DOP @visador

CONSEJERÍA DE JUSTICIA Y ADMINISTRACIÓN PÚBLICA

Documento elaborado por: Francisco José Cantero Villar

Revisado por: Miguel J. Vázquez Rebollo

Versión: 1.0.0
Lugar y fecha: Sevilla, 21 de Julio de 2005

Contenido:

1 Procedimientos de operación de los componentes del sistema.

En el presente documento se van a detallar dos procedimientos de operación, la instalación del sistema partiendo desde cero, así como el reajuste de los parámetros de configuración.

1.1 Creación usuario de base de datos

1. El primer requisito que debe cumplir es la versión de la base de datos. @visador es en actualidad compatible con base de datos Oracle 8i y 9i en los que se hayan habilitado clases Java en el servidor (extensión Java en la base de datos).

2. En la instancia de base de datos donde queramos instalar el sistema, tenemos que dar de alta tres "tablespaces":

   - TS_AVISA_INDICES
   - TS_AVISA _DATOS
   - TS_AVISA _BLOB

   Los tamaños de los mismos quedan a elección del administrador, siendo recomendables de partida 16, 64 y 128 respectivamente, con posibilidad de crecimiento automático. Por temas de rendimiento se recomienda además que el fichero de "tablespace" de índices esté en una unidad de disco física distinta de la de datos normales o columnas BLOB´s (documentos binarios).

3. Dar de alta un usuario Oracle, recomendamos "AVISAMG" pues será el que empleemos como referencia en el resto del documento, con los roles "CONNECT" y "RESOURCE". El "tablespace" por defecto del usuario debe ser "TS_AVISA_DATOS", y el temporal el "TEMP", o el definido en el sistema como temporal. Una vez creado el usuario, con el usuario "SYSTEM", hay que concederle un permiso adicional, con la instrucción "GRANT CREATE TYPE TO AVISAMG", así como "GRANT CREATE ROLE TO AVISAMG" si vamos a permitir el acceso a las API´s mediante PLSQL.

1.2 Script´s de creación de tablas y objetos

El siguiente paso, una vez creado el usuario y los tablespaces es probar dicho usuario, conectando mediante un SQLPLUS o un cliente Oracle tipo TOAD o TORA.

Si la conexión es correcta podemos pasar a crear todas las entidades necesarias para el @visador, en este caso, tablas, índices, constraints, secuencias, tipos y vistas en este orden.

Para ello pensemos a partir de ahora que tenemos el CD del kit @visador en la unidad D, y por tanto desde un SQLPLUS debemos poner:

SQL>@D:\KitAvisador\Scripts\tablas\avisadorGeneral.tab

Y pulsamos intro, si todo va bien ira apareciendo diálogos indicando “tabla creada”. Ejecutamos el resto de los scripts en el siguiente orden:

SQL>@D:\KitAvisador\Scripts\tablas\avisadorGeneral.ind

SQL>@D:\KitAvisador\Scripts\tablas\avisadorGeneral.con

SQL>@D:\KitAvisador\Scripts\tablas\avisadorGeneral.sqs

SQL>@D:\KitAvisador\Scripts\tablas\avisadorGeneral.typ

SQL>@D:\KitAvisador\Scripts\tablas\avisadorGeneral.vw

Una vez terminado de ejecutar los scripts el modelo de datos ya está generado en el usuario propietario. Estos mismos scripts se pueden lanzar desde un cliente TOAD o TORA sin problemas, teniendo solo cuidado en ejecutarlo en el mismo orden.

1.3 Carga de datos

El siguiente paso es cargar los datos básicos necesarios en las tablas de codificación, el proceso es similar al anterior. Seguimos con el ejemplo en el caso de SQLPLUS. El orden en este caso es el siguiente:

SQL> @D:\KitAvisador\Scripts\datos\AR_CONSTANTES.dat

Pulsamos intro y obtendremos los siguientes mensaje:

...

1 fila creada.

1 fila creada.

Validación terminada.

