Junta de Andalucía

Manual de operación

Trew@ 2.6.11.0

Versión: v01r00

Fecha: 15/03/2025

Queda prohibido cualquier tipo de explotación y, en particular, la reproducción, distribución, comunicación pública y/o transformación, total o parcial, por cualquier medio, de este documento sin el previo consentimiento expreso y por escrito de la Junta de Andalucía.

HOJA DE CONTROL

Título	Trew@ 2.6.11.0		
Entregable	Manual de operación		
Nombre del Fichero	TRW26110E_MIC_Manual_de_operación_v01r00.odt		
Autor	UTE		
Versión/Edición	v01r00	**Fecha Versión**	15/03/2025
Aprobado por	ADA	**Fecha Aprobación**	
		N.º Total Páginas	57

REGISTRO DE CAMBIOS

Versión	Causa del cambio	Responsable del cambio	Área	Fecha del cambio
v01r00	Versión inicial	UTE	Sv. Informática	15/03/2025

CONTROL DE DISTRIBUCIÓN

Nombre y Apellidos	Cargo	Área
Juan Luis Ceada Ramos	Subdirector	Servicios Digitales y Analítica de Datos
Manuel Escobar Montes	Jefe de Servicio	Administración Digital
José Antonio Frutos Beltrán	Técnico	Gabinete Estratégico
Almudena López Maraver	Directora de Proyecto	PMO
José Luis Romero Dorado	Jefe de proyecto	UTE
Juan Carlos Gómez de Tejada Díaz	Consultor	UTE

ÍNDICE

1 INTRODUCCIÓN....5
1.1 Objeto....5
1.2 Alcance....5
2 REQUISITOS PREVIOS A LA INSTALACIÓN....6
2.1 Recursos hardware....6
2.1.1 Estaciones cliente....6
2.1.2 Conectividad....6
2.2 Recursos software....7
2.2.1 Restricciones técnicas del sistema....7
2.2.2 Relación entre recursos....7
3 INSTALACIÓN....8
3.1 Base de datos....8
3.1.1 Instalación del motor completo....8
3.1.1.1 Creación de tablas de espacio (Tablespaces)....9
3.1.1.2 Creación del esquema TREWA....10
3.1.1.3 Creación de objetos del esquema....10
3.1.2 Actualización desde un motor existente....12
3.1.2.1 Migración de notificaciones....12
3.1.2.2 Migración de pagos....13
3.1.2.3 Generación de CSV de expedientes....14
3.1.3 Revisión de objetos y datos en el motor....16
3.1.4 Tablas necesarias....16
3.1.5 Secuencias....17
3.1.6 Constantes generales (tabla “GN_CONSTANTES”)....17
3.1.7 Datos de Sistemas por defecto....17
3.1.8 Datos de Componentes por defecto....18
3.1.9 Configuración de los sistemas (tabla TR_CONSTANTES)....19
3.1.10 Generación de documentos mediante OpenOffice/ LibreOffice....20
3.1.11 Generación de documentos a PDF mediante itext....20
3.1.12 Perfiles de usuario....20
3.2 Empaquetado distribuido....21

3.2.1 Web.xml....21
3.2.1.1 Configuración de niveles de permisos....21
3.2.1.2 Conexiones....22
3.2.1.3 Configuración de perfiles de conexión....22
3.2.2 Trazabilidad Log4Java....25
3.3 Servidor de aplicaciones (WildFly 15.0.1.Final)....27
3.3.1 Configuración del servidor de aplicaciones....27
3.3.1.1 Creación de usuarios de WildFly....27
3.3.1.2 Validar configuración inicial....30
3.3.1.3 Modificación de la IP de acceso público y de acceso para la administración....31
3.3.1.4 Configuración del modo de arranque....34
3.3.1.5 Inclusión del módulo para la conexión con Oracle....34
3.3.1.6 Cambios en el fichero de configuración standalone....35
3.4 Instalación del empaquetado....39
3.5 Configuración de la Herramienta de Administración de Trew@....39
3.5.1 Configuración de la firma delegada por EJB....39
3.5.2 Configuración de EnidocWS y EniexpWS....41
3.5.2.1 Configuración de la Herramienta Centralizada de Verificación....41
3.5.2.2 Configuración de Port@firmas v3....43
3.5.3 Configuración del componente DIR3....46
3.5.4 Configuración de la fachada de @firma....47
3.6.1 Configuración de WebOffice....47
3.6.2 Condiciones, acciones, variables y selectores de plantillas....48
3.6.3.1 Configuración de los servlets....50
4 MANTENIMIENTO....53
4.1 Base de datos....53
4.2 Pantallas de Administración de Trew@....53
4.3 Condiciones y acciones....53
4.4 Variables de los documentos....53

1 INTRODUCCIÓN

1.1 Objeto

El presente documento recoge los manuales de operación, instalación y datos de configuración de diversos componentes pertenecientes a un motor Trew@.

1.2 Alcance

Este documento se encuentra dirigido a:

- Dirección de Proyecto.
- El colectivo de usuarios y desarrolladores de sistemas que usan Trew@.

2 REQUISITOS PREVIOS A LA INSTALACIÓN

Para poder instalar y asegurar un buen funcionamiento de Trew@, se deben cumplir los siguientes requisitos:

- Servidor de Aplicaciones **Wildfly 15.0.1.FINAL**
- **JDK 8**
- Base de datos Oracle versión 11g para el modelo de datos propio de Trew@
- Acceso a una instalación de @firma, para la autenticación de usuarios y la firma electrónica de documentos y expedientes. Para el uso de la

Figura 1- Advertencia de requisitos

2.1 Recursos hardware

2.1.1Estaciones cliente

Dato	Valor mínimo	Valor recomendado
Procesador	-	-
Memoria RAM	-	-
Tamaño Almacenamiento	-	-
Navegadores web	Mozilla Firefox, Google Chrome	Mozilla Firefox, Google Chrome
Sistemas operativos	Windows, Linux	Windows, Linux
Otros	-	-

Tabla 1- Recursos hardware para estaciones de cliente

2.1.2Conectividad

Dato	Valor mínimo	Valor recomendado
Tarjeta de red	-	-
Tipo de red	Ethernet 10/100/1000	Ethernet 10/100/1000
Otros	-	-

Tabla 2- Recursos hardware de conectividad

2.2 Recursos software

2.2.1Restricciones técnicas del sistema

Elemento	Descripción
Sistema operativo	Windows, Linux
Servidor de aplicaciones	Wildfly-15.0.1-final
Servidor de base de datos	Oracle 11g Release 2 (11.2.0.4)
Compilador	JDK 1.8
JVM	JVM 8
Otros	-

Tabla 3- Restricciones técnicas del sistema

2.2.2Relación entre recursos

Recurso Software	Recurso Hardware
Archivos con extensión *.ear (aplicaciones web)	Servidor de aplicaciones
Archivos con extensión *.xml (ficheros de definición de recursos)	Servidor de aplicaciones
Archivos con extensión *.sql (scripts para base de datos)	Servidor de base de datos
Archivos con extensión *.jar	Servidor de aplicaciones

Tabla 4- Trazabilidad de relación entre recursos

3 INSTALACIÓN

En los siguientes apartados se describirán los pasos a seguir para realizar una instalación completa de Trew@. Describiendo todos los niveles: base de datos, empaquetado distribuido, servidor de aplicaciones y configuración de la aplicación desplegable.

3.1 Base de datos

En este apartado se describen los cambios necesarios a realizar para tener una base de datos actualizada a la versión 2.6.11.0 de Trew@. Se indicarán los pasos partiendo, tanto de una base de datos completamente nueva, como partiendo de una con una versión anterior de Trew@.

3.1.1 Instalación del motor completo

NOTA: *Los apartados descritos a continuación solo se deberán llevar a cabo, sí y solo sí, se requiere la instalación de un motor de tramitación desde cero. En caso contrario, continúe en el apartado 3.1.2.*

Para la instalación inicial del motor de tramitación, se facilitan un conjunto de scripts *.sql.

Los scripts de base de datos se encuentran con codificación UTF-8. Se deben configurar previamente para evitar posibles fallos.

3.1.1.1 Creación de tablas de espacio (Tablespaces)

Para su correcto funcionamiento, Trew@ hace uso de tres tablas de espacio destinando a cada uno de ellos una utilidad diferente.

Tablespace[1]	Utilidad	Tamaño mínimo recomendado
TS_TREWA	Almacenamiento de datos	100 Mb
TS_TREWA_I	Almacenamiento de índices	200 Mb
TS_TREWA_BLOB	Almacenamiento de valores de columnas BLOB	200 Mb

Tabla 5- Descripción de tablas de espacio

Para la creación de las tablas de espacio se debe ejecutar el fichero *Creacion_tablespace_Trewa.sql*, distribuido con el CD de esta versión bajo la ruta *Base de datos/Oracle/Scripts tablespaces.*

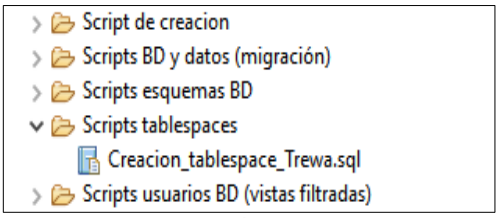

Figura 2- Localización del script de creación de tablas de espacio

La ejecución del script debe ser realizada con el usuario SYSTEM de base de datos y, en el script, se deberán sustituir las variables definidas por los valores deseados.

NOTA: *Por ejemplo, para unos 100 procedimientos definidos, con unas 2000 fases posibles, unas 3500 transiciones entre fases, unos 2000 expedientes, y unos 1000 documentos almacenados en blob, una estimación para el tamaño de los tablespaces sería:*

Tablespace	Tamaño estimado
TS_TREWA	700 Mb
TS_TREWA_I	1 Gb
TS_TREWA_BLOB	1 Gb

Tabla 6- Estimación de tamaño futuro de tablas de espacio

1 Los nombres indicados son recomendaciones.

El administrador de la base de datos debe tener en cuenta el crecimiento de las tablas de espacios conforme se vayan incluyendo datos en la misma.

3.1.1.2Creación del esquema TREWA

El modelo de datos de Trew@ está basado en un esquema único que hace uso de las tablas de espacio que se crearon en el apartado anterior.

Se creará un esquema de base de datos que será el propietario de todos los objetos. Para ello, se debe ejecutar el fichero *Creacion_esquema_Trewa.sql*, distribuido con el CD de esta versión bajo la ruta *Base de datos/Oracle/Scripts esquemas BD*

La ejecución del script debe realizarse con el usuario SYSTEM de base de datos y, en el script, se deberán sustituir las variables definidas por los valores deseados.

NOTA: *Mediante el script se crea un esquema llamado TREWA que hace uso de las tablas de espacio con nombres TS_TREWA, TS_TREWA_I y TS_TREWA_BLOB. Estos son los valores por defecto y si es necesario pueden ser modificados.*

3.1.1.3Creación de objetos del esquema

Se deberán ejecutar los scripts distribuidos con el CD de esta versión bajo la ruta *Base de datos/Oracle/Script de creación* en el orden indicado en el archivo *00. Leeme.txt* del mismo directorio.

