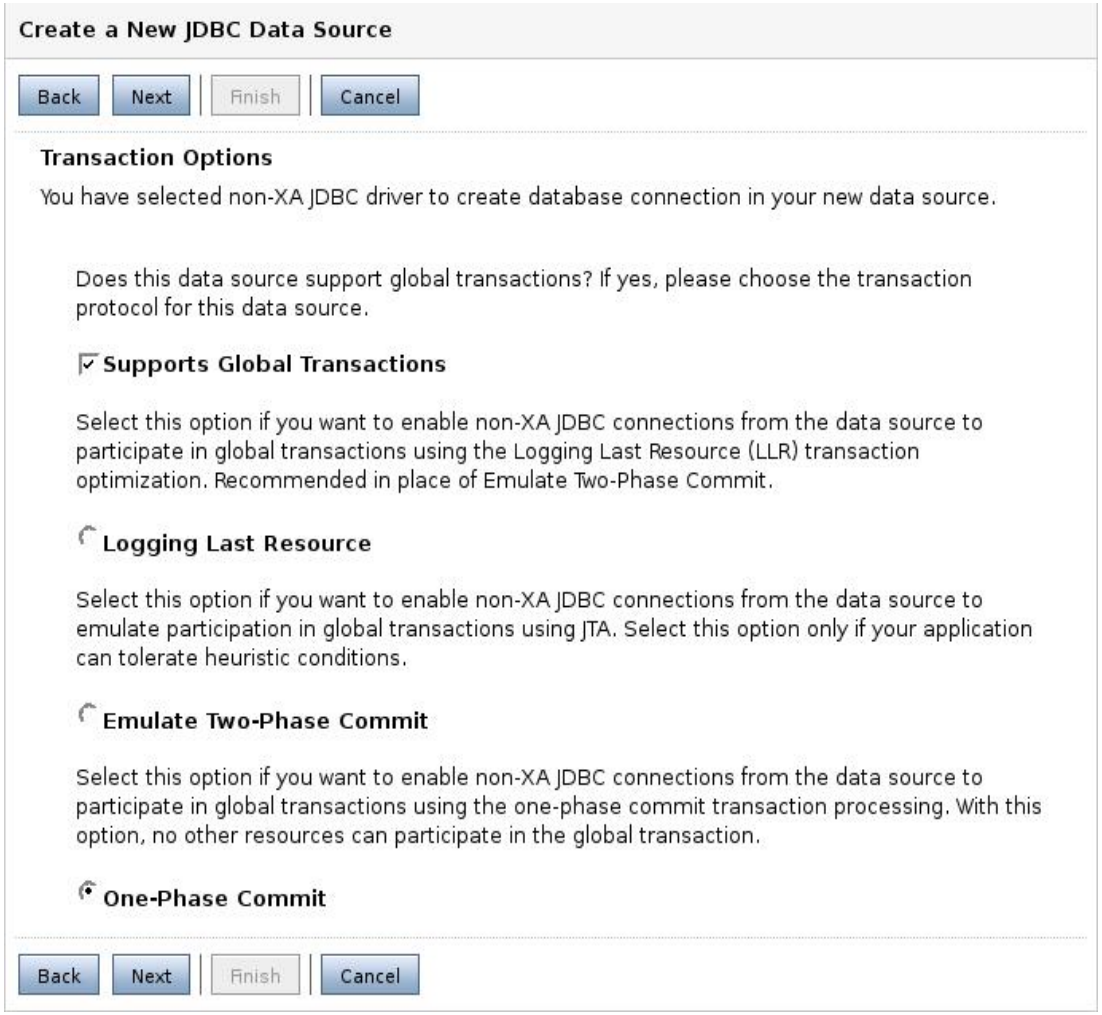
- En versiones anteriores a la 9.0, para trabajar con bases de datos se debía definir primero el pool de conexiones y luego crear una fuente de datos asociada a ese pool
- A partir de la 9.0 sólo es necesario definir la fuente de datos. Ésta lleva asociada internamente un pool.
- Pinchamos en *Services->DataSources.*

Definición de una fuente de datos (I)