Procedemos con el resto de los scripts:

@D:\KitAvisador\Scripts\datos\AR_ESTADOS.DAT

@D:\KitAvisador\Scripts\datos\AR_EVENTOS.DAT

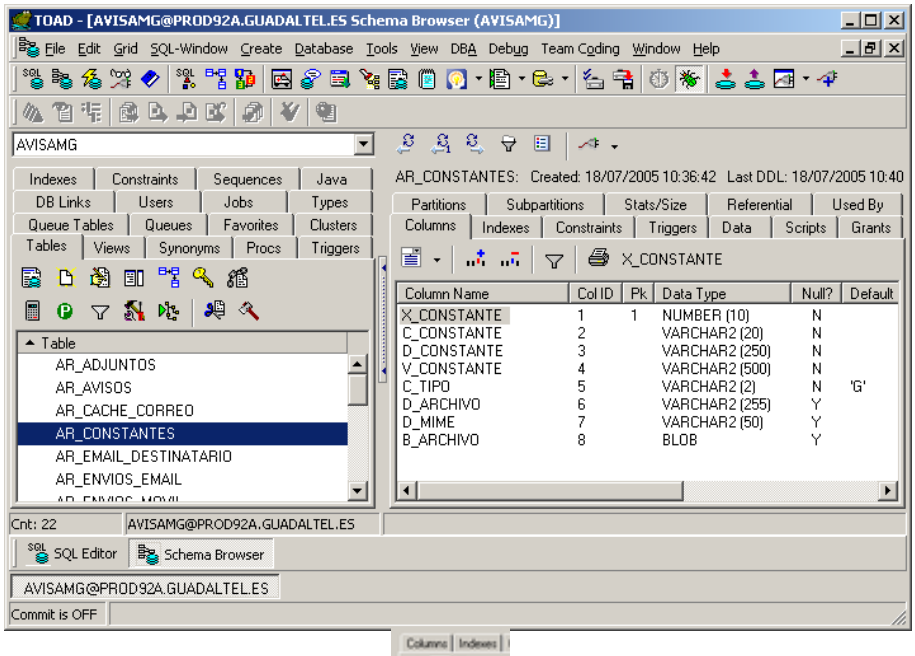
@D:\KitAvisador\Scripts\datos\AR_VARIABLES.dat

Al igual que el caso anterior estos scripts se pueden lanzar como tales en el TOAD o similar.

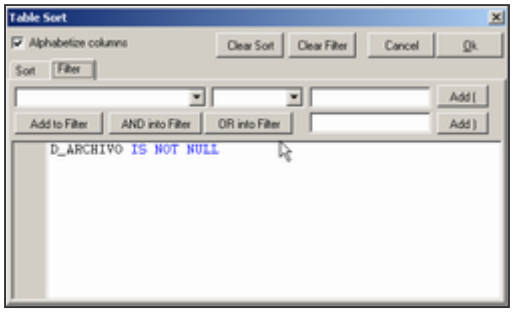
1.3.1 Carga de documentos binarios mediante TOAD

Hay una serie de constantes en el @visador que tienen adjunto algo tipo de documento binario, según el caso, imágenes o documentos html como plantillas. Vamos a ver como se cargarían estos documentos mediante el TOAD.

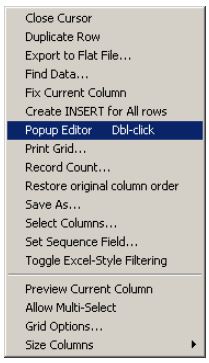
- Abrimos un navegador de objetos de TOAD y nos vamos a la pestaña de tablas, y seleccionamos la tabla constantes AR_CONSTANTES.

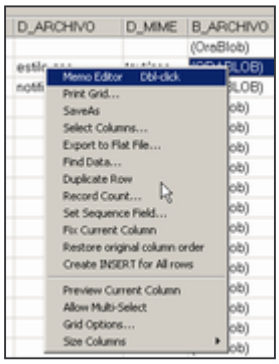


- Pulsamos el siguiente botón  y introducimos el siguiente criterio de filtrado.