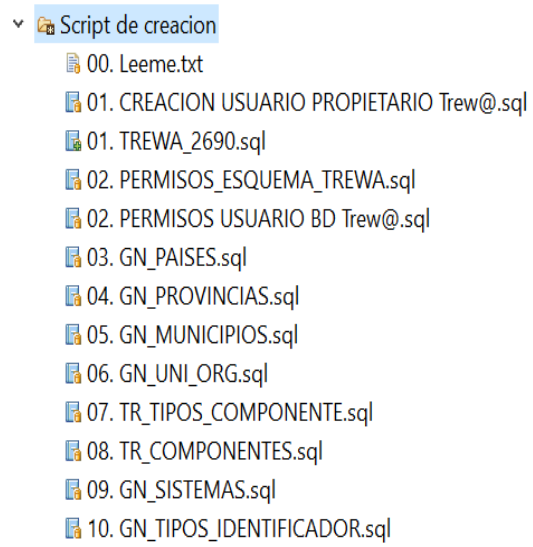

Figura 3: Localización de scripts de creación

Estos scripts deben ser ejecutados con el usuario del esquema que se creó en el apartado anterior y, en el script, se deberá sustituir la variable “OWNER” por el nombre de este esquema. Adicionalmente, hay algunos scripts que tienen definidas variables que deben ser sustituidas por los valores adecuados; estos scripts son los destacados en negrita entre los siguientes:

- 01. TREWA_26100.sql

- 02. PERMISOS_ESQUEMA_TREWA.sql
- 03. GN_PAISES.sql
- 04. GN_PROVINCIAS.sql
- 05. GN_MUNICIPIOS.sql
- 06. GN_UNI_ORG.sql
- 07. TR_TIPOS_COMPONENTE.sql
- 08. TR_COMPONENTES.sql
- 09. GN_SISTEMAS.sql
- 10. GN_TIPOS_IDENTIFICADOR.sql
- 11. GN_TIPOS_ORGANISMO.sql
- 12. TR_ERRORES.sql
- 13. GN_TIPOS_ORGANIZACION.sql
- 14. GN_TIPOS_VIA.sql
- 15. MIGRACION_INT_JURIDICOS.sql
- 16. GN_USUARIOS.sql
- 17. GN_CONSTANTES.sql
- 18. TR_CONSTANTES.sql
- 19. TR_PERFILES_USUARIOS.sql
- 20. TR_USUARIOS_X_PERFILES_USUARIO.sql
- 21. TR_TIPOS_PARRAFOS.sql
- 22. TR_TIPOS_CERTIF.sql
- 23. TR_TIPOS_NORMATIVA.sql
- 24. TR_DATOS_COMPONENTE.sql
- 25. TR_ESTELAB_ENI.sql
- 26. TR_FORMATODOC_ENI.sql
- 27. TR_NUMERADORES.sql
- 28. TR_PARAMETROS.sql
- 29. TR_PLANTILLAS.sql
- 30. TR_RAZONES_INTERES.sql
- 31. TR_TEXTOS_DISPOSICIONES.sql
- 32. TR_TIPODOC_ENI.sql
- 33. TR_TIPOS_CONTACTO.sql
- 34. TR_TIPOS_INDICACION.sql
- 35. TR_METADATOS.sql

- **36. MOVE_BLOB.sql**
- **37. MOVE_INDEX.sql**

3.1.2Actualización desde un motor existente.

Para facilitar la actualización, se facilitan una serie de scripts consecutivos permitiendo la actualización desde diferentes versiones anteriores.

Para ello, se deben ejecutar los scripts distribuidos con el CD de esta versión bajo la ruta *Oracle/Scripts BD y datos (migración)*, en el orden indicado en el fichero *00.Leeme.txt*, con el usuario del esquema y, en los scripts, se deberá sustituir la variable "OWNER" por el nombre del propietario del esquema. Hay algunos scripts que tienen definidas variables que deben ser sustituidas por los valores adecuados; estos scripts son los siguientes:

- **trewa_v2690_a_v26100.sql** – Debe sustituirse la variable USUARIO_PROPIETARIO por el valor de la columna C_USUARIO de la tabla GN_USUARIOS del usuario que será el propietario.

NOTA: *A continuación, se describe cómo deben realizarse estas ejecuciones de una forma mas detallada.*

Para la ejecución hay que tener en cuenta algunos detalles, dependiendo de la versión desde la que se provenga:

- En el caso de que se provenga de una versión 2.1.* o anterior, se tendrá que ejecutar la aplicación "Migrador de Notificaciones". Esta ejecución es obligatoria y necesaria si existen notificaciones en el motor de tramitación antes de iniciar la migración a la siguiente versión.
- En el caso de que se provenga de una versión 2.2.* o anterior, se tendrá que ejecutar la aplicación "Migrador de pagos". Esta ejecución es obligatoria y necesaria.
- En el caso de que se provenga de una versión 2.4.* o anterior, se tendrá que ejecutar la aplicación "Generador CSV expedientes". Esta ejecución es obligatoria y necesaria.

3.1.2.1Migración de notificaciones

NOTA: *Este paso se realizará exclusivamente al finalizar la actualización de la versión 2.1.0 a la 2.2.0 (2.1.1 – 2.2.1).*

Debido a la modificación realizada en el modelo de datos de Trew@, en su versión 2.2.0, para la gestión de notificaciones es de obligado cumplimiento realizar acciones adicionales.

En caso de que se haga uso del componente Notific@dor, para la gestión de notificaciones, se deberá ejecutar la aplicación de escritorio *Migrador de Notificaciones*, distribuida bajo la ruta *Aplicaciones J2EE/ Migrador_Notificaciones*.

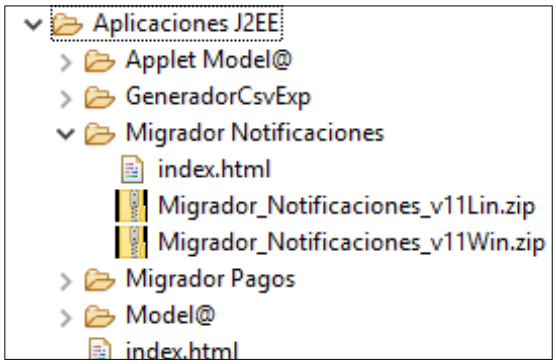

Figura 4: Localización migrador de notificaciones

Se remite al lector al documento de “Soporte Trew@ a notificaciones”, que puede encontrar en el entregable en la subcarpeta “Documentación”.

En caso de que el sistema no haga uso de la gestión de notificaciones, componente Notific@dor, tan solo hará falta lanzar el script *Script_notificaciones.sql* suministrado en la carpeta *Oracle/Scripts BD y datos (migración)*.

3.1.2.2Migración de pagos

NOTA: *Este paso se realizará exclusivamente al finalizar la actualización de la versión 2.2.0 / 2.2.1 a la 2.3.0 /2.3.1.*

Debido a la modificación realizada en el modelo de datos de Trew@, en su versión 2.3.0, para la gestión de pagos, es de obligado cumplimiento realizar acciones adicionales.

Se deberá ejecutar la aplicación de escritorio “Migrador de Pagos”, distribuida en el entregable de dicha versión. Para asegurar el correcto funcionamiento de la aplicación de migración, previamente se habrá tenido que actualizar la versión a la 2.3.0 mediante los scripts distribuidos, en caso contrario, se mostrará un error informando de la situación.

La aplicación incluye el fichero *conexion.properties*, ubicado en el directorio donde se encuentra el fichero migradorPagos.jar, donde se configuran los siguientes parámetros:

Campo	Obligatorio	Descripción
tipo	Si	Tipo de Base de Datos del esquema de Trew@ (Oracle,…).
nombreMaquina	Si	IP de la máquina donde se aloja el esquema de Base de Datos de Trew@.
puerto	Si	Puerto de acceso para la conexión con el esquema de Base de Datos de Trew@.
nombreBD	Si	Nombre de la Base de Datos donde se aloja el modelo de datos de Trew@.
nombreUsuario	Si	Nombre de Usuario del esquema de Trew@.
claveUsuario	Si	Clave de acceso del esquema de Trew@.

Tabla 7: Propiedades para el migrador de pagos

La aplicación se encuentra bajo la ruta *Aplicaciones J2EE/ Migrador_Notificaciones*.

- Aplicación en Windows: se ejecutará el fichero *MigradorPagos.bat*
- Aplicación en Linux: se ejecutará el fichero *MigradorPagos.sh*.

Si se realiza correctamente la migración se mostrará el siguiente mensaje: “MIGRACIÓN COMPLETADA CORRECTAMENTE”, en caso contrario, se mostrará un mensaje indicando el error y se restaurarán los cambios realizados.

3.1.2.3Generación de CSV de expedientes

NOTA: *Este paso se realizará exclusivamente al finalizar la actualización de la versión 2.4.1 a la 2.5.0* .

En la versión 2.5.0 se realizó una modificación en el modelo de datos de Trew@, siendo de obligado cumplimiento realizar acciones adicionales para la gestión de expedientes.

Se deberá ejecutar la aplicación *Generador CSV expedientes*, ubicada en la ruta *Aplicaciones J2EE/GeneradorCsvExp*.

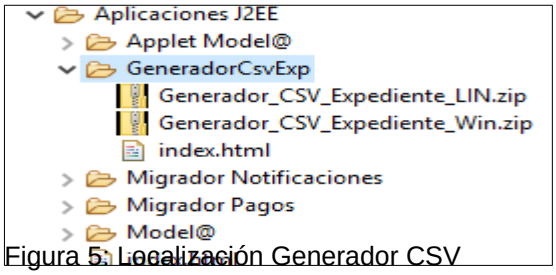

Figura 5: Localización Generador CSV expedientes

La aplicación incluye el fichero *conexion.properties*, ubicado en el directorio donde se encuentra el fichero *generadorCSVExp.jar*, donde se configuran los siguientes parámetros:

Campo	Obligatorio	Descripción
tipo	Si	Tipo de Base de Datos del esquema de Trew@ (Oracle,…).
nombreMaquina	Si	IP de la máquina donde se aloja el esquema de Base de Datos de Trew@.
puerto	Si	Puerto de acceso para la conexión con el esquema de Base de Datos de Trew@.
nombreBD	Si	Nombre de la Base de Datos donde se aloja el modelo de datos de Trew@.
nombreUsuario	Si	Nombre de Usuario del esquema de Trew@.
claveUsuario	Si	Clave de acceso del esquema de Trew@.

Tabla 8: Propiedades generador de CSV de expedientes

- Aplicación en Windows: se ejecutará el fichero *GeneradorCsvExp.bat*
- Aplicación en Linux: se ejecutará el fichero *GeneradorCsvExp.sh*

3.1.3Revisión de objetos y datos en el motor

En este apartado se tratarán los principales aspectos por revisar y/o actualizar para poder validar como correcta la instalación de Trew@.

Una vez realizada la importación de los objetos de Trew@ satisfactoriamente, se puede verificar su correcta creación. Para ello se puede ejecutar la siguiente sentencia sql:

```
SELECT COUNT(*), STATUS, OBJECT_TYPE FROM USER_OBJECTS GROUP BY STATUS, OBJECT_TYPE ORDER BY COUNT(*) DESC;
```

Figura 6: Sentencia de revisión de objetos del esquema Trew@

El resultado de esta sentencia debe ser :

COUNT(*)	STATUS	OBJECT_TYPE
426	VALID	INDEX
137	VALID	TRIGGER
131	VALID	TABLE
131	VALID	VIEW
100	VALID	SEQUENCE
22	VALID	LOB
1	VALID	FUNCTION

Tabla 9: Resultados de la sentencia de revisión de objetos del esquema Trew@

3.1.4Tablas necesarias

Se pueden distinguir dos tipos de tablas según el comienzo de su nombre:

- “TR_”: Constituyen el “metamodelo” en el que se apoya Trew@ y debería estudiarse cualquier modificación en las mismas, ya que afectan a otros objetos necesarios para el correcto funcionamiento.
- “GN_”: Serían un conjunto de tablas necesarias para que Trew@ funcione, pero que no se consideran parte integrante del “metamodelo” de Trew@ sino que, más bien, se consideran tablas “generales” a todas las aplicaciones. Por ejemplo, se encuentran entre ellas aquellas tablas que guardan la estructura organizativa (Unidades Orgánicas de la Junta de Andalucía, por ejemplo), las que guardan los usuarios que utilizan Trew@, etc.