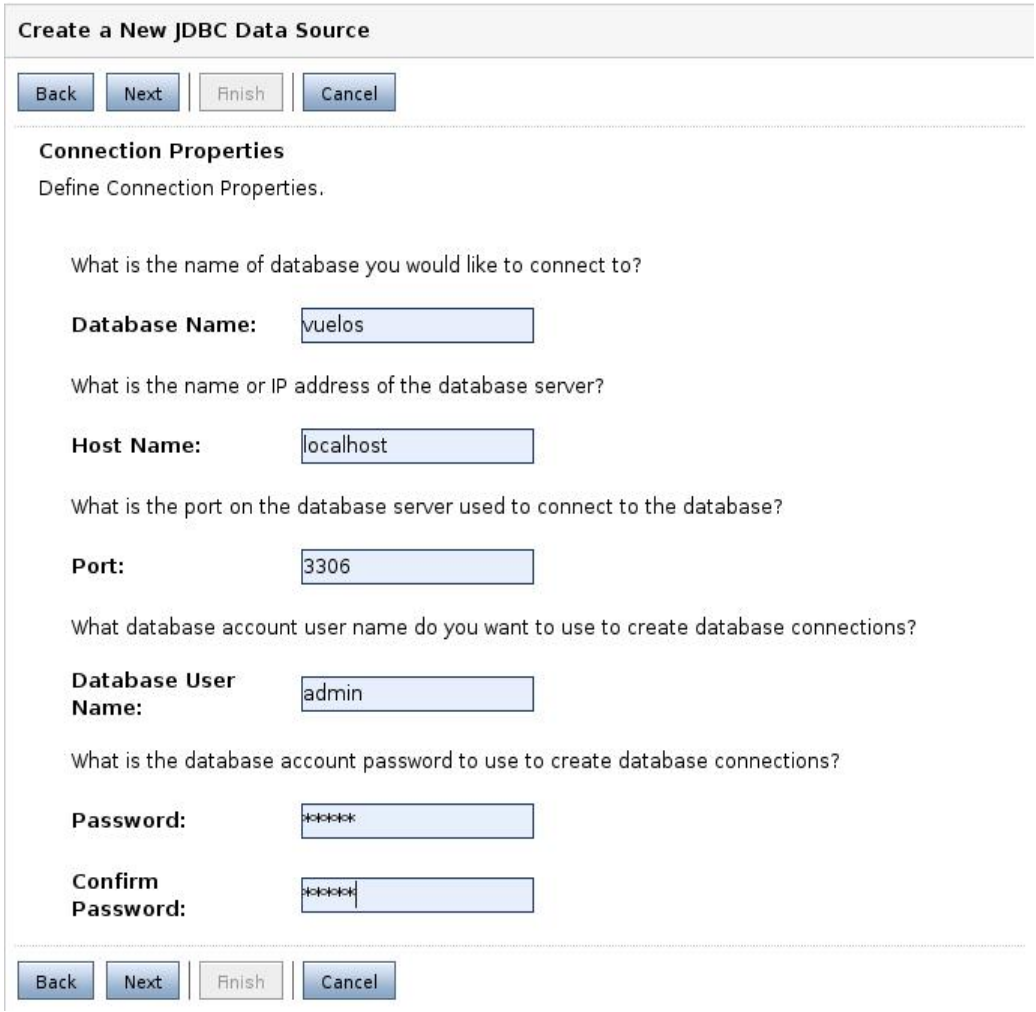
- No incluye *drivers* de todos los SGBD que aparecen
- Debemos incluir el *driver* en el CLASSPATH