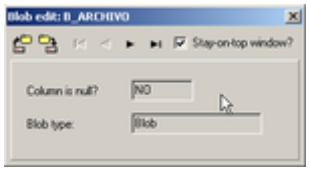
- Pulsamos “OK” y solo nos aparecerán los registros en los que debemos incorporar los archivos. Nos situamos sobre la columna B_ARCHIVO y pulsamos el botón derecho del ratón, y seleccionaremos la opción:





(Ojo, según la versión del TOAD puede que la entrada del menú varíe de nombre)

- Nos aparece la siguiente ventana

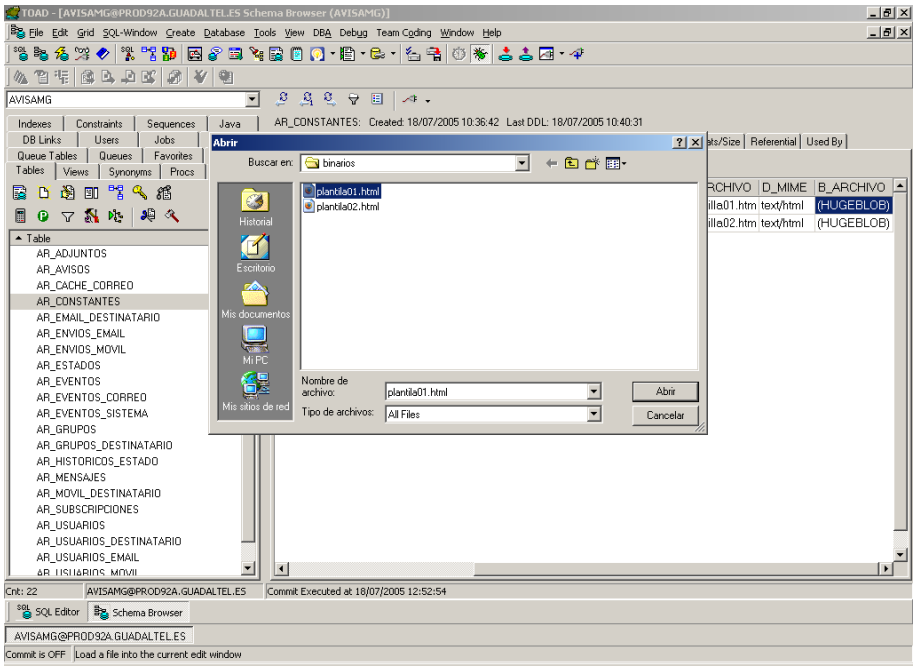


En este menú emergente que nos aparece usaremos los siguientes botones:

-  Para seleccionar un archivo y subirlo a la base de datos.
-  Navegar al primer registro.
-  Navegar al registro anterior.
-  Navegar al registro siguiente.
-  Navegar al último registro.

- Ahora iremos cargando los dos documentos existentes en la carpeta:

D:\KitAvisador\Scripts\datos\binarios



1.3.2 Carga de documentos binarios mediante SQLPLUS

El proceso en este caso es más complejo por eso lo indicamos en el último caso dado que hay que subir a un directorio del servidor de base de datos el contenido de la carpeta **D:\KitAvisador\Scripts\datos\binarios**, supongamos **/tmp**, y seguir los siguientes pasos.

- Conceder de forma temporal el permiso de creación de directorio al usuario AVISAMG con el usuario SYSTEM.

SQL> grant create any directory to avisamg;

- Tras esto nos conectaremos con el usuario AVISAMG, pondremos la siguiente instrucción:

SQL> create directory READDIR as '/tmp';.