3.1.5Secuencias

- “TR_S...”: Secuencias para el buen funcionamiento de las pantallas de mantenimiento de Trew@
- “GN_S...”: Secuencias para el buen funcionamiento de las tablas generales.

3.1.6Constantes generales (tabla “GN_CONSTANTES”)

La tabla de constantes generales debe contener unos datos mínimos:

- TRVERS: Almacena el valor de la versión de Trew@, en este caso 2.6.11.0.
- TRSTMA: Almacena el nombre del sistema por defecto para inicializar el motor de tramitación. Por defecto, su valor será “TREW@”, que es el nombre del sistema por defecto durante la instalación de Trew@. Este valor permite que al instanciarse la API se inicialice el motor con las constantes por defecto.
- TRUSUPROP: Usuario propietario de los objetos de Trew@. Debe contener como valor el nombre del usuario, existente en la tabla GN_USUARIOS, que actuará como propietario de los objetos de Trew@.

  En esta versión se introduce el acceso a la Herramienta de Administración de Trew@ utilizando la autenticación mediante certificado electrónico. Debido a esto, será necesario tener un registro en la tabla GN_USUARIOS, teniendo en cuenta que el identificador que se le asigne debe ser válido. Para ello, si no existe ningún registro, se debe ejecutar el fichero *Creacion_usuario_propietario.sql*, distribuido con el CD de esta versión bajo la ruta *Base de datos/Oracle/Usuario.*

  NOTA: *Se recomienda consultar el apartado 3.5.4 de este manual.*

3.1.7Datos de Sistemas por defecto

Debe existir al menos un sistema (por defecto "TREW@") en la tabla GN_SISTEMAS para poder iniciar el motor con los valores por defecto. El nombre de este sistema debe corresponder con la constante "TRSTMA" de la tabla GN_CONSTANTES descrita en el apartado anterior. Se recomienda que este sistema por defecto haga referencia al propio motor de tramitación, ya que así se define cierta información que por defecto es necesaria para poder trabajar con el motor, así como otros datos que pueden ser compartidos por otros sistemas por el hecho de estar asociadas a este sistema por defecto.

3.1.8Datos de Componentes por defecto

Debe existir al menos un componente (por defecto "TREW@") en la tabla TR_COMPONENTES para poder utilizar ciertas APIs orientadas al funcionamiento de Trew@ dentro de un entorno w@ndA, necesario para poder interactuar con otros componentes w@ndA. Esta información contendrá los datos del componente de tramitación que representa el Trew@ instalado, así como otros datos de configuración contenidos en la tabla TR_DATOS_COMPONENTE, necesarios para la integración con otros componentes. El nombre de este valor puede ser en principio cualquiera, y será etiquetado en w@ndA como un componente de tramitación con un "código de identificación" en un entorno w@ndA.

De la misma forma, en esta versión se incluyen datos por defecto de un componente “BUS” que representa el bus de comunicación en un entorno w@ndA y que tendremos que configurar para poder hacer uso de él (para más detalles sobre el funcionamiento en un entorno w@ndA se remite al lector al documento Guía de integración w@ndA).

En la tabla TR_CONSTANTES, asociado al sistema por defecto, debe existir un valor "MOTOR_TRAMITA" que representa al nombre del componente de tramitación por defecto definido con la que ciertas APIs que ofrece Trew@ puede obtener la información del componente de tramitación definido por defecto necesaria, por ejemplo, en el XML de exportación de expedientes.

Además del componente por defecto, también existirán como datos creados durante la instalación, otros componentes (así como datos asociados a éstos) tales como "WEBOFFICE", para el caso en el que se prefiera el tratamiento de documentos mediante Open Office.

3.1.9 Configuración de los sistemas (tabla TR_CONSTANTES)

Las constantes asignadas al sistema por defecto (típicamente "TREW@", ver apartado anterior) son las siguientes:

- "MOTOR_TRAMITA" → Define el motor de tramitación que se va a utilizar. Por defecto "TREW@" (ver apartado anterior).
- "FORMATO_FECHA" → Define el formato de la fecha que va a ser utilizada por el motor de tramitación, por ejemplo “DD/MM/YYYY”. Este formato será el que utilicen, también, todas las APIs de Trew@ cuando trabajen con fechas.
- "COMP_PERMISOS_DOC" → Define la posibilidad de habilitar, o no, la comprobación de los permisos sobre documentos de los que un usuario dispone para una determinada fase en un expediente. El valor configurado por defecto es "N", indicando que no se habilite.

  Si se configura a "N", Trew@ solo verifica que el documento esté permitido al perfil de tramitación del usuario, independientemente del permiso concreto asignado en la definición del procedimiento, todos los usuarios con el perfil adecuado tienen el mismo permiso sobre el documento, sin distinguir si es para "generar", "editar", "firmar", etc...

  Si se configura a "S", Trew@ verificará que el usuario tenga el permiso sobre tareas de manipulación de escritos exactamente igual a como se haya definido en el procedimiento, además del perfil de tramitación adecuado.

  Por ejemplo, si para un tipo de tarea sólo se han asignado permisos de "Editar", las APIs destinadas a la edición de los párrafos o datos de documentos permitirán realizar la edición del mismo sólo en el caso de que el usuario tenga el perfil de tramitación al que se ha asignado este permiso. Además, no se permitirán llamadas a otras APIs que no sean para editar el documento (por ejemplo, no se podría generar documentos aunque sí editarlos). De esta forma un usuario que no tenga este perfil no podrá editar los datos del documento aunque tenga otros perfiles con otros permisos sobre el documento (por ejemplo de firma o generación).
- "URL_REPORT_JAVA" → Define la url donde se encuentran los ficheros de configuración para la generación de documentos PDF, mediante Java. Por ejemplo “http://nombreMaquina:puerto/agendaTrewa/editor/config/". Esta url es usada por las APIs java para la generación de documentos PDF, mediante itext, para localizar las plantillas .xml que formatean el documento.
- “REPOSITORIO_DOC_EXT” → Define la posibilidad de habilitar, o no, la utilización de un repositorio externo de documentos. El valor configurado por defecto es “N”, indicando que no se habilite.

  Si se configura a “S”, indica a la API de Trew@ que el almacenamiento de los documentos físicos se hará en un repositorio externo, siempre que se esté usando un gestor documental externo.

NOTA: *Aunque estas constantes deben existir por defecto asociadas al sistema “Trew@”, se aconseja que cada sistema nuevo defina las suyas propias (concretamente todas excepto la*

1ª) para así disponer de un "juego de configuración" concreto (independiente del resto) para dicho sistema.

3.1.10 Generación de documentos mediante OpenOffice/ LibreOffice

Para la utilización de la funcionalidad de generación de documentos mediante OpenOffice/LibreOffice se deberán ajustar los datos necesarios del componente "WEBOFFICE". Para más detalle sobre la instalación de los componentes necesarios para la utilización de esta funcionalidad ver la documentación relativa al componente WebOffice[2].

3.1.11 Generación de documentos a PDF mediante itext

Si se desea esta nueva funcionalidad disponible desde Trew@ v1.3.0, debe definirse para el sistema la constante "URL_REPORT_JAVA", que debe contener la ruta donde se encuentren los ficheros xml que contienen los datos para el formato a PDF de los documentos basados en párrafos, o lo que es lo mismo, las "plantillas" para este tipo de generación.

De la misma forma, si se va a utilizar la generación de documentos que ofrece Trew@, la tabla "TR_TIPOS_PARRAFOS" debe contener los datos necesarios para la correcta visualización de los documentos generados. Estos datos permiten el formateo del documento con el estilo de los párrafos que componen el documento.

3.1.12 Perfiles de usuario

Deben existir por defecto dos perfiles de usuarios asignados al sistema "TREW@" denominados "TR_R_USUARIO" y "TR_R_ADMINISTRADOR", necesarios en el caso de utilizar Trew@ con aplicaciones basadas en usuarios que no son de base de datos (aplicaciones que utilizan los mismos datos de conexión independientemente del usuario de la aplicación). Estos perfiles se darán a los usuarios definidos en el sistema a modo de perfiles de tramitación desde la pantalla de administración habilitada al efecto. Si un usuario que no es de base de datos no tiene alguno de estos perfiles no podrá ejecutar las APIs de Trew@.

3.1.13 Creación de otros usuarios administradores distintos al propietario

La herramienta de administración de Trew@, permite utilizar distintos usuarios administradores, típicamente, para separar la administración de los distintos sistemas que hagan uso del motor de tramitación.

Para crear estos usuarios administradores se pueden seguir dos métodos:

- Ejecutar el fichero *Creacion_usuario_administrador.sql*, distribuido con el CD de esta versión bajo la ruta *Base de datos/Oracle/Usuario*, sustituyendo las variables definidas por los valores apropiados.
- Crear un usuario desde la herramienta de administración de Trew@ y darle los perfiles apropiados.

2 Consulte la segunda referencia, en la tabla de referencias.

A partir de entonces el usuario administrador creado podrá entrar con las pantallas de administración de Trew@, pero sólo podrá administrar los sistemas en los que se haya dado permiso.

3.2 Empaquetado distribuido

Para el correcto funcionamiento de la Herramienta de Administración de Trew@ y de los componentes que se despliegan junto a ésta, es necesario realizar algunas configuraciones en el propio empaquetado TrewaAdm-2.6.11.0.ear que se distribuye en este CD.

3.2.1Web.xml

Para el correcto funcionamiento de la Herramienta de Administración de Trew@, se deberán realizar configuraciones en el fichero web.xml, que se encuentra bajo la ruta *TrewaAdm-2.6.11.0.ear/TrewaAdm-2.6.11.0.war/WEB-INF/.*

3.2.1.1Configuración de niveles de permisos

La Herramienta de Administración de Trew@ permite, en función de los perfiles de los administradores, configurar diferentes niveles de permisos con los que se podrá acceder a sistemas y configuraciones del menú. Estos niveles se configuran con dos variables en el fichero web.xml:

- **TrAdminTrew@:** Indica si los usuarios administradores de Trew@ pueden administrar el sistema por defecto.
- **TrPermisosPropietario:** Indica si los usuarios administradores de Trew@ pueden realizar el mantenimiento de componentes, constantes generales y errores.

Para facilitar la comprensión se proporciona una tabla de los diferentes niveles de acceso en función de la configuración de estas variables y los perfiles de los usuarios administradores.

	Gestión del sistema por defecto	Gestión de componentes, constantes generales y errores.
TrAdminTrew@ (SI) **TrPermisosPropietario (SI)**	- Super-administrador (propietario) - Administrador - Administrador Delegado	- Super-administrador (propietario) - Administrador
TrAdminTrew@ (SI) **TrPermisosPropietario (NO)**	- Super-administrador (propietario) - Administrador - Administrador Delegado	- Super-administrador (propietario)
TrAdminTrew@ (NO) **TrPermisosPropietario (SI)**	- Super-administrador (propietario)	- Super-administrador (propietario) - Administrador
TrAdminTrew@ (NO) **TrPermisosPropietario (NO)**	- Super-administrador (propietario)	- Super-administrador (propietario)

Tabla 10: Niveles de acceso

NOTA: *Por defecto, la configuración de estas variables es la correspondiente a la configuración más permisiva.*

3.2.1.2Conexiones

La ventana de conexión de esta aplicación permite obtener una lista de perfiles de conexión definidos previamente si se desea. Para que los perfiles aparezcan en la lista deben ser configurados en el fichero *web.xml* del empaquetado web (TrewaAdm-2.6.11.0.war). Para ello, se añadirá un parámetro del contexto cuyo nombre debe comenzar con el prefijo "*TrPerfil_"* por cada uno de los perfiles necesarios, por ejemplo, "*TrPerfil_Default*".

Figura 7: Configuración de perfiles en web.xml

Las partes en color negro de la configuración serán las que deben ajustarse a la configuración propia de cada instalación.