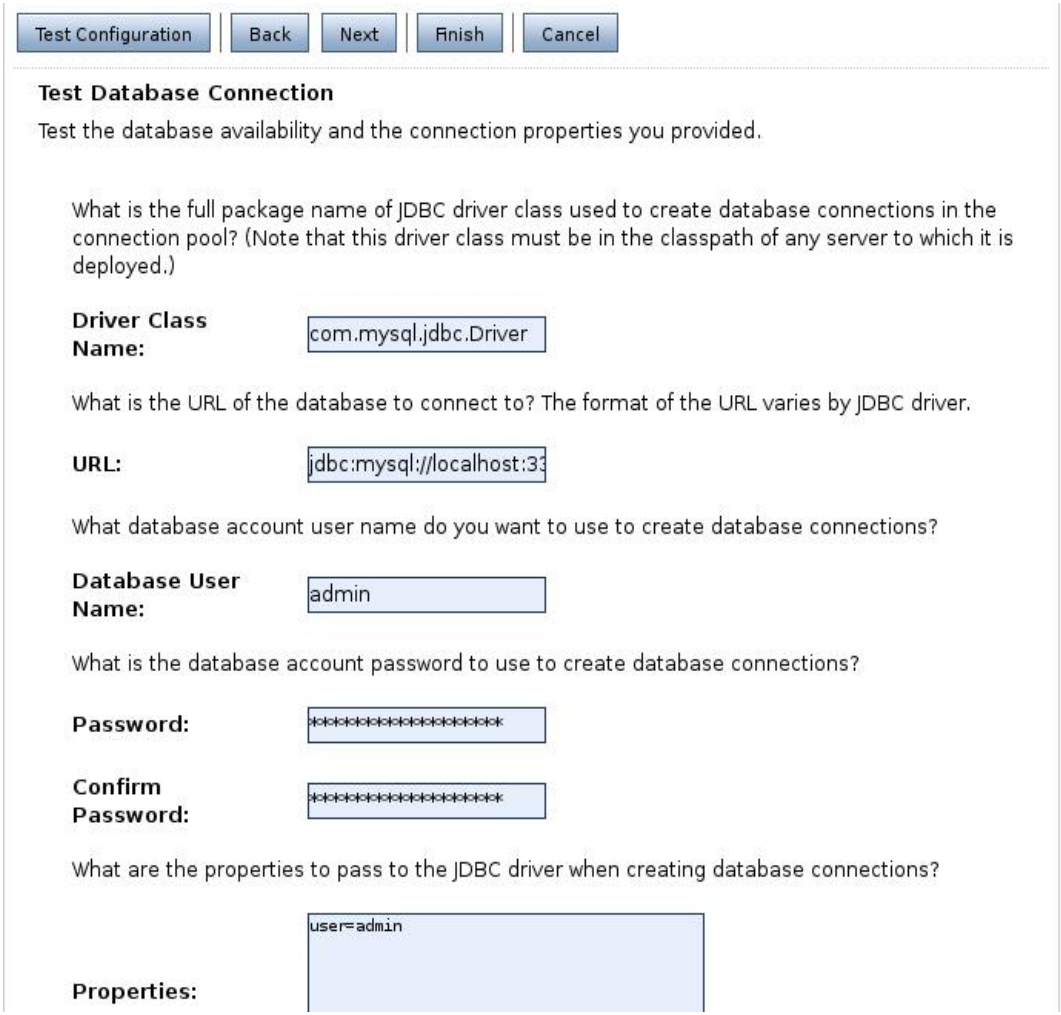
Definición de una fuente de datos (II)



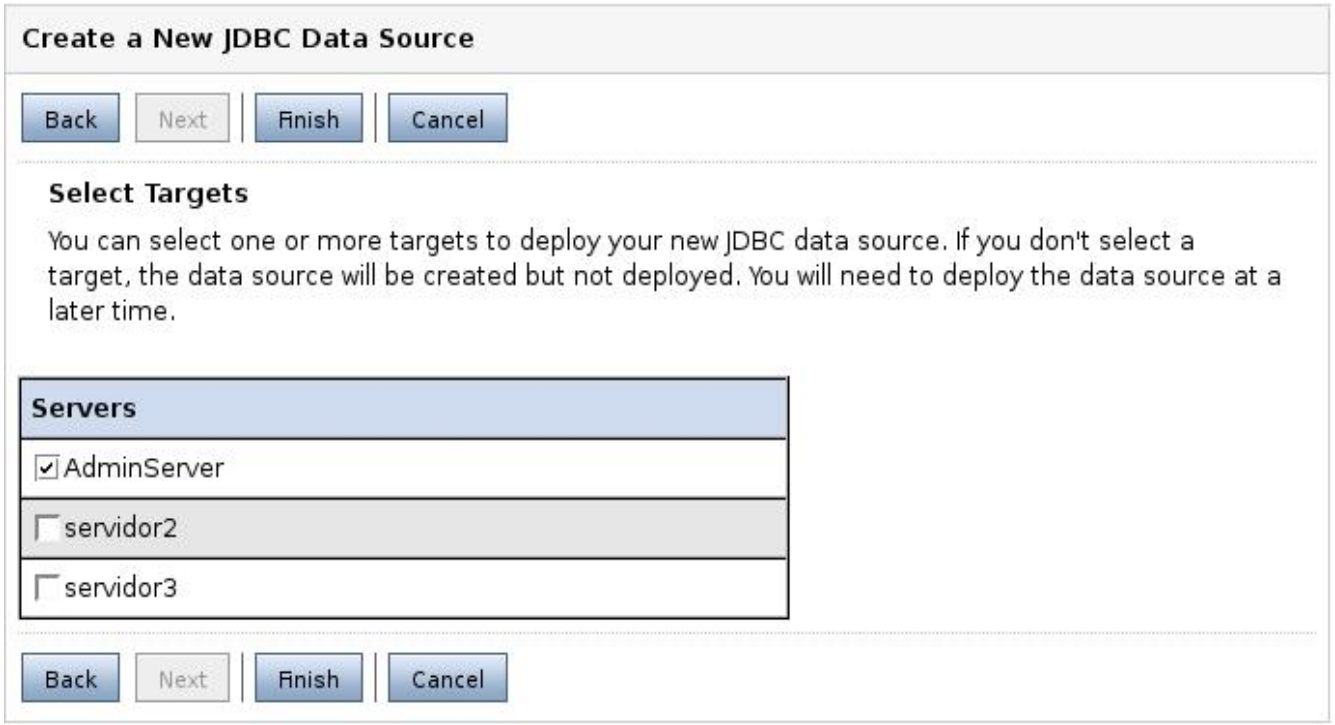
Definición de una fuente de datos (III)