- Crear el siguiente procedimiento

```
CREATE OR REPLACE PROCEDURE CARGA_IMAGENES AS
CURSOR IMAGENES IS
SELECT X_CONSTANTE,D_ARCHIVO
FROM AR_CONSTANTES WHERE D_ARCHIVO IS NOT NULL;
V_IMAGENES IMAGENES%ROWTYPE;
AUX_B_ARCHIVO AR_CONSTANTES.B_ARCHIVO%TYPE;
FILS BFILE;
AMT INTEGER;
BEGIN
UPDATE AR_CONSTANTES SET B_ARCHIVO=EMPTY_BLOB() WHERE D_ARCHIVO IS NOT NULL;
COMMIT;
FOR V_IMAGENES IN IMAGENES LOOP
        FILS:= BFILENAME('READDIR',V_IMAGENES.D_ARCHIVO);
        DBMS_LOB.FILEOPEN(FILS, DBMS_LOB.FILE_READONLY);
        AMT:=DBMS_LOB.GETLENGTH(FILS);
        IF AMT>0 THEN
         SELECT B_ARCHIVO INTO AUX_B_ARCHIVO FROM AR_CONSTANTES WHERE
X_CONSTANTE=V_IMAGENES.X_CONSTANTE FOR UPDATE;
         DBMS_LOB.LOADFROMFILE(AUX_B_ARCHIVO, FILS, AMT);
         COMMIT;
        END IF;
        DBMS_LOB.FILECLOSE(FILS);
END LOOP;

END;
```

- Ejecutar el procedimiento anterior:

SQL> exec CARGA_IMAGENES;

- Eliminar el procedimiento y quitarle los permisos de creación de directorios al usuario.

1.4 Carga de procedimientos API PL/SQL

Una vez que hemos cargado todos las tablas de codificación, vamos a proceder a cargar el resto de script´s, en este caso, las clases java y paquetes necesarios si queremos ofrecer una interfaz PL/SQL a nuestras aplicaciones.

1.4.1 Clase Base64

Esta clase es necesaria para realizar operaciones de codificación descodificación de BASE64 a binario y viceversa. Hacemos uso de ella debido a que en Oracle 8i no existen clases y paquetes a tal efecto.

Como en los casos anteriores, desde la ventana del SQLPLUS ponemos las siguientes instrucciones:

SQL> set scan off;

SQL> @D:\KitAvisador\Scripts\base64\Base64plsqljava.sql

1.4.2 Especificaciones de paquete

Vamos primero a cargar las especificaciones de los paquetes, para ello en la ventana del SQLPLUS ejecutamos las siguientes intrucciones:

SQL>set scan off;

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_TYPE.PKS

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_API.PKS

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_TOOL.PKS

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_EVENTO.PKS

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_BASE64.PKS

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_CORREO.PKS

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_AVISADOR.PKS

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_DAEMON.PKS

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_EMISOR.PKS

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_LOOKUP.PKS

Si todo va bien iran saliendo diálogos:

Paquete creado.

1.4.3 Cuerpos de paquete

Al igual que antes, ejecutamos las siguientes lineas en el SQLPLUS.

SQL>set scan off;

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_TYPE.PKB

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_API.PKB

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_TOOL.PKB

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_EVENTO.PKB

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_BASE64.PKB

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_CORREO.PKB

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_AVISADOR.PKB

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_DAEMON.PKB

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_EMISOR.PKB

SQL>@D:\KitAvisador\Scripts\paquetes\AR_PQ_LOOKUP.PKB

Y con esto hemos terminado de cargar todos los objetos necesarios en la base de datos. Al igual que en los casos anteriores estos script´s pueden ser lanzados con el TOAD teniendo cuidado en respetar el orden de ejecución indicado.

1.5 Creación de Rol para API PL/SQL

A continuación vamos a ver como se crearía el rol, para así luego poder concederle este rol a otros usuarios Oracle a los que queramos dar acceso al @visador.

Desde la ventana del SQLPLUS ejecutamos la siguiente línea:

SQL> @"D:\KitAvisador\Scripts\roles y permisos\CREACION ROL AR_R_GESTION.sql"

Tras esto ya se habrá creado el Rol para poder concederselo a un usuario Oracle que vaya a acceder al @visador para la grabación de avisos a través de la API PL/SQL.

En el manual de Administración se describe como se ha de crear un usuario para poder acceder a través de este Rol a los métodos de @visador.