NOTA: *Como se puede apreciar en la figura, la etiqueta <param-value> tiene valores con formatos diferentes, esto se debe a que, a partir de la versión 2.6.0, Trew@ acepta la configuración de los perfiles de conexión de dos formas diferentes, como se describe en el apartado siguiente. Se recomienda tener esto en cuenta antes de configurar los perfiles.*

3.2.1.3Configuración de perfiles de conexión

Un perfil de conexión para las J-TrAPIs no es más que una configuración de conexión a una base de datos.

En la actualidad las J-TrAPIs permiten dos posibilidades de configuración para un perfil de conexión:

- Mediante fichero de properties
- Mediante JNDI

A nivel de aplicación el uso de un perfil u otro va a ser totalmente transparente y no se va a conocer exactamente donde se establece la conexión. Lo único que van a necesitar las aplicaciones será el **nombre del perfil** que deberá ser aportado por el administrador del servidor de aplicaciones donde se desplieguen las J-TrAPIs.

3.2.1.3.1 Configuración mediante JNDI (Recomendada)

NOTA: *La configuración relacionada con este apartado, debe realizarse en el fichero de configuración standalone del servidor de aplicaciones y se describirá más adelante. En caso de que desee realizar las configuraciones de esta manera, lo cual es lo recomendado por el equipo de Trew@, ignore el siguiente apartado (Configuración mediante fichero de properties) y continúe con los pasos descritos en los siguientes apartados.*

3.2.1.3.2 Configuración mediante fichero de properties

En el caso en que se decida realizar la configuración de esta manera, es muy importante tener en cuenta que las configuraciones de los properties de conexión deben realizarse en **todos** los módulos .war que conforman el .ear distribuido:

- carpetaCiudadanaJA-2.6.11.0.war
- hazelcast-2.6.11.0.war
- TrewaAdm-2.6.11.0.war
- wopi-host-2.6.11.0.war

NOTA:*La configuración de los perfiles debe ser igual en todos los módulos y no deben existir perfiles solo configurados para un módulo concreto.*

NOTA:*La configuración, en todos los módulos, de estos perfiles se debe realizar bajo la ruta: " *.war\WEB-INF\classes\trewa\conf\perfiles\"*

En este caso el nombre del perfil va a coincidir con el nombre del archivo de properties donde se realice la configuración. Así por ejemplo si el administrador crea un archivo *trOracle.properties*, el nombre del perfil será *trOracle.*

Los perfiles deben ubicarse en el paquete *trewa.conf.perfiles* para que las J-TrAPIs los puedan localizar. En esa carpeta debe existir configurado al menos un perfil **default** (default.properties), en el caso de que no exista el equivalente como JNDI, que es el que intentarán usar las J-TrAPIs cuando no se le indique perfil alguno.

Cada fichero properties, que define la conexión con una base de datos, debe contener los siguientes campos de configuración:

Campo	Obligatorio	Descripción
tipo	Sí	Tipo de base de datos a la que se va a acceder. Por defecto ORACLE
nombreMaquina	Sí	Máquina
puerto	Sí	Puerto de acceso
nombreBD	Sí	Nombre de la base de datos
nombreUsuario	Si[3]	Nombre de usuario
claveUsuario	Sí[4]	Clave de acceso

3 No será obligatorio, sí y solo sí, se configura una url de conexión JDBC completa.

4 No será obligatorio, sí y solo sí, se configura una url de conexión JDBC completa.

usaPool	No	Indica si se quiere hacer uso del pool interno que ofrecen las J-TrAPIs. Posibles valores: **true** o **false**

Tabla 11: Campos de configuración de fichero properties para perfiles de conexión

Se proporciona un ejemplo del contenido de un fichero properties:

```
#Archivo de propiedades
tipo=ORACLE
#Datos para realizar la conexion
nombreMaquina=desa.ja.es
puerto=1521
nombreBD=bdOracle
nombreUsuario=usuario
claveUsuario=clave

#Indica si se quiere crear la conexión usando pool de conexiones
usaPool=false

#Máximo de conexiones abiertas
maxConexionesActivas=4

#Máximo de tiempo de espera por conexión
maxEsperaConexion=5000

#Máximo de conexiones ociosas
maxConexionesOciosas=2
```

Figura 8: Ejemplo de configuración de perfil de conexión

Los valores de color negro de la figura anterior deben ser modificados para adaptarlos a la configuración necesaria para cada conexión con un servidor de base de datos.

Se permite indicar en el properties si se hace uso del pool de conexiones que ofrecen las J-TrAPIs, para ello se definen en el fichero properties los atributos usaPool, maxConexionesActivas, maxEsperaConexion y maxConexionesOciosas.

También se permite indicar los valores de nombreMaquina, puerto y nombreBD en uno solo, formando así la url de conexión JDBC. Para ello basta con indicar la url de conexión JDBC en el atributo nombreMaquina. De este modo se permiten realizar conexiones remotas tan solo indicando la url de conexión JDBC. Por ejemplo:

```
#Archivo de propiedades
tipo=ORACLE
#Datos para realizar la conexión
      nombreMaquina=jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS_LIST=(ADDRESS= (PROTOCOL=TCP)
      (Host=nombreMaquina)(Port=puerto))(ADDRESS=(PROTOCOL = TCP)(Host=nombreMaquina)(Port = puerto)))
      (CONNECT_DATA= (SID=nombreBD))(SOURCE_ROUTE=YES))
puerto=
nombreBD=
nombreUsuario=
claveUsuario=
```

Figura 9: Ejemplo de configuración de perfil de conexión mediante url de conexión JDBC

Los valores de color negro de la figura anterior deben ser modificados para adaptarlos a la configuración necesaria para cada conexión con un servidor de base de datos.

3.2.2Trazabilidad Log4Java

Para habilitar la trazabilidad de log4java en Trew@, bastará con modificar un único archivo dentro del fichero .war, el archivo *log4j2.xml*, en la ruta TrewaAdm-2.6.11.0.war\WEB-INF\classes\

A través de este archivo se podrá configurar tanto el nivel de trazabilidad que se desea habilitar sobre Trew@, como la ruta donde se despliega el fichero log.

Para establecer la ruta del fichero log se debe modificar la propiedad con nombre *directorio* indicando la ruta en la que se desea crear el fichero. El valor por defecto configura su almacenamiento en la carpeta de log del servidor de aplicaciones(WildFly15.0.1.Final/standalone/log).

Para cambiar la configuración de los niveles de trazabilidad sobre Trew@, se modificará la etiqueta *level* estableciendo algún nivel de los posibles que se describen a continuación:

DEBUG: Se utiliza para escribir mensajes de depuración, este log no debe estar activado cuando la aplicación se encuentre en producción.

INFO: Se utiliza para mensajes que proporcionan información sobre el progreso y la información de estado elegida. Es similar al modo "verbose" en otras aplicaciones.

WARN: Se utiliza para mensajes de alerta sobre eventos que se desea mantener constancia, pero que no afectan el correcto funcionamiento del programa.

ERROR: Se utiliza en mensajes de error de la aplicación que se desea guardar, estos eventos afectan al programa pero lo dejan seguir funcionando.

FATAL: Se utiliza para mensajes críticos del sistema, generalmente luego de guardar el mensaje el programa abortará.

Adicionalmente a estos niveles de log, existen dos niveles extra que solo se utilizan en el archivo de configuración, estos son:

ALL: este es el nivel más bajo posible, habilita todos los logs.

OFF: este es el nivel más alto posible, deshabilita todos los logs.

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration>
    <Properties>
        <Property name="directorio">${sys:jboss.server.base.dir}</Property>
    </Properties>
    <Appenders>
        <RollingFile name="TREWA" fileName="${directorio}/log/TrewaAdm.log"
            filePattern="${directorio}/log/$${date:yyyy-MM}/TrewaAdm-%d{yyyy-MM-dd}-%i.log">
            <PatternLayout>
```

```
                    <Pattern>%d %p %c{1.} [%t] %m%n</Pattern>
                </PatternLayout>
                <Policies>
                    <TimeBasedTriggeringPolicy interval="1" modulate="true"/>
                    <SizeBasedTriggeringPolicy size="50 MB"/>
                    </Policies>
            </RollingFile>
            <Console name="CONSOLE" target="SYSTEM_OUT">
                <PatternLayout pattern="[%-5level] %d{yyyy-MM-dd HH:mm:ss.SSS}
          [%t] %c{1} - %msg%n" />
            </Console>
            <Async name="ASYNC">
                <AppenderRef ref="TREWA" level="info" />
                <AppenderRef ref="CONSOLE" level="info" />
            </Async>
        </Appenders>
        <Loggers>
            <Root level="debug">
                <AppenderRef ref="ASYNC" />
            </Root>
        </Loggers>
</Configuration>
```

Figura 10: Configuración de log4j2

Los valores de color negro de la figura anterior deben ser modificados para adaptarlos a la configuración necesaria en caso de querer modificar la ruta de almacenamiento o el nivel de log a mostrar.

3.3 Servidor de aplicaciones (WildFly 15.0.1.Final)

Hasta ahora, se han descrito todas las configuraciones relacionadas con los aspectos previos al despliegue de la aplicación y, por tanto, previas a la configuración del servidor de aplicaciones. En este apartado se describe en detalle cada uno de los pasos a seguir para ello.

En primer lugar, será necesario contar con una instalación del contenedor de aplicaciones WildFly 15.0.1.Final, con el fin de desplegar en él la aplicación.

Para su instalación, existirá un usuario que será el encargado de administrar y ejecutar WildFly, y que deberá ser el propietario de la aplicación, ya que se le asignaran permisos de lectura, escritura y ejecución sobre las diversas carpetas y aplicaciones afectadas.

Por tanto, a la hora de instalar WildFly en el entorno, se creará el usuario propietario, por ejemplo “trewa”, cuyo directorio asignado es la carpeta de instalación de WildFly, por ejemplo */usr/share/WildFly-15.0.1.Final*, considerando una distribución Linux. Por motivos de seguridad es recomendable que este usuario esté autorizado para ejecutar todas las operaciones sobre la aplicación.

Será necesario obtener el paquete oficial del servidor de la página oficial de WildFly. La descarga provee un binario válido tanto para Windows como para Linux. Para descargar en formato .zip pulse aquí o, para descargar en formato .tgz, aquí.

Una vez descargado y descomprimido en la ruta deseada, se deben realizar una serie de configuraciones que se describen a continuación.

3.3.1 Configuración del servidor de aplicaciones

3.3.1.1 Creación de usuarios de WildFly

Con el fin de poder llevar a cabo la administración del servidor, así como el uso de ciertas funcionalidades de Trew@, es necesario crear dos usuarios de WildFly.

3.3.1.1.1 Creación de usuario administrador de WildFly

Con el fin de poder llevar a cabo la administración del servidor, es necesario que se establezca un usuario propio del servidor WildFly que tenga permisos para realizar estas acciones.

Para crear un usuario de WildFly, es necesario ejecutar la aplicación de creación de usuarios *add-user*. Dependiendo del sistema operativo, se deberá ejecutar la aplicación *add-user.bat (Windows)* o *add-user.sh (Linux)*, que se encuentran en el directorio *WildFly-15.0.1-Final/bin* del servidor de aplicaciones.

Una vez ejecutada la aplicación, se deben realizar las siguientes configuraciones mediante línea de comandos, cambiando tanto el usuario como la contraseña por los valores deseados:

```
// Cuando aparezca el siguiente texto indicar “a”

What type of user do you wish to add?
 a) Management User (mgmt-users.properties)
 b) Application User (application-users.properties)
(a): a

// Aparecerá el siguiente mensaje para indicar el Real que se quiere utilizer. Se pulsará enter, para dejar la opción por defecto (ManagementRealm)

Enter the details of the new user to add.
Realm (ManagementRealm) :

// A continuación se pide que se indique el nombre del usuario. En este caso se indicará el que se estime oportuno.