Definición de una fuente de datos (IV)



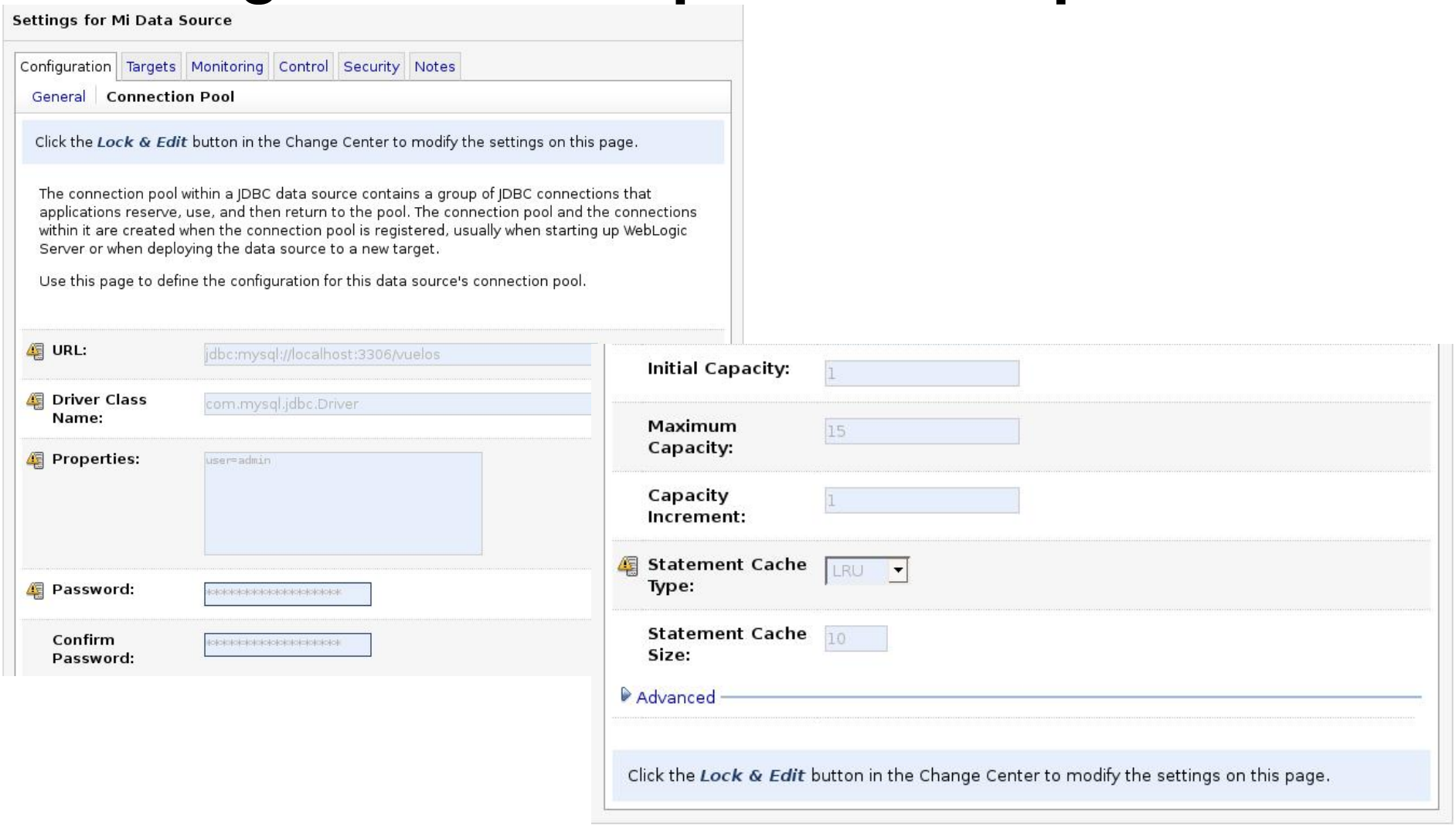
Definición de una fuente de datos (final)