2 Procedimientos de operaciones de producción e implantación.

A continuación vamos a explicar como desplegar la aplicación dentro de un servidor de aplicaciones Java y también a configurar los procesos de base de datos periódicos para lanzar los demonios de envío y recepción de correos.

@visador tiene una doble interfaz, PL/SQL y Java lo cual permite quedarnos solo con una de ellas o bien montar ambas fachadas para que puedan ser atacadas desde la base de datos o bien desde Java mediante Web Services.

Cuando se despliegen estos servicios hay que tener cuidado en solo habilitar los “demonios” o procesos periódicos en uno de los entornos para evitar colisiones entre ambos. Dentro de este documento se explica como arrancar o parar estos demonios según el entorno.

2.1 Despliegue en el servidor de aplicaciones Tomcat

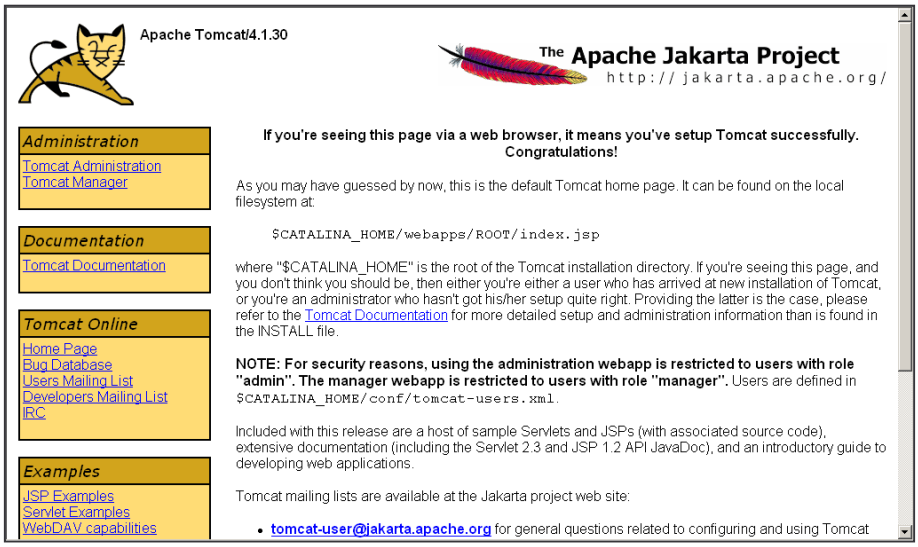
Seguidamente vamos a explicar como se despliega la aplicación de @visador en el Tomcat si queremos disponer de la fachada Java del @visador, es por tanto opcional y no necesaria para actuar únicamente a través de la API PL/SQL. En ese caso solo hay que remitirse en el manual al punto 2.2.1 donde se detalla la activación de los demonios PL/SQL para finalizar la instalación.

Si queremos desplegar la API Java del avisador partimos de la base de que disponemos de un servidor de aplicaciones Tomcat 4.1 o 5.0 y JDK1.4.2 en nuestro servidor.

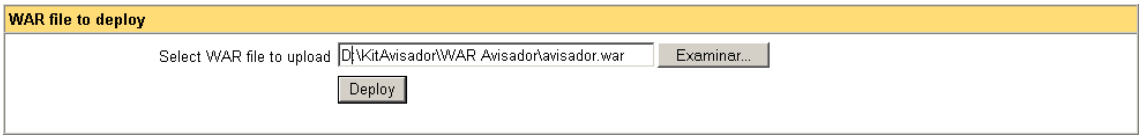
Procedemos a desplegar los wars, para ellos desde una ventana del navegador escribimos la dirección:

http://servidorAplicaciones:8080/

Con lo que nos saldrá la siguiente ventana:



Entramos en “Tomcat Manager” y tras indicar el usuario y clave administrador del Tomcat con el que dimos de alta el servidor, accederemos al módulo de configuración de aplicaciones. Nos vamos a la parte de Instalación (Install) y puslamos el botón examinar, escogemos el fichero WAR del @visador y pulsamos “instalar” (install)



Si todo ha ido bien la nueva aplicación en la lista de aplicaciones de la parte superior de la pantalla. Es mejor pararla por el momento, dando al botón Parar (Stop) para poder ajustar los parámetros de configuración.

Accedemos al directorio:

\Tomcat4\webapps\avisador\WEB-INF\classes

Y modificamos el archivo de propiedades “avisador.properties” con los datos de la conexión:

```
driver=oracle.jdbc.driver.OracleDriver
url=jdbc:oracle:thin:@servidorBaseDeDatos:PuertoBaseDeDatos:SID
username=AVISAMG
password=AVISAMG
```

Seguidamente tenemos que entrar en la base de datos para ajustar los parametros de configuración que hay en la tabla AR_CONSTANTES. Los valores a modificar son:

EMAIL	Dirección de correo del @visador
EMAIL_HELO	Servidor de correo
EMAIL_NOMBRE	Nombre como remitente
EMAIL_POP3	Servidor POP3
EMAIL_SMTP	Servidor SMTP
EMAIL_USER	Usuario de correo
EMAIL_PASS	Clave de correo
T_ERROR_02	Cuenta de administrador en el caso de emitir correos de error.

Con esto podriamos arrancar la aplicación de nuevo la aplicación pulsando el botón Empezar (Start) del Tomcat Manager o bien reiniciando el Tomcat. Si queremos comprobar el correcto funcionamiento de la misma podemos poner en la barra de direcciones de nuestro navegador la siguiente url:

http://servidorAplicaciones:puerto/avisador/services/ArServicioWS?wsdl

Con lo que nos saldría la siguiente dirección:

2.1.1 Ampliación del parámetro de memoria de la máquina virtual de Java

Si tenemos una instalación típica de Tomcat, este se estará ejecutando con un límite de memoria RAM de 64 megas, las cuales se quedan cortas para poder funcionar

correctamente con @visador y resto de aplicaciones web desplegadas en el mismo produciendo pasado un tiempo que se dé errores “memoryAllocation error”.

Recomendamos por tanto realizar el siguiente cambio en la configuración del mismo.

Para ello debemos editar el archivo que se encuentra en el directorio “\Tomcat4\bin” el archivo “catalina.bat”, buscar el siguiente trozo de código e incluir las partes marcadas en amarillo:

```
rem Execute Java with the applicable properties

if not "%JPDA%" == "" goto doJpda

if not "%SECURITY_POLICY_FILE%" == "" goto doSecurity

%_EXECJAVA% %JAVA_OPTS% %CATALINA_OPTS% %DEBUG_OPTS% -Xms32m -Xmx256m -
Djava.endorsed.dirs="%JAVA_ENDORSED_DIRS%" -classpath "%CLASSPATH%" -
Dcatalina.base="%CATALINA_BASE%" -Dcatalina.home="%CATALINA_HOME%" -
Djava.io.tmpdir="%CATALINA_TMPDIR%" %MAINCLASS% %CMD_LINE_ARGS% %ACTION%

goto end

:doSecurity

%_EXECJAVA% %JAVA_OPTS% %CATALINA_OPTS% %DEBUG_OPTS% -Xms32m -Xmx256m -
Djava.endorsed.dirs="%JAVA_ENDORSED_DIRS%" -classpath "%CLASSPATH%" -Djava.security.manager -
Djava.security.policy=="%SECURITY_POLICY_FILE%" -Dcatalina.base="%CATALINA_BASE%" -
Dcatalina.home="%CATALINA_HOME%" -Djava.io.tmpdir="%CATALINA_TMPDIR%" %MAINCLASS%
%CMD_LINE_ARGS% %ACTION%

goto end

:doJpda

if not "%SECURITY_POLICY_FILE%" == "" goto doSecurityJpda

%_EXECJAVA% %JAVA_OPTS% %CATALINA_OPTS% -Xdebug -Xms32m -Xmx256m -
Xrunjdwp:transport=%JPDA_TRANSPORT%,address=%JPDA_ADDRESS%,server=y,suspend=n %DEBUG_OPTS% -
Djava.endorsed.dirs="%JAVA_ENDORSED_DIRS%" -classpath "%CLASSPATH%" -
Dcatalina.base="%CATALINA_BASE%" -Dcatalina.home="%CATALINA_HOME%" -
Djava.io.tmpdir="%CATALINA_TMPDIR%" %MAINCLASS% %CMD_LINE_ARGS% %ACTION%

goto end

:doSecurityJpda

%_EXECJAVA% %JAVA_OPTS% %CATALINA_OPTS% -Xms32m -Xmx256m -
Xrunjdwp:transport=%JPDA_TRANSPORT%,address=%JPDA_ADDRESS%,server=y,suspend=n %DEBUG_OPTS% -
Djava.endorsed.dirs="%JAVA_ENDORSED_DIRS%" -classpath "%CLASSPATH%" -Djava.security.manager -
Djava.security.policy=="%SECURITY_POLICY_FILE%" -Dcatalina.base="%CATALINA_BASE%" -
Dcatalina.home="%CATALINA_HOME%" -Djava.io.tmpdir="%CATALINA_TMPDIR%" %MAINCLASS%
%CMD_LINE_ARGS% %ACTION%

goto end

:end
```

Con esto habremos pasado de 64 megas a un máximo de 256 megas. Si se siguen apreciando caídas por que existan gran número de aplicaciones desplegadas seguir ampliando dicho parámetro en función a las características del servidor y recursos

disponibles.

Para más información es mejor acceder a la FAQ de Apache/Tomcat y buscar en relación a este tema.

2.2 Configuración Demonios

En este punto se va a explicar como podemos configurar los procesos periódicos que revisan los mensajes, y gestionan los envíos y recepciones de correo. Como ya se ha comentado con anterioridad estos demonios se puede habilitar tanto en PLSQL o bien Java, pero nunca los dos al mismo tiempo.

2.2.1 Demonios PL/SQL

Para arrancar o parar los demonios PL/SQL no tenemos más que lanzar un script para cada acción desde un SQLPLUS.

Arrancar los demonios

SQL> @C:\KitAvisador\Scripts\demonios\start.sql

Parar los demonios

SQL> @C:\KitAvisador\Scripts\demonios\stop.sql

2.2.2 Demonios Java

Para arrancar o parar los demonios Java hay que comentar/descomentar los servletsListener del web.xml de la aplicación Java. Por tanto para que tomen efecto los cambios, una vez que hayamos hecho las modificaciones, tenemos que reiniciar la aplicación a nivel de Tomcat Manager o bien el Tomcat.

Arrancar los demonios

Por defecto la aplicación viene con los demonios activos, solo hay que revisar que las siguientes lineas no están comentadas:

```
<listener>
  <listener-class>avisador.servlet.ArDescargaCorreoListener</listener-class>
</listener>
<listener>
  <listener-class>avisador.servlet.ArMensajesRevisaListener</listener-class>
</listener>
<listener>
  <listener-class>avisador.servlet.ArRevisaAvisosListener</listener-class>
</listener>
```

```
<listener>
  <listener-class>avisador.servlet.ArRevisaMensajesListener</listener-class>
</listener>
<listener>
  <listener-class>avisador.servlet.ArSpoolListener</listener-class>
</listener>
```

Parar los demonios

Para parar los demonios debemos comentar los servlet´s arriba indicados de la siguiente forma:

```
<!--Demonios parados
<listener>
  <listener-class>avisador.servlet.ArDescargaCorreoListener</listener-class>
</listener>
<listener>
  <listener-class>avisador.servlet.ArMensajesRevisaListener</listener-class>
</listener>
<listener>
  <listener-class>avisador.servlet.ArRevisaAvisosListener</listener-class>
</listener>
<listener>
  <listener-class>avisador.servlet.ArRevisaMensajesListener</listener-class>
</listener>
<listener>
  <listener-class>avisador.servlet.ArSpoolListener</listener-class>
</listener> -->
```

3 Historia de versiones

VERSION	FECHA	AUTOR	DESCRIPCIÓN
1.0.0	21/07/2005	FJCV	Primera versión del documento, @visador v1.0.0