Username : trewa

// El siguiente paso sera indicar la clave, teniendo en cuenta que no sea la misma que el nombre de usuario. A continuación se repetirá la clave por seguridad.

Password: trewa260

// Wildfly te ofrece recomendaciones de seguridad para la pass sin necesidad que estas sean obligatorias.

WFLYDM0102: Password should have at least 1 non-alphanumeric symbol.

Are you sure you want to use the password entered yes/no? yes
```

```
Re-enter Password: trewa260
// Se pulsará enter, para dejar la opción por defecto.
What groups do you want this user to belong to? (Please enter a comma separated list, or leave blank for none)[  ]:
// Avisará del resultado de la operación y será necesario indicar “yes” para que guarde los cambios.
About to add user 'trewa' for realm 'ManagementRealm'
Is this correct yes/no? yes
// Se indica que se han actualizado correctamente los ficheros internos de Wildfly.
Added user 'trewa' to file 'C:\dev\wildfly-15.0.1.Final\standalone\configuration\mgmt-users.properties'
Added user 'trewa' to file 'C:\dev\wildfly-15.0.1.Final\domain\configuration\mgmt-users.properties'
Added user 'trewa' with groups  to file 'C:\dev\wildfly-15.0.1.Final\standalone\configuration\mgmt-groups.properties'
Added user 'trewa' with groups  to file 'C:\dev\wildfly-15.0.1.Final\domain\configuration\mgmt-groups.properties'
// Por último se indica si se quiere usar el usuario creado para conexiones remotas. No es necesario hacer uso del usuario para conexiones remotas.
Is this new user going to be used for one AS process to connect to another AS process?
e.g. for a slave host controller connecting to the master or for a Remoting connection for server to server EJB calls.
yes/no? no
Presione una tecla para continuar . . .
```

Figura 11: Configuración usuario de administración de WildFly

3.3.1.1.2 Creación de usuario de WildFly para la gestión de peticiones EJB

Con el fin de poder llevar a cabo invocaciones al módulo EJB contenido en el EAR de TrewaAdm, es necesario que se establezca un usuario propio del servidor WildFly que tenga permisos para realizar estas acciones.

Para crear este usuario en WildFly, es necesario ejecutar la aplicación de creación de usuarios del apartado anterior, introduciendo el tipo de usuario *b.*

Una vez ejecutada la aplicación, se deben realizar las siguientes configuraciones mediante línea de comandos, cambiando tanto el usuario como la contraseña por los valores deseados:

```
// Cuando aparezca el siguiente texto indicar “b”

What type of user do you wish to add?
 a) Management User (mgmt-users.properties)
 b) Application User (application-users.properties)
(a): b

// Aparecerá el siguiente mensaje para indicar el Real que se quiere utilizer. Se pulsará enter, para dejar la opción por defecto (ApplicationRealm)

Enter the details of the new user to add.

Using realm 'ApplicationRealm' as discovered from the existing property files.

// A continuación se pide que se indique el nombre del usuario. En este caso se indicará el que se estime oportuno.

Username : ejb

// El siguiente paso sera indicar la clave, teniendo en cuenta que no sea la misma que el nombre de usuario. A continuación se repetirá la clave por seguridad.

Password: ejbTrewa

// Wildfly te ofrece recomendaciones de seguridad para la pass sin necesidad que estas sean obligatorias.

WFLYDM0101: Password should have at least 1 digit.

Are you sure you want to use the password entered yes/no? yes

Re-enter Password: ejbTrewa

// Se pulsará enter, para dejar la opción por defecto.

What groups do you want this user to belong to? (Please enter a comma separated list, or leave blank for none)[  ]:
// Avisará del resultado de la operación y será necesario indicar “yes” para que guarde los cambios.

About to add user 'ejb' for realm 'ApplicationRealm'

Is this correct yes/no? Yes

// Se indica que se han actualizado correctamente los ficheros internos de Wildfly.

Added user 'ejb' to file '%WILDFLY_HOME%/wildfly-15.0.1.Final/standalone/configuration/application-users.properties'

Added user 'ejb' to file '%WILDFLY_HOME%/wildfly-15.0.1.Final/domain/configuration/application-users.properties'

Added user 'ejb' with groups  to file '%WILDFLY_HOME%\wildfly-15.0.1.Final\standalone\configuration\application-roles.properties'

Added user 'ejb' with groups  to file '%WILDFLY_HOME%\wildfly-15.0.1.Final\domain\configuration\application-roles.properties'

// En el último paso WildFly nos permite indicar si el usuario creado puede establecer conexiones remotas entre servidores.
```

Se debe indicar que sí.

Is this new user going to be used for one AS process to connect to another AS process?

e.g. for a slave host controller connecting to the master or for a Remoting connection for server to server EJB calls.

yes/no? Yes

To represent the user add the following to the server-identities definition <secret value="[redacted]" />

Presione una tecla para continuar . . .

Tabla 12: Creación de usuario de WildFly para EJB

3.3.1.2Validar configuración inicial

Para validar la configuración inicial del servidor de aplicaciones, se arrancará para comprobar que todo funciona correctamente. Para ello se realizará un **primer y único arranque con una configuración especial** que permite, además, la accesibilidad al servidor y a su administración, de forma remota. El comando a utilizar en la consola es el siguiente, y deberá ejecutarse desde el directorio *WildFly-15.0.1.Final/bin* del servidor de aplicaciones:

Figura 12: Comando de arranque inicial de WildFly

Una vez arrancado sin problemas, será necesario acceder a la consola de administración, indicando la URL http://IP_SERVIDOR:9990/, y se deberá acceder a la pantalla de acceso de la administración del servidor de aplicaciones.

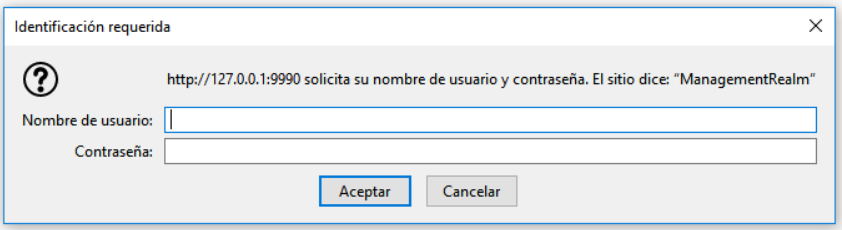

Figura 13: Pantalla de acceso a la administración de WildFly

Para acceder, se utilizarán las credenciales del usuario creado en el apartado 3.3.1.1.1. Tras esto se mostrará la pantalla de inicio de la administración de WildFly.

Tras acceder a la administración sin problemas, se puede confirmar la validez de la configuración básica del servidor de aplicaciones. En el siguiente apartado se detallarán las configuraciones a realizar desde la administración de WildFly.

3.3.1.3 Modificación de la IP de acceso público y de acceso para la administración

Tal y como se ha indicado en el comando para el primer arranque de WildFly, es necesario indicar la IP 0.0.0.0 para que se pueda acceder tanto a la parte pública como a la privada del servidor de aplicaciones desde una IP distinta a la localhost. Para que este cambio sea permanente y no sea necesario indicar el parámetro correspondiente en cada arranque, se llevará a cabo la configuración necesaria a través de la consola de administración.

Para ello, en el menú superior de la consola de administración, se debe acceder a *Configuration*, y en el desplegable que aparecerá en la parte izquierda, marcar la opción *Interfaces*.

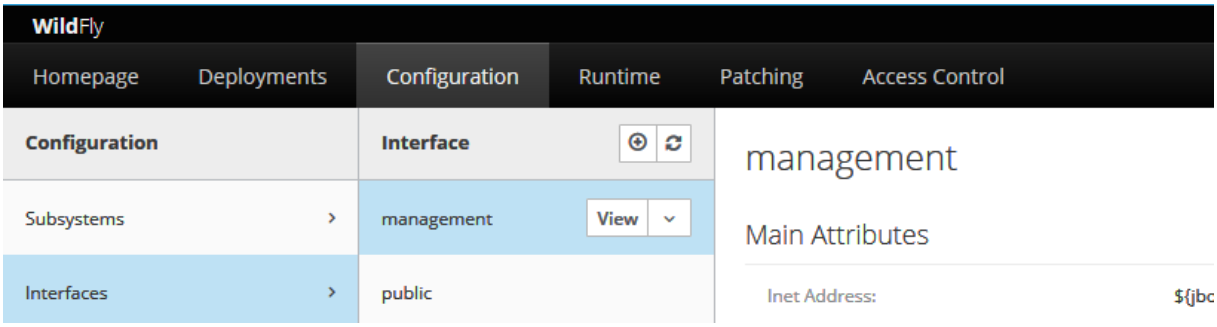

Figura 14: Sección de interfaces de la administración de WildFly

Una vez aparezcan las interfaces, se seleccionara la interfaz *management* y se hará click sobre el botón *View*. De esta forma, se abrirá el detalle de la interfaz.

Desde el detalle de la interfaz se seleccionará la opción *Edit* para modificar las propiedades del la interfaz.

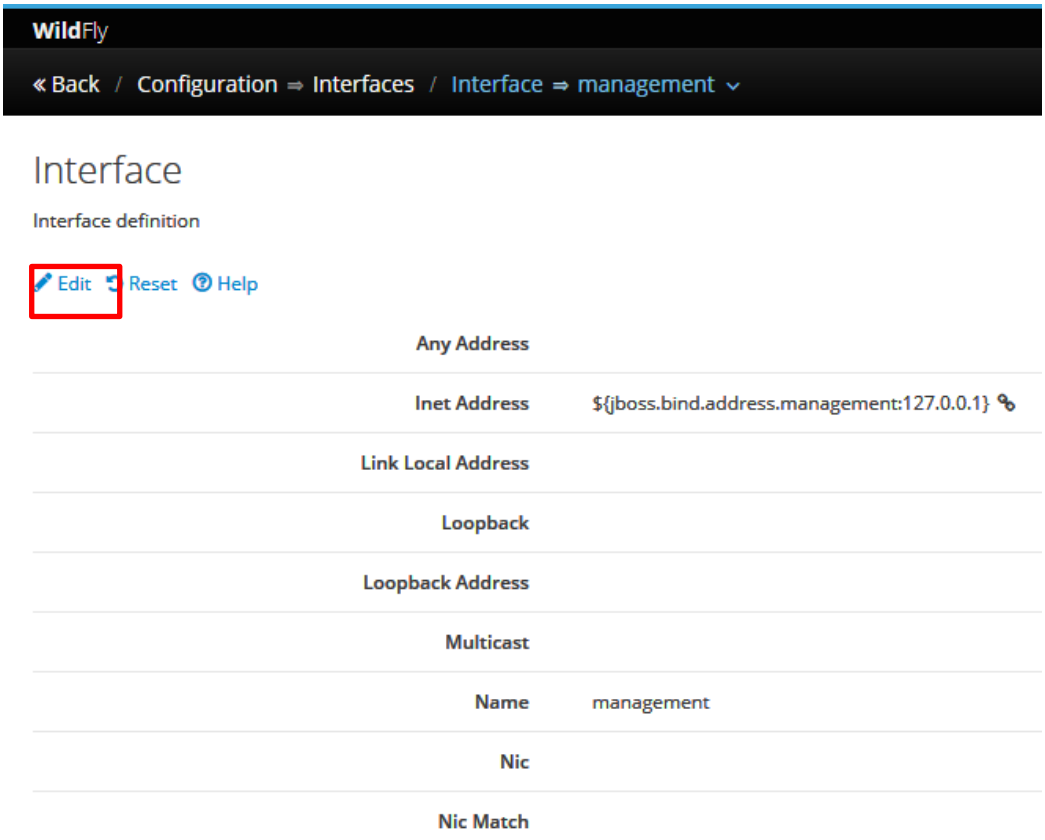

Figura 15: Detalle de la interfaz de administración de WildFly

Se editará el campo **Inet Address** sustituyendo la IP de localhost (127.0.0.1) por la IP del servidor donde está alojado WildFly o, de forma genérica, por la IP 0.0.0.0.

Una vez realizado el cambio, se pulsará el botón *Save*.

Estos cambios también serán necesarios realizarlos sobre la configuración de la interfaz pública del servidor.

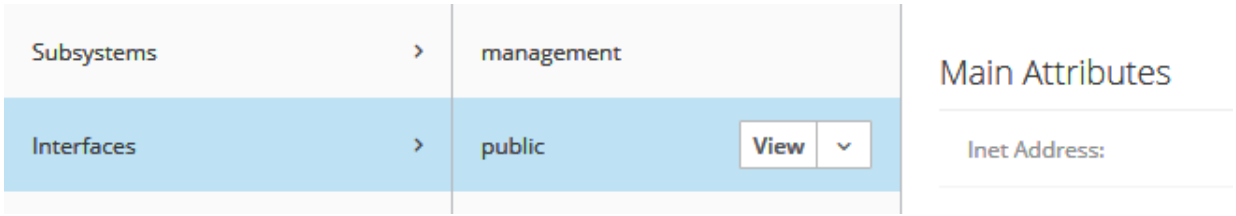

Figura 16: Sección de interfaz pública de WildFly

Se hará click en el botón *View* . Después de ésto se mostrará el detalle de la interfaz pública, donde se pulsará sobre *Edit* para configurar el campo **Inet Address** sustituyendo la IP de localhost (127.0.0.1) por la IP del servidor donde está alojado WildFly o, de forma genérica, por la IP 0.0.0.0.

Una vez realizado el cambio, se pulsará el botón *Save.*

NOTA: *En los siguientes arranques se ejecutará directamente* ***standalone.sh*** *o* ***standalone.bat****, para Linux y Windows, respectivamente. Para parar el servidor, bastará con ejecutar* ***jboss-cli.sh****, o* ***jboss-cli.bat****, para Linux y Windows, respectivamente.*

3.3.1.4Configuración del modo de arranque

NOTA: *Antes de iniciar las acciones indicadas en este apartado, asegúrese de que el servidor de aplicaciones esta parado. Para ello, puede ejecutar* ***jboss-cli.sh*** *o* ***jboss-cli.bat*** *para Linux y Windows, respectivamente.*

Para el correcto funcionamiento de la Herramienta de Administración de Trew@, como se mencionó anteriormente, es obligatorio arrancar el modo standalone-full ó standalone-ha del servidor de aplicaciones. Por esto, es necesario editar el fichero de configuración presente en el directorio *WildFly-15.0.1.Final/bin*. Dependiendo de si la instalación se hace en Linux o Windows, los ficheros son standalone.conf y standalone.conf.bat, respectivamente. Para ello, se deben realizar los siguientes cambios:

Figura 17: Configuración de modo standalone-full

3.3.1.5Inclusión del módulo para la conexión con Oracle

NOTA: *Antes de iniciar las acciones indicadas en este apartado, asegúrese de que el servidor de aplicaciones esta parado. Para ello, puede ejecutar* ***jboss-cli.bat*** *o* ***jboss-cli.sh****, para Windows y Linux, respectivamente.*

Para el correcto funcionamiento de la implantación de Trew@, es necesario incluir en el servidor de aplicaciones un módulo para la conexión con el servidor de base de datos. Este módulo se trata de ojdbc7.jar[5], que puede ser descargado en la página oficial de Oracle.

Para ello es necesario seguir los siguientes pasos:

1. Crear el directorio *oracle/jdbc/main* en el repositorio de módulos del servidor WildFly, bajo la ruta WidlFly-15.0.1.Final/*modules/system/layers/base*.
2. Incluir la librería ojdbc7.jar en el directorio creado.
3. Crear un nuevo fichero de nombre module.xml en el mismo directorio, con el siguiente contenido:

5 La información sobre compatibilidad se puede encontrar en https://www.oracle.com/database/technologies/faq-jdbc.html

```
<?xml version="1.0" encoding="UTF-8"?>
<module xmlns="urn:jboss:module:1.1" name="oracle.jdbc">
      <resources>
   <resource-root path="ojdbc7.jar"/>
      </resources>
      <dependencies>
   <module name="javax.api" />
   <module name="javax.transaction.api" />
      </dependencies>
</module>
```

Figura 18: Configuración del módulo de conexión con Oracle, jdbc

4. Referenciar el módulo creado en el fichero de configuración de WildFly. (Los cambios necesarios para ello se tratarán en el apartado Cambios en el fichero de configuración standalone).

3.3.1.6 Cambios en el fichero de configuración standalone

NOTA: *En este apartado, todos los cambios indicados son para el fichero standalone-full.xml. En el caso de que se quiera configurar en alta disponibilidad, todas estas configuraciones se llevarán a cabo en el fichero standalone-ha.xml.*

En este apartado se han concentrado, por facilidad de lectura y seguimiento, todas las configuraciones necesarias a realizar sobre el fichero de configuración *standalone-full.xml*. Por tanto, se realizarán cambios referentes a diferentes aspectos del funcionamiento de la implantación de Trew@.

Este fichero se puede encontrar bajo la ruta *WildFly-15.0.1.Final/standalone/configuration/*.

3.3.1.6.1 Configuración del módulo de oracle

Será necesario incluir la referencia al módulo de oracle. Para ello se debe modificar el subsistema *urn:jboss:domain:datasources:5.0* incluyendo los cambios resaltados en negro.

3.3.1.6.2Aumentar tiempo de arranque

Será necesario aumentar el tiempo en el que el servidor despliega las aplicaciones, para asegurar de que no se producen errores por tiempo insuficiente. Para ello se debe modificar el subsistema *deployment-scanner:2.0*:

```
<subsystem xmlns="urn:jboss:domain:deployment-scanner:2.0">
    <deployment-scanner path="deployments" relative-to="jboss.server.base.dir" scan-
     Interval="5000" deployment-timeout="3000"
     runtime-failure-causes-rollback="${jboss.deployment.scanner.rollback.on.failure:false}"/>
</subsystem>
```

Figura 19: Cambios para aumentar el tiempo de arranque del servidor, resaltados en negro

3.3.1.6.3Configuración del identificador del servidor

Será necesario establecer un identificador único al nodo del servidor de aplicaciones. Para ello, se debe modificar los siguientes subsistemas, incluyendo los cambios resaltados en negro.

```
<subsystem xmlns="urn:jboss:domain:transactions:5.0">
    <core-environment node-identifier=”localhost”>
        <process-id>
            <uuid/>
        </process-id>
    </core-environment>
    <recovery-environment socket-binding="txn-recovery-environment" status-socket-binding="txn-
     status-manager"/>
</subsystem>

<subsystem xmlns="urn:jboss:domain:undertow:7.0" instance-id="${jboss.host.name}" ...>
  ...
</subsystem>
```

Figura 20: Cambios para la identificación del servidor, resaltados en negro

3.3.1.6.4Configuración de la autenticación de usuarios

Será necesario desactivar la autenticación de las conexiones entrantes en el servidor para que no necesiten ser autenticadas contra un usuario de WildFly existente, para ello se introducirán los cambios resaltados en negro en el contenido del subsistema *urn:jboss:domain:undertow:7.0*:

```
<subsystem xmlns="urn:jboss:domain:undertow:7.0"...>
  . . .

  <servlet-container name="default" proactive-authentication="false">
    <jsp-config/>
    <websockets/>
  </servlet-container>

</subsystem>
```

Figura 21: Cambios para la configuración de la autenticación, resaltados en negro

3.3.1.6.5Configuración de perfiles de conexión usando JNDI

Como se ha mencionado en apartados anteriores, a partir de esta versión de Trew@ se permite configurar los perfiles de conexión a base de datos mediante JNDI. Para ello, será necesario incluir fragmentos como el siguiente, modificando las partes resaltadas en negro por la configuración deseada:

Figura 22: Configuración de conexión JNDI

3.3.1.6.6Configuración de propiedades de sistema

Para el correcto funcionamiento de la Herramienta de Administración de Trew@ y de algunos componentes de la misma se deben configurar una serie de propiedades de sistema. Para ello, se deberá introducir el siguiente fragmento dentro de la etiqueta *<system-properties>* ,sustituyendo las partes resaltadas en negro por las configuraciones propias:

Figura 23: Configuración de propiedades de sistema

A continuación se explican los valores de las propiedades definidas en la figura anterior entre corchetes:

- **{NOMBRE_GRUPO}:** Se debe introducir un nombre único no compartido entre entornos. E
- Esto asegura que Hazelcast no se comunique automáticamente con nodos de otros entornos.
- **{NODO_SERVIDOR}** y **{PUERTO}:** Debido a que Hazelcast crea un nodo en cada uno de los desplegados de Trew@, debe configurarse uno de ellos como “nodo servidor”. Para esto se debe configurar la IP de uno de los nodos de Trew@ para que haga esta función de “nodo servidor”. Por ejemplo, considerando la existencia de dos nodos, 192.0.0.84 y 192.0.0.85 con puerto libre para Hazelcast 8580. Tomando el primero como “nodo servidor”, se deberá configurar en el fichero standalone, de ambos nodos, {NODO_SERVIDOR} = 192.0.0.84 y {PUERTO} = 8580.
- **{PASSWORD}:** Se debe introducir una contraseña única no compartida por entornos. Esto asegura que Hazelcast no se comunique automáticamente con nodos de otros entornos.
- **{CONTEXT}:** Se debe introducir el contexto en el que se despliega el módulo Hazelcast. Por defecto “hazelcast”.

3.4 Instalación del empaquetado

Desde la versión 2.6.2 de la Herramienta de Administración de Trew@ se distribuye para su despliegue con una arquitectura **EAR** (TrewaAdm.ear). El contenido de **TrewaAdm.ear** está formado por cinco módulos que se detallan a continuación:

- carpetaCiudadanaJA-war: Módulo que contiene la capa lógica del servicio web ofrecido para las consultas de expedientes desde Carpeta Ciudadana.
- hazlecast-war: Módulo que contiene la capa lógica de la caché distribuida de Trew@.
- signerDelegateEJB.jar: Módulo EJB para realizar firmas utilizando el componente de firma delegada de @firma.
- TrewaAdm-war: Módulo que contiene toda la capa lógica y de presentación de la Herramienta de Administración de Trew@.
- wopi-host-war: Módulo que contiene toda la capa lógica para la integración del protocolo wopi, utilizado por LibreOffice Online, con Trew@.

3.5 Configuración de la Herramienta de Administración de Trew@

En este apartado se describen las configuraciones que se deben realizar una vez se arranque el servidor de aplicaciones con el empaquetado en su interior.

3.5.1 Configuración de la firma delegada por EJB

Debido a la actualización de WildFly, los protocolos de comunicación EJB han variado con respecto a versiones anteriores de Trew@.

Para poder realizar conexiones mediante EJB, con la funcionalidad de firma delegada de la Herramienta de Administración de Trew@, se contemplan dos casos:

- Herramientas que hacen uso de la API de Trew@. En este caso, será necesario configurar algunos datos para el componente @firma, adicionales a los que se indican en el documento *TRW26110_OTR_Soporte_Trewa_a_expedientes_ENI_v01r00*, distribuido en el CD de esta versión:

Atributo	Valor
EJB.USER	Usuario de WildFly configurado para EJB. Por ejemplo: ejb.
EJB.PASSWORD	Contraseña del usuario de WildFly configurado para EJB. Por ejemplo: ejbTrewa.
EJB.URL.PROVIDER	IP/URL del despliegue de TrewaAdm. Por ejemplo: 192.127.12.1

EJB.URL.PROVIDER.PORT	Puerto http donde está desplegado TrewaAdm. Por ejemplo: 8180.

Tabla 13: Configuración para las comunicaciones EJB de firma delegada

- Herramientas que no hacen uso de la API de Trew@. En este caso, será necesario asegurar la configuración del contexto inicial para las comunicaciones mediante EJB. Las propiedades de configuración deben ser las siguientes:

Atributo	**Valor**
java.naming.factory.url.pkgs	org.jboss.ejb.client.naming
java.naming.factory.initial	org.jboss.naming.remote.client.InitialContextFactory
java.naming.factory.url.pkgs	org.jboss.ejb.client.naming
java.naming.provider.url	http-remoting://**192.127.12.1:8180**
java.naming.security.principal	ejb
java.naming.security.credentials	ejbTrewa
jboss.naming.client.ejb.context	true
remote.connectionprovider.create.options.org.xnio.Options.SSL_ENABLED	false
remote.connection.default.connect.options.org.xnio.Options.SASL_POLICY_NOANONYMOUS	false
jboss.naming.client.connect.options.org.xnio.Options.SASL_POLICY_NOPLAINTEXT	false
jboss.naming.client.ejb.context	false
remote.connection.default.host	**192.127.12.1**
remote.connection.default.port	**8180**
remote.connections	true

Tabla 14: Propiedades necesarias para la configuración del contexto EJB en aplicaciones cliente que no utilicen la API de Trew@

Los valores resaltados en negrita deben ser adaptados para utilizar los necesarios.

El cómo se lleve a cabo la configuración de estas propiedades es decisión del administrador de la herramienta que consumirá. No obstante, se recomienda la configuración mediante fichero properties o a nivel de código.

3.5.2 Configuración de EnidocWS y EniexpWS

Para la correcta integración de Trew@ con la Herramienta Centralizada de Verificación , en adelante HCV, se debe crear un sistema repositorio en HCV, donde se establezca el código, el usuario, la contraseña y la dirección IP del servidor donde se encuentra desplegado el EnidocWS que realiza la integración.

El sistema repositorio creado para los documentos puede usarse también para los expedientes, ya que para nuestro caso el EniexpWS se encuentra desplegado en el mismo servidor que el EnidocWS. De esta forma cuando se realice el alta del sistema repositorio en HCV se puede indicar que el repositorio es para ambos.

Si se requiere un nuevo sistema repositorio en HCV con diferente código, usuario y contraseña para la integración con el EniexpWS, se necesita la configuración de tres nuevos parámetros de configuración.

La dirección IP a establecer en HCV está compuesta por la dirección donde se encuentra desplegada la Herramienta de Administración de Trew@ añadiéndole el sufijo '/rest/eni/', sufijo configurado en el fichero web.xml de la aplicación web TrewaAdm.war. Ejemplo de Dirección IP: http://host:port/TrewaAdm/rest/eni/

En la versión 2.6.2 de TrewaAdm es obligatoria la configuración de EnidocWS tanto para el uso de la Herramienta Centralizada de Verificación como para el uso de Port@firmas.

3.5.2.1 Configuración de la Herramienta Centralizada de Verificación

3.5.2.1.1 Configuración de constantes

Para la integración de Trew@ con HCV mediante el EnidocWS se debe crear tres nuevas constantes de sistema en la Herramienta de Administración de Trew@. Las nuevas constantes para la configuración de credenciales de HCV son:

Constante	Denominación
USUARIO_HCV	Usuario definido para el repositorio en HCV
PASSWORD_HCV	Contraseña definida para el repositorio en HCV
CODIGO_HCV	Código asociado al repositorio de HCV

Tabla 15: Constantes para la consulta de documentos ENI

Para la integración de Trew@ con HCV mediante el EniexpWS se puede utilizar el mismo repositorio creado para el EnidocWS no siendo necesario la configuración de nuevas constantes. En contrario, si se utiliza un sistema/repositorio distinto para el EniexpWS es necesario crear las siguientes constantes para la configuración de credenciales de HCV para los expedientes:

Constante	Denominación
USUARIO_HCV_EXP	Usuario definido para el repositorio de expedientes en HCV
PASSWORD_HCV_EXP	Contraseña definida para el repositorio de expedientes en HCV
CODIGO_HCV_EXP	Código asociado al repositorio de expedientes de HCV

Tabla 16: Constantes para la consulta de expedientes ENI

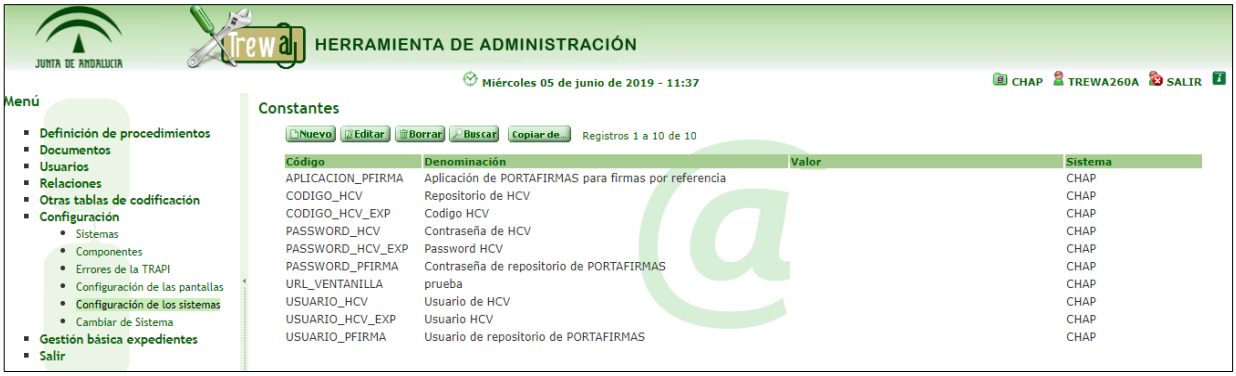

Figura 24: Captura de la pantalla de constantes

3.5.2.1.2Configuración del componente HCV

Para el correcto funcionamiento de Trew@, es necesaria y obligatoria la creación de un componente para HCV para el tipo de componente TREW@. En la creación se debe establecer de forma obligatoria la IP, el organismo y los siguientes datos del componente:

Parámetro	Denominación
PROTOCOLO	Protocolo de comunicación con HCV
PUERTO	Puerto de la aplicación HCV
RUTA_ENIDOC_CLIENT	Ruta utilizada para la obtención de documento existentes en repositorios externos. Por defecto su valor es: **/hcv/rest/eni/**
RUTA_VERIFICACION	Ruta utilizada para formar la URL de verificación de un documento que se incorpora en el pdf resumen de un expediente ENI. Por defecto su valor es: **/hcv/**

Tabla 17: Configuración del componente de HCV

La configuración del componente HCV se establece para el uso de la integración que realiza la obtención de documentos existentes en repositorios externos y para formar la URL de

verificación que se incorpora en el pdf resumen que se incorpora en la visualización del índice de un expediente ENI a la hora de su exportación.

3.5.2.2Configuración de Port@firmas v3

Trew@ hace uso del envío de firmas por referencias. Con el envío por referencia, en lugar de remitir el contenido binario de los documentos y anexos, se envía su resumen HASH y su código de verificación.

De esta forma evitamos que se realice en Port@firmas la custodia de los documentos y así, esta función recaerá íntegramente en Trew@. Por ello se debe configurar de forma correcta el EnidocWS que será consumido por Port@firmas para rescatar los documentos cuando sea necesario.

Para realizar el envío de forma correcta de las peticiones de firma se debe dar de alta una nueva aplicación en Port@firmas donde se indique que la custodia de los documentos se realice mediante repositorio, indicando los siguientes parámetros de configuración:

Parámetro	Denominación
URL repositorio	Repositorio configurado en la Herramienta Centralizada de Verificación (HCV)
Usuario repositorio	Usuario dado de alta en HCV para el repositorio
Clave repositorio	Contraseña dada de alta en HCV para el repositorio

Tabla 18: Configuración de aplicación en Port@firmas

Una vez realizada el alta de aplicación en Port@firmas se debe realizar una serie de acciones en la Herramienta de Administración de Trew@.

1. Alta de tres nuevas constantes a nivel de sistema para la configuración de credenciales:

Código	Denominación
USUARIO_PFIRMA	Usuario de repositorio de Port@firmas
PASSWORD_PFIRMA	Contraseña de repositorio de Port@firmas
APLICACION_PFIRMA	Aplicación de Port@firmas para firmas por referencia

Tabla 19: Constantes para el funcionamiento de Port@firma

2. En caso de establecer una contraseña en la aplicación dada de alta en Port@firmas se debe establecer en el componente de PORT@FIRMAS los siguientes parámetros:

Parámetro	Valor
USUARIO	Aplicación de Port@firmas para firmas por referencia.

PASSWORD	Contraseña de la aplicación de Port@firmas para firmas por referencia

Tabla 20: Configuración de usuario y contraseña para Port@firmas

Figura 25: Captura de la pantalla de componentes

3. Para finalizar la configuración se debe dar de alta tres datos del componente PORT@FIRMAS

Parámetro	**Valor**
PROTOCOLO	Protocolo de comunicación con los servicios web de la fachada web v2.
PUERTO	Puerto de la aplicación Port@firmas.
RUTA	Ruta donde están desplegados los servicios v2

Tabla 21: Datos del componente de @firma

3.5.3Configuración del componente DIR3

En esta versión de Trew@, para mejorar el funcionamiento de la integración con Carpeta Ciudadana se ha introducido una integración con el servicio web de consulta de DIR3. Por esto, es necesario realizar las siguientes configuraciones:

1. Creación del componente DIR3, para el tipo de componente TREW@, con los siguientes atributos:

Parámetro	Valor
Nombre	DIR3
Dirección IP	URL del servicio web. Por ejemplo, https://cancanaprun1.chap.junta-andalucia.es/cmpru/rest/DIR3Services/
Usuario	Nombre del usuario, si fuera necesario.
Password	Contraseña del usuario, si fuera necesario

Tabla 22: Atributos del componente DIR3

2. Creación de los siguientes datos de componente:

Parámetro	Valor
all.justice.units	Nombre del método de obtención de unidades organizativas de justicia: getAllJusticeOrganizationalUnits
all.organizational.units	Nombre del método de obtención de unidades organizativas: getAllOrganizationalUnits
dir3.parent	Código DIR3 del organismo padre, por defecto, para consultas: A01002820
justice.unit.detail	Nombre del método de obtención del detalle de una unidad organizativa de justicia: getJusticeOrganizationalUnitDetail
organizational.unit.children	Nombre del método de obtención de los hijos de una unidad organizativa: getOrganizationalUnitChildren
organizational.unit.detail	Nombre del método de obtención del detalle de una unidad organizativa: getOrganizationalUnitDetail

Tabla 23: Datos del componente DIR3

3.5.4Configuración de la fachada de @firma

En esta versión de Trew@, como se ha mencionado anteriormente, se introduce el acceso a la Herramienta de Administración por certificado electrónico, mediante la fachada de @firma. Para ello es necesario configurar los siguiente valores como datos del componente @firma.

Parámetro	Valor
AFIRMA.TICKETS.AUTH.HOST	Url del servicio web. Por ejemplo, ws159.juntadeandalucia.es
AFIRMA.ENTORNO	PRODUCCION
AFIRMA.TICKETS.URL.SERVLET	ReturnAuthenticationServlet
AFIRMA.TICKETS.URL.APP	login.do

Tabla 24: Atributos del componente @firma para el uso de la fachada.

NOTA: *Esta configuración deberá ser realizada directamente en base de datos. Ya que sin esto no se podrá acceder a la herramienta. Además de esta configuración, será necesario configurar el usuario y contraseña de @firma en el componente.*

3.6 Otras configuraciones

3.6.1 Configuración de WebOffice

Para el correcto funcionamiento de WebOffice (versiones anteriores a v3.0.0) se precisa el establecimiento de un perfil para el mismo. Dicho perfil se especificará a través del archivo de **weboffice.properties** que se ubicará en el paquete **trewa.conf** de la aplicación cliente con el siguiente formato:

perfil=default

Parámetro en Trew@	Obligatorio	Descripción
perfil	No	Perfil de acceso para WebOffice

Tabla 25: Configuración WebOffice

3.6.2 Condiciones, acciones, variables y selectores de plantillas

Una de las características que aporta la interfaz de acceso a Trew@ es la posibilidad de ejecutar clases java desplegadas en el servidor de aplicaciones. Para poder habilitar esta característica se deben tener en cuenta los siguientes puntos:

1. Se debe definir alguna condición/acción/variable y/o selector de plantilla en un procedimiento con *Implementación = "Java"*. Dicha definición debe llevar el nombre de la clase, que incluirá el paquete completo al que pertenezca (miApp.trewa.condiciones.ClaseCondiciones), y el nombre de la función a ejecutar. Dicha función debe aparecer definida como *public* en la clase de forma que pueda ser accesible por trew@.
2. Los métodos asociados a condiciones o acciones deben devolver un valor numérico (objeto java.lang.Integer,…):
   - **0**: si no se cumple la condición
   - **1**: si se cumple la condición. Los parámetros en este caso deben llevar el siguiente orden y deben ser del tipo que corresponda según la siguiente tabla:

Parámetro en Trew@	Parámetro en método
Id Expediente	java.math.BigDecimal
Id Transición	java.math.BigDecimal
Id Documento permitido	java.math.BigDecimal
Id Expediente en fase	java.math.BigDecimal
Id del procedimiento	java.math.BigDecimal
Fecha	java.sql.Timestamp
Usuario	java.lang.String
Id Fase	java.math.BigDecimal
Id del tipo de documento	java.math.BigDecimal

Tabla 26: Parámetros de condiciones

3. Los métodos asociados a variables deben devolver un java.lang.String. Los parámetros en este caso deben ser equivalentes a los definidos en los parámetros asociados a la variable, siendo la correspondencia la indicada en la siguiente tabla:

Tipo de parámetro en Trew@	Tipo de parámetro en método
NUMÉRICO	java.lang.Integer
CADENA	java.lang.String

FECHA	java.sql.Timestamp

Tabla 27: Parámetros de variable

4. Los métodos asociados a selectores de plantillas deben devolver un **java.lang.String** con el valor del código de plantilla Office asociado al tipo de documento que se deseé generar. Los parámetros en este caso deben llevar el siguiente orden y deben ser del tipo que corresponda según la siguiente tabla:

Parámetro en Trew@	Parámetro en método
Id Expediente	java.math.BigDecimal
Id Documento permitido	java.math.BigDecimal
Id Expediente en fase	java.math.BigDecimal
Id del procedimiento	java.math.BigDecimal
Fecha	java.sql.Timestamp
Usuario	java.lang.String
Id del tipo de documento	java.math.BigDecimal

Tabla 28: Parámetros de plantillas

5. La clase debe aparecer accesible a las clases de Trew@.
6. Si se desea que la función a ejecutar en la condición o variable reutilice la interfaz TrAPIUI, la clase a la que pertenezca se tendrá que derivar de la clase trewa.ext.TrAccesoUI. Heredando de la clase TrAccesoUI se tendrá acceso al método **getApiUI()** a través del cual se podrá obtener la interfaz TrAPIUI y reutilizar la conexión actual.

   Además al extender de la clase TrAccesoUI tendremos acceso al método **getNombre()** con el que podemos obtener el nombre de la variable, condición u acción que se esté ejecutando.

IMPORTANTE: *Hasta la versión 1.4.2 de Trew@, los atributos apiUI y nombre de la clase TrAccesoUI eran static, a partir de esta versión no son static, para corregir algunas causísticas. Al actualizar de una versión anterior o igual a la 1.4.2 a una superior se recomienda revisar el código de las variables, condiciones y acciones.*

```
import trewa.ext.TrAccesoUI;
public class ClaseVariable extends TrAccesoUI
{
  // Método de variable sin parámetros
  public String metodoVar( java.lang.String val ) throws TrException
  {
      String valRet = "";
      return str;
```

```
  }
}


import trewa.ext.TrAccesoUI;
public class ClaseCondicion extends TrAccesoUI
{
  // Método de condición
  public Integer existenDatos( BigDecimal idExpediente ) throws TrException
  {
      Integer intRet = new Integer(0);
      apiUI = getApiUI();
      if ( apiUI == null ) return intRet;
      ....
      return str;
  }
}
```

Figura 26: Fragmentos de código de variable y condición

En el caso de condiciones complejas, se permite que la condición/es que componen dicha condición compleja devuelvan un String. Con esto podremos comparar el valor de una condición con un valor numérico o una cadena para decidir si la condición se cumple o no. Para ello debemos usar el prefijo $$VAL$$ si queremos que se compare por el valor de la condición en lugar de su resultado lógico.

Por ejemplo, suponemos que tenemos definidas dos condiciones, COND_COMPLEJA y COND_SIMPLE. La COND_SIMPLE estará definida como una condición de implementación java que devolverá un String con el valor “2000”. La COND_COMPLEJA estará definida como una condición compleja de implementación java cuya expresión podría ser “$$VAL$ $COND_SIMPLE$$ > 1000”, esto quiere decir que si el valor de COND_SIMPLE es mayor que 1000 se cumplirá la condición.

3.6.3Configuración de las J-TrAPIs

La configuración que se muestra en este apartado es exclusivamente para aplicaciones que se integren con el API de Trew@. No es necesaria la configuración para la Herramienta de Administración de Trew@.

3.6.3.1Configuración de los servlets

Para que una aplicación pueda acceder a los servlets de subida y descarga de documentos (TrAPIUI), a los ficheros de subida y descarga de XML (TrAPIUTL) y a los de subida de documento WebOffice y descarga de XML WebOffice, debe existir un paso previo de configuración a nivel del servidor de aplicaciones. Básicamente la configuración consistirá en modificar el archivo **web.xml** de la aplicación tercera que haga uso del API para registrar la información de los servlets.

Se editará el archivo **web.xml** y en la primera parte *Built In Servlet Definitions* definimos los servlets de la siguiente forma:

```
<servlet>
   <servlet-name>TrAPIUIDescarga</servlet-name>
        <servlet-class>trewa.bd.trapi.trapiui.servlet.DescargaDocumento</servlet-class>

</servlet>


<servlet>
  <servlet-name>TrAPIUISubida</servlet-name>
  <servlet-class>trewa.bd.trapi.trapiui.servlet.SubidaDocumento</servlet-class>
</servlet>
<servlet>
   <servlet-name>TrAPIUTLDescarga</servlet-name>
   <servlet-class>trewa.bd.trapi.trapiutl.servlet.DescargaXML</servlet-class>
</servlet>


<servlet>
   <servlet-name>TrAPIUTLSubida</servlet-name>
   <servlet-class>trewa.bd.trapi.trapiutl.servlet.SubidaXML</servlet-class>
</servlet>


<servlet>
   <servlet-name>TrAPIUTLSubidaZip</servlet-name>
   <servlet-class>trewa.bd.trapi.trapiutl.servlet.SubidaXMLzip</servlet-class>
</servlet>


<!-- Servlets para WebOffice -->


<servlet>
   <servlet-name>TrAPIUISubidaOO</servlet-name>
   <servlet-class>trewa.bd.trapi.trapiui.servlet.OOSubidaDocumento</servlet-class>
</servlet>


<servlet>
   <servlet-name>TrAPIUIDescargaXMLOO</servlet-name>
   <servlet-class>trewa.bd.trapi.trapiui.servlet.OODescargaXML</servlet-class>
</servlet>
```

Figura 27: Definición de servlets

Además se debe añadir información referente a desde dónde se puede acceder a los servlets. Esta información se denomina *Servlet Mappings*, y para configurarla se incluirán las siguientes líneas en el apartado *Built In Servlet Mappings*:

```
<servlet-mapping>
   <servlet-name>TrAPIUIDescarga </servlet-name>
   <url-pattern>/trewa/DescargaDoc</url-pattern>
 </servlet-mapping>

<servlet-mapping>
  <servlet-name>TrAPIUISubida </servlet-name>
  <url-pattern>/trewa/SubidaDoc</url-pattern>
</servlet-mapping>

<servlet-mapping>
   <servlet-name>TrAPIUTLDescarga </servlet-name>
   <url-pattern>/trewa/DescargaXML</url-pattern>
</servlet-mapping>

<servlet-mapping>
   <servlet-name>TrAPIUTLSubida </servlet-name>
   <url-pattern>/trewa/SubidaXML</url-pattern>
</servlet-mapping>

<servlet-mapping>
   <servlet-name>TrAPIUTLSubidaZip</servlet-name>
   <url-pattern>/trewa/SubidaXMLzip</url-pattern>
</servlet-mapping>

<servlet-mapping>
   <servlet-name>TrAPIUISubidaOO</servlet-name>
   <url-pattern>/trewa/OOSubidaDoc/*</url-pattern>
</servlet-mapping>

<servlet-mapping>
   <servlet-name>TrAPIUIDescargaXMLOO</servlet-name>
   <url-pattern>/trewa/OODescargaXML/*</url-pattern>
</servlet-mapping>
```

Figura 28: Definición bindings de servlets

Siendo *\standalone\deployments\ aplicacion.war\WEB_INF* la ubicación por defecto dónde se encuentra archivo **web.xml**.

4 MANTENIMIENTO

4.1 Base de datos

La base de datos de Trew@ no requiere especial mantenimiento, aparte de las tareas típicas de administración de base de datos. Basta con ir actualizando cuando sea necesario a futuras nuevas versiones.

4.2 Pantallas de Administración de Trew@

Estas pantallas no sólo nos van a permitir definir procedimientos en Trew@, sino que también nos van a permitir mantener los datos de configuración del motor (tablas de constantes nombradas anteriormente).

Mediante estas pantallas podremos por ejemplo definir el "juego" de constantes que va a utilizar cada sistema (siempre que sea necesario) así como modificar los permisos y perfiles asociados a cada usuario.

Además nos van a permitir la importación al sistema y exportación desde el sistema de ficheros de definición de procedimientos en XML.

4.3 Condiciones y acciones

Cabe destacar en este apartado que las condiciones y acciones pl/sql que se definen en el procedimiento a nivel de transiciones y de tareas, deben estar definidas como funciones PL/SQL que devuelven un valor numérico (0 para cuando no se cumplan y 1 cuando se ejecuten correctamente). También deben darse permisos de ejecución sobre las mismas al usuario propietario del motor Trew@ (o al usuario de conexión en caso de utilizar usuarios distintos al propietario).

4.4 Variables de los documentos

Al igual que las condiciones las variables que se definen en los documentos, deben estar definidas como funciones que devuelven un valor cadena de caracteres (varchar2 o String). También en el caso de variables pl/sql deben darse permisos de ejecución sobre las mismas al usuario propietario del motor Trew@ (o al usuario de conexión en caso de utilizar usuarios distintos al propietario).

Para que Trew@ haga la sustitución de variables por el valor de la función que representan, éstas deben ponerse en los párrafos de los documentos como "$$*NOMBRE_VARIABLE*$$".

En el caso de utilizar WebOffice, además de cumplir esta norma en el texto de la plantilla del documento, las variables se deben asociar al tipo de documento independientemente de si están implementadas en pl/sql o java.

5 BIBLIOGRAFÍA Y REFERENCIAS

Referencia	Título	Código
Ref.1	Manual integración Weboffice	TRW261E_MIC_WebOfficev412_Manual_De_Integracion_v01r00.pdf
Ref.2	Manual de pantallas de Trew@	TRW26110_MUS_Manual_de_pantallas_Trewa_v01r00.odt

Tabla 29: Bibliografía y Referencias