Configuración de opciones del pool

- Pinchamos sobre el nombre de la fuente de datos
- Nos vamos a la solapa *Configuration->Connection pool*
- Podemos definir la capacidad inicial, máxima capacidad y el incremento

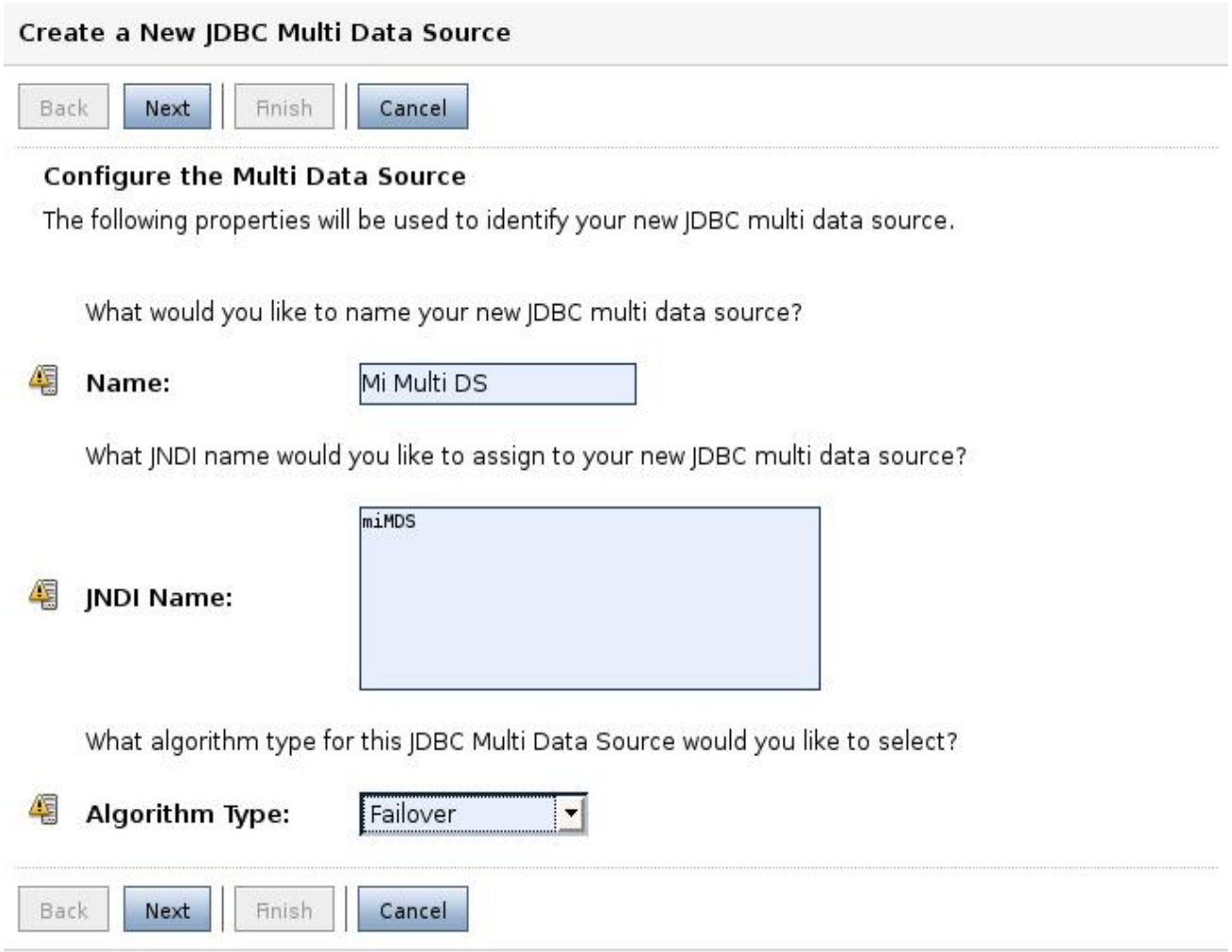
Configuración de opciones del pool



Multi fuentes de datos

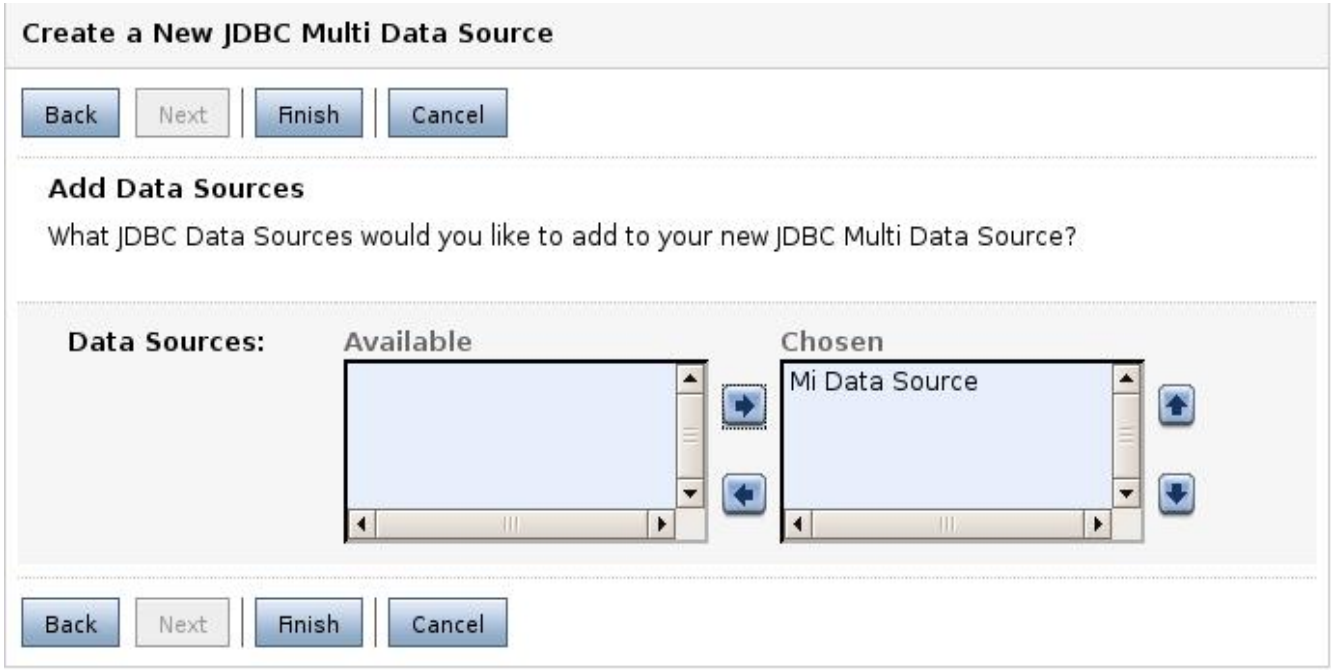
- Conjunto de fuentes de datos que permiten dar soporte de balanceo de carga y recuperación ante fallos
- Simplemente es una agrupación de distintas fuentes de datos (previamente creadas)

Creación de multidatasources



Creación de multidatasources

Creación de multidatasources



Uso de fuentes de datos en nuestra aplicación

- Haremos uso de JNDI para acceder a una fuente de datos
- Si queremos ejecutar externamente al servidor (aplicación separada del servidor) debemos incluir en el CLASSPATH la ruta hasta $HOME_BEA/weblogic92/server/lib/weblogic.jar

Código a incluir en nuestro programa

- Importación de clases:
  import javax.sql.DataSource;
  import javax.naming.;*
  import java.util.Hashtable;
- Definición del contexto inicial:

```
Context miContexto = null;
Hashtable ht = new Hashtable ();
ht.put(Context.INITIAL_CONTEXT_FACTORY,
                    "weblogic.jndi.WLInitialContextFactory");
ht.put(Context.PROVIDER_URL, "t3://localhost:7001");
miContexto = new InitialContext (ht);
```

Servidor que contenga la fuente de datos

Código a incluir en nuestro programa

- Obtención de la fuente de datos:
  DataSource ds = (DataSource) miContexto.lookup ("MySQLDataSource");

- Obtención de la conexión:
  Connection con = ds.getConnection ();