Trew@ v1.2.0 - Guía de referencia de J-TRAPI

CONSEJERÍA DE JUSTICIA Y ADMINISTRACIÓN PÚBLICA

29 de diciembre de 2006

GuadalTEL
INGENIERÍA DE SISTEMAS

ÍNDICE

Introducción

TREW@ como tramitador de procedimientos, dispone de una serie de funcionalidades mediante las cuáles un sistema externo puede, en general:

- gestionar la tramitación de expedientes conforme a la definición de los procedimientos que en TREW@ se han implementado
- obtener todos los datos que sobre dicha tramitación se recoge en TREW@ a modo de “historia” de tramitación
- obtener otro tipo de información acerca de los procedimientos definidos

Esta funcionalidad se ofrece a los sistemas que utilizan TREW@ mediante un conjunto de métodos, en adelante **TrAPIs**, las cuáles permiten interactuar con el metamodelo de datos que da soporte a TREW@.

Una visión global de las J-TrAPIs

Las **J-TrAPIs** son el punto de acceso de las aplicaciones al motor de tramitación TREW@. El objetivo de las mismas es abstraer a las aplicaciones del complejo modelo de datos y aportar una serie de funcionalidades enfocadas a tener un uso común de acceso para todas las aplicaciones.

Se han definido una serie de APIs según su funcionalidad:

- Interfaz de acceso UI – TrAPIUI

  API para la gestión de operaciones de tramitación.

- Interfaz de acceso ADM – TrAPIADM

  API para la gestión de operaciones de definición de procedimientos.

- Interfaz de acceso UTL – TrAPIUTL

  API para el intercambio de definiciones de procedimiento entre diferentes sistemas.

Generalidades

El conjunto de APIs que ofrece TREW@ se encuentran implementadas en forma de métodos que acceden al modelo de datos para realizar las operaciones necesarias.

Datos devueltos

Un gran número de las APIs que existen tratan de devolver en algunos de sus parámetros información en forma de un conjunto de datos. Para ello, se aportan una serie de ***clases de entidad*** a través de las cuales se producirá el intercambio de información entre aplicaciones y el propio motor de TREW@. Se recomienda comprobar si se han obtenido datos preguntando si el array es distinto de null y su tamaño es mayor a cero. Por ejemplo:

```
...
TrFase[] datosFase = apiUI.obtenerDatosFase(new TpoPK(123),null,null);
if(datosFase!=null && datosFase.length > 0)
{
     //hay datos
     ...
}
```

Filtros y orden sobre los datos

La mayoría de los métodos de las APIs que devuelven un array de elementos permiten hacer filtros y ordenaciones sobre los mismos según se necesite. Esto se hace a través del paso de parámetro de objetos de 2 clases implementadas para montar sentencias Where y OrderBy: **ClausulaWhere** y **ClausulaOrderBy**. En muchos casos aparecerán sentencias del tipo:

Nombre_API_TREW@ (..., ClausulaWhere ***where****, ClausulaOrderBy* ***orderby*** *)*

En general para todas aquellas consultas sobre las que no se quiera aplicar filtros u ordenación, bastará con pasar "*null*" en el paso de estos parámetros.

Para una descripción completa de estas clases y su forma de uso, así como de las clases involucradas, se remite al apartado: Clases SQL.

Errores

En general, los errores devueltos por TREW@ se harán en forma de excepciones que deberán ser capturadas por las aplicaciones para conseguir el mensaje de error.

Para una descripción completa de las clases de Excepción, se remite al apartado: Clases de Excepción.

Estructura de paquetes

Las TrAPIs presentan una estructura organizada en paquetes según la funcionalidad expuesta en los mismos. La siguiente tabla describe los principales paquetes en los que se descompone la librería:

paquete	Descripción
trewa	Paquete principal del que cuelga toda la implementación de las J-TrAPIS.
trewa.bd	Paquete para clases de acceso a base de datos.
trewa.bd.sql	Paquete para clases SQL.
trewa.bd.tpo	Paquete para clases de enmascaramiento de tipos.
trewa.bd.trapi	Paquete principal de acceso a las J-TrAPIs.
trewa.bd.trapi.trapiui	Paquete para clases de acceso a la interfaz TrAPIUI.
trewa.bd.trapi.trapiui.tpo	Paquete para clases de entidad asociadas a la interfaz TrAPIUI.
trewa.bd.trapi.trapiadm	Paquete para clases de acceso a la interfaz TrAPIADM.
trewa.bd.trapi.trapiutl	Paquete para clases de acceso a la interfaz TrAPIUTL.
trewa.conf	Paquete para clases de configuración de Trew@.
trewa.conf.perfiles	Paquete para la especificación de perfiles.
trewa.exception	Paquete para clases de excepciones Trew@.
trewa.ext	Paquete para clases de extensión Trew@.
trewa.util	Paquete para clases de utilidad.

Apis que usan w@rdA

Las apis de la interfaz TrAPIUI, marcadas en esta Guía de referencia con una (w) intercambian información de los documentos con el componente w@rdA, siempre y cuando se haya definido dicho componente en el sistema con el que estemos trabajando. Para que la información enviada al componente w@rdA sea correcta deben estar rellenos los siguientes campos en Trew@:

- El código w@ndA del componente w@rdA.
- El ciwa del organismo al que pertenece el expediente.
- El código w@ndA del procedimiento.
- El código w@ndA del tipo de documento.
- El código w@ndA de los interesados del documento y de sus razones de interés .

Clases SQL. El paquete trewa.bd.sql

Las clases SQL se usan para realizar filtros y ordenaciones en las peticiones a las TrAPIs aportando un mecanismo para montar cláusulas Where y Order By. Todas estas clases pertenecen al paquete ***trewa.bd.sql***.

Básicamente una ***cláusula Where*** aparece implementada en la librería como un conjunto de "***Expresiones Where***" y otras "*cláusulas Where*" enlazadas unas con otras a tavés de "***Operadores lógicos***" .

→ Una expresión Where presenta el siguiente formato:

< *ExpresionWhere* > = < *Campo* > + < *OperadorWhere* > + < *Valor* >

→ Una ClausulaWhere presenta alguno de los siguientes formatos:

< *ClausulaWhere* > = < *ExpresionWhere* | *ClausulaWhere* > + < *OperadorLogico* > + < *ExpresionWhere* | *ClausulaWhere* >

La **cláusula "Order By"** aperece implementada como un conjunto de "Expresiones Order By".

→ Una ClausulaOrderBy presenta el siguiente formato:

< *ClausulaOrderBy* > = < *ExpresionOrderBy*> + <*ExpresionOrderBy*> + ...

→ Una expresión OrderBy presenta el siguiente formato:

< *ExpresionOrderBy* > = < *Campo* > + < *OperadorOrderBy* >

Operadores

Los operadores intervienen en las diferentes expresiones Where y OrderBy

OperadorWhere

trewa.bd.sql.OperadorWhere

Esta clase define un operador válido para una expresión where. Los posibles operadores habilitados son:

OperadorWhere	Operador
OP_LIKE	“LIKE”
OP_NOT_LIKE	“NOT LIKE”
OP_IGUAL	“=”
OP_DISTINTO	“!=”
OP_IS_NULL	“IS NULL”
OP_IS_NOT_NULL	“IS NOT NULL”
OP_MAYOR	“>”
OP_MENOR	“<”
OP_MAYOR_IGUAL	“>=”
OP_MENOR_IGUAL	“<=”
OP_IN	“IN”

OperadorOrderBy

trewa.bd.sql.OperadorOrderBy

Esta clase define un operador válido para una expresión Order By. Los posibles operadores habilitados son:

OperadorWhere	Operador
ASCENDENTE	“ASC”
DESCENDENTE	“DESC”

OperadorLogico

trewa.bd.sql.OperadorLogico

Esta clase define un operador lógico usado entre expresiones Where. Los posibles operadores habilitados son:

OperadorWhere	Operador
AND	"AND"
OR	"OR"

Campos y valores

Un Campo para la librería es una columna sobre la que se va a poder filtrar u ordenar.

Los Campos son usados tanto por las expresiones Where como por las OrderBy y se obtendrán a través de las clases de entidad, para las cuales se han definido una serie de campos susceptibles de ser usados en los filtros y ordenaciones. Para ver una relación de los campos disponibles y los posibles valores de uso, consultar el apartado *Clases de Entidad*.

Campo

trewa.bd.Campo

Interfaz para aportar información sobre un campo.

Métodos

→ `String getNombreCampo( )`

Obtiene el nombre de un campo

→ `TipoCampo getTipo( )`

Obtiene el tipo de campo

TipoCampo

trewa.bd.TipoCampo

Clase que representa un tipo de campo.

Tipo de campo	**Tipo**
TIPO_NUMBER	“NUMBER”
TIPO_VARCHAR2	“VARCHAR2”
TIPO_DATE	“DATE”
TIPO_BOOLEAN	“BOOLEAN”

Expresiones

ExpresionWhere

trewa.bd.sql.ExpresionWhere

Clase que define una expresión simple en una cláusula where.

Constructores

→ `ExpresionWhere( Campo campoNuevo, OperadorWhere opNuevo, Object sValNuevo )`

Crea una expresión where con formato <campo> + <Operador> + <valor>.

→ `ExpresionWhere( Campo campoNuevo, OperadorWhere opNuevo )`

Crea una expresión where con formato <campo> + <Operador>.
Este constructor habilita la creación de Expresiones que incluyan operadores del tipo "IS NULL" , "IS NOT NULL".

→ `ExpresionWhere( Campo campoNuevo, ArrayList sValNuevo )`

Crea una expresión where con formato <campo> + OperadorWhere.OP_IN + <Lista de valores>.
Este constructor habilita la creación de expresiones que incluyan el operador "**IN**" acompañado de una lista de valores.

ExpresionOrderBy

trewa.bd.sql.ExpresionOrderBy

Clase que define una expresión en una cláusula Order By.

Constructores

→ `ExpresionOrderBy( Campo nuevoCampo, OperadorOrderBy nuevoOrden )`

Crea una expresión order by con formato <campo> + <Operador>

Ejemplo:

```
ExpresionOrderBy exp = new ExpresionOrderBy(TrMensaje.CAMPO_FECHA,OperadorOrderBy.ASCENDENTE );
```

Cláusulas

ClausulaWhere

trewa.bd.sql.ClausulaWhere

Clase que define una cláusula where para aplicar un filtro en algún método de la TrAPI. Una cláusula where puede incluir a otras cláusulas where y a expresiones where unidas por operadores lógicos (‘and’ u ‘or’).

Constructores

→ `ClausulaWhere( )`

Crea una cláusula where sin expresiones pero con el operador lógico ‘AND’ por defecto como nexo de unión.

→ `ClausulaWhere( OperadorLogico nuevoOpLog )`

Crea una cláusula where sin expresiones que hace uso del operador lógico recibido como nexo de unión.

Métodos para inclusión de expresiones en la cláusula where

→ `boolean addExpresion( Campo campo, OperadorWhere op )`

Crea una expresión basada en operadores “IS NULL” o “IS NOT NULL” y la añade a la cláusula where.

→ `boolean addExpresion( Campo campo, OperadorWhere op, String strValor )`

Crea una expresión y la añade a la cláusula where.

→ `boolean addExpresionFecha( Campo campo, OperadorWhere op, TpoFecha fecha )`

Crea una expresión con valor de tipo fecha y la añade a la cláusula where.

→ `boolean addExpresionIn( Campo campo, ArrayList aValor)`

Crea una expresión basada en el operador “IN” y la añade a la cláusula where.

→ `boolean addExpresion( ExpresionWhere nuevaExpresion )`

Añade una nueva expresión a la cláusula where.

→ `boolean addExpresion(ClausulaWhere cWhere)`

Añade otra clausula where a la clausula where actual.
Con este método se habilitan mecanimos de anidamiento en los filtros.

Ejemplos de uso de ClausulaWhere:

```
cWhere.addExpresion(TrMensaje.CAMPO_FECHA,OperadorWhere.OP_MAYOR_IGUAL,"10/10/2004");

cWhere.addExpresion(TrMensaje.CAMPO_REFMENSAJE, OperadorWhere.OP_IS_NOT_NULL);
```

ClausulaOrderBy

trewa.bd.sql.ClausulaOrderBy

Clase que define una cláusula order by para aplicar ordenación en algún método de la TrAPI.

Constructores

→ `ClausulaOrderBy( )`

Crea una cláusula order by sin expresiones.

Métodos para inclusión de expresiones en la cláusula order by

→ `boolean addExpresion( Campo campo, OperadorOrderBy op )`

Crea una expresión order by y la añade a la cláusula order by.

Ejemplo:

```
// ORDER BY CAMPO_FECHA DESC, CAMPO_TEXTOMENSAJE ASC
ClausulaOrderBy orderMsj = new ClausulaOrderBy();
orderMsj.addExpresion(TrMensaje.CAMPO_FECHA,OperadorOrderBy.DESCENDENTE);
orderMsj.addExpresion(TrMensaje.CAMPO_TEXTOMENSAJE,OperadorOrderBy.ASCENDENTE);
```

→ `boolean addExpresion( ExpresionOrderBy nuevaExpresion )`

Añade una nueva expresión a la cláusula order by.

Ejemplo:

```
// ORDER BY CAMPO_FECHA DESC
ClausulaOrderBy orderMsj = new ClausulaOrderBy();
ExpresionOrderBy exp = new ExpresionOrderBy(TrMensaje.CAMPO_FECHA,OperadorOrderBy.DESCENDENTE );
orderMsj.addExpresion( exp );
```

Clases de Excepción.

Las TrAPIs avisan de errores producidos en las mismas, disparando excepciones del tipo TrException.

TrException

trewa.exception.TrException

Clase de excepción para avisar de errores producidos en las TrAPIs.

Métodos

→ `String getMessage()`

Obtiene el mensaje de error asociado a la excepción.

Ejemplo:

```
try{
      ...
} catch( TrException ex ){
      System.out.println( ex.getMessage() );
}
```

→ `long getErrorCode()`

Obtiene el código de error Trew@.

Ejemplo:

```
try{
      ...
} catch( TrException ex ){
      System.out.println( "--> " + ex. getErrorCode () + " -- " + ex.getMessage() );
}
```

Descripción de la TrAPIUI

Descripción de clases involucradas

Interfaz J-TrAPIUI

La interfaz J-TrAPIUI define todos los métodos de acceso al motor de tramitación TREW@. En la librería este acceso se realiza a través de la implementación de la interfaz **TrAPIUI** que se describe en este apartado.

TrAPIUI

trewa.bd.trapi.trapiui.TrAPIUI

Interfaz de acceso al motor de tramitación TREW@. Todos los métodos quedan descritos en el apartado "***Métodos del TrAPIUI***".

Para poder trabajar con la TrAPIUI es preciso crear una instancia de la misma a través de la clase Factoría: ***TrAPIUIFactory***, descrita en el apartado siguiente.

Clase Factoría

La clase Factoría se encarga de crear una instancia correcta de la J-TrAPIUI de forma que los métodos de esta última aparezcan disponibles para las aplicaciones.

TrAPIUIFactory

trewa.bd.trapi.trapiui.TrAPIUIFactory

Clase que aporta funcionalidad para la correcta creación de una instancia de la TrAPIUI.

Métodos para la creación de instancias del TrAPIUI

→ `TrAPIUI crearAPIUI(String stma)`

Devuelve una instancia del TrAPIUI. La información para el establecimiento de la conexión la obtiene a través del perfil "default".

Entradas	*String stma*	*Nombre del sistema con el que va a trabajar el TrAPIUI*. Si se pasa un valor *null* se creará un TrAPIUI sin sistema asociado y habrá métodos del TrAPIUI que no funcionarán correctamente. Se deben establecer los parámetros posteriormente a través del método *establecerConfiguracionSistema* del TrAPIUI. Si el sistema pasado no existe devolverá *null*.
Salidas	*TrAPIUI*	Instancia del TrAPIUI o "null" si no se pudo crear.

→ `TrAPIUI crearAPIUI(String usuario, String clave, String stma)`

Devuelve una instancia del TrAPIU. La información para el establecimiento de la conexión la obtiene a través del perfil "default" pero en este caso hace uso del usuario y password que recibe como parámetros.

Entradas	*String usuario*	Usuario
	String clave	Clave de usuario
	String stma	*Nombre del sistema con el que va a trabajar el TrAPIUI*. Si se pasa un valor *null* se creará un TrAPIUI sin sistema asociado y habrá métodos del TrAPIUI que no funcionarán correctamente. Se deben establecer los parámetros posteriormente a través del método *establecerConfiguracionSistema* del TrAPIUI. Si el sistema pasado no existe devolverá *null*.
Salidas	*TrAPIUI*	Instancia del TrAPIUI o "null" si no se pudo crear.

→ `TrAPIUI crearAPIUI(String strPerfil,String stma)`

Devuelve una instancia del TrAPIU. La información para el establecimiento de la conexión la obtiene a través del perfil especificado en la llamada como parámetro.

Entradas	*String strPerfil*	*Nombre del perfil para el establecimiento de la conexión.*
	String stma	*Nombre del sistema con el que va a trabajar el TrAPIUI*. Si se pasa un valor *null* se creará un TrAPIUI sin sistema asociado y habrá métodos del TrAPIUI que no funcionarán correctamente. Se deben establecer los parámetros posteriormente a través del método *establecerConfiguracionSistema* del TrAPIUI. Si el sistema pasado no existe devolverá *null*.
Salidas	*TrAPIUI*	*Instancia del TrAPIUI o "null" si no se pudo crear.*

→ `TrAPIUI crearAPIUI(String strPerfil, String usuario, String clave,String stma)`

Devuelve una instancia del TrAPIUI. La información para el establecimiento de la conexión la obtiene a través del perfil especificado en la llamada como parámetro, pero en este caso hace uso del usuario y password que recibe como parámetros.

Entradas	*String strPerfil*	*Nombre del perfil para el establecimiento de la conexión.*
	String usuario	*Usuario*
	String clave	*Clave de usuario*
	String stma	*Nombre del sistema con el que va a trabajar el TrAPIUI*. Si se pasa un valor *null* se creará un TrAPIUI sin sistema asociado y habrá métodos del TrAPIUI que no funcionarán correctamente. Se deben establecer los parámetros posteriormente a través del método *establecerConfiguracionSistema* del TrAPIUI. Si el sistema pasado no existe devolverá *null*.
Salida	*TrAPIUI*	*Instancia del TrAPIUI o "null" si no se pudo crear.*

Ejemplo de uso

En el siguiente ejemplo se muestra cómo obtener una instancia del TrAPIUI a través de la clase TrAPIUIFactory:

```
TrAPIUI apiUI = null;
TpoPK pK = null;

apiUI = TrAPIUIFactory.crearAPIUI( "perfilTrewa", "SISTEMA_TREWA" );
if ( apiUI != null ){
       // Ya se puede acceder a los métodos del TrAPIUI
       pK = new TpoPK( BigDecimal.valueOf(39) );
       try{
          TrParrafo[] arrParrafos = apiUI.obtenerParrafosDocumento( pK, null, null );
       ...
      }
     catch(TrException ex)
     {
         ex.printStackTrace();
     }
}
```

Clases de entidad

Estas clases son el mecanismo de intercambio de información entre las aplicaciones y el motor Trew@ a través de las J-TrAPIs.

Las clases de entidad asociadas al TrAPIUI se encuentran en el paquete **trewa.bd.trapi.trapiui.tpo**.

TrAccion

trewa.bd.trapi.trapiui.tpo.TrAccion

Clase que representa la información asociada a una acción.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **NOMBRE**	`String`	Nombre de la acción
→ **DESCRIPCION**	`String`	Descripción de la acción
→ **STMA**	`TrSistema`	Sistema

TrAccionDocumento

trewa.bd.trapi.trapiui.tpo.TrAccionDocumento

Clase que representa la información relacionada con una acción asociada a un documento.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **ACCION**	`TrAccion`	Acción
→ **COMPROBAR**	`String`	Comprobación → "**G**" – La acción se produce al Generar el documento → "**I**" – La acción se produce al Incorporar el documento → "**T**" – La acción se produce al Generar e Incorporar el documento

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_NOMBRE**	`CADENA(50)`	Nombre de la acción
→ **CAMPO_DESCRIPCION**	`CADENA(250)`	Descripción de la acción
→ **CAMPO_COMPROBAR**	`CADENA(1)`	Comprobación → "**G**" – La acción se produce al Generar el documento → "**I**" – La acción se produce al Incorporar el documento → "**T**" – La acción se produce al Generar e Incorporar el documento
→ **CAMPO_REFSTMA**	`NUMÉRICO`	Identificador del sistema asociado a la acción

Métodos para aplicación de los filtros → `obtenerAccionesDocumento`

TrAccionDocumentoPortafirmas

trewa.bd.trapi.trapiui.tpo.TrAccionDocumentoPortafirmas

Clase que representa la información de una acción que se ejecutará al cambiar el estado del documento en port@firmas. Esta información se envía a port@firmas mediante el método enviaDocumentoExpediente y port@firmas se encargará de ejecutar la acción cuando se cumpla el estado indicado.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **ESTADO**	`String`	Estado del documento cuando se ejecutará la acción. → "**FIRMADO**" – Firmado → "**DEVUELTO**" – Devuelto → "**LEIDO**" – Leído → "**NUEVO**" – Nuevo
→ **ACCION**	`String`	Procedimiento o url que se ejecutará.
→ **TIPO**	`String`	Tipo de la acción → "**PLSQL**" – Procedimiento PLSQL → "**WEB**" – Url web.

TrAccionTransicion

trewa.bd.trapi.trapiui.tpo.TrAccionTransicion

Clase que representa la información referente a acciones a realizar en una transición.

El modelo de referencia w@nda define una acción en transición como *"aquellas acciones que son realizadas por el tramitador de procedimientos sobre la base de unas condiciones establecidas, cuando se produce una transición concreta entre fases*".

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **ACCION**	`TrAccion`	Acción
→ **COMPROBAR**	`String`	Comprobación → "**D**" – La acción se produce al Deshacer la transición → "**T**" – La acción se produce al Tramitar la transición → "**A**" – La acción se produce al Tramitar y Deshacer la transición

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_NOMBRE**	`CADENA(50)`	Nombre de la acción
→ **CAMPO_DESCRIPCION**	`CADENA(250)`	Descripción de la acción
→ **CAMPO_COMPROBAR**	`CADENA(1)`	Comprobación → "**D**" – La acción se produce al Deshacer la transición → "**T**" – La acción se produce al Tramitar la transición → "**A**" – La acción se produce al Tramitar y Deshacer la transición
→ **CAMPO_REFSTMA**	`NUMÉRICO`	Identificador del sistema asociado a la acción

Métodos para aplicación de los filtros → `obtenerAccionesTransicion`

TrAvisoCaducidad

trewa.bd.trapi.trapiui.tpo.TrAvisoCaducidad

Clase que representa la información referente a un aviso establecido sobre una caducidad de un expediente.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**FECHA**	`Timestamp`	Fecha en la que se dio de alta el aviso
→	**FECHAAVISO**	`Timestamp`	Fecha de aviso de la caducidad al usuario
→	**USUARIO**	`String`	Usuario al que hay que avisar
→	**TRANSICION**	`TrTransicion`	Transición en la que hay que avisar al usuario

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_FECHA**	`DATE`	Fecha en la que se dio de alta el aviso
→	**CAMPO_FECHAAVISO**	`DATE`	Fecha de aviso de la caducidad al usuario
→	**CAMPO_USUARIO**	`CADENA(10)`	Usuario al que hay que avisar
→	**CAMPO_DESCTRANS**	`CADENA(50)`	Descripción de la transición en la que hay que avisar al usuario
→	**CAMPO_REFTIPOACTO**	`NUMERICO`	Identificador del tipo de acto administrativo que representa la transición en la que hay que avisar al usuario.

Métodos para aplicación de los filtros → `obtenerAvisosCaducidadExpediente`

TrBloque

trewa.bd.trapi.trapiui.tpo.TrBloque

Clase que representa la información referente a un bloque o módulo al que se puede acceder en cualquier momento del procedimiento.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFBLOQUE**	`TpoPK`	Identificador del bloque
→	**NOMBRE**	`String`	Nombre del bloque
→	**DESCRIPCION**	`String`	Descripción del bloque
→	**TIPO**	`String`	Tipo de bloque → "**W**" – Web → "**R**" – Informe → "**F**" – Pantalla
→	**STMA**	`TrSistema`	Sistema
→	**INFORMAR**	`String`	Indica si se informa al bus: → "**S**" – Sí → "**N**" – No

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFBLOQUE**	`NUMERICO`	Identificador del bloque
→	**CAMPO_NOMBRE**	`CADENA(20)`	Nombre del bloque
→	**CAMPO_DESCRIPCION**	`CADENA(250)`	Descripción del bloque
→	**CAMPO_TIPO**	`CADENA(1)`	Tipo del bloque → "**W**" – Web → "**R**" – Informe → "**F**" – Pantalla
→	**CAMPO_REFSTMA**	`NUMERICO`	Identificador del sistema
→	**CAMPO_INFORMAR**	`CADENA(1)`	Indica si se informa al bus → "**S**" – Sí → "**N**" – No

Métodos para aplicación de los filtros → `obtenerBloquesDefinidos`

TrBloquePermitido

trewa.bd.trapi.trapiui.tpo.TrBloquePermitido

Clase que representa la información referente a las tareas de tipo "MANIPULAR_DATOS" definidas en una fase de un procedimiento.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **BLOQUEFIN**	`TrBloque`	Bloque final
→ **ETIQUETA**	`String`	Etiqueta del bloque permitido
→ **DESCRIPCION**	`String`	Descripción del bloque permitido
→ **ETIQLARGA**	`String`	Etiqueta larga para la tarea en fase
→ **INFORMAR**	`String`	Indica si se informa al bus de esta tarea → "**S**" – Sí → "**N**" – No
→ **OBLIGATORIO**	`String`	Indica si es obligatoria la realización de esta tarea → "**S**" – Sí → "**N**" – No
→ **PLAZO**	`TrPlazo`	Plazo de realización de la tarea en fase
→ **REFBLOQUEPER**	`TpoPK`	Identificador del bloque permitido.
→ **BLOQUEINI**	`TrBloque`	Bloque inicial
→ **ORDEN**	`Integer`	Orden del bloque permitido.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFBLOQUE**	`NUMERICO`	Identificador del bloque
→ **CAMPO_ETIQUETA**	`CADENA(11)`	Etiqueta del bloque permitido
→ **CAMPO_DESCRIPCION**	`CADENA(50)`	Descripción del bloque permitido
→ **CAMPO_TIPO**	`CADENA(1)`	Tipo del bloque → "**W**" – Web → "**R**" – Informe → "**F**" – Pantalla
→ **CAMPO_NOMBRE**	`CADENA(20)`	Nombre del bloque
→ **CAMPO_ETIQLARGA**	`CADENA(25)`	Etiqueta larga para la tarea en fase
→ **CAMPO_INFORMAR**	`CADENA(1)`	Indica si se informa al bus de esta tarea → "**S**" – Sí → "**N**" – No

→	**CAMPO_OBLIGATORIO**	CADENA(1)	Indica si es obligatoria la realización de esta tarea → "**S**" – Sí → "**N**" – No
→	**CAMPO_UNIDAD**	CADENA(1)	Indica el tipo de unidad de tiempo → "**D**" – Días → "**M**" – Meses → "**A**" – Años
→	**CAMPO_NUMUNIDADES**	NUMERICO	Indica el número de unidades
→	**CAMPO_DESCFECHALIMITE**	CADENA(100)	Descripción del significado de la fecha límite de realización de la tarea.
→	**CAMPO_REFBLOQUEINI**	NUMERICO	Identificador del bloque inicial
→	**CAMPO_ETIQUETABLQINI**	CADENA(11)	Etiqueta del bloque inicial
→	**CAMPO_DESCRIPCIONBLQINI**	CADENA(50)	Descripción del bloque inicial
→	**CAMPO_NOMBREBLQINI**	CADENA(20)	Nombre del bloque inicial
→	**CAMPO_TIPOBLQINI**	CADENA(1)	Tipo del bloque → "**W**" – Web → "**R**" – Informe → "**F**" – Pantalla
→	**CAMPO_ORDEN**	NUMERICO	Orden del bloque permitido.

Métodos para aplicación de los filtros → `obtenerBloquesPermitidos`

TrCaducidad

trewa.bd.trapi.trapiui.tpo.TrCaducidad

Clase que representa la información referente a caducidades que se pueden definir para un procedimiento.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFCADU**	`TpoPK`	Identificador de la caducidad
→ **ABREVIATURA**	`String`	Abreviatura de la caducidad
→ **DESCRIPCION**	`String`	Descripción de la caducidad
→ **UNIDAD**	`String`	Unidad de medida de tiempo → "**D**" – Días → "**M**" – Meses → "**A**" – Años
→ **NUMUNIDADES**	`long`	Número de unidades que se cuentan (nº de días, nº de años,...)
→ **TIPO**	`String`	Tipo de caducidad → "**P**" - Plazo de ejecución → "**O**" - Otro
→ **VIGENTE**	`String`	Indica si la caducidad está vigente o activa en el procedimiento → "**S**" – Sí → "**N**" – No

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFCADU**	`NUMERICO`	Identificador de la caducidad
→ **CAMPO_ABREVIATURA**	`CADENA(10)`	Abreviatura de la caducidad
→ **CAMPO_DESCRIPCION**	`CADENA(50)`	Descripción de la caducidad
→ **CAMPO_UNIDAD**	`CADENA(1)`	Unidad de medida de tiempo → "**D**" – Días → "**M**" – Meses → "**A**" – Años
→ **CAMPO_NUMUNIDADES**	`NUMERICO(2)`	Número de unidades que se cuentan (nº de días, nº de años,...)
→ **CAMPO_TIPO**	`CADENA(1)`	Tipo de caducidad → "**P**" - Plazo de ejecución → "**O**" - Otro
→ **CAMPO_VIGENTE**	`CADENA(1)`	Indica si la caducidad está vigente o activa en el procedimiento

→ "**S**" – Sí
→ "**N**" – No

Métodos para aplicación de los filtros → `obtenerCaducidadesDefProcedimiento`

TrCaducidadExpediente

trewa.bd.trapi.trapiui.tpo.TrCaducidadExpediente

Clase que representa la información referente de caducidades asociadas a un expediente.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFCADEXP**	TpoPK	Identificador de la caducidad del expediente
→	**CADUCIDAD**	TrCaducidad	Caducidad que representa
→	**FECHA**	Timestamp	Fecha en la que comenzó a contar el tiempo para la caducidad
→	**FECHALIMITE**	Timestamp	Límite de tiempo para la caducidad
→	**REFEXP**	TpoPK	Identificador del expediente

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFCADEXP**	NUMÉRICO	Identificador de la caducidad del expediente
→	**CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura de la caducidad
→	**CAMPO_DESCRIPCION**	CADENA(50)	Descripción de la caducidad
→	**CAMPO_TIPO**	CADENA(1)	Tipo de caducidad → "**P**" - Plazo de ejecución → "**O**" - Otro
→	**CAMPO_FECHA**	DATE	Fecha en la que comenzó a contar el tiempo para la caducidad
→	**CAMPO_FECHALIMITE**	DATE	Límite de tiempo para la caducidad
→	**CAMPO_REFCADU**	NUMÉRICO	Identificador de la caducidad que representa (la definida en el procedimiento)
→	**CAMPO_REFEXP**	NUMÉRICO	Identificador del expediente

Métodos para aplicación de los filtros → `obtenerCaducidadesExpediente`

TrCambioProcedimientoExpediente

trewa.bd.trapi.trapiui.tpo.TrCambioProcedimientoExpediente

Clase que representa la información referente a los datos de un procedimiento que ha seguido o sigue un expediente.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFTIPOEXP**	TpoPK	Identificador del tipo de expediente
→	**REFDEFPROC**	TpoPK	Identificador de la definición de procedimiento que sigue o ha seguido el expediente
→	**OBSERVACIONES**	String	Observaciones anotadas en la "historia de evoluciones" por las que ha pasado el expediente
→	**USUARIO**	String	Usuario que dio de alta el expediente o cambió el procedimiento y tipo de expediente del expediente
→	**FECHA**	Timestamp	Fecha de alta o cambio de flujo del expediente en TREW@
→	**VIGENTE**	String	Indica si los datos están vigentes o no → "**S**" – Los datos sí están vigentes → "**N**" – Los datos no están vigentes
→	**NOMBREUSU**	String	Nombre y apellidos del usuario
→	**DESCTIPOEXP**	String	Descripción del tipo de expediente
→	**DESCDEFPROC**	String	Descripción de la definición del procedimiento

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFTIPOEXP**	NUMÉRICO	Identificador del tipo de expediente
→	**CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición de procedimiento que sigue o ha seguido el expediente
→	**CAMPO_OBSERVACIONES**	CADENA(250)	Observaciones anotadas en la "historia de evoluciones" por las que ha pasado el expediente
→	**CAMPO_USUARIO**	CADENA(10)	Usuario que dio de alta el expediente o cambió el procedimiento y tipo de expediente del expediente
→	**CAMPO_FECHA**	DATE	Fecha de alta o cambio de flujo del expediente en TREW@
→	**CAMPO_VIGENTE**	CADENA(1)	Indica si los datos están vigentes o no → "**S**" – Los datos sí están vigentes → "**N**" – Los datos no están vigentes
→	**CAMPO_NOMBREUSU**	CADENA(98)	Nobre y apellidos del usuario

→	**CAMPO_DESCTIPOEXP**	CADENA(50)	Descripción del tipo de expediente
→	**CAMPO_DESCDEFPROC**	CADENA(50)	Descripción de la definición del procedimiento

Métodos para aplicación de los filtros → obtenerDatosExpediente

TrComponente

trewa.bd.trapi.trapiui.tpo.TrComponente

Clase que representa la información referente a componentes que pueden intervenir en w@ndA.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFCOMPONENTE**	TpoPK	Identificador del componente.
→	**REFORGANISMO**	TpoPK	Identificador del organismo en el que está el componente
→	**NOMBRE**	String	Nombre del componente
→	**DESCRIPCION**	String	Descripción del componente
→	**TIPOCOMPONENTE**	TrTipoComponente	Tipo de componente
→	**CODWANDA**	Long	Identificador del componente en w@ndA

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFCOMPONENTE**	NUMÉRICO	Identificador del componente
→	**CAMPO_REFORGANISMO**	NUMÉRICO	Identificador del organismo en el que está el componente
→	**CAMPO_NOMBRE**	CADENA(50)	Nombre del componente
→	**CAMPO_DESCRIPCION**	CADENA(250)	Descripción del componente
→	**CAMPO_REFTIPOCOMP**	NUMERICO	Identificador del tipo de componente
→	**CAMPO_CODWANDA**	NUMERICO	Identificador del componente en w@ndA

Métodos para aplicación de los filtros → obtenerComponentes

TrCondicion

trewa.bd.trapi.trapiui.tpo.TrCondicion

Clase que representa la información referente a una condición.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **NOMBRE**	`String`	Nombre de la condición
→ **DESCRIPCION**	`String`	Descripción de la condición
→ **STMA**	`TrSistema`	Sistema

TrCondicionAccionAviso

trewa.bd.trapi.trapiui.tpo.TrCondicionAccionAviso

Clase que representa la información de una condición, acción o aviso.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **NOMBRE**	`String`	Nombre de la condición, acción o aviso.
→ **DESCRIPCION**	`String`	Descripción de la condición, acción o aviso.
→ **COMPROBAR**	`String`	Indica cuando se comprueba la condición, acció o aviso.
→ **OBLIGATORIA**	`String`	Indica si es obligatoria. → "**S**" – Sí → "**N**" – No
→ **TIPO**	`String`	Tipo → "**C**" – Condición → "**A**" – Acción → "**W**" – Aviso
→ **REFSTMA**	`TpoPK`	Identificador del sistema

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso

→	**CAMPO_NOMBRE**	CADENA(50)	Nombre de la condición, acción o aviso.
→	**CAMPO_DESCRIPCION**	CADENA(250)	Descripción de la condición, acción o aviso.
→	**CAMPO_COMPROBAR**	CADENA(1)	Indica cuando se comprueba la condición, acció o aviso.
→	**CAMPO_OBLIGATORIA**	CADENA(1)	Indica si es obligatoria. → "**S**" – Sí → "**N**" – No
→	**CAMPO_TIPO**	CADENA(1)	Tipo → "**C**" – Condición → "**A**" – Acción → "**W**" – Aviso
→	**CAMPO_REFSTMA**	NUMERICO	Identificador del sistema

Métodos para aplicación de los filtros → `obtenerCondicionesAccionesAvisos`

TrCondicionDocumento

trewa.bd.trapi.trapiui.tpo.TrCondicionDocumento

Clase que representa la información referente a condiciones asociadas a un documento.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**CONDICION**	TrCondicioon	Condición
→	**COMPROBAR**	String	Comprobación → "**G**" – La condición se produce al Generar el documento → "**I**" – La condición se produce al Incorporar el documento → "**T**" – La condición se produce al Generar e Incorporar el documento → "**V**" – La condición se produce al Visualizar el documento
→	**OBLIGATORIA**	String	Indica si la condición es o no obligatoria → "**S**" – La condición es obligatoria → "**N**" – La condición no es obligatoria

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso

→	**CAMPO_NOMBRE**	CADENA(50)	Nombre de la condición
→	**CAMPO_DESCRIPCION**	CADENA(250)	Descripción de la condición
→	**CAMPO_COMPROBAR**	CADENA(1)	Comprobación → "**G**" – La condición se produce al Generar el documento → "**I**" – La condición se produce al Incorporar el documento → "**T**" – La condición se produce al Generar e Incorporar el documento → "**V**" – La condición se produce al Visualizar el documento
→	**CAMPO_OBLIGATORIA**	CADENA(1)	Indica si la condición es o no obligatoria → "**S**" – La condición es obligatoria → "**N**" – La condición no es obligatoria
→	**CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema asociado a la condición

Métodos para aplicación de los filtros → `obtenerCondicionesDocumento`

TrCondicionTransicion

trewa.bd.trapi.trapiui.tpo.TrCondicionTransicion

Clase que representa la información referente a condiciones a evaluar en una transición.

El modelo de referencia w@nda define una condición de transición como "el conjunto de circustancias que se evalúan por el sistema cada vez que se intenta producir una transición en el procedimiento".

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**CONDICION**	TrCondicion	Condición
→	**COMPROBAR**	String	Comprobación → "**D**" – La condición se produce al Deshacer la transición → "**T**" – La condición se produce al Tramitar la transición → "**A**" – La condición se produce al Tramitar y Deshacer la transición → "**V**" – La condición se produce al Visualizar la transición
→	**OBLIGATORIA**	String	Indica si la condición es o no obligatoria → "**S**" – La condición es obligatoria → "**N**" – La condición no es obligatoria

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_NOMBRE**	CADENA(50)	Nombre de la condición
→ **CAMPO_DESCRIPCION**	CADENA(250)	Descripción de la condición
→ **CAMPO_COMPROBAR**	CADENA(1)	Comprobación → "**D**" – La condición se produce al Deshacer la transición → "**T**" – La condición se produce al Tramitar la transición → "**A**" – La condición se produce al Tramitar y Deshacer la transición → "**V**" – La condición se produce al Visualizar la transición (ver transiciones_posibles-tr_pr_transiciones_posibles)
→ **CAMPO_OBLIGATORIA**	CADENA(1)	Indica si la condición es o no obligatoria → "**S**" – La condición es obligatoria → "**N**" – La condición no es obligatoria
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema asociado a la condición

Métodos para aplicación de los filtros → obtenerCondicionesTransicion

TrDatoComponente

trewa.bd.trapi.trapiui.tpo.TrDatoComponente

Clase que representa otros datos necesarios para el componente.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFDATOCOMP**	TpoPK	Identificador del dato del componente
→ **COMPONENTE**	TrComponente	Componente
→ **ATRIBUTO**	String	Nombre del atributo del componente
→ **VALOR**	String	Valor del atributo del componente

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFDATOCOMP**	NUMERICO	Identifiador del dato del componente
→ **CAMPO_REFCOMP**	NUMERICO	Identificador del componente
→ **CAMPO_ATRIBUTO**	CADENA(32)	Nombre del atributo del componente
→ **CAMPO_VALOR**	CADENA(255)	Valor del atributo del componente

Métodos para aplicación de los filtros → `obtenerDatosComponente`

TrDatosContacto

trewa.bd.trapi.trapiui.tpo.TrDatosContacto

Clase que representa los datos de contacto de los interesados.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFDATOCONT**	`TpoPK`	Identificador del dato de contacto
→ **MUNICIPIO**	`TrMunicipio`	Municipio
→ **PAIS**	`TrPais`	País
→ **TIPOVIA**	`TrTipoVia`	Tipo de vía
→ **NOMBREVIA**	`String`	Nombre de la vía
→ **NUMERO**	`Integer`	Número del domicilio
→ **LETRA**	`String`	Letra del domicilio
→ **ESCALERA**	`String`	Escalera del domicilio
→ **PISO**	`Integer`	Piso del domicilio
→ **PUERTA**	`String`	Puerta del domicilio
→ **CODPOSTAL**	`Integer`	Código postal del domicilio
→ **TELEFONO**	`String`	Número(s) de teléfono de contacto
→ **TLFMOVIL**	`String`	Número(s) de teléfono móvil
→ **FAX**	`String`	Número(s) de fax
→ **EMAIL**	`String`	Correo electrónico
→ **PORDEFECTO**	`String`	Indica si tiene un interesado por defecto → "**S**" – Tiene interesado por defecto → "**N**" – No tiene interesado por defecto

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFDATOCONT**	`NUMERICO`	Identificador del dato de contacto
→ **CAMPO_CODMUNICIPIO**	`CADENA(3)`	Código del municipio
→ **CAMPO_MUNICIPIO**	`CADENA(32)`	Nombre del municipio
→ **CAMPO_CODPROVINCIA**	`CADENA(3)`	Código de la provincia

→	**CAMPO_PROVINCIA**	CADENA(32)	Nombre de la provincia
→	**CAMPO_CODPAIS**	CADENA(3)	Código del país
→	**CAMPO_PAIS**	CADENA(32)	Nombre del país
→	**CAMPO_TIPOVIA**	CADENA(10)	Típo de vía
→	**CAMPO_NOMBREVIA**	CADENA(100)	Nombre de la vía
→	**CAMPO_NUMERO**	NUMERICO	Número del domicilio
→	**CAMPO_LETRA**	CADENA(2)	Letra del domicilio
→	**CAMPO_ESCALERA**	CADENA(2)	Escalera del domicilio
→	**CAMPO_PISO**	NUMERICO	Piso del domicilio
→	**CAMPO_PUERTA**	CADENA(2)	Puerta del domicilio
→	**CAMPO_CODPOSTAL**	NUMERICO	Código postal del domicilio
→	**CAMPO_TELEFONO**	CADENA(25)	Número(s) de teléfono de contacto
→	**CAMPO_TLFMOVIL**	CADENA(25)	Número(s) de teléfono móvil
→	**CAMPO_FAX**	CADENA(25)	Número(s) de Fax
→	**CAMPO_EMAIL**	CADENA(64)	Correo electrónico
→	**CAMPO_PORDEFECTO**	CADENA(1)	Indica si tiene un interesado por defecto → "**S**" – Tiene interesado por defecto → "**N**" – No tiene interesado por defecto

Métodos para aplicación de los filtros → `obtenerDatosContactoInteresado`

TrDefProcedimiento

trewa.bd.trapi.trapiui.tpo.TrDefProcedimiento

Clase que representa la información referente a una definición de procedimientos en TREW@.

El modelo de referencia W@nda define la definición de procedimiento como "el resultado del modelado de un determinado procedimiento".

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFDEFPROC**	TpoPK	Identificador de la definición del procedimiento
→	**ABREVIATURA**	String	Abreviatura de la definición del procedimiento
→	**DESCRIPCION**	String	Descripción de la definición del procedimiento
→	**STMA**	TrSistema	Sistema
→	**INFORMAR**	String	Indica si se tendrá que informar al bus de este procedimiento y de los expedientes que se tramiten

			→ "**S**" – Sí → "**N**" – No
→	**VIGENTE**	String	Indica si el procedimiento está vigente. → "**S**" – Si está vigente → "**N**" – No está vigente
→	**CATEGORIA**	String	Indica la categoría del procedimiento → "**F**" – Familia → "**S**" – Subfamilia → "**P**" – Procedimiento
→	**DESCRIPCIONAMP**	String	Descripción ampliada del procedimiento.
→	**OTROSDATOS**	String	Indica si el procedimiento tiene otros datos. → "**S**" – Sí → "**N**" – No
→	**FAMILIA**	TrFamiliaSubfamilia	Familia
→	**SUBFAMILIA**	TrFamiliaSubfamilia	Subfamilia
→	**ORGANISMO**	TrOrganismo	Organismo a la que pertenece.
→	**ORGCOMPETENTE**	TrOrganismo	Organismo competente.
→	**ORGRESUELVE**	TrOrganismo	Organismo que resuelve.
→	**ORGTRAMITA**	TrOrganismo	Organismo que tramita.
→	**CODWANDA**	String	Valor en w@ndA del procedimiento
→	**COMENTARIOS**	String	Comentarios al procedimiento/familia/subfamilia

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFDEFPROC**	NUMERICO	Identificador de la definición del procedimiento.
→	**CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura de la definición del procedimiento
→	**CAMPO_DESCRIPCION**	CADENA(50)	Descripción de la definición del procedimiento
→	**CAMPO_REFSTMA**	NUMERICO	Identificador del sistema
→	**CAMPO_SISTEMA**	CADENA(10)	Nombre del sistema
→	**CAMPO_INFORMAR**	CADENA(1)	Indica si se tendrá que informar al bus de este procedimiento y de los expedientes que se tramiten → "**S**" – Sí → "**N**" – No
→	**CAMPO_VIGENTE**	CADENA(1)	Indica si el procedimiento está vigente. → "**S**" – Si está vigente → "**N**" – No está vigente
→	**CAMPO_CATEGORIA**	CADENA(1)	Indica la categoría del procedimiento → "**F**" – Familia → "**S**" – Subfamilia → "**P**" – Procedimiento
→	**CAMPO_DESCRIPCIONAMP**	CADENA(250)	Descripción ampliada del procedimiento.

→	**CAMPO_OTROSDATOS**	CADENA(1)	Indica si el procedimiento tiene otros datos. → "**S**" – Sí → "**N**" – No
→	**CAMPO_REFSUBFAMILIA**	NUMERICO	Identificador de la familia/subfamilia
→	**CAMPO_REFORGANISMO**	NUMERICO	Identificador del organismo a la que pertenece.
→	**CAMPO_NOMBREORG**	CADENA(32)	Nombre del organismo a la que pertenece.
→	**CAMPO_REFORGCOMP**	NUMERICO	Identificador del organismo competente.
→	**CAMPO_NOMBREORGCOMP**	CADENA(32)	Nombre del organismo competente.
→	**CAMPO_REFORGRES**	NUMERICO	Identificador del organismo que resuelve.
→	**CAMPO_NOMBREORGRES**	CADENA(32)	Nombre del organismo que resuelve.
→	**CAMPO_REFORGTRAM**	NUMERICO	Identificador del organismo que tramita.
→	**CAMPO_NOMBREORGTRAM**	CADENA(32)	Nombre del organismo que tramita.
→	**CAMPO_CODWANDA**	CADENA(15)	Valor en w@ndA del procedimiento
→	**CAMPO_COMENTARIOS**	CADENA(250)	Comentarios al procedimiento/familia/subfamilia

Métodos para aplicación de los filtros → obtenerDefProcedimientosDefinidos

TrDocumentoExpediente

trewa.bd.trapi.trapiui.tpo.TrDocumentoExpediente

Clase que representa la información asociada a un documento generado y/o incorporado a un expediente.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFDOCEXP**	TpoPK	Identificador del documento del expediente
→	**TIPODOC**	TrTipoDocumento	Tipo de documento
→	**FECHA**	Timestamp	Fecha de generación o incorporación
→	**PRESENTADO**	String	Para los documentos a "incorporar" indica si el documento se ha presentado o no. → "**S**" – El documento se ha presentado → "**N**" – El documento no se ha presentado
→	**CORRECTO**	String	Para los documentos a "incorporar" indica si el documento se ha presentado correcto o no. Esto no significa que no se pueda utilizar con los documentos generados. → "**S**" – El documento se ha presentado correcto → "**N**" – El documento no se ha presentado correctamente
→	**FECHALIMITE**	Timestamp	Fecha límite de presentación. (Sólo para los documentos a

			incorporar)
→	**ESTADO**	String	Estado en el que se encuentra el documento. (Este dato no tiene sentido para documentos incorporados) → "**R**" – El documento se encuentra en realización → "**E**" – El documento se encuentra pendiente de firma → "**F**" – El documento se encuentra firmado → "**D**" – El documento está descartado → "**T**" – El documento está terminado → "**V**" – El documento está versionado
→	**OBSERVACIONES**	String	Observaciones del documento
→	**REFEXPXFAS**	TpoPK	Identificador del paso en la evolución del expediente en el que se generó y/o incorporó.
→	**USUARIO**	String	Usuario que generó y/o incorporó el documento al expediente
→	**FECHAFIRMA**	Timestamp	Fecha de firma. Sólo para documentos generados
→	**REGISTRODOC**	TrRegistroDocumento	Datos de registro que existen para un documento
→	**FECHAFIN**	Timestamp	Fecha de terminación, anulación, descarte o firma del documento.
→	**FECHAARCHIVO**	Timestamp	Fecha de archivo del documento.
→	**FECHACADUCIDAD**	Timestamp	Fecha de caducidad del documento.
→	**INFORMADO**	String	Indica si se ha informado al bus del documento. → "**S**" – Si se ha informado → "**N**" – No se ha informado
→	**REUTILIZABLE**	String	Indica si el documento es reutilizable → "**S**" – Sí → "**N**" – No
→	**FIRMADIG**	String	Indica si el documento es firmable digitalmente. → "**S**" – Si es firmable digitalmente → "**N**" – No es firmable digitalmente
→	**VERSION**	String	Versión del documento.
→	**REFDOCEXPVERS**	TpoPK	Identificador del documento al que versiona
→	**MODOGEN**	String	Modo de generación del documento → "**R**" – Reports Server Oracle → "**P**" – Java PDF → "**O**" – OpenOffice Bean
→	**REFFASE**	TpoPK	Identificador de la fase
→	**NOMBREFASE**	String	Nombre de la fase
→	**REFDEFPROC**	TpoPK	Identificador de la definición del procedimiento
→	**NOMBREUSU**	String	Nombre y apellidos del usuario
→	**CODHASH**	String	Código de identificación de documento para notific@, port@firma
→	**REFEXPEDIENTE**	TpoPK	Identificador del expediente
→	**REFWARDA**	String	Identificador del documento en w@rdA
→	**REFWARDAANX**	String	Identificador del anexo del documento en w@rdA
→	**REFCOMPONENTE**	TpoPK	Identificador del componente en el que está guardado.

→	**NOMBREFICHERO**	String	Nombre del fichero del documento físico.
→	**FORMATO**	String	Tipo de formato del fichero del documento.

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFDOCEXP**	NUMÉRICO	Identificador del documento del expediente
→	**CAMPO_ETIQUETA**	CADENA(11)	Abreviatura del tipo de documento
→	**CAMPO_NOMBRE**	CADENA(50)	Nombre del tipo de documento
→	**CAMPO_DESCRIPCION**	CADENA(250)	Descripción del tipo de documento
→	**CAMPO_FECHA**	DATE	Fecha de generación o incorporación
→	**CAMPO_PRESENTADO**	CADENA(1)	Para los documentos a "incorporar" indica si el documento se ha presentado o no. → "**S**" – El documento se ha presentado → "**N**" – El documento no se ha presentado
→	**CAMPO_CORRECTO**	CADENA(2)	Para los documentos a "incorporar" indica si el documento se ha presentado correcto o no. Esto no significa que no se pueda utilizar con los documentos generados. → "**S**" – El documento se ha presentado correcto → "**N**" – El documento no se ha presentado correctamente
→	**CAMPO_FECHALIMITE**	DATE	Fecha límite de presentación. (Sólo para los documentos a incorporar)
→	**CAMPO_ESTADO**	CADENA(1)	Estado en el que se encuentra el documento. (Este dato no tiene sentido para documentos incorporados) → "**R**" – El documento se encuentra en realización → "**E**" – El documento se encuentra pendiente de firma → "**F**" – El documento se encuentra firmado → "**D**" – El documento está descartado → "**T**" – El documento está terminado → "**V**" – El documento está versionado
→	**CAMPO_OBSERVACIONES**	CADENA(250)	Observaciones del documento
→	**CAMPO_ENTRADASALIDA**	CADENA(2)	Indica cómo está definido el documento en TREW@ → "**E**" – Entrada → "**S**" – Salida → "**ES**" – Entrada / Salida
→	**CAMPO_INCGEN**	CADENA(1)	Indica cómo está definido el documento → "**I**" – Documento a Incorporar al expediente → "**G**" – Documento a Generar
→	**CAMPO_USUARIO**	CADENA(10)	Usuario que generó y/o incorporó el documento al expediente
→	**CAMPO_FECHAFIRMA**	DATE	Fecha de la firma del documento
→	**CAMPO_REFEXPXFAS**	NUMÉRICO	Identificador del paso en la evolución del expediente en el que se generó y/o incorporó.
→	**CAMPO_FECHAFIN**	DATE	Fecha de terminación, anulación, descarte o firma del documento.

→	**CAMPO_FECHAARCHIVO**	DATE	Fecha de archivo del documento.
→	**CAMPO_FECHACADUCIDAD**	DATE	Fecha de caducidad del documento.
→	**CAMPO_INFORMADO**	CADENA(1)	Indica si se ha informado al bus del documento. → "**S**" – Si se ha informado → "**N**" – No se ha informado
→	**CAMPO_VERSIONABLE**	CADENA(1)	Indica si el documento es versionable → "**S**" – Sí → "**N**" – No
→	**CAMPO_REGISTRABLE**	CADENA(1)	Indica si el documento es registrable → "**S**" – Sí → "**N**" – No
→	**CAMPO_NOTIFICABLE**	CADENA(1)	Indica si el documento es notificable → "**S**" – Sí → "**N**" – No
→	**CAMPO_REUTILIZABLE**	CADENA(1)	Indica si el documento es reutilizable → "**S**" – Sí → "**N**" – No
→	**CAMPO_FIRMADIG**	CADENA(1)	Indica si el documento es firmable digitalmente. → "**S**" – Si es firmable digitalmente → "**N**" – No es firmable digitalmente
→	**CAMPO_TIPOFIRMA**	CADENA(1)	Tipo de firma para el documento → "**C**" – Firma en cascada → "**P**" – Firma en paralelo
→	**CAMPO_VERSION**	CADENA(9)	Versión del documento.
→	**CAMPO_REFDOCEXPVERS**	NUMERICO	Identificador del documento al que versiona
→	**CAMPO_MULTIPLE**	CADENA(1)	Indica si el tipo de documento es múltiple → "**S**" – Sí → "**N**" – No
→	**CAMPO_MODOGEN**	CADENA(1)	Modo de generación del documento → "**R**" – Report Server Oracle → "**P**" – Generación PDF → "**O**" – OpenOffice Bean
→	**CAMPO_TEXTOAUXILIAR**	CADENA(50)	Texto auxiliar del tipo de documento
→	**CAMPO_REFFASE**	NUMERICO	Identificador de la fase
→	**CAMPO_NOMBREFASE**	CADENA(50)	Nombre de la fase
→	**CAMPO_REFDEFPROC**	NUMERICO	Identificador de la definición del procedimiento
→	**CAMPO_NOMBREUSU**	CADENA(98)	Nombre y apellidos del usuario
→	**CAMPO_CODHASH**	CADENA(20)	Código de identificación de documento para notific@, port@firma
→	**CAMPO_REFTIPODOC**	NUMERICO	Identificador del tipo de documento
→	**CAMPO_REFEXPEDIENTE**	NUMERICO	Identificador del expediente
→	**CAMPO_REFWARDA**	CADENA(50)	Identificador del documento en w@rdA
→	**CAMPO_REFWARDAANX**	CADENA(50)	Identificador del anexo del documento en w@rdA
→	**CAMPO_REFCOMPONENTE**	NUMERICO	Identificador del componente en el que está guardado.

→	**CAMPO_NOMBREFICHERO**	CADENA(64)	Nombre del fichero del documento físico.
→	**CAMPO_FORMATO**	CADENA(128)	Formato del fichero del documento.
→	**CAMPO_REFSISTEMA**	NUMÉRICO	Identificador del sistema
→	**CAMPO_CODSISTEMA**	CADENA(10)	Código del sistema
→	**CAMPO_DESCRIPCION**	CADENA(128)	Descripción del sistema
→	**CAMPO_CONEXIONBUS**	CADENA(1)	Indica si se tiene conexión al bus. → '**S**' – Sí → '**N**' – No → '**O'** – Opcional
→	**CAMPO_REFCOMPINF**	NUMÉRICO	Identificador del componente a informar.
→	**CAMPO_REFCOMPAVI**	NUMÉRICO	Identificador del componente del @visador.
→	**CAMPO_REFCOMPGUAR**	NUMÉRICO	Identificador del componente que guarda los documentos.
→	**CAMPO_REFCOMPARCH**	NUMÉRICO	Identificador del componente que archiva.
→	**CAMPO_REFCOMPREG**	NUMÉRICO	Identificador del componente que registra los documentos.
→	**CAMPO_REFCOMPFIRMA**	NUMÉRICO	Identificador del componente que firma los documentos.
→	**CAMPO_REFCOMPNOTI**	NUMÉRICO	Identificador del componente que notifica los documentos.

Métodos para aplicación de los filtros → `obtenerDocumentosExpediente`

TrDocumentoPermitido

trewa.bd.trapi.trapiui.tpo.TrDocumentoPermitido

Clase que representa la información referente a las tareas de tipo "GENERAR_DOCUMENTO" o "INCORPORAR_DOCUMENTO" definidas en una fase de un procedimiento.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFDOCPER**	TpoPK	Identificador del documento permitido
→	**TIPODOC**	TrTipoDocumento	Tipo de documento
→	**OBLIGATORIO**	String	Indica si el documento es obligatorio generarlo o incorporarlo. → "**S**" – El documento es obligatorio → "**N**" – El documento no es obligatorio
→	**PERMISO**	String	Permiso que tiene el usuario sobre el documento → "**G**" – Permiso para generar → "**E**" – Permiso para editar → "**F**" – Permiso para firmar → "**I**" – Permiso para incorporar → "**T**" – Todos
→	**INFORMAR**	String	Indica si se informa al bus del documento → "**S**" – Sí → "**N**" – No
→	**ETIQUETA**	String	Texto con el que se etiqueta el documento permitido en la fase
→	**DESCRIPCION**	String	Descripción del documento permitido
→	**ETIQLARGA**	String	Etiqueta larga para el documento permitido
→	**PLAZO**	TrPlazo	Plazo de realización
→	**ORDEN**	Integer	Orden del documento permitido

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFDOCPER**	NUMÉRICO	Identificador del documento permitido
→	**CAMPO_REFTIPODOC**	NUMÉRICO	Identificador del tipo de documento
→	**CAMPO_ETIQUETA**	CADENA(11)	Abreviatura del tipo de documento
→	**CAMPO_NOMBRE**	CADENA(50)	Nombre del tipo de documento
→	**CAMPO_DESCRIPCION**	CADENA(250)	Descripción del tipo de documento
→	**CAMPO_OBLIGATORIO**	CADENA(1)	Indica si el documento es obligatorio generarlo o incorporarlo.

			→ "**S**" – El documento es obligatorio
			→ "**N**" – El documento no es obligatorio
→	**CAMPO_ENTRADASALIDA**	CADENA(2)	Indica cómo está definido el documento en TREW@
			→ "**E**" – Entrada
			→ "**S**" – Salida
			→ "**ES**" – Entrada / Salida
→	**CAMPO_INCGEN**	CADENA(1)	Indica cómo está definido el documento
			→ "**I**" – Documento a Incorporar al expediente
			→ "**G**" – Documento a Generar
→	**CAMPO_PERMISO**	CADENA(1)	Permiso que tiene el usuario sobre el documento
			→ "**G**" – Permiso para generar
			→ "**E**" – Permiso para editar
			→ "**F**" – Permiso para firmar
			→ "**I**" – Permiso para incorporar
			→ "**T**" – Todos
→	**CAMPO_MULTIPLE**	CADENA(1)	Indica si el tipo de documento es múltiple o no.
			→ "**S**" – Tipo de documento es múltiple
			→ "**N**" – Tipo de documento no es múltiple
→	**CAMPO_TEXTOAUXILIAR**	CADENA(50)	Texto de libre uso para las aplicaciones que utilizan TREW@
→	**CAMPO_INFORMAR**	CADENA(1)	Indica si se informa al bus del documento
			→ "**S**" – Sí
			→ "**N**" – No
→	**CAMPO_ETIQUETADOC**	CADENA(11)	Etiqueta del documento permitido
→	**CAMPO_DESCRIPCIONDOC**	CADENA(50)	Descripción del documento permitido
→	**CAMPO_ETIQLARGA**	CADENA(25)	Etiqueta larga para el documento permitido
→	**CAMPO_UNIDAD**	CADENA(1)	Unidad de tiempo
			→ "**D**" – Días
			→ "**M**" – Meses
			→ "**A**" – Años
→	**CAMPO_NUMUNIDADES**	NUMERICO	Número de unidades
→	**CAMPO_DESCFECHALIMITE**	CADENA(100)	Descripción de la fecha límite de realización
→	**CAMPO_ORDEN**	NUMERICO	Orden del documento permitido

Métodos para aplicación de los filtros → obtenerDocumentosPermitidos

TrEmpleado

trewa.bd.trapi.trapiui.tpo.TrEmpleado

Clase que representa la información referente a los usuarios que ocupan los puestos de trabajo en los organismos definidos.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **USUARIO**	`TrUsuario`	Usuario
→ **ORGANISMO**	`TrOrganismo`	Organismo
→ **PTOTRABAJO**	`TrPuestoTrabajo`	Puesto de trabajo
→ **TIPO**	`String`	Tipo de empleado → "**F**" – Funcionario → "**L**" – Laboral
→ **FECHANOMBRAMIENTO**	`Timestamp`	Fecha de ocupación del puesto por el usuario.
→ **FECHACESE**	`Timestamp`	Fecha de baja del usuario en el puesto
→ **TRATAMIENTO**	`String`	Texto de tratamiento para los documentos (D., Dª, Ilmo., etc.)
→ **TXTREFERENCIA**	`String`	Texto de referencia para el usuario.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_USUARIO**	`CADENA(10)`	Código del usuario
→ **CAMPO_REFORGANISMO**	`NUMERICO`	Identificador del organismo
→ **CAMPO_ORGANISMO**	`CADENA(32)`	Nombre del organismo
→ **CAMPO_CODPTOTRAB**	`CADENA(10)`	Código del puesto de trabajo
→ **CAMPO_PTOTRABAJO**	`CADENA(32)`	Nombre del puesto de trabajo
→ **CAMPO_TIPO**	`CADENA(1)`	Tipo de empleado → "**F**" – Funcionario → "**L**" – Laboral
→ **CAMPO_FECHANOMBRAMIENTO**	`DATE`	Fecha de ocupación del puesto por el usuario.
→ **CAMPO_FECHACESE**	`DATE`	Fecha de baja del usuario en el puesto
→ **CAMPO_TRATAMIENTO**	`CADENA(64)`	Texto de tratamiento para los documentos (D., Dª, Ilmo., etc.)
→ **CAMPO_TXTREFERENCIA**	`CADENA(10)`	Texto de referencia para el usuario.

Métodos para aplicación de los filtros → `obtenerEmpleados`

TrEnviarA

trewa.bd.trapi.trapiui.tpo.TrEnviarA

Clase que representa la información referente a un paso para llevar a un expediente a una situación concreta.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFTRANSICION**	TpoPK	Identificador de transición
→	**REFFASE**	TpoPK	Identificador de fase en la que se quedaría el expediente
→	**REFFASEPADRE**	TpoPK	Identificador de fase de la que parte si es un flujo reutilizable

TrEvolucionExpediente

trewa.bd.trapi.trapiui.tpo.TrEvolucionExpediente

Clase que representa la información referente a situaciones por las que pasa un expediente.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFEXPXFAS**	TpoPK	Identificador del paso en la evolución
→	**FASE**	TrFase	Fase o situación
→	**FECHAENTRADA**	Timestamp	Fecha en la que se entró en la fase
→	**FECHASALIDA**	Timestamp	Fecha en la que se salió de la fase
→	**FECHALIMITE**	Timestamp	Fecha límite de salida de la fase
→	**USUARIO**	String	Usuario que lo dejó en la fase
→	**USUSARIOBLQ**	String	Usuario que lo tiene bloqueado en la fase
→	**OBSERVACIONES**	String	Observaciones del usuario que lo dejó en la fase
→	**REFEXPXFASPADRE**	TpoPK	Para los pasos pertenecientes a módulos reutilizables, indica el paso que dejó al expediente en la fase que representa a dicho módulo.
→	**TRANSICION**	TrTransicion	Transición
→	**INFORMADABUS**	String	Indica si se informa al bus de la entrada en la fase

			→ "**S**" – Sí
			→ "**N**" – No
→	**INFORMADAFINBUS**	String	Indica si se informa al bus de la salida de la fase
			→ "**S**" – Sí
			→ "**N**" – No
→	**REFFASEPADRE**	TpoPk	Identificador de la fase padre
→	**NOMBREUSU**	String	Nombre y apellidos del usuario
→	**NOMBREUSUBLQ**	String	Nombre y apellidos del usuario que bloquea
→	**REFDEFPROC**	TpoPK	Identificador de la definición del procedimiento
→	**ABIERTAEVENTO**	String	Indica si la fase viene propagada de un evento
			→ "**S**" – Sí
			→ "**N**" – No

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFEXPXFAS**	NUMÉRICO	Identificador del paso en la evolución
→	**CAMPO_REFFASE**	NUMÉRICO	Identificador de la fase o situación
→	**CAMPO_FASE**	CADENA(50)	Nombre de la fase
→	**CAMPO_METAFASE**	CADENA(50)	Nombre de la metafase
→	**CAMPO_FECHAENTRADA**	DATE	Fecha en la que se entró en la fase
→	**CAMPO_FECHASALIDA**	DATE	Fecha en la que se salió de la fase
→	**CAMPO_FECHALIMITE**	DATE	Fecha límite de salida de la fase
→	**CAMPO_USUARIO**	CADENA(10)	Usuario que lo dejó en la fase
→	**CAMPO_USUARIOBLQ**	CADENA(10)	Usuario que lo tiene bloqueado en la fase
→	**CAMPO_OBSERVACIONES**	CADENA(250)	Observaciones del usuario que lo dejó en la fase
→	**CAMPO_REFTRANSICION**	NUMÉRICO	Identificador de la transición
→	**CAMPO_DESCTRANSICION**	CADENA(50)	Descripción de la transición
→	**CAMPO_REFTIPOACTO**	NUMÉRICO	Identificador del tipo de acto administrativo que representa la transición
→	**CAMPO_REFEXPXFASPADRE**	NUMÉRICO	Para los pasos pertenecientes a módulos reutilizables, indica el paso que dejó al expediente en la fase que representa a dicho módulo.
→	**CAMPO_TIPOTRANS**	CADENA(2)	Indica el tipo de la transición
			→ "**N**" – Transición normal
			→ "**D**" – Transición de divisisón
			→ "**U**" – Transición de unión
			→ "**ES**" – Evento que provoca salida
			→ "**EN**" – Evento que no provoca la salida
→	**CAMPO_REFFASEPADRE**	NUMÉRICO	Identificador de la fase padre
→	**CAMPO_DESCFASE**	CADENA(250)	Descripción de la fase
→	**CAMPO_NOMBREUSU**	CADENA(98)	Nombre y apellidos del usuario
→	**CAMPO_NOMBREUSUBLQ**	CADENA(98)	Nombre y apellidos del usuario que bloquea
→	**CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento

→	**CAMPO_ABIERTAEVENTO**	CADENA(1)	Indica si la fase viene propagada de un evento → "**S**" – Sí → "**N**" – No
→	**CAMPO_INFORMADABUS**	CADENA(1)	Indica si se informa al bus de la entrada en la fase → "**S**" – Sí → "**N**" – No
→	**CAMPO_INFORMADAFINBUS**	CADENA(1)	Indica si se informa al bus de la salida de la fase → "**S**" – Sí → "**N**" – No
→	**CAMPO_TEXTOAUXILIAR**	CADENA(500)	Texto auxiliar de la fase

Métodos para aplicación de los filtros → obtenerEvolucionExpediente

TrExpediente

trewa.bd.trapi.trapiui.tpo.TrExpediente

Clase que representa la información referente a un expediente.

El modelo de referencia w@nda define un expediente como "cada materialización de un determinado procedimiento".

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFEXP**	TpoPK	Identificador del expediente
→	**STMA**	TrSistema	Sistema
→	**TIPOEXP**	TrTipoExpediente	Tipo de expediente
→	**DEFPROC**	TrDefProcedimiento	Definición del procedimiento
→	**FECHAALTA**	Timestamp	Fecha de alta
→	**NUMEXP**	String	Número del expediente en la aplicación "cliente"
→	**TITULOEXP**	String	Título del expediente en la aplicación "cliente"
→	**OBSERVACIONES**	String	Observaciones
→	**ORGANISMO**	TrOrganismo	Organismo al que pertenece
→	**ORGENVIA**	TrOrganismo	Organismo que lo ha enviado
→	**OTROSDATOS**	String	Indica si tiene otros datos → "**S**" – Sí → "**N**" – No
→	**FECHAARCHIVO**	Timestamp	Fecha de archivo del expediente
→	**INFORMADO**	String	Indica si se ha informado del expediente al bus

			→ "**S**" – Sí
			→ "**N**" – No
→	**REFCOMPONENTE**	TpoPK	Identificador del componente en el que está archivado
→	**URLWANDA**	String	Url del expediente en w@ndA

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFEXP**	NUMÉRICO	Identificador del expediente
→	**CAMPO_NUMEXP**	CADENA(30)	Número del expediente
→	**CAMPO_TITULOEXP**	CADENA(250)	Título del expediente
→	**CAMPO_FECHAALTA**	DATE	Fecha de alta del expediente
→	**CAMPO_OBSERVACIONES**	CADENA(250)	Observaciones del expediente
→	**CAMPO_REFORGANISMO**	NUMÉRICO	Identificador del organismo al que pertenece el expediente
→	**CAMPO_NOMBREORGANISMO**	CADENA(32)	Nombre del organismo al que pertenece el expediente
→	**CAMPO_URLWANDA**	CADENA(255)	Url del expediente en w@ndA
→	**CAMPO_REFORGENVIA**	NUMÉRICO	Identificador del organismo que envía el expediente
→	**CAMPO_NOMBREORGENVIA**	CADENA(32)	Nombre del organismo que envía el expediente
→	**CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento
→	**CAMPO_DESCDEFPROC**	CADENA(50)	Descripción de la definición del procedimiento
→	**CAMPO_REFTIPOEXP**	NUMÉRICO	Identificador del tipo de expediente
→	**CAMPO_DESCTIPOEXP**	CADENA(50)	Descripción del tipo de expediente
→	**CAMPO_REFSTMADEFPROC**	NUMÉRICO	Identificador del sistema de la definición del procedimiento
→	**CAMPO_CODSTMADEFPROC**	CADENA(10)	Código o nombre del sistema de la definición del procedimiento
→	**CAMPO_OTROSDATOS**	CADENA(1)	Indica si el expediente tiene otros datos
			→ "**S**" – Sí
			→ "**N**" – No
→	**CAMPO_FECHAARCHIVO**	DATE	Fecha de archivo del expediente
→	**CAMPO_REFCOMPONENTE**	NUMÉRICO	Identificador del componente dónde está archivado el expediente
→	**CAMPO_INFORMADO**	CADENA(1)	Indica si se ha informado al bus del expediente
			→ "**S**" – Sí
			→ "**N**" – No
→	**CAMPO_ABREVTIPOEXP**	CADENA(10)	Abreviatura del tipo de expediente
→	**CAMPO_ABREVDEFPROC**	CADENA(10)	Abreviatura de la definición del procedimiento

Métodos para aplicación de los filtros → `obtenerExpedientesCaducados`

TrExpedienteCaducado

trewa.bd.trapi.trapiui.tpo.TrExpedienteCaducado

Clase que representa la información referente a un expediente caducado.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**EXPEDIENTE**	`TrExpediente`	Expediente
→	**CADUCIDADEXP**	`TrCaducidadExpediente`	Caducidad del expediente
→	**FASE**	`TrFase`	Fase que ha superado el tiempo máximo de estancia

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFEXP**	`NUMÉRICO`	Identificador del expediente
→	**CAMPO_REFCADEXP**	`NUMÉRICO`	Identificador de la caducidad
→	**CAMPO_REFFASE**	`NUMÉRICO`	Identificador de la fase que ha caducado
→	**CAMPO_REFSTMA**	`NUMÉRICO`	Identificador del sistema al que pertenece el expediente

Métodos para aplicación de los filtros → `obtenerExpedientesCaducados`

TrExplorador

trewa.bd.trapi.trapiui.tpo.TrExplorador

Clase que representa la información asociada al estado en el que se encuentra un expedientes.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**EXPEDIENTE**	`TrExpediente`	Expediente
→	**FASE**	`TrFase`	Fase en la que se encuentra el expediente
→	**PENDIENTES**	`String`	Indica si el expediente se encuentra pendiente de alguna tarea o cambio de fase → '**S**' – Sí → '**N**' – No
→	**CADUCADOS**	`String`	Indica si el expediente se encuentra caducado, ya sea por algún plazo definido en el procedimiento o porque ha vencido el plazo de estancia en una fase → '**S**' – Sí → '**N**' – No
→	**RESERVADOS**	`String`	Indica si el expediente se encuentra reservado por el usuario, ya sea una fase del expediente o el expediente completo → '**S**' – Sí → '**N**' – No

TrFamiliaSubfamilia

trewa.bd.trapi.trapiui.tpo.TrFamiliaSubfamilia

Clase que representa la información de una familia o subfamilia de un procedimiento.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFDEFPROC**	`TpoPK`	Identificador de la definición del procedimiento
→	**ABREVIATURA**	`String`	Abreviatura de la definición del procedimiento.
→	**DESCRIPCION**	`String`	Descripción de la definición del procedimiento.

→	**CATEGORIA**	String	Indica la categoría del procedimiento → '**F**' – Familia → '**S**' – Subfamilia → '**P**' – Procedimiento
→	**DESCRIPCIONAMP**	String	Descripción ampliado
→	**CODWANDA**	String	Valor en w@ndA del procedimiento

TrFase

trewa.bd.trapi.trapiui.tpo.TrFase

Clase que representa la información asociada a una fase.

El modelo de referencia w@nda define una fase como "un conjunto homogéneo de tareas desde el punto de vista del tramitador, cuya sucesión en el tiempo componen un determinado procedimiento una vez ha sido modelado y constituyen la unidad elemental de tramitación dentro del mismo".

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFFASE**	TpoPK	Identificador de la fase
→	**NOMBRE**	String	Nombre de la fase
→	**DESCRIPCION**	String	Descripción de la fase
→	**METAFASE**	TrMetafase	Metafase
→	**DEFPROC**	TrDefProcedimiento	Definición del procedimiento
→	**STMA**	TrSistema	Sistema
→	**INFORMARBUS**	String	Indica si se informa al bus de la entrada en la fase → '**S**' – Sí → '**N**' – No
→	**TEXTOAUXILIAR**	String	Texto auxiliar de la fase
→	**ORDEN**	Integer	Orden de la fase

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFFASE**	NUMÉRICO	Identificador de la fase
→	**CAMPO_NOMBRE**	CADENA(50)	Nombre de la fase
→	**CAMPO_DESCRIPCION**	CADENA(250)	Descripción de la fase

→	**CAMPO_METAFASE**	CADENA(50)	Nombre de la metafase
→	**CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema
→	**CAMPO_TEXTOAUXILIAR**	CADENA(500)	Texto auxiliar para la fase
→	**CAMPO_ORDENFASE**	NUMÉRICO	Orden de la fase
→	**CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento
→	**CAMPO_DESCDEFPROC**	CADENA(50)	Descripción de la definición del procedimiento
→	**CAMPO_REFMETAFASE**	NUMÉRICO	Identificador de la metafase
→	**CAMPO_DESCMETAFASE**	CADENA(250)	Descripción de la metafase
→	**CAMPO_ORDENMETAFASE**	NUMÉRICO	Orden de la metafase
→	**CAMPO_INFORMARBUSFASE**	CADENA(1)	Indica si se informa al bus de la entrada en la fase → '**S**' – Sí → '**N**' – No
→	**CAMPO_CAMPO_INFORMARBUS METAFASE**	CADENA(1)	Indica si se informa al bus de la entrada en la metafase → '**S**' – Sí → '**N**' – No

Métodos para aplicación de los filtros → `obtenerDatosFase, obtenerFasesFinTransicion`

TrFaseActualExpediente

trewa.bd.trapi.trapiui.tpo.TrFaseActualExpediente

Clase que representa una situación en la que se encuentra un expediente.

El modelo de referencia w@nda define una fase actual como "la fase activa en estos momentos en el tramitador de procedimientos para el expediente de todas las posibles definidas en el procedimiento".

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**FASE**	TrFase	Fase
→	**FECHAENTRADA**	Timestamp	Fecha en la que se entró en la fase
→	**FECHALIMITE**	Timestamp	Fecha de salida de la fase
→	**USUARIO**	String	Usuario que lo dejó en la fase
→	**USUARIOBLQ**	String	Usuario que lo tiene bloqueado en la fase
→	**OBSERVACIONES**	String	Observaciones del usuario que lo dejó en la fase
→	**REFEXPXFAS**	TpoPK	Identificador de la situación actual de la evolución total del expediente. Es el id del paso concreto de la tramitación.
→	**REFFASEPADRE**	TpoPK	Para el caso en que la situación pertenezca a un módulo

			reutilizable, indica la fase que representa el módulo reutilizable
→	**REFDEFPROC**	TpoPK	Para el caso que la fase pertenezca a un módulo reutilizable, indica cuál es dicho módulo.
→	**REFEXPXFASPADRE**	TpoPK	Identificador del expediente en fase padre
→	**REFTRANSICION**	TpoPK	Identificador de la transición
→	**DESCTRANSICION**	String	Descripción de la transición
→	**REFTIPOACTO**	TpoPK	Identificador del tipo de acto
→	**TIPOTRANS**	String	Tipo de la transición. → "**N**" – Transición normal → "**D**" – Transición de divisisón → "**U**" – Transición de unión → "**ES**" – Evento que provoca salida → "**EN**" – Evento que no provoca la salida
→	**NOMBREUSU**	String	Nombre y apellidos del usuario
→	**NOMBREUSUBLQ**	String	Nombre y apellidos del usuario que bloquea
→	**ABIERTAEVENTO**	String	Indica si la fase viene propagada de un evento → "**S**" – Sí → "**N**" – No
→	**INFORMADABUS**	String	Indica si se ha informado al bus → "**S**" – Sí → "**N**" – No

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFFASE**	NUMÉRICO	Identificador de la fase o situación
→	**CAMPO_NOMBFASE**	CADENA(50)	Nombre de la fase
→	**CAMPO_REFMETAFASE**	NUMÉRICO	Identificador de la metafase
→	**CAMPO_NOMBMETAFASE**	CADENA(50)	Nombre de la metafase
→	**CAMPO_FECHAENTRADA**	DATE	Fecha en la que se entró en la fase
→	**CAMPO_FECHALIMITE**	DATE	Fecha de salida de la fase
→	**CAMPO_USUARIO**	CADENA(10)	Usuario que lo dejó en la fase
→	**CAMPO_USUARIOBLQ**	CADENA(10)	Usuario que lo tiene bloqueado en la fase
→	**CAMPO_OBSERVACIONES**	CADENA(250)	Observaciones del usuario que lo dejó en la fase
→	**CAMPO_REFEXPXFAS**	NUMÉRICO	Identificador de la situación actual en la evolución total del expediente. Es el identificador del paso concreto en la tramitación.
→	**CAMPO_REFFASEPADRE**	NUMÉRICO	Identificador de la fase padre
→	**CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento
→	**CAMPO_REFEXPXFASPADRE**	NUMÉRICO	Identificador del expediente en fase padre
→	**CAMPO_REFTRANSICION**	NUMÉRICO	Identificador de la transición
→	**CAMPO_DESCTRANSICION**	CADENA(50)	Descripción de la transición
→	**CAMPO_REFTIPOACTO**	NUMÉRICO	Identificador del tipo de acto
→	**CAMPO_TIPOTRANS**	CADENA(2)	Tipo de la transición.

			→ "**N**" – Transición normal
			→ "**D**" – Transición de divisisón
			→ "**U**" – Transición de unión
			→ "**ES**" – Evento que provoca salida
			→ "**EN**" – Evento que no provoca la salida
→	**CAMPO_DESCFASE**	CADENA(250)	Descripción de la fase
→	**CAMPO_NOMBREUSU**	CADENA(98)	Nombre y apellidos del usuario
→	**CAMPO_NOMBREUSUBLQ**	CADENA(98)	Nombre y apellidos del usuario que bloquea
→	**CAMPO_ABIERTAEVENTO**	CADENA(1)	Indica si la fase viene propagada de un evento
			→ "**S**" – Sí
			→ "**N**" – No
→	**CAMPO_TEXTOAUXILIAR**	CADENA(500)	Texto auxiliar de la fase
→	**CAMPO_INFORMADABUS**	CADENA(1)	Indica si se informa al bus

Métodos para aplicación de los filtros → obtenerFaseActualExpediente

TrFirmaDocumentoExpediente

trewa.bd.trapi.trapiui.tpo.TrFirmaDocumentoExpediente

Clase que representa la información referente a los datos de la firma de un documento.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**FECHA**	Timestamp	Fecha de firma
→	**FIRMANTEPPAL**	TrFirmante	Firmante principal
→	**USUARIO**	String	Clave del usuario
→	**USUARIODIG**	String	Usuario al que pertenece la unidad orgánica
→	**FIRMANTEDELEG**	TrFirmante	Firmante delegado
→	**FIRMANTESUST**	TrFirmante	Firmante sustituto
→	**EDITABLE**	String	Indica si es editable el pie de firma
			→ 'S' – El pie de firma es editable
			→ 'N' – El pie de firma no es editable
→	**TXTPIE**	String	Texto que va antes de la rúbrica de la firma en el pie de firma
→	**TXTFDO**	String	Texto que va después de la rúbrica de la firma en el pie de firma
→	**USUARIODIGANT**	String	Código del usuario digital de la firma anterior
→	**REFUNIORGANT**	TpoPK	Identificador de la unidad orgánica de la firma anterior
→	**PTOTRABANT**	String	Código del puesto de trabajo de la firma anterior

→	**HASH**	String	Código de petición para port@firmas
→	**FECHAFIRMADIG**	Timestamp	Fecha de la firma digital
→	**CODTRANSACCION**	String	Código de transacción de firma
→	**PKCS7**	byte[]	Recibo de la firma digital
→	**ORDEN**	Integer	Orden de la firma
→	**FIRMA**	TrFirma	Firma del usuario digital
→	**PUESTO**	String	Nombre del puesto de trabajo del usuario digital
→	**ORGANISMO**	String	Nombre del organismo del usuario digital
→	**NOMBREUSUDIGI**	String	Nombre y apellidos del usuario digital.

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_FECHA**	DATE	Fecha de firma
→	**CAMPO_PTOTRABAJO**	CADENA(10)	Código que identifica el puesto de trabajo
→	**CAMPO_UNIORGANICA**	NUMÉRICO	Código que identifica la unidad orgánica
→	**CAMPO_USUARIO**	CADENA(10)	Clave de usuario
→	**CAMPO_USUARIODIG**	CADENA(10)	Usuario al que pertenece la unidad orgánica
→	**CAMPO_PTOTRABDELG**	CADENA(10)	Código que identifica el puesto de trabajo del delegado
→	**CAMPO_UNIORGDELEG**	NUMÉRICO	Código que identifica la unidad orgánica del delegado
→	**CAMPO_PTOTRABSUST**	CADENA(10)	Código que identifica el puesto de trabajo del sustituto
→	**CAMPO_UNIORGSUST**	NUMÉRICO	Código que identifica la unidad orgánica del sustituto
→	**CAMPO_EDITABLE**	CADENA(1)	Indica si es editable el pie de firma → '**S**' – El pie de firma es editable → '**N**' – El pie de firma no es editable
→	**CAMPO_TXTPIE**	CADENA(1000)	Texto que va antes de la rúbrica de la firma en el pie de firma
→	**CAMPO_TXTFDO**	CADENA(250)	Texto que va después de la rúbrica de la firma en el pie de firma
→	**CAMPO_USUARIODIGANT**	CADENA(10)	Código del usuario digital de la firma anterior
→	**CAMPO_REFUNIORGANT**	NUMÉRICO	Identificador de la unidad orgánica de la firma anterior
→	**CAMPO_PTOTRABANT**	CADENA(10)	Código del puesto de trabajo de la firma anterior
→	**CAMPO_HASH**	CADENA(20)	Código de petición para port@firmas
→	**CAMPO_FECHAFIRMADIG**	DATE	Fecha de la firma digital
→	**CAMPO_CODTRANSACCION**	CADENA(250)	Código de transacción de firma
→	**CAMPO_ORDEN**	NUMÉRICO	Orden de la firma
→	**CAMPO_PUESTO**	CADENA(32)	Puesto de trabajo del usuario que firma digitalmente
→	**CAMPO_ORGANISMO**	CADENA(32)	Organismo al que pertenece el usuario que firma digitalmente
→	**CAMPO_NOMBREUSUDIGI**	CADENA(100)	Nombre y apellidos del usuario digital.
→	**CAMPO_FORMATO**	CADENA(128)	Formato de la firma digital
→	**CAMPO_NOMBREARCHIVO**	CADENA(64)	Nombre del archivo de la firma digital.
→	**CAMPO_NOMBREUSUDIGI**	CADENA(98)	Nombre y apellidos del usuario digital.

Métodos para aplicación de los filtros → `obtenerFirmasDocumento`

TrFirmante

trewa.bd.trapi.trapiui.tpo.TrFirmante

Clase que representa la información referente a un firmante.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFFIRMANTE**	TpoPK	Identificador del firmante definido
→	**PTOTRABAJO**	String	Código que identifica el puesto de trabajo
→	**UNIORGANICA**	TpoPK	Código que identifica a la unidad organica
→	**TIPO**	String	Tipo de firmante definido → '**P**' – Principal → '**D**' – Delegado → '**S**' – Sustituto
→	**FECHAINIVIG**	Timestamp	Fecha de inicio de vigencia
→	**FECHAFINVIG**	Timestamp	Fecha de fin de vigencia
→	**TEXTODISPOSICION**	TrTextoDisposicion	Texto de la disposición
→	**REFDELEGSUST**	TpoPK	Identificador del firmante que actúa como delegado o sustituto
→	**PUESTO**	String	Nombre del puesto de trabajo
→	**ORGANISMO**	String	Nombre del organismo o unidad orgánica

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFFIRMANTE**	NUMÉRICO	Identificador del firmante definido
→	**CAMPO_PTOTRABAJO**	CADENA(10)	Código que identifica el puesto de trabajo
→	**CAMPO_UNIORGANICA**	NUMÉRICO(10)	Código que identifica a la unidad organica
→	**CAMPO_TIPO**	CADENA(1)	Tipo de firmante definido → '**P**' – Principal → '**D**' – Delegado → '**S**' – Sustituto
→	**CAMPO_FECHAINIVIG**	DATE	Fecha de inicio de vigencia
→	**CAMPO_FECHAFINVIG**	DATE	Fecha de fin de vigencia
→	**CAMPO_TEXTODISPOSICION**	NUMÉRICO	Identificador del texto de la disposición
→	**CAMPO_REFDELEGSUST**	NUMÉRICO	Identificador del firmante que actúa como delegado o sustituto
→	**CAMPO_PUESTO**	CADENA(32)	Nombre del puesto de trabajo
→	**CAMPO_ORGANISMO**	CADENA(32)	Nombre del organismo o unidad orgánica

→	**CAMPO_DESCTEXTODISP**	CADENA(100)	Texto de la disposición
→	**CAMPO_TIPODISP**	CADENA(1)	Tipo de la disposición → '**D**' – Decreto → '**O'** – Orden → '**T'** – Otros

Métodos para aplicación de los filtros → obtenerFirmantesDefinidos

TrFirmanteTipoDocumento

trewa.bd.trapi.trapiui.tpo.TrFirmanteTipoDocumento

Clase que representa la información referente a un firmante definido para un tipo de documento.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**FIRMANTE**	TrFirmante	Firmante definido
→	**TIPODOC**	TrTipoDocumento	Tipo de documento
→	**EDITABLE**	String	Indica si el pie de firma es editable
→	**ETIQUETA**	String	Permite poner una etiqueta en el pie de firma
→	**ORDEN**	long	Orden de la firma en el documento

TrInteresado

trewa.bd.trapi.trapiui.tpo.TrInteresado

Clase que representa los datos principales de los interesados que se recogen en el sistema.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFINTERESADO**	TpoPK	Identificador del interesado
→	**TIPOIDENT**	String	Tipo de identificación
→	**NUMIDENT**	String	Número de identificación.

→	**APELLIDO1**	String	Primer apellido del interesado.
→	**APELLIDO2**	String	Segundo apellido del interesado.
→	**NOMBRE**	String	Nombre del interesado.
→	**SEXO**	String	Sexo del interesado. → '**F**' – Femenino → '**M**' – Masculino
→	**DIGCONTROL**	String	Dígito de control para el número de identificación.
→	**CIWA**	String	Código de identificación w@ndA.
→	**FECHANACIM**	Timestamp	Fecha de nacimiento.
→	**FECHABAJA**	Timestamp	Fecha de baja en el sistema.
→	**NUMREGMER**	String	Número del registro mercantil.
→	**FECHARCA**	Timestamp	Fecha de RCA de la entidad.
→	**EPIGRAFEIAE**	String	Epígrafe IAE.
→	**EPIGRAFECNAE**	String	Epígrafe CNAE.
→	**COMENTARIOS**	String	Comentarios.
→	**OTROSDATOS**	String	Indica si el interesado tiene otros datos → '**S**' – Sí → '**N**' – No
→	**TIPOORGZ**	String	Tipo de organización.
→	**ANAGRAMAFISCAL**	String	Anagrama fiscal del interesado
→	**REFDATOCONT**	TpoPK	Identificador de los datos de contacto

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFINTERESADO**	NUMÉRICO	Identificador del interesado
→	**CAMPO_TIPOIDENT**	CADENA(10)	Tipo de identificación
→	**CAMPO_NUMIDENT**	CADENA(15)	Número de identificación.
→	**CAMPO_APELLIDO1**	CADENA(50)	Primer apellido del interesado.
→	**CAMPO_APELLIDO2**	CADENA(50)	Segundo apellido del interesado.
→	**CAMPO_NOMBRE**	CADENA(100)	Nombre del interesado.
→	**CAMPO_SEXO**	CADENA(1)	Sexo del interesado. → '**F**' – Femenino → '**M**' – Masculino
→	**CAMPO_DIGCONTROL**	CADENA(1)	Dígito de control para el número de identificación.
→	**CAMPO_CIWA**	CADENA(15)	Código de identificación w@ndA.
→	**CAMPO_FECHANACIM**	DATE	Fecha de nacimiento.
→	**CAMPO_FECHABAJA**	DATE	Fecha de baja en el sistema.
→	**CAMPO_NUMREGMER**	CADENA(30)	Número del registro mercantil.
→	**CAMPO_FECHARCA**	DATE	Fecha de RCA de la entidad.
→	**CAMPO_EPIGRAFEIAE**	CADENA(30)	Epígrafe IAE.
→	**CAMPO_EPIGRAFECNAE**	CADENA(30)	Epígrafe CNAE.
→	**CAMPO_COMENTARIOS**	CADENA(100)	Comentarios.

→	**CAMPO_OTROSDATOS**	CADENA(1)	Indica si el interesado tiene otros datos → '**S**' – Sí → '**N**' – No
→	**CAMPO_TIPOORGZ**	CADENA(1)	Tipo de organización.
→	**CAMPO_ANAGRAMAFISCAL**	CADENA(20)	Anagrama fiscal del interesado
→	**CAMPO_REFDATOCONT**	NUMÉRICO	Identificador de los datos de contacto

Métodos para aplicación de los filtros → obtenerInteresados

TrInteresadoDocumento

trewa.bd.trapi.trapiui.tpo.TrInteresadoDocumento

Clase que representa la información de los interesados en un documento del expediente.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFDOCEXP**	TpoPK	Identificador del documento del expediente
→	**REFTIPODOC**	TpoPK	Identificador del tipo de documento
→	**NOMBRETIPODOC**	String	Nombre del tipo de documento
→	**INTERESADOEXP**	TrInteresadoExpediente	Interesado del expediente
→	**RAZONINT**	TrRazonInteres	Razón de interés.
→	**OBSERVACIONES**	String	Observaciones

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFDOCEXP**	NUMÉRICO	Identificador del documento del expediente
→	**CAMPO_REFTIPODOC**	NUMÉRICO	Identificador del tipo de documento
→	**CAMPO_NOMBRETIPODOC**	CADENA(50)	Nombre del tipo de documento
→	**CAMPO_REFEXPEDIENTE**	NUMÉRICO	Identificador del expediente.
→	**CAMPO_REFINTERESADO**	NUMÉRICO	Identificador del interesado
→	**CAMPO_REFRAZONINTEXP**	NUMÉRICO	Identificador de la razón de interés del interesado en el expediente.
→	**CAMPO_REFRAZONINTDOCU**	NUMÉRICO	Identificador de la razón de interés.
→	**CAMPO_OBSERVACIONES**	CADENA(250)	Observaciones del interesado en el documento
→	**CAMPO_TIPOIDENT**	CADENA(10)	Tipo de identificación

→	**CAMPO_NUMIDENT**	CADENA(15)	Número de identificación.
→	**CAMPO_APELLIDO1**	CADENA(50)	Primer apellido del interesado.
→	**CAMPO_APELLIDO2**	CADENA(50)	Segundo apellido del interesado.
→	**CAMPO_NOMBRE**	CADENA(100)	Nombre del interesado.
→	**CAMPO_SEXO**	CADENA(1)	Sexo del interesado. → '**F**' – Femenino → '**M'** – Masculino
→	**CAMPO_DIGCONTROL**	CADENA(1)	Dígito de control para el número de identificación.
→	**CAMPO_CIWAINTE**	CADENA(15)	Código de identificación w@ndA del interesado
→	**CAMPO_FECHANACIM**	DATE	Fecha de nacimiento.
→	**CAMPO_FECHABAJA**	DATE	Fecha de baja en el sistema.
→	**CAMPO_NUMREGMER**	CADENA(30)	Número del registro mercantil.
→	**CAMPO_FECHARCA**	DATE	Fecha de RCA de la entidad.
→	**CAMPO_EPIGRAFEIAE**	CADENA(30)	Epígrafe IAE.
→	**CAMPO_EPIGRAFECNAE**	CADENA(30)	Epígrafe CNAE.
→	**CAMPO_COMENTARIOS**	CADENA(100)	Comentarios.
→	**CAMPO_TIPOORGZ**	CADENA(1)	Tipo de organización.
→	**CAMPO_ANAGRAMAFISCAL**	CADENA(20)	Anagrama fiscal del interesado
→	**CAMPO_REFDATOCONT**	NUMÉRICO	Identificador de los datos de contacto
→	**CAMPO_ABREVRAZONINTEXP**	CADENA(10)	Abreviatura de la razón de interés del interesado en el expediente.
→	**CAMPO_DESCRAZONINTEXP**	CADENA(100)	Descripción de la razón de interés del interesado en el expediente.
→	**CAMPO_OBSOLETORAINEXP**	CADENA(1)	Indica si la razón de interés del interesado en el expediente está obsoleta → '**S**' – Sí → '**N**' – No
→	**CAMPO_CODWANDARAZONINTEXP**	CADENA(10)	Valor en w@ndA de la razón de interés del interesado en el expediente.
→	**CAMPO_ABREVRAZONINTDOC**	CADENA(10)	Abreviatura de la razón de interés del interesado en el documento.
→	**CAMPO_DESCRAZONINTDOC**	CADENA(100)	Descripción de la razón de interés del interesado en el documento.
→	**CAMPO_OBSOLETORAINDOC**	CADENA(1)	Indica si la razón de interés del interesado en el documento está obsoleta → '**S**' – Sí → '**N**' – No
→	**CAMPO_CODWANDARAZONINTDOC**	CADENA(10)	Valor en w@ndA de la razón de interés del interesado en el documento.

Métodos para aplicación de los filtros → obtenerInteresadosDocumento

TrInteresadoExpediente

trewa.bd.trapi.trapiui.tpo.TrInteresadoExpediente

Clase que representa la información de los interesados en un expediente.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFEXPEDIENTE**	TpoPK	Identificador del expediente
→ **INTERESADO**	TrInteresado	Interesado del expediente.
→ **RAZONINT**	TrRazonInteres	Razón de interés.
→ **OBSERVACIONES**	String	Observaciones del interesado en el expediente.
→ **REFDATOCONT**	TpoPK	Identificador de los datos de contacto

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFEXPEDIENTE**	NUMÉRICO	Identificador del expediente.
→ **CAMPO_REFINTERESADO**	NUMÉRICO	dentificador del interesado.
→ **CAMPO_REFRAZONINT**	NUMÉRICO	Identificador de la razón de interés.
→ **CAMPO_OBSERVACIONES**	CADENA(250)	Observaciones
→ **CAMPO_REFDATOCONT**	NUMÉRICO	Identificador de los datos de contacto
→ **CAMPO_CODTIPOIDENT**	CADENA(10)	Tipo de identificación
→ **CAMPO_NUMIDENT**	CADENA(15)	Número de identificación.
→ **CAMPO_APELLIDO1**	CADENA(50)	Primer apellido del interesado.
→ **CAMPO_APELLIDO2**	CADENA(50)	Segundo apellido del interesado.
→ **CAMPO_NOMBRE**	CADENA(100)	Nombre del interesado.
→ **CAMPO_SEXO**	CADENA(1)	Sexo del interesado. → '**F**' – Femenino → '**M**' – Masculino
→ **CAMPO_DIGCONTROL**	CADENA(1)	Dígito de control para el número de identificación.
→ **CAMPO_CIWA**	CADENA(15)	Código de identificación w@ndA del interesado
→ **CAMPO_FECHANACIM**	DATE	Fecha de nacimiento.
→ **CAMPO_FECHABAJA**	DATE	Fecha de baja en el sistema.
→ **CAMPO_NUMREGMER**	CADENA(30)	Número del registro mercantil.
→ **CAMPO_FECHARCA**	DATE	Fecha de RCA de la entidad.
→ **CAMPO_EPIGRAFEIAE**	CADENA(30)	Epígrafe IAE.
→ **CAMPO_EPIGRAFECNAE**	CADENA(30)	Epígrafe CNAE.
→ **CAMPO_COMENTARIOS**	CADENA(100)	Comentarios.
→ **CAMPO_REFTIPOORGZ**	CADENA(1)	Tipo de organización.

→	**CAMPO_ANAGRAMAFISCAL**	CADENA(20)	Anagrama fiscal del interesado
→	**CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura de la razón de interés del interesado en el expediente.
→	**CAMPO_DESCRIPCION**	CADENA(100)	Descripción de la razón de interés del interesado en el expediente.
→	**CAMPO_OBSOLETO**	CADENA(1)	Indica si la razón de interés del interesado en el expediente está obsoleta → '**S**' – Sí → '**N**' – No
→	**CAMPO_CODWANDA**	CADENA(10)	Valor en w@ndA de la razón de interés del interesado en el expediente.

Métodos para aplicación de los filtros → obtenerInteresadosExpediente

TrMensaje

trewa.bd.trapi.trapiui.tpo.TrMensaje

Clase que representa la información referente a mensajes.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFMENSAJE**	TpoPK	Identificador del mensaje
→	**TEXTOMENSAJE**	String	Texto del mensaje
→	**LEIDO**	String	Indica si el mensaje ha sido o no leído. → '**S**' – El mensaje ha sido leído → '**N'** – El mensaje no ha sido leído
→	**FECHA**	Timestamp	Fecha en la que se envió el mensaje
→	**PRIORIDAD**	String	Prioridad del mensaje → '**A**' – Prioridad alta → '**M'** – Prioridad media → '**B**' – Prioridad baja
→	**USUARIOENV**	String	Usuario origen del mensaje
→	**USUARIOREC**	String	Usuario destino del mensaje
→	**EXPEDIENTE**	TrExpediente	Expediente al que está asociado

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFMENSAJE**	NUMÉRICO	Identificador del mensaje
→ **CAMPO_TEXTOMENSAJE**	CADENA(250)	Texto del mensaje
→ **CAMPO_LEIDO**	CADENA	Indica si el mensaje ha sido o no leído. → '**S**' – El mensaje ha sido leído → '**N'** – El mensaje no ha sido leído
→ **CAMPO_FECHA**	DATE	Fecha en la que se envió el mensaje
→ **CAMPO_PRIORIDAD**	CADENA(1)	Prioridad del mensaje → '**A**' – Prioridad alta → '**M'** – Prioridad media → '**B**' – Prioridad baja
→ **CAMPO_USUARIOENV**	CADENA(10)	Usuario origen del mensaje
→ **CAMPO_USUARIOREC**	CADENA(10)	Usuario destino del mensaje
→ **CAMPO_REFEXP**	NUMÉRICO	Identificador del expediente al que está asociado

Métodos para aplicación de los filtros
→ obtenerMensajesUsuario
→ obtenerMensajesEnviados

TrMensajeCondicionAccion

trewa.bd.trapi.trapiui.tpo.TrMensajeCondicionAccion

Clase que representa la información referente a un mensaje asociado a una acción, condición o un mensaje de aviso.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **NOMBRE**	String	Nombre de la condición o acción
→ **DESCRIPCION**	String	Descripción de la condición o acción
→ **MENSAJE**	String	Resultado de evaluar la condición o realizar la acción
→ **TIPO**	String	Indica si el mensaje corresponde a una condición, a una acción o a un mensaje de aviso. → '**C**' – El mensaje corresponde a una condición → '**A'** – El mensaje corresponde a una acción → '**W'** – Se trata de un mensaje de aviso

TrMetafase

trewa.bd.trapi.trapiui.tpo.TrMetafase

Clase que representa la información referente a una metafase.

El modelo de referencia W@nda define una metafase como “el conjunto de fases dentro de un procedimiento que comparten una serie de características comunes que permiten al Tramitador dar un tratamiento diferenciado del resto pero común a todas ellas”.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFMETAFASE**	TpoPK	Identificador de la metafase
→	**NOMBRE**	String	Nombre de la metafase
→	**DESCRIPCION**	String	Descripción de la metafase
→	**ORDEN**	long	Orden
→	**STMA**	TrSistema	Sistema
→	**INFORMARBUS**	String	Indica si se informa al bus al entrar en la metafase → ‘**S’** – Sí → ‘**N’** – No

TrModificacionCaducidadExpediente

trewa.bd.trapi.trapiui.tpo.TrModificacionCaducidadExpediente

Clase que representa la información referente a una modificación de la caducidad de un expediente.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**TIPO**	String	Tipo de la modificación → ‘**A**’ – Ampliación → ‘**R’** – Reducción → ‘**S** – Suspensión
→	**UNIDAD**	String	Unidad de medida de tiempo → ‘**D**’ – Días → ‘**M’** – Meses → ‘**A’** – Años
→	**NUMUNIDADES**	long	Número de unidades que se cuentan (nº de días, nº de años,...)

→	**FECHA**	Timestamp	Fecha en la que se produjo la modificación sobre la caducidad
→	**FECHAFINAL**	Timestamp	Fecha de finalización para las modificaciones del tipo "suspensión". Es la fecha en la que sea reanuda la cuenta del tiempo.
→	**USUARIO**	String	Usuario que hizo la modificación

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_TIPO**	CADENA(1)	Tipo de la modificación → '**A**' – Ampliación → '**R'** – Reducción → '**S** – Suspensión
→	**CAMPO_UNIDAD**	CADENA(1)	Unidad de medida de tiempo → '**D**' – Días → '**M'** – Meses → '**A'** – Años
→	**CAMPO_NUMUNIDADES**	NUMÉRICO	Número de unidades que se cuentan (nº de días, nº de años,...)
→	**CAMPO_FECHA**	DATE	Fecha en la que se produjo la modificación sobre la caducidad
→	**CAMPO_FECHAFINAL**	DATE	Fecha de finalización para las modificaciones del tipo "suspensión". Es la fecha en la que sea reanuda la cuenta del tiempo.
→	**CAMPO_USUARIO**	CADENA(10)	Usuario que hizo la modificación

Métodos para aplicación de los filtros → `obtenerModificacionesCaducidadExpediente`

TrMunicipio

trewa.bd.trapi.trapiui.tpo.TrMunicipio

Clase que representa la información de un municipio

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**CODMUNICIPIO**	String	Código del municipio.
→	**PROVINCIA**	TrProvincia	Provincia a la que pertenece el municipio.
→	**NOMBRE**	String	Nombre del municipio.
→	**DESCRIPCION**	String	Descripción del municipio.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_CODMUNICIPIO**	CADENA(3)	Código del municipio.
→ **CAMPO_CODPROVINCIA**	CADENA(2)	Provincia a la que pertenece el municipio.
→ **CAMPO_NOMBRE**	CADENA(32)	Nombre del municipio.
→ **CAMPO_DESCRIPCION**	CADENA(64)	Descripción del municipio.

Métodos para aplicación de los filtros → obtenerMunicipios

TrNotificacionInteresado

trewa.bd.trapi.trapiui.tpo.TrNotificacionInteresado

Clase que representa la información de las notificaciones de los interesados.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFDOCEXP**	TpoPK	Identificador del documento del expediente
→ **REFEXPEDIENTE**	TpoPK	Identificador del expediente
→ **REFINTERESADO**	TpoPK	Identificador del interesado
→ **REFRAZONINTEXP**	TpoPK	Identificador de la razón de interés del interesado del expediente
→ **REFRAZONINTDOCU**	TpoPK	Identificador de la razón de interés del interesado del documento
→ **FECHAPRIMERA**	Timestamp	Fecha de la primera notificación
→ **MEDIOPRIMERA**	String	Medio de comunicación de la primera notificación
→ **RECHAZOPRIMERA**	String	Causa del rechazo de la primera notificación
→ **FECHASEGUNDA**	Timestamp	Fecha de la segunda notificación
→ **MEDIOSEGUNDA**	String	Medio de comunicación de la segunda notificación
→ **RECHAZOSEGUNDA**	String	Causa del rechazo de la segunda notificación
→ **FECHATERCERA**	Timestamp	Fecha de la tercera notificación
→ **MEDIOTERCERA**	String	Medio de comunicación de la tercera notificación
→ **RECHAZOTERCERA**	String	Causa del rechazo de la tercera notificación
→ **OBSERVACIONES**	String	Observaciones de la notificación.
→ **HASHNOTIF**	String	Código de la notificación.
→ **CODSERVICIO**	Long	Código del servicio notifi@
→ **FECHAACUSE**	Timestamp	Fecha de acuse de la notificación

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFDOCEXP**	NUMÉRICO	Identificador del documento del expediente
→ **CAMPO_REFEXPEDIENTE**	NUMÉRICO	Identificador del expediente
→ **CAMPO_REFINTERESADO**	NUMÉRICO	Identificador del interesado
→ **CAMPO_REFRAZONINTEXP**	NUMÉRICO	Identificador de la razón de interés del interesado del expediente
→ **CAMPO_REFRAZONINTDOCU**	NUMÉRICO	Identificador de la razón de interés del interesado del documento
→ **CAMPO_FECHAPRIMERA**	DATE	Fecha de la primera notificación
→ **CAMPO_MEDIOPRIMERA**	CADENA(100)	Medio de comunicación de la primera notificación
→ **CAMPO_RECHAZOPRIMERA**	CADENA(250)	Causa del rechazo de la primera notificación
→ **CAMPO_FECHASEGUNDA**	DATE	Fecha de la segunda notificación
→ **CAMPO_MEDIOSEGUNDA**	CADENA(100)	Medio de comunicación de la segunda notificación
→ **CAMPO_RECHAZOSEGUNDA**	CADENA(250)	Causa del rechazo de la segunda notificación
→ **CAMPO_FECHATERCERA**	DATE	Fecha de la tercera notificación
→ **CAMPO_MEDIOTERCERA**	CADENA(100)	Medio de comunicación de la tercera notificación
→ **CAMPO_RECHAZOTERCERA**	CADENA(250)	Causa del rechazo de la tercera notificación
→ **CAMPO_OBSERVACIONES**	CADENA(250)	Observaciones de la notificación.
→ **CAMPO_HASHNOTIF**	CADENA(20)	Código de la notificación.
→ **CAMPO_CODSERVICIO**	NUMÉRICO	Código del servicio notifi@
→ **CAMPO_FECHAACUSE**	DATE	Fecha de acuse de la notificación

Métodos para aplicación de los filtros → obtenerNotificacionesInteresado

TrOrganismo

trewa.bd.trapi.trapiui.tpo.TrOrganismo

Clase que representa la información de los distintos organismos o unidades orgánicas.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFORGANISMO**	TpoPK	Identificador del organismo
→ **CODORG**	String	Código del organismo
→ **NOMBRE**	String	Nombre del organismo
→ **DESCRIPCION**	String	Descripción del organismo
→ **TIPO**	String	Tipo de organismo → '**DP**' – Delegación Provincial

			→ '**SC**' – Servicios Centrales
			→ '**CE**' – Centro Adscrito
			→ '**EE**' – Empresa Externa
→	**DATOSCONTACTO**	TrDatosContacto	Datos de contacto
→	**CIWA**	String	Código de identificación w@ndA.
→	**IDARIES**	String	Identificador de aries
→	**IDENTIFICADOR**	String	Número de identificación
→	**DIGCONTROL**	String	Dígito de control
→	**FECHAINIVIG**	Timestamp	Fecha de inicio de vigencia del organismo
→	**FECHAFINVIG**	Timestamp	Fecha de fin de vigencia del organismo
→	**REFORGPADRE**	TpoPK	Identificador del organismo padre
→	**TIPOORG**	TrTipoOrganismo	Tipo de Organismo
→	**TIPOORGZ**	TrTipoOrganizacion	Tipo de Organización

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFORGANISMO**	NUMÉRICO	Identificador del organismo
→	**CAMPO_CODMUNICIPIO**	CADENA(3)	Código del municipio
→	**CAMPO_CODPROVINCIA**	CADENA(2)	Código de la provincia
→	**CAMPO_TIPOVIA**	CADENA(10)	Tipo de vía
→	**CAMPO_CODORG**	CADENA(16)	Código del organismo
→	**CAMPO_NOMBRE**	CADENA(32)	Nombre del organismo
→	**CAMPO_DESCRIPCION**	CADENA(200)	Descripción del organismo
→	**CAMPO_TIPO**	CADENA(1)	Tipo de organismo
			→ '**DP**' – Delegación Provincial
			→ '**SC**' – Servicios Centrales
			→ '**CE**' – Centro Adscrito
			→ '**EE**' – Empresa Externa
→	**CAMPO_NOMBREVIA**	CADENA(64)	Nombre de la vía
→	**CAMPO_NUMERO**	NUMÉRICO	Número del domicilio
→	**CAMPO_LETRA**	CADENA(2)	Letra del domicilio
→	**CAMPO_ESCALERA**	CADENA(2)	Escalera del domicilio
→	**CAMPO_PISO**	NUMERICO	Piso del domicilio
→	**CAMPO_PUERTA**	CADENA(2)	Puerta del domicilio
→	**CAMPO_CODPOSTAL**	NUMERICO	Código postal del domicilio
→	**CAMPO_TELEFONO**	CADENA(25)	Número(s) de teléfono de contacto
→	**CAMPO_TLFMOVIL**	CADENA(25)	Número(s) de teléfono móvil
→	**CAMPO_FAX**	CADENA(25)	Número(s) de Fax
→	**CAMPO_EMAIL**	CADENA(64)	Correo electrónico
→	**CAMPO_CIWA**	CADENA(15)	Código de identificación w@ndA
→	**CAMPO_IDARIES**	CADENA(5)	Identificador de aries
→	**CAMPO_IDENTIFICADOR**	CADENA(15)	Número de identificador

→	**CAMPO_DIGCONTROL**	CADENA(1)	Dígito de control
→	**CAMPO_FECHAINIVIG**	DATE	Fecha de inicio de vigencia del organismo
→	**CAMPO_FECHAFINVIG**	DATE	Fecha de fin de vigencia del organismo
→	**CAMPO_REFORGPADRE**	NUMÉRICO	Identificador del organismo superior al que pertenece.
→	**CAMPO_REFTIPOORG**	NUMÉRICO	Identificador del tipo de organismo.
→	**CAMPO_REFTIPOORGZ**	NUMÉRICO	Identificador del tipo de organización.

Métodos para aplicación de los filtros → obtenerOrganismos

TrPais

trewa.bd.trapi.trapiui.tpo.TrPais

Clase que representa la información de un país

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**CODPAIS**	String	Identificador del país
→	**NOMBRE**	String	Nombre del país
→	**DESCRIPCION**	String	Descripción del país

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_CODPAIS**	CADENA(3)	Identificador del país
→	**CAMPO_NOMBRE**	CADENA(32)	Nombre del país
→	**CAMPO_DESCRIPCION**	CADENA(64)	Descripción del país

Métodos para aplicación de los filtros → obtenerPaises

TrParametro

trewa.bd.trapi.trapiui.tpo.TrParametro

Clase que representa la información referente a un parámetro.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFPARAM**	TpoPK	Identificador del parámetro
→	**NOMBRE**	String	Nombre del parámetro
→	**DESCRIPCION**	String	Descripción del parámetro
→	**TIPO**	String	Tipo de parámetro → '**C**' – Carácter → '**N'** – Numérico → '**F'** – Fecha
→	**TAMANIO**	long	Tamaño del parámetro
→	**STMA**	TrSistema	Sistema asociado al parámetro

TrParametroBloque

trewa.bd.trapi.trapiui.tpo.TrParametroBloque

Clase que representa la información referente a un parámetro definido para un bloque.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**PARAMETRO**	TrParametro	Parámetro
→	**ORDEN**	long	Orden del parámetro en la llamada al módulo

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_NOMBRE**	CADENA(20)	Nombre del parámetro
→	**CAMPO_DESCRIPCION**	CADENA(100)	Descripción del parámetro
→	**CAMPO_TIPO**	CADENA(1)	Tipo de parámetro → '**C**' – Carácter → '**N'** – Numérico → '**F'** – Fecha
→	**CAMPO_TAMANIO**	NUMÉRICO	Tamaño del parámetro
→	**CAMPO_ORDEN**	NUMÉRICO	Orden del parámetro en la llamada al módulo
→	**CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema asociado al parámetro

Métodos para aplicación de los filtros → `obtenerParametrosBloque`

TrParrafo

trewa.bd.trapi.trapiui.tpo.TrParrafo

Clase que representa la información referente a un párrafo de un documento.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFPARRDOC**	TpoPK	Identificador defl párrafo del documento del expediente
→	**ETIQUETA**	String	Etiqueta del párrafo del documento
→	**PARRAFO**	String	Texto del párrafo
→	**ORDEN**	long	Orden del párrafo
→	**ALINEACION**	String	Alineación del párrafo → "**I**" – A la izquierda → "**C**" – Centrada → "**D**" – A la derecha → "**J**" – Justificada
→	**ESTILO**	String	Estilo del párrafo
→	**ESTILOETIQ**	String	Estilo de la etiqueta del párrafo
→	**EDITABLE**	String	Indica si el párrafo es editable o no → '**S**' – El párrafo es editable → '**N'** – El párrafo no es editable
→	**TIPOPARRAFO**	TrTipoParrafo	Tipo de párrafo
→	**IMAGEN**	byte[]	Imagén del párrafo
→	**FORMATO**	String	Tipo de formato (tipo mime)
→	**NOMBREFICHERO**	String	Nombre del fichero

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFPARRDOC**	NUMÉRICO	Identificador del párrafo del documento del expediente
→	**CAMPO_ETIQUETA**	CADENA(15)	Etiqueta del párrafo del documento
→	**CAMPO_PARRAFO**	CADENA(4000)	Texto del párrafo
→	**CAMPO_ORDEN**	NUMÉRICO	Orden del párrafo
→	**CAMPO_ALINEACION**	CADENA(1)	Alineación del párrafo → "**I**" – A la izquierda

			→ "**C**" – Centrada
			→ "**D**" – A la derecha
			→ "**J**" – Justificada
→	**CAMPO_ESTILO**	CADENA(8)	Estilo del párrafo
→	**CAMPO_ESTILOETIQ**	CADENA(8)	Estilo de la etiqueta del párrafo
→	**CAMPO_EDITABLE**	CADENA(1)	Indica si el párrafo es editable o no
			→ '**S**' – El párrafo es editable
			→ '**N'** – El párrafo no es editable
→	**CAMPO_REFTIPOPARR**	NUMÉRICO	Identificador del tipo de párrafo
→	**CAMPO_ABREVTIPOPARR**	CADENA(15)	Abreviatura del tipo de párrafo
→	**CAMPO_DESCTIPOPARR**	CADENA(250)	Descripción del tipo de párrafo
→	**CAMPO_FORMATO**	CADENA(128)	Tipo de formato (tipo mime)
→	**CAMPO_NOMBREFICHERO**	CADENA(64)	Nombre del fichero

Métodos para aplicación de los filtros → obtenerParrafosDocumento

TrPerfilUsuario

trewa.bd.trapi.trapiui.tpo.TrPerfilUsuario

Clase que representa la información de los distintos perfiles de usuario que pueden existir.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFPERFILUSU**	TpoPK	Identificador del perfil de usuario
→	**NOMBRE**	String	Nombre del perfil de usuario
→	**DESCRIPCION**	String	Descripción del perfil de usuario
→	**STMA**	TrSistema	Sistema del perfil de usuario.

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFPERFILUSU**	NUMÉRICO	Identificador del perfil de usuario
→	**CAMPO_NOMBRE**	CADENA(25)	Nombre del perfil de usuario
→	**CAMPO_DESCRIPCION**	CADENA(250)	Descripción del perfil de usuario
→	**CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema del perfil de usuario
→	**CAMPO_CODSTMA**	CADENA(10)	Código o nombre del sistema del perfil de usuario

→	**CAMPO_DESCSTMA**	CADENA(128)	Descripción del sistema del perfil de usuario

Métodos para aplicación de los filtros → obtenerPerfilesUsuario

TrPlazo

trewa.bd.trapi.trapiui.tpo.TrPlazo

Clase que representa la unidad de tiempo, el número de unidades y la descripción de la fecha en la que cumple el plazo.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**UNIDAD**	String	Indica la unidad de tiempo → "**D**" – Días → "**M**" – Meses → "**A**" – Años
→	**NUMUNIDADES**	Integer	Número de unidades que se cuentan (nº de días, nº de años,...)
→	**DESCFECHALIMITE**	String	Descripción de la fecha límite

TrProvincia

trewa.bd.trapi.trapiui.tpo.TrProvincia

Clase que representa la información de una provincia.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**CODPROVINCIA**	String	Identificador de la provincia
→	**NOMBRE**	String	Nombre de la provincia.
→	**DESCRIPCION**	String	Descripción de la provincia.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso

→	**CAMPO_CODPROVINCIA**	CADENA(2)	Identificador de la provincia
→	**CAMPO_NOMBRE**	CADENA(32)	Nombre de la provincia.
→	**CAMPO_DESCRIPCION**	CADENA(64)	Descripción de la provincia.

Métodos para aplicación de los filtros → obtenerProvincias

TrPtoTrabOrganismo

trewa.bd.trapi.trapiui.tpo.TrPtoTrabOrganismo

Clase que representa los distintos puestos de trabajo que existen en cada organismo.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**ORGANISMO**	TrOrganismo	Organismo
→	**PUESTOTRABAJO**	TrPuestoTrabajo	Puesto de trabajo

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_CODPTOTRAB**	CADENA(10)	Identificador del puesto de trabajo
→	**CAMPO_NOMBREPTOTRAB**	CADENA(32)	Nombre del puesto de trabajo
→	**CAMPO_DESCRIPCIONPTOTRAB**	CADENA(64)	Descripción del puesto de trabajo
→	**CAMPO_REFORGANISMO**	NUMÉRICO	Identificador del organismo
→	**CAMPO_CODMUNICIPIO**	CADENA(3)	Código del municipio
→	**CAMPO_CODPROVINCIA**	CADENA(2)	Código de la provincia
→	**CAMPO_TIPOVIA**	CADENA(10)	Tipo de vía
→	**CAMPO_CODORG**	CADENA(16)	Código del organismo
→	**CAMPO_NOMBRE**	CADENA(32)	Nombre del organismo
→	**CAMPO_DESCRIPCION**	CADENA(200)	Descripción del organismo
→	**CAMPO_TIPO**	CADENA(1)	Tipo de organismo → '**DP**' – Delegación Provincial → '**SC**' – Servicios Centrales → '**CE**' – Centro Adscrito → '**EE**' – Empresa Externa
→	**CAMPO_NOMBREVIA**	CADENA(64)	Nombre de la vía
→	**CAMPO_NUMERO**	NUMÉRICO	Número del domicilio
→	**CAMPO_LETRA**	CADENA(2)	Letra del domicilio

→	**CAMPO_ESCALERA**	CADENA(2)	Escalera del domicilio
→	**CAMPO_PISO**	NUMERICO	Piso del domicilio
→	**CAMPO_PUERTA**	CADENA(2)	Puerta del domicilio
→	**CAMPO_CODPOSTAL**	NUMERICO	Código postal del domicilio
→	**CAMPO_TELEFONO**	CADENA(25)	Número(s) de teléfono de contacto
→	**CAMPO_TLFMOVIL**	CADENA(25)	Número(s) de teléfono móvil
→	**CAMPO_FAX**	CADENA(25)	Número(s) de Fax
→	**CAMPO_EMAIL**	CADENA(64)	Correo electrónico
→	**CAMPO_CIWA**	CADENA(15)	Código de identificación w@ndA
→	**CAMPO_IDARIES**	CADENA(5)	Identificador de aries
→	**CAMPO_IDENTIFICADOR**	CADENA(15)	Número de identificador
→	**CAMPO_DIGCONTROL**	CADENA(1)	Dígito de control
→	**CAMPO_FECHAINIVIG**	DATE	Fecha de inicio de vigencia del organismo
→	**CAMPO_FECHAFINVIG**	DATE	Fecha de fin de vigencia del organismo
→	**CAMPO_REFORGPADRE**	NUMÉRICO	Identificador del organismo superior al que pertenece.
→	**CAMPO_REFTIPOORG**	NUMÉRICO	Identificador del tipo de organismo.
→	**CAMPO_REFTIPOORGZ**	NUMÉRICO	Identificador del tipo de organización.

Métodos para aplicación de los filtros → `obtenerPuestosTrabajoOrganismo`

TrPuestoTrabajo

trewa.bd.trapi.trapiui.tpo.TrPuestoTrabajo

Clase que representa los distintos puestos de trabajo.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**CODPTOTRAB**	String	Identificador del puesto de trabajo
→	**NOMBRE**	String	Nombre del puesto de trabajo
→	**DESCRIPCION**	String	Descripción del puesto de trabajo

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_CODPTOTRAB**	CADENA(10)	Identificador del puesto de trabajo
→	**CAMPO_NOMBRE**	CADENA(32)	Nombre del puesto de trabajo
→	**CAMPO_DESCRIPCION**	CADENA(64)	Descripción del puesto de trabajo

Métodos para aplicación de los filtros → obtenerPuestosTrabajoOrganismo

TrRazonInteres

trewa.bd.trapi.trapiui.tpo.TrRazonInteres

Clase que recoge las distintas razones de interés de los interesados en los expedientes y documentos.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFRAZONINT**	TpoPK	Identificador de la razón de interés
→	**ABREVIATURA**	String	Abreviatura de la razón de interés
→	**DESCRIPCION**	String	Descripción de la razón de interés
→	**OBSOLETO**	String	Indica si la razón de interés está obsoleta. → '**S**' – Está obsoleta → '**N**' – No está obsoleta
→	**CODWANDA**	String	Valor en w@ndA de la razón de interés

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFRAZONINT**	NUMÉRICO	Identificador de la razón de interés
→	**CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura de la razón de interés
→	**CAMPO_DESCRIPCION**	CADENA(100)	Descripción de la razón de interés
→	**CAMPO_OBSOLETO**	CADENA(1)	Indica si la razón de interés está obsoleta. → '**S**' – Está obsoleta → '**N**' – No está obsoleta
→	**CAMPO_CODWANDA**	CADENA(10)	Valor en w@ndA de la razón de interés

Métodos para aplicación de los filtros → obtenerRazonesInteres

TrRegistroDocumento

trewa.bd.trapi.trapiui.tpo.TrRegistroDocumento

Clase que representa la información de registro para un documento.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**NUMREGSALIDA**	long	Número de registro de salida del documento
→	**FECHASALIDA**	Timestamp	Fecha de salida
→	**REMITENTE**	String	Remitente del documento
→	**OFICINASALIDA**	String	Oficina de salida del documento
→	**ASUNTO**	String	Asunto del documento
→	**DESTINATARIO**	String	Destinatario del documento
→	**NUMREGENTRADA**	long	Número de registro de entrada del documento
→	**FECHAENTRADA**	Timestamp	Fecha de entrada del documento
→	**OFICINAENTRADA**	String	Oficina de entrada del documento
→	**REGISTRO**	String	Datos del registro (típicamente XML)

TrRelacionDefinida

trewa.bd.trapi.trapiui.tpo.TrRelacionDefinida

Clase que representa la información referente a relaciones establecidas entre fases, transiciones, definición de procedimientos,... definidas en el tramitador.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**TIPORELACION**	TrTipoRelacion	Tipo de relación
→	**REFRELACION**	TpoPK	Identificador de la relación
→	**DESCRIPCION**	String	Descripción de la relación
→	**REFFASEA**	TpoPK	Identificador de la fase A
→	**REFFASEB**	TpoPK	Identificador de la fase B
→	**TRANSICION**	TrTransicion	Transición
→	**TIPODOC**	TrTipoDocumento	Tipo de documento
→	**DEFPROC**	TrDefProcedimiento	Definición del procedimiento

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFTIPORELA**	NUMÉRICO	Identificador del tipo de relación

→	**CAMPO_NOMBRETIPORELA**	CADENA(25)	Nombre del tipo de relación
→	**CAMPO_DESCRIPCIONTIPORELA**	CADENA(250)	Descripción del tipo de relación
→	**CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema
→	**CAMPO_REFRELACION**	NUMÉRICO	Identificador de la relación
→	**CAMPO_DESCRIPCION**	CADENA(250)	Descripción de la relación
→	**CAMPO_REFFASEA**	NUMÉRICO	Identificador de la fase A
→	**CAMPO_REFFASEB**	NUMÉRICO	Identificador de la fase B
→	**CAMPO_REFTRANSICION**	NUMÉRICO	Identificador de la transición
→	**CAMPO_REFTIPODOC**	NUMÉRICO	Identificador del tipo de documento
→	**CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento

Métodos para aplicación de los filtros → `obtenerRelacionesDefinidas`

TrRelacionExpediente

trewa.bd.trapi.trapiui.tpo.TrRelacionExpediente

Clase que recoge las posibles relaciones entre los distintos expedientes del sistema para dar soporte al anidamiento, encadenamiento o simple relación entre expedientes..

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFRELACIONEXP**	TpoPK	Identificador de la relación entre expedientes.
→	**REFEXPEDIENTE**	TpoPK	Identificador del expediente (interno o externo).
→	**TIPORELACION**	String	Tipo de relación → '**M**' – Maesto → '**D**' – Detalle → '**I**' – Igualdad
→	**OBSERVACIONES**	String	Observaciones a la relación.
→	**COMPONENTE**	TrComponente	Componente
→	**NUMEXP**	String	Número del expediente en la aplicación "cliente".
→	**TITULOEXP**	String	Título del expediente en la aplicación cliente.

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFRELACIONEXP**	NUMÉRICO	Identificador de la relación entre expedientes.
→	**CAMPO_REFEXPEDIENTE**	NUMÉRICO	Identificador del expediente (interno o externo).

→	**CAMPO_TIPORELACION**	CADENA(1)	Tipo de relación → '**M**' – Maesto → '**D**' – Detalle → '**I**' – Igualdad
→	**CAMPO_OBSERVACIONES**	CADENA(250)	Observaciones a la relación.
→	**CAMPO_REFCOMP**	NUMÉRICO	Identificador del componente
→	**CAMPO_COMPONENTE**	CADENA(50)	Nombre del componente
→	**CAMPO_NUMEXP**	CADENA(30)	Número del expediente en la aplicación "cliente".
→	**CAMPO_TITULOEXP**	CADENA(250)	Título del expediente en la aplicación cliente.

Métodos para aplicación de los filtros → obtenerRelacionesExpediente

TrRelacionInteresado

trewa.bd.trapi.trapiui.tpo.TrRelacionInteresado

Clase que recoge las posibles relaciones entre los ciudadanos interesados.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**INTERESADO**	TrInteresado	Interesado
→	**TIPOCONTACTO**	TrTipoContacto	Tipo de contacto
→	**SENTIDORELACION**	String	Sentido de la relación

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFTIPOCONT**	NUMÉRICO	Identificador del tipo de contato
→	**CAMPO_ABREVTIPOCONTACTO**	CADENA(10)	Abreviatura del tipo de contacto
→	**CAMPO_REFINTERESADO**	NUMÉRICO	Identificador del interesado
→	**CAMPO_CODTIPOIDENT**	CADENA(10)	Tipo de identificación
→	**CAMPO_NUMIDENT**	CADENA(15)	Número de identificación.
→	**CAMPO_APELLIDO1**	CADENA(50)	Primer apellido del interesado.
→	**CAMPO_APELLIDO2**	CADENA(50)	Segundo apellido del interesado.
→	**CAMPO_NOMBRE**	CADENA(100)	Nombre del interesado.
→	**CAMPO_SEXO**	CADENA(1)	Sexo del interesado. → '**F**' – Femenino → '**M**' – Masculino

→	**CAMPO_DIGCONTROL**	CADENA(1)	Dígito de control para el número de identificación.
→	**CAMPO_CIWA**	CADENA(15)	Código de identificación w@ndA.
→	**CAMPO_FECHANACIM**	DATE	Fecha de nacimiento.
→	**CAMPO_FECHABAJA**	DATE	Fecha de baja en el sistema.
→	**CAMPO_NUMREGMER**	CADENA(30)	Número del registro mercantil.
→	**CAMPO_FECHARCA**	DATE	Fecha de RCA de la entidad.
→	**CAMPO_EPIGRAFEIAE**	CADENA(30)	Epígrafe IAE.
→	**CAMPO_EPIGRAFECNAE**	CADENA(30)	Epígrafe CNAE.
→	**CAMPO_COMENTARIOS**	CADENA(100)	Comentarios.
→	**CAMPO_TIPOORGZ**	CADENA(1)	Tipo de organización.
→	**CAMPO_ANAGRAMAFISCAL**	CADENA(20)	Anagrama fiscal del interesado
→	**CAMPO_REFDATOCONT**	NUMÉRICO	Identificador de los datos de contacto

Métodos para aplicación de los filtros → obtenerRelacionesInteresado

TrSistema

trewa.bd.trapi.trapiui.tpo.TrSistema

Clase que representa la información referente a un sistema.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFSTMA**	TpoPK	Identificador del sistema
→	**CODSTMA**	String	Código del sistema
→	**DESCRIPCION**	String	Descripción del sistema

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema
→	**CAMPO_CODSTMA**	CADENA(10)	Código del sistema
→	**CAMPO_DESCRIPCION**	CADENA(128)	Descripción del sistema
→	**CAMPO_CONEXIONBUS**	CADENA(1)	Indica si se tiene conexión al bus. → '**S**' – Sí → '**N**' – No → '**O**' – Opcional

→	**CAMPO_REFCOMPINF**	NUMÉRICO	Identificador del componente a informar.
→	**CAMPO_NOMBRECOMPINF**	CADENA(50)	Nombre del componente que informa al bus
→	**CAMPO_REFCOMPAVI**	NUMÉRICO	Identificador del componente del @visador.
→	**CAMPO_NOMBRECOMPAVI**	CADENA(50)	Nombre del componente @visador
→	**CAMPO_REFCOMPGUAR**	NUMÉRICO	Identificador del componente que guarda los documentos.
→	**CAMPO_NOMBRECOMPGUAR**	CADENA(50)	Nombre del componente w@rdA
→	**CAMPO_REFCOMPARCH**	NUMÉRICO	Identificador del componente que archiva.
→	**CAMPO_NOMBRECOMPARCH**	CADENA(50)	Nombre del componente que archiva
→	**CAMPO_REFCOMPREG**	NUMÉRICO	Identificador del componente que registra los documentos.
→	**CAMPO_NOMBRECOMPREG**	CADENA(50)	Nombre del componente de registro
→	**CAMPO_REFCOMPFIRMA**	NUMÉRICO	Identificador del componente que firma los documentos.
→	**CAMPO_NOMBRECOMPFIRMA**	CADENA(50)	Nombre del componente de firma
→	**CAMPO_REFCOMPNOTI**	NUMÉRICO	Identificador del componente que notifica los documentos.
→	**CAMPO_NOMBRECOMPNOTI**	CADENA(50)	Nombre del componente de notificaciones

TrTareaExpediente

trewa.bd.trapi.trapiui.tpo.TrTareaExpediente

Clase que representa una tarea del expediente.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFTAREAEXP**	TpoPK	Identificador de la tarea del expediente
→	**REFTAREA**	TpoPK	Identificador de la tarea
→	**NOMBRETAREA**	String	Nombre de la tarea
→	**TIPO**	String	Tipo de la tarea → "**G**" – Generar → "**I**" – Incorporar → "**M**" – Manipular datos → "**O**" – Otras
→	**FECHACOMIENZO**	Timestamp	Fecha de comienzo de la tarea
→	**FECHALIMITE**	Timestamp	Fecha límite de realización de la tarea
→	**FECHAFINAL**	Timestamp	Fecha de finalización de la tarea
→	**ESTADO**	String	Estado de la tarea → "**I**" – Iniciada → "**D**" – Descartada → "**F**" – Finalizada
→	**OBSERVACIONES**	String	Observaciones de la tarea

→	**REFFASE**	TpoPK	Identificador de la fase
→	**NOMBREFASE**	String	Nombre de la fase
→	**REFEXPFASE**	TpoPK	Identificador del expediente en fase
→	**REFDEFPROC**	TpoPK	Identificador de la definición del procedimiento
→	**USUARIO**	String	Usuario
→	**NOMBREUSU**	String	Nombre y apellidos del usuario
→	**MULTIPLE**	String	Indica si es múltiple → "**S**" – Sí → "**N**" – No
→	**TEXTOAUXILIAR**	String	Texto auxiliar
→	**REFTAREALLAMANTE**	TpoPK	Identificador de la tarea llamante
→	**NOMBRETAREALLAMANTE**	String	Nombre de la tarea llamante
→	**INFORMADABUS**	String	Indica si se informa al bus del inicio de la tarea → "**S**" – Sí → "**N**" – No

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFTAREAEXP**	NUMÉRICO	Identificador de la tarea del expediente
→	**CAMPO_REFTAREA**	NUMÉRICO	Identificador de la tarea
→	**CAMPO_NOMBRETAREA**	CADENA(50)	Nombre de la tarea
→	**CAMPO_TIPO**	CADENA(1)	Tipo de la tarea → "**G**" – Generar → "**I**" – Incorporar → "**M**" – Manipular datos → "**O**" – Otras
→	**CAMPO_FECHACOMIENZO**	DATE	Fecha de comienzo de la tarea
→	**CAMPO_FECHALIMITE**	DATE	Fecha límite de realización de la tarea
→	**CAMPO_FECHAFINAL**	DATE	Fecha de finalización de la tarea
→	**CAMPO_ESTADO**	CADENA(1)	Estado de la tarea → "**I**" – Iniciada → "**D**" – Descartada → "**F**" – Finalizada
→	**CAMPO_OBSERVACIONES**	CADENA(250)	Observaciones de la tarea
→	**CAMPO_REFFASE**	NUMÉRICO	Identificador de la fase
→	**CAMPO_NOMBREFASE**	CADENA(50)	Nombre de la fase
→	**CAMPO_REFEXPFASE**	NUMÉRICO	Identificador del expediente en fase
→	**CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento
→	**CAMPO_USUARIO**	CADENA(10)	Usuario
→	**CAMPO_NOMBREUSU**	CADENA(98)	Nombre y apellidos del usuario
→	**CAMPO_MULTIPLE**	CADENA(1)	Indica si es múltiple → "**S**" – Sí

			→ "**N**" – No
→	**CAMPO_TEXTOAUXILIAR**	CADENA(50)	Texto auxiliar
→	**CAMPO_REFTAREALLAMANTE**	NUMÉRICO	Identificador de la tarea llamante
→	**CAMPO_NOMBRETAREALLAMANTE**	CADENA(50)	Nombre de la tarea llamante
→	**CAMPO_INFORMADABUS**	CADENA(1)	Indica si se informa al bus del inicio de la tarea
			→ "**S**" – Sí
			→ "**N**" – No

Métodos para aplicación de los filtros → obtenerTareasExpediente

TrTareaPermitida

trewa.bd.trapi.trapiui.tpo.TrTareaPermitida

Clase que representa la información de una tarea permitida.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFTAREAFASE**	TpoPK	Identificador de la tarea en fase
→	**REFTAREA**	TpoPK	Identificador de la tarea
→	**TAREA**	String	Nombre de la tarea
→	**ETIQUETA**	String	Etiqueta de la tarea
→	**ETIQUETALARGA**	String	Etiqueta larga para la tarea
→	**DESCRIPCION**	String	Descripción de la tarea
→	**TIPO**	String	Tipo de la tarea permitida
			→ "**G**" – Generar
			→ "**I**" – Incorporar
			→ "**M**" – Manipular datos
			→ "**O**" – Otras
→	**PERMISO**	String	Permiso de la tarea
→	**OBLIGATORIA**	String	Indica si es obligatoria su realización
			→ "**S**" – Sí
			→ "**N**" – No
→	**PLAZO**	TrPlazo	Plazo de realización
→	**MULTIPLE**	String	Indica si es múltiple
			→ "**S**" – Sí
			→ "**N**" – No
→	**TEXTOAUXILIAR**	String	Texto auxiliar
→	**REFTAREALLAMANTE**	TpoPK	Identificador de la tarea llamante

→	**TAREALLAMANTE**	String	Nombre de la tarea llamante
→	**ORDEN**	Integer	Orden de la tarea
→	**INFORMARBUS**	String	Indica si se informa al bus de la realización de la tarea → "**S**" – Sí → "**N**" – No

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFTAREAFASE**	NUMÉRICO	Identificador de la tarea en fase
→	**CAMPO_REFTAREA**	NUMÉRICO	Identificador de la tarea
→	**CAMPO_TAREA**	CADENA(50)	Nombre de la tarea
→	**CAMPO_ETIQUETA**	CADENA(11)	Etiqueta de la tarea
→	**CAMPO_ETIQUETALARGA**	CADENA(25)	Etiqueta larga para la tarea
→	**CAMPO_DESCRIPCION**	CADENA(50)	Descripción de la tarea
→	**CAMPO_TIPO**	CADENA(1)	Tipo de la tarea permitida → "**G**" – Generar → "**I**" – Incorporar → "**M**" – Manipular datos → "**O**" – Otras
→	**CAMPO_PERMISO**	CADENA(1)	Permiso de la tarea
→	**CAMPO_OBLIGATORIA**	CADENA(1)	Indica si es obligatoria su realización → "**S**" – Sí → "**N**" – No
→	**CAMPO_MULTIPLE**	CADENA(1)	Indica si es múltiple → "**S**" – Sí → "**N**" – No
→	**CAMPO_TEXTOAUXILIAR**	CADENA(50)	Texto auxiliar
→	**CAMPO_REFTAREALLAMANTE**	NUMÉRICO	Identificador de la tarea llamante
→	**CAMPO_TAREALLAMANTE**	CADENA(50)	Nombre de la tarea llamante
→	**CAMPO_ORDEN**	NUMÉRICO	Orden de la tarea
→	**CAMPO_INFORMARBUS**	CADENA(1)	Indica si se informa al bus de la realización de la tarea → "**S**" – Sí → "**N**" – No
→	**CAMPO_DESCPLAZO**	CADENA(100)	Descripción de la fecha límite del plazo
→	**CAMPO_UNIDAD**	CADENA(1)	Unidad de tiempo → "**D**" – Días → "**M**" – Meses → "**A**" – Años
→	**CAMPO_NUMUNIDADES**	NUMÉRICO	Número de unidades

Métodos para aplicación de los filtros → `obtenerTareasPermitidas`

TrTextoDisposicion

trewa.bd.trapi.trapiui.tpo.TrTextoDisposicion

Clase que representa la información referente al texto de una disposición.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFDISPOSICION**	TpoPK	Identificador del texto de la disposición
→	**DESCRIPCION**	String	Texto de la disposición
→	**TIPO**	String	Tipo de disposición → "**D**" – Decreto → "**T**" – Otros → "**O**" – Orden

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFDISPOSICION**	NUMÉRICO	Identificador del texto de la disposición
→	**CAMPO_DESCRIPCION**	CADENA(100)	Texto de la disposición
→	**CAMPO_TIPO**	CADENA(1)	Tipo de disposición → "**D**" – Decreto → "**T**" – Otros → "**O**" – Orden

Métodos para aplicación de los filtros → obtenerDisposicionesFirmante

TrTipoActo

trewa.bd.trapi.trapiui.tpo.TrTipoActo

Clase que representa a la información asociada a un tipo de acto administrativo.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFTIPOACTO**	TpoPK	Identificador del acto administrativo
→ **ABREVIATURA**	String	Abreviatura del acto
→ **DESCRIPCION**	String	Descripción del acto administrativo
→ **STMA**	TrSistema	Sistema

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFTIPOACTO**	NUMÉRICO	Identificador del acto administrativo
→ **CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura del acto
→ **CAMPO_DESCRIPCION**	CADENA(200)	Descripción del acto administrativo
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema del tipo de acto

Métodos para aplicación de los filtros → obtenerTiposActoAdmDefProcedimiento

TrTipoContacto

trewa.bd.trapi.trapiui.tpo.TrTipoContacto

Recoge los distintos tipos de contacto que pueden existir entre ciudadanos.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFTIPOCONT**	TpoPK	Identificador del tipo de contacto.
→ **ABREVIATURA**	String	Abreviatura del tipo de contacto.
→ **DESCRIPCION**	String	Descripción del tipo de contacto.
→ **CODWANDA**	String	Valor en w@ndA del tipo de contacto.
→ **OBSOLETO**	String	Indica si el tipo de contacto está obsoleto. → "**S**" – Sí → "**N**" – No

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_ REFTIPOCONT**	NUMÉRICO	Identificador del tipo de contacto.

→	**CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura del tipo de contacto.
→	**CAMPO_DESCRIPCION**	CADENA(50)	Descripción del tipo de contacto.
→	**CAMPO_CODWANDA**	NUMÉRICO	Valor en w@ndA del tipo de contacto.
→	**CAMPO_OBSOLETO**	CADENA(1)	Indica si el tipo de contacto está obsoleto. → "**S**" – Sí → "**N**" – No

Métodos para aplicación de los filtros → obtenerTiposContacto

TrTipoDocumento

trewa.bd.trapi.trapiui.tpo.TrTipoDocumento

Clase que representa la información referente a tipo de documento.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFTIPODOC**	TpoPK	Identificador del tipo de documento
→	**ETIQUETA**	String	Etiqueta del tipo de documento
→	**NOMBRE**	String	Nombre del tipo de documento
→	**DESCRIPCION**	String	Descripción del tipo de documento
→	**ENTRADASALIDA**	String	Indica cómo está definido el documento en TREW@ → "**E**" – Entrada → "**S**" – Salida → "**ES**" – Entrada / Salida
→	**INCGEN**	String	Indica cómo está definido el documento → "**I**" – Documento a Incorporar al expediente → "**G**" – Documento a Generar
→	**STMA**	TrSistema	Sistema
→	**MULTIPLE**	String	Indica si el tipo de documento es múltiple o no. → "**S**" – Tipo de documento es múltiple → "**N**" – Tipo de documento no es múltiple
→	**TEXTO_AUXILIAR**	String	Texto auxiliar de libre uso para las aplicaciones que utilizan TREW@
→	**INFORMAR**	String	Indica si se informa al bus del tipo de documento → "**S**" – Sí → "**N**" – No
→	**VERSIONABLE**	String	Indica si el tipo de documento es versionable o no. → "**S**" – Tipo de documento es versionable

			→ "**N**" – Tipo de documento no es versionable
→	**REUTILIZABLE**	String	Indica si el tipo de documento es reutilizable o no. → "**S**" – Tipo de documento es reutilizable → "**N**" – Tipo de documento no es reutilizable
→	**MODOGEN**	String	Modo de generación del documento → "**R**" – Report Server Oracle → "**P**" – Generación PDF → "**O**" – OpenOffice Bean
→	**FUSIONARVAR**	String	Indica si se fusionan las variables al generar → "**S**" – Sí → "**N**" – No
→	**FIRMADIGI**	String	Indica si se puede firmar digitalmente. → "**S**" – Sí → "**N**" – No
→	**TIPOFIRMA**	String	Tipo de firma para el documento. → "**C**" – Cascada → "**P**" – Paralelo
→	**PLANTILLAOFFICE**	byte[]	Plantilla binaria para Open Office.
→	**FORMATO**	String	Tipo de formato de la plantilla (tipo mime).
→	**NOMBREFICHERO**	String	Nombre del fichero de la plantilla.
→	**REGISTRABLE**	String	Indica si el tipo de documento es registrable o no. → "**S**" – Tipo de documento es registrable → "**N**" – Tipo de documento no es registrable
→	**NOTIFICABLE**	String	Indica si el tipo de documento es notificable o no. → "**S**" – Tipo de notificable es registrable → "**N**" – Tipo de notificable no es registrable
→	**OBSOLETO**	String	Indica si el tipo de documento está obsoleto. → "**S**" – Sí → "**N**" – No
→	**CODWANDA**	String	Valor en w@ndA del tipo de documento.
→	**PLANTILLA**	TrPlantilla	Plantilla utilizada para la generación de documentos Report y Java PDF.

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFTIPODOC**	NUMÉRICO	Identificador del tipo de documento
→	**CAMPO_ETIQUETA**	CADENA(10)	Abreviatura del tipo de documento
→	**CAMPO_NOMBRE**	CADENA(200)	Nombre del tipo de documento
→	**CAMPO_DESCRIPCION**	CADENA	Descripción del tipo de documento
→	**CAMPO_ENTRADASALIDA**	CADENA(1)	Indica cómo está definido el documento en TREW@ → "**E**" – Entrada

			→ "**S**" – Salida
			→ "**ES**" – Entrada / Salida
→	**CAMPO_INCGEN**	CADENA(1)	Indica cómo está definido el documento
			→ "**I**" – Documento a Incorporar al expediente
			→ "**G**" – Documento a Generar
→	**CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema
→	**CAMPO_MULTIPLE**	CADENA(1)	Indica si el tipo de documento es múltiple o no.
			→ "**S**" – Tipo de documento es múltiple
			→ "**N**" – Tipo de documento no es múltiple
→	**CAMPO_TEXTOAUXILIAR**	CADENA(50)	Texto auxiliar de libre uso para las aplicaciones que utilizan TREW@
→	**CAMPO_INFORMAR**	CADENA(1)	Indica si se informa al bus del tipo de documento
			→ "**S**" – Sí
			→ "**N**" – No
→	**CAMPO_VERSIONABLE**	CADENA(1)	Indica si el tipo de documento es versionable o no.
			→ "**S**" – Tipo de documento es versionable
			→ "**N**" – Tipo de documento no es versionable
→	**CAMPO_REUTILIZABLE**	CADENA(1)	Indica si el tipo de documento es reutilizable o no.
			→ "**S**" – Tipo de documento es reutilizable
			→ "**N**" – Tipo de documento no es reutilizable
→	**CAMPO_MODOGEN**	CADENA(1)	Modo de generación del documento
			→ "**R**" – Report Server Oracle
			→ "**P**" – Generación PDF
			→ "**O**" – OpenOffice Bean
→	**CAMPO_FUSIONAR**	CADENA(1)	Indica si se fusionan las variables al generar
			→ "**S**" – Sí
			→ "**N**" – No
→	**CAMPO_FIRMADIGI**	CADENA(1)	Indica si se puede firmar digitalmente.
			→ "**S**" – Sí
			→ "**N**" – No
→	**CAMPO_TIPOFIRMA**	CADENA(1)	Tipo de firma para el documento.
			→ "**C**" – Cascada
			→ "**P**" – Paralelo
→	**CAMPO_FORMATO**	CADENA(128)	Tipo de formato de la plantilla (tipo mime).
→	**CAMPO_NOMBREFICHERO**	CADENA(64)	Nombre del fichero de la plantilla.
→	**CAMPO_REGISTRABLE**	CADENA(1)	Indica si el tipo de documento es registrable o no.
			→ "**S**" – Tipo de documento es registrable
			→ "**N**" – Tipo de documento no es registrable
→	**CAMPO_NOTIFICABLE**	CADENA(1)	Indica si el tipo de documento es notificable o no.
			→ "**S**" – Tipo de notificable es registrable
			→ "**N**" – Tipo de notificable no es registrable
→	**CAMPO_OBSOLETO**	CADENA(1)	Indica si el tipo de documento está obsoleto.
			→ "**S**" – Sí
			→ "**N**" – No

→	**CAMPO_CODWANDA**	CADENA(30)	Valor en w@ndA del tipo de documento.
→	**CAMPO_REFPLANTILLA**	NUMÉRICO	Identificador de la plantilla para la generación de documentos Report o JavaPDF
→	**CAMPO_NOMBREPLANTILLA**	CADENA(15)	Nombre de la plantilla
→	**CAMPO_DESCPLANTILLA**	CADENA(250)	Descripción de la plantilla
→	**CAMPO_NOMBINFORME**	CADENA(15)	Nombre del informe de la plantilla

Métodos para aplicación de los filtros → obtenerTiposDocumento

TrTipoExpediente

trewa.bd.trapi.trapiui.tpo.TrTipoExpediente

Clase que representa la información referente a tipos de expediente.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFTIPOEXP**	TpoPK	Identificador del tipo de expediente
→	**ABREVIATURA**	String	Abreviatura del tipo de expediente
→	**DESCRIPCION**	String	Descripción del tipo de expediente
→	**VIGENTE**	String	Indica si el tipo de expediente está vigente en TREW@ → "**S**" – El tipo de expediente Sí está vigente → "**N**" – El tipo de expediente No está vigente
→	**STMA**	TrSistema	Sistema al que está asociado el tipo de expediente.

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFTIPOEXP**	NUMÉRICO	Identificador del tipo de expediente
→	**CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura del tipo de expediente
→	**CAMPO_DESCRIPCION**	CADENA(50)	Descripción del tipo de expediente
→	**CAMPO_VIGENTE**	CADENA(1)	Indica si el tipo de expediente está vigente en TREW@ → "**S**" – El tipo de expediente Sí está vigente → "**N**" – El tipo de expediente No está vigente
→	**CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema al que está asociado el tipo de expediente.

Métodos para aplicación de los filtros → `obtenerTiposExpediente`

TrTipoIdentificador

trewa.bd.trapi.trapiui.tpo.TrTipoIdentificador

Clase que representa la información de los distintos tipos de identificadores (DNI, NIF...).

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **CODTIPOIDENT**	`String`	Código del tipo de identificador.
→ **DESCRIPCION**	`String`	Descripción del tipo de identificador.
→ **TIPOIDENT**	`String`	Indica si se trata de una persona física o jurídica. → "**F**" – Física → "**J**" – Jurídica
→ **CODWANDA**	`String`	Valor en w@ndA del tipo de identificador
→ **OBSOLETO**	`String`	Indica si el tipo de identificador está obsoleto. → "**S**" – Física → "**N**" – Jurídica

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CODTIPOIDENT**	`CADENA(10)`	Código del tipo de identificador.
→ **CAMPO_DESCRIPCION**	`CADENA(100)`	Descripción del tipo de identificador.
→ **CAMPO_TIPOIDENT**	`CADENA(1)`	Indica si se trata de una persona física o jurídica. → "**F**" – Física → "**J**" – Jurídica
→ **CAMPO_CODWANDA**	`CADENA(30)`	Valor en w@ndA del tipo de identificador
→ **CAMPO_OBSOLETO**	`CADENA(1)`	Indica si el tipo de identificador está obsoleto. → "**S**" – Sí → "**N**" – No

Métodos para aplicación de los filtros → `obtenerTiposIdentificador`

TrTipoOrganismo

trewa.bd.trapi.trapiui.tpo.TrTipoOrganismo

Clase que representa la información de los distintos tipos de organismos.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFTIPOORG**	TpoPK	Identificador del tipo de organismo.
→ **DESCRIPCION**	String	Descripción del tipo de organismo.
→ **ABREVIATURA**	String	Abreviatura del tipo de organismo.
→ **CODWANDA**	String	Valor en w@ndA del tipo de organismo
→ **OBSOLETO**	String	Indica si el tipo de organismo está obsoleto. → "**S**" – Sí → "**N**" – No

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFTIPOORG**	NUMÉRICO	Identificador del tipo de organismo.
→ **CAMPO_DESCRIPCION**	CADENA(100)	Descripción del tipo de organismo.
→ **CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura del tipo de organismo.
→ **CAMPO_CODWANDA**	CADENA(1)	Valor en w@ndA del tipo de organismo
→ **CAMPO_OBSOLETO**	CADENA(1)	Indica si el tipo de organismo está obsoleto. → "**S**" – Sí → "**N**" – No

Métodos para aplicación de los filtros → `obtenerTiposOrganismo`

TrTipoOrganizacion

trewa.bd.trapi.trapiui.tpo.TrTipoOrganizacion

Clase que representa la información de los distintos tipos de organización.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **CODTIPOORGZ**	`String`	Identificador del tipo de organización.
→ **DESCRIPCION**	`String`	Descripción del tipo de organización.
→ **VIGENTE**	`String`	Indica si el tipo de organización está vigente. → "**S**" – Sí → "**N**" – No

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_CODTIPOORGZ**	`CADENA(1)`	Identificador del tipo de organización.
→ **CAMPO_DESCRIPCION**	`CADENA(100)`	Descripción del tipo de organización.
→ **CAMPO_VIGENTE**	`CADENA(1)`	Indica si el tipo de organización está vigente. → "**S**" – Sí → "**N**" – No

Métodos para aplicación de los filtros → `obtenerTiposOrganizacion`

TrTipoParrafo

trewa.bd.trapi.trapiui.tpo.TrTipoParrafo

Clase que representa la información referente a un tipo de párrafo.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFTIPOPARR**	`TpoPK`	Identificador del tipo de párrafo
→ **ABREVIATURA**	`String`	Abreviatura del tipo de párrafo
→ **DESCRIPCION**	`String`	Descripción del tipo de párrafo
→ **ETIQUETA**	`String`	Etiqueta del tipo de párrafo
→ **PARRAFO**	`String`	Texto del tipo de párrafo
→ **ALINEACION**	`String`	Tipo de alineación del tipo de párrafo → "**I**" – A la izquierda → "**C**" – Centrada → "**D**" – A la derecha

			→ "**J**" – Justificada
→	**ESTILO**	String	Estilo del tipo de párrafo
→	**ESTILOETIQ**	String	Estilo de la etiqueta del tipo de párrafo
→	**UBICACIÓN**	String	Ubicación del tipo de párrafo
			→ "**A**" – Cabecera
			→ "**C**" – Cuerpo
			→ "**P**" – Pie
			→ "**O**" – Otro
→	**EDITABLE**	String	Indica si el tipo de párrafo es editable o no
			→ "**S**" – El tipo de párrafo Sí es editable
			→ "**N**" – El tipo de párrafo No es editable
→	**STMA**	TrSistema	Sistema al que pertenece el tipo de párrafo
→	**NOMBREFICHERO**	String	Nombre del fichero de la imagen
→	**FORMATO**	String	Formato o tipo mime del fichero
→	**IMAGEN**	byte[]	Imagen del tipo de párrafo

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFTIPOPARR**	NUMÉRICO	Identificador del tipo de párrafo
→	**CAMPO_ABREVIATURA**	CADENA(15)	Abreviatura del tipo de párrafo
→	**CAMPO_DESCRIPCION**	CADENA(250)	Descripción del tipo de párrafo
→	**CAMPO_ETIQUETA**	CADENA(15)	Etiqueta del tipo de párrafo
→	**CAMPO_PARRAFO**	CADENA(4000)	Texto del tipo de párrafo
→	**CAMPO_ALINEACION**	CADENA(1)	Tipo de alineación del tipo de párrafo
			→ "**I**" – A la izquierda
			→ "**C**" – Centrada
			→ "**D**" – A la derecha
			→ "**J**" – Justificada
→	**CAMPO_ESTILO**	CADENA(8)	Estilo del tipo de párrafo
→	**CAMPO_ESTILOETIQ**	CADENA(8)	Estilo de la etiqueta del tipo de párrafo
→	**CAMPO_UBICACIÓN**	CADENA(1)	Ubicación del tipo de párrafo
			→ "**A**" – Cabecera
			→ "**C**" – Cuerpo
			→ "**P**" – Pie
			→ "**O**" – Otro
→	**CAMPO_EDITABLE**	CADENA(1)	Indica si el tipo de párrafo es editable o no
			→ "**S**" – El tipo de párrafo Sí es editable
			→ "**N**" – El tipo de párrafo No es editable
→	**CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema al que pertenece el tipo de párrafo
→	**CAMPO_NOMBREFICHERO**	CADENA(64)	Nombre del fichero de la imagen
→	**CAMPO_FORMATO**	CADENA(128)	Formato o tipo mime del fichero

Métodos para aplicación de los filtros → obtenerTiposParrafo

TrTipoRelacion

trewa.bd.trapi.trapiui.tpo.TrTipoRelacion

Clase que representa la información referente a un tipo de relación.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFTIPORELA**	TpoPK	Identificador del tipo de relación
→	**NOMBRE**	String	Nombre del tipo de relación
→	**DESCRIPCION**	String	Descripción del tipo de relación
→	**STMA**	TrSistema	Sistema

TrTipoVia

trewa.bd.trapi.trapiui.tpo.TrTipoVia

Clase que representa la información de los tipos de vía.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**CODTIPOVIA**	String	Identificador del tipo de vía.
→	**DESCRIPCION**	String	Descripción del tipo de vía.
→	**CODWANDA**	String	Valor en w@ndA del tipo de vía
→	**OBSOLETO**	String	Indica si el tipo de vía está obsoleto. → "**S**" – Sí → "**N**" – No

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_CODTIPOVIA**	CADENA(10)	Identificador del tipo de vía.

→	**CAMPO_DESCRIPCION**	CADENA(32)	Descripción del tipo de vía.
→	**CAMPO_CODWANDA**	CADENA(15)	Valor en w@ndA del tipo de vía
→	**CAMPO_OBSOLETO**	CADENA(1)	Indica si el tipo de vía está obsoleto. → "**S**" – Sí → "**N**" – No

Métodos para aplicación de los filtros → obtenerTiposVia

TrTransicion

trewa.bd.trapi.trapiui.tpo.TrTransicion

Clase que representa la información referente a un transición.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFTRANSICION**	TpoPK	Identificador de la transición
→	**ETIQUETA**	String	Etiqueta de la transición
→	**DESCRIPCION**	String	Descripción de la transición
→	**TIPOACTO**	TrTipoActo	Tipo de acto que representa la transición
→	**FASEFIN**	TrFase	Fase final
→	**ETIQUETALARGA**	String	Etiqueta larga para la transición
→	**TIPO**	String	Tipo de la transición → "**N**" – Transición normal → "**D**" – Transición de divisisón → "**U**" – Transición de unión → "**ES**" – Evento que provoca salida → "**EN**" – Evento que no provoca la salida
→	**DESCFECHA**	String	Descripción de lo que significa la fecha de realización de la transición
→	**PLAZO**	TrPlazo	Plazo de realización
→	**REFTRANPROV**	TpoPK	Identificador de la transición provocada por finalizar el plazo
→	**NUMMAX**	Integer	Número máximo de posibles repeticiones para la transición en el procedimiento
→	**ORDEN**	Integer	Orden entre las posibles transiciones que se pueden dar desde una determinada fase
→	**INFORMARBUS**	String	Indica si se informa al bus del cambio de fase → "**S**" – Sí → "**N**" – No

→ **REFFASEINI** TpoPK Identificador de la fase inicial

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFTRANSICION**	NUMÉRICO	Identificador de la transición
→ **CAMPO_ETIQUETA**	CADENA(11)	Etiqueta de la transición
→ **CAMPO_DESCRIPCION**	CADENA(50)	Descripción de la transición
→ **CAMPO_REFTIPOACTO**	NUMÉRICO	Identificador del tipo de acto que representa la transición
→ **CAMPO_ABREVTIPOACTO**	CADENA(10)	Abreviatura del tipo de acto
→ **CAMPO_DESCTIPOACTO**	CADENA(200)	Descripción del tipo de acto
→ **CAMPO_ETIQUETALARGA**	CADENA(25)	Etiqueta larga de la transición
→ **CAMPO_TIPO**	CADENA(2)	Tipo de la transición → "**N**" – Transición normal → "**D**" – Transición de divisisón → "**U**" – Transición de unión → "**ES**" – Evento que provoca salida → "**EN**" – Evento que no provoca la salida
→ **CAMPO_DESCFECHA**	CADENA(50)	Descripción de lo que significa la fecha de realización de la transición
→ **CAMPO_REFFASEFIN**	NUMÉRICO	Identificador de la fase fin de la transición
→ **CAMPO_REFFASEINI**	NUMÉRICO	Identificador de la fase inicial de la transición
→ **CAMPO_UNIDAD**	CADENA(1)	Unidad de tiempo → "**D**" – Días → "**M**" – Meses → "**A**" – Años
→ **CAMPO_NUMUNIDADES**	NUMÉRICO	Número de unidades de tiempo
→ **CAMPO_DESCFECHALIMITE**	CADENA(100)	Descripción de la fecha límite
→ **CAMPO_REFTRANPROV**	NUMÉRICO	Identificador de la transición provocada por la fecha límite
→ **CAMPO_NUMMAX**	NUMÉRICO	Número máximo de posibles repeticiones para la transición en el procedimiento
→ **CAMPO_ORDEN**	NUMÉRICO	Orden de la transición
→ **CAMPO_INFORMARBUS**	CADENA(1)	Indica si se informa al bus del cambio de fase → "**S**" – Sí → "**N**" – No

Métodos para aplicación de los filtros
→ obtenerTransicionesPermitidas
→ obtenerEventosPosibles
→ obtenerDatosTransicion

TrTransicionDefProcedimiento

trewa.bd.trapi.trapiui.tpo.TrTransicionDefProcedimiento

Clase que representa la información referente a una transición de un procedimiento.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**TRANSICION**	TrTransicion	Transición
→	**REFFASEPADRE**	TpoPK	Para fases pertenecientes a módulos reutilizables, indica la fase que representa dicho procedimiento.
→	**DEFPROC**	TrDefProcedimiento	Si la transición pertenece a un módulo reutilizable, contiene la definición del procedimiento que representa dicho módulo
→	**NUMMAX**	Integer	Número máximo de veces por la que se puede pasar por una transición en un procedimiento
→	**REFTRANPROV**	TpoPK	Identificador de la transición provocada por la fecha límite

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFTRANSICION**	NUMÉRICO	Identificador de la transición
→	**CAMPO_ETIQUETA**	CADENA(11)	Etiqueta de la transición
→	**CAMPO_DESCRIPCION**	CADENA(50)	Descripción de la transición
→	**CAMPO_REFTIPOACTO**	NUMÉRICO	Identificador del tipo de acto que representa la transición
→	**CAMPO_REFFASE**	NUMÉRICO	Identificador de la fase en la que se quedaría el expediente con la transición
→	**CAMPO_FASE**	CADENA(50)	Nombre de la fase o situación en la que quedaría el expediente con la transición
→	**CAMPO_METAFASE**	CADENA(50)	Nombre la metafase a la que corresponde la fase en la que quedaría el expediente con la transición
→	**CAMPO_UNIDAD**	CADENA(1)	Unidad de tiempo → "**D**" – Días → "**M**" – Meses → "**A**" – Años
→	**CAMPO_NUMUNIDADES**	NUMÉRICO	Número de unidades de tiempo
→	**CAMPO_DESCFECHALIM**	CADENA(100)	Descripción de la fecha límite
→	**CAMPO_REFTRANPROV**	NUMÉRICO	Identificador de la transición provocada por la fecha límite
→	**CAMPO_NUMMAX**	NUMÉRICO	Número máximo de posibles repeticiones para la transición en el procedimiento

Métodos para aplicación de los filtros → obtenerTransicionesDefProcedimiento

TrUsuario

trewa.bd.trapi.trapiui.tpo.TrUsuario

Clase que representa la información del usuario.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**CODUSUARIO**	String	Identificador del usuario
→	**TIPOIDENT**	String	Tipo de identificador. → "**D**" – DNI / NIF → "**P**" – Pasaporte → "**N**" – NIE → "**C**" – CIF → "**O**" – Otros
→	**IDENTIFICADOR**	String	Código de identificación
→	**NOMBRE**	String	Nombre del usuario
→	**APELLIDO1**	String	Primer apellido del usuario.
→	**APELLIDO2**	String	Segundo apellido del usuario.
→	**SEXO**	String	Sexo del usuario. → "**F**" – Femenino → "**M**" – Masculino
→	**EMAIL**	String	Dirección de correo electrónico del usuario.
→	**CLAVE**	String	Clave del usuario (típicamente encriptada).
→	**FECHAALTA**	Timestamp	Fecha de alta en el sistema.
→	**FECHABAJA**	Timestamp	Fecha de baja en el sistema.
→	**ANAGRAMAFISCAL**	String	Anagrama fiscal del usuario
→	**FIRMA**	TrFirma	Datos de la firma del usuario

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_CODUSUARIO**	CADENA(10)	Identificador del usuario
→	**CAMPO_TIPOIDENT**	CADENA(1)	Tipo de identificador. → "**D**" – DNI / NIF → "**P**" – Pasaporte

			→ "**N**" – NIE
			→ "**C**" – CIF
			→ "**O**" – Otros
→	**CAMPO_IDENTIFICADOR**	CADENA(15)	Código de identificación
→	**CAMPO_NOMBRE**	CADENA(32)	Nombre del usuario
→	**CAMPO_APELLIDO1**	CADENA(32)	Primer apellido del usuario.
→	**CAMPO_APELLIDO2**	CADENA(32)	Segundo apellido del usuario.
→	**CAMPO_SEXO**	CADENA(1)	Sexo del usuario.
			→ "**F**" – Femenino
			→ "**M**" – Masculino
→	**CAMPO_EMAIL**	CADENA(64)	Dirección de correo electrónico del usuario.
→	**CAMPO_CLAVE**	CADENA(32)	Clave del usuario (típicamente encriptada).
→	**CAMPO_FECHAALTA**	DATE	Fecha de alta en el sistema.
→	**CAMPO_FECHABAJA**	DATE	Fecha de baja en el sistema.
→	**CAMPO_ANAGRAMAFISCAL**	VARCHAR2(50)	Anagrama fiscal del usuario
→	**CAMPO_REFFIRMA**	NUMÉRICO	Identificador de la firma del usuario
→	**CAMPO_FORMATO**	VARCHAR2(128)	Formato de la imagen de la firma
→	**CAMPO_NOMBREFICHERO**	VARCHAR2(64)	Nombre del fichero de la imagen de la firma

Métodos para aplicación de los filtros → obtenerUsuarios

TrUsuarioAsignado

trewa.bd.trapi.trapiui.tpo.TrUsuarioAsignado

Clase que representa la información de los usuarios asignados al expediente.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**USUARIO**	TrUsuario	Usuario
→	**TIPO**	String	Tipo de usuario asignado.
			→ "**P**" – Principal
			→ "**S**" – Secundario
→	**FECHAALTA**	Timestamp	Fecha de asignación al expediente.
→	**FECHABAJA**	Timestamp	Fecha de baja.
→	**RAZONASIGNACION**	String	Razón de la asignación.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_USUARIO**	CADENA(10)	Usuario
→ **CAMPO_TIPO**	CADENA(1)	Tipo de usuario asignado. → "**P**" – Principal → "**S**" – Secundario
→ **CAMPO_FECHAALTA**	DATE	Fecha de asignación al expediente.
→ **CAMPO_FECHABAJA**	DATE	Fecha de baja.
→ **CAMPO_RAZONASIGNACION**	CADENA(250)	Razón de la asignación.

Métodos para aplicación de los filtros → obtenerUsuariosAsignados

TrValorParametro

trewa.bd.trapi.trapiui.tpo.TrValorParametro

Clase que representa la información referente a un valor de un parámetro.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **PARAMETRO**	String	Nombre del parámetro
→ **VALOR**	String	Valor del parámetro

TrVariable

trewa.bd.trapi.trapiui.tpo.TrVariable

Clase que representa la información referente a una variable de un sistema.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFVARIABLE**	TpoPK	Identificador de la variable
→ **NOMBRE**	String	Nombre de la variable
→ **DESCRIPCION**	String	Descripción de la variable
→ **FUNCION**	String	Nombre de la función
→ **STMA**	TrSistema	Sistema

→	**TIPOACTO**	TrTipoActo	Tipo de acto
→	**PAQUETE**	String	Nombre del paquete que contiene la función almacenada o método java.
→	**IMPLEMENTACION**	String	Tipo de implementación → "**F**" – Implementación PL-SQL → "**J**" – Implementación Java

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_REFVARIABLE**	NUMÉRICO	Identificador de la variable
→	**CAMPO_NOMBRE**	CADENA(25)	Nombre de la variable
→	**CAMPO_DESCRIPCION**	CADENA(250)	Descripción de la variable
→	**CAMPO_REFTIPOACTO**	NUMÉRICO	Identificador del tipo de acto
→	**CAMPO_FUNCION**	CADENA(50)	Nombre de la función
→	**CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema
→	**CAMPO_PAQUETE**	CADENA(100)	Nombre del paquete que contiene la función almacenada o método java.
→	**CAMPO_IMPLEMENTACION**	CADENA(1)	Tipo de implementación → "**F**" – Implementación PL-SQL → "**J**" – Implementación Java
→	**CAMPO_ABREVTIPOACTO**	CADENA(10)	Abreviatura del tipo de acto administrativo
→	**CAMPO_DESCTIPOACTO**	CADENA(200)	Descripcion del tipo de acto

Métodos para aplicación de los filtros → obtenerVariablesSistema

TrVariableDocExp

trewa.bd.trapi.trapiui.tpo.TrVariableDocExp

Clase que recoge las variables del documento. Sólo válido para plantillas de Open Office.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**REFVARIABLE**	TpoPK	Identificador de la variable
→	**NOMBRE**	String	Nombre de la variable
→	**DESCRIPCION**	String	Descripción de la variable

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFVARIABLE**	NUMÉRICO	Identificador de la variable
→ **CAMPO_NOMBRE**	CADENA(25)	Nombre de la variable
→ **CAMPO_DESCRIPCION**	CADENA(250)	Descripción de la variable

Métodos para aplicación de los filtros → obtenerVariablesDocