APÉNDICE D

CONFIGURACIÓN DE STRUTS

[McClanahan, 2002]

CONFIGURACIÓN DE STRUTS

Antes de poder construir una aplicación en Struts es necesario realizar varias configuraciones. Estas configuraciones incluyen la de componentes en el archivo de configuración de Struts así como configuraciones al archivo descriptor de despliegue de aplicaciones web.

EL ARCHIVO DE CONFIGURACIÓN DE STRUTS (STRUTS-CONFIG.XML)

Struts incluye un módulo Digester que es capaz de leer la descripción basada en XML de los mapeos deseados, creando los objetos apropiados de la misma forma. La responsabilidad del desarrollador es crear un archivo XML llamado struts-config.xml, y situarlo en el directorio WEB-INF de su aplicación. Este formato de documento está restringido por su definición en "struts-config_1_0.dtd". El elemento XML raíz debe ser `<struts-config>` [McClanahan, 2002].

A continuación en la figura D.1 se verá el código de una implementación del archivo struts-config.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE struts-config PUBLIC "-//Apache Software Foundation//DTD
Struts                    Configuration                    1.0//EN"
"http://jakarta.apache.org/struts/dtds/struts-config_1_0.dtd">
<struts-config>
 <!-- ========== Definiciones de Form Beans ============== -->
  <form-beans>
    <form-bean
      name="logonForm"
      type="org.apache.struts.example.LogonForm" />
  </form-beans>
```

```
  <!-- ========== Definiciones de Global Forward ========== -->
  <global-forwards
      type="org.apache.struts.action.ActionForward" />
    <forward name="logon" path="/logon.jsp"
 redirect="false" />
  </global-forwards>
<!-- ========== Definiciones de Action Mapping ========== -->
  <action-mappings>
    <action
path="/logon"
type="org.apache.struts.example.LogonAction"
name="logonForm"
       scope="request"
       input="/logon.jsp"
     unknown="false"
    validate="true" />
  </action-mappings>
</struts-config>
```

Figura D.1 Código de implementación del archivo struts-config.xml

Las etiquetas `<form-beans>` se usan por cada *bean* de formulario que se use en la aplicación. En el atributo *name* se define el nombre lógico del *bean* de formulario, y en el atributo *type* se define el nombre completo de la clase.

Las etiquetas `<global-forwards>` se usan para crear mapeos de nombres lógicos para páginas JSP usadas comunmente. Cada uno de estos reenvíos está disponible a través de una llamada a nuestro ejemplar de mapeo de *action*.

Las etiquetas `<action-mappings>` entre estas etiquetas definen nuestras clases **Action**; se usas un elemento `<action>` para cada una de las clases *action* que se quieran definir. En el atributo *path* se define el *path* de la clase **Action** en relación al contexto de la aplicación, en el atributo *type* se define el nombre totalmente calificado de la clase **Action** y en el atributo *name* se define el nombre del elemento `<form-bean>` correspondiente a esta acción (clase **Action**).

En esta sección se ha explicado de forma muy básica como crear el archivo de configuración de Struts. Si desea consultar sobre opciones de configuración más avanzada y adentrarse más en lo que es el marco de trabajo Struts se recomienda [McClanaham, 2002] ,[Wienser, 2002], [Bellas, 2002].

Configuración del descriptor de despliegue de la aplicación web (web.xml)

El último paso en la configuración de una aplicación Struts es configurar el archivo web.xml, para incluir todos los componentes de Struts que sean necesarios. McClanahan divide la configuración del web.xml en tres secciones que veremos a continuación.

Configuración del Action Servlet

En la figura D.2 podremos ver un fragmento de código sobre la configuración del Action Servlet. Primero defimos el propio *servlet action* (etiquetas `<servlet-name>` y `<servlet-class>`) y después definimos los parámetros de inicialización soportados por **Action** Servlet (etiquetas `<init-param>`), los cuales se describen a continuación. La información que aparece entre corchetes cuadrados describe los valores por defecto.

```
<servlet>
  <servlet-name>action</servlet-name>
  <servlet-class>
    org.apache.struts.action.ActionServlet
  </servlet-class>
  <init-param>
    . . .
  </init-param>
  <init-param>
    . . .
  </init-param>
  . . .
</servlet>
```

Figura D.2 Fragmento de código de configuración de Action Servlet en el archivo web.xml

Los parámetros de inicialización soportados por **Action Servlet** son los siguientes:

- *application* - El nombre de la clase Java para la clase base del paquete de recursos de la aplicación. [NONE].
- *bufferSize* - El tamaño del buffer de entrada usado para procesar *uploads* de archivos. [4096].
- *config* - *Path* relativo al contexto del recurso XML que contiene la información de configuración.[/WEB-INF/struts-config.xml].
- *content* - Tipo de contenido por defecto y codificación de caracteres a seleccionar en cada respuesta; podría ser sobrescrito por un *servlet* re-enviado o una página JSP. [text/html].
- *debug* - El nivel de detalle de depuración para este *servlet*, que controla cuanta información se pone en el *log*. [0].
- *detail* - El nivel de detalles de depuración para el **Digester** que utilizamos en `initMapping()`, que sale por System.out en lugar de servlet log. [0].
- *factory* - El nombre de la clase Java del **MessageResourcesFactory** usado para crear el objeto *MessageResources* de la aplicación. [**org.apache.struts.util.PropertyMessageResourcesFactory**].
- *formBean* - El nombre de la clase Java de la implementación de **ActionFormBean** a utilizar. [**org.apache.struts.action.ActionFormBean**].
- *forward* - el nombre de la clase Java de la implementación de **ActionForward** a utilizar. [**org.apache.struts.action.ActionForward**]. Se puede usar aquí dos clases de conveniencia:
    - **org.apache.struts.action.ForwardingActionForward** - Subclase de **org.apache.struts.action.ActionForward** que por defecto pone la propiead *redirect* a *false* (lo mismo que el valor por defecto de **ActionForward**).
    - **org.apache.struts.action.RedirectingActionForward** - Subclase de **org.apache.struts.action.ActionForward** que por defecto pone la propiedad *redirect* a *true*.

- *locale* - Si se selecciona a *true*, y hay una sesión de usuario, indentifica y almacena un objeto **java.util.Locale** apropiado (bajo la clave estándard

indentificada por **Action.LOCALE_KEY**) en la sesión de usuario si no hay ya un objeto **Locale**. [*true*]

- *mapping* - El nombre de la clase Java de la implementación del **ActionMapping** a utilizar. [**org.apache.struts.action.ActionMapping**]. Se puede usar aquí dos clases de conveniencia:
    - **org.apache.struts.action.RequestActionMapping** - Subclase de **org.apache.struts.action.ActionMapping** que por defecto deja la propiedad *scope* a "request".
    - **org.apache.struts.action.SessionActionMapping** - Subclase de **org.apache.struts.action.ActionMapping** que por defecto deja la propiedad *scope* a "session". (Igual que el valor por defecto de **ActionMapping**).
- *maxFileSize* - El tamaño máximo (en *bytes*) para que un archivo sea aceptado para *upload*. Puede expresarse como un número seguido por una K" "M", o "G", que serán interpretadas como kilobytes, megabytes, o gigabytes, respectivamente. [250M].
- *multipartClass* - El nombre totalmente cualificado de la clase de la implementación de **MultipartRequestHandler** usado para procesar *uploads* de archivos. [**org.apache.struts.upload.DiskMultipartRequestHandler**].
- *nocache* - Si se selecciona a *true*, añade cabeceras HTTP a cada respuesta para evitar que el navegador almacene en el caché cualquier respuesta generado o reenviada. [*false*].
- *null* - Si se selecciona a *true*, configura los recursos de la aplicación a devolver *null* si se usa una clave de mensaje desconocida. De otra forma, se devolverá un mensaje de error incluyendo la clave errónea. [*true*].
- *tempDir* - El directorio de trabajo temporal usado cuando se procesan *uploads* de archivos. [El directorio de trabajo proporcionado para esta aplicación web como atributo contexto del *servlet*].
- *validate* – Para indicar si se esta usando el nuevo formato de archivo de configuración [*true*].
- *validating* – Para indicar si se debe usar un *parser* de XML para procesar el archivo de configuración [*true*].

Configuración del mapeo del servlet action

La configuración del mapeo de *servlets* está definida en la Java Servlet Especification. Cabe señalar que está sección no es tema específico de *struts* sino de la tecnología Java Servlets. Aquí se explicará como definir las URL's que serán procesadas por el *servlet* controlador.

En este proyecto de tesis se utilizará el mapeo por extensión, en el cual se reenvian las URIs solicitadas al *servlet action* basándose en el hecho de que la URI termine en un punto seguido por un conjunto definido por caracteres. Por ejemplo, el *servlet* de procesamiento JSP está *mapeado* al patrón *.jsp para que sea llamado cada vez que se solicite una página JSP. Para usar la extensión *.do, el fragmento de código para definir la entrada de mapeo se puede ver en la figura D.3. Así una URI que corresponda con el path /logon descrito anteriormente puede ser esta: "http://www/miAplicacion/logon.do"

```
<servlet-mapping>
   <servlet-name>action</servlet-name>
   <url-pattern>*.do</url-pattern>
 </servlet-mapping>
```

Figura D.3 Fragmento de código de configuración de mapeo de la clase Action en el archivo web.xml

Configuración de las librerías de etiquetas (tag libraries) de struts

En la figura D.4 se puede ver el código para definir las librerías de etiquetas de Struts en el descriptor de despliegue de la aplicación (web.xml). Esto es necesario para que dichas librerías de *tags* puedan ser utilizadas por la aplicación. En el fragmento de código sólo se especifica que se usará la librería de *tags* "struts-bean", en caso de usarse otras librerías se debe reutilizar el mismo código por cada librería de *tags* que se quiera utilizar sustituyendo solamente el texto "struts-bean.tld" por el de la librería a utilizar. Se recomienda especificar solamente las librerías que van a ser utilizadas por la aplicación.

Para poder usar Struts en nuestra aplicación se deben copiar los archivos con extensión ".tld" que necesitemos en el directorio /WEB-INF y copiar struts.jar (y todos los otros archivos commons-*.jar) en el directorio /WEB-INF/lib de la aplicación.

```
<taglib>
  <taglib-uri>
    /WEB-INF/struts-bean.tld
  </taglib-uri>
  <taglib-location>
    /WEB-INF/struts-html.tld
  </taglib-location>
</taglib>
```

Figura D.4 Fragmento de código de definición de las librerías de tags de Struts en el archivo web.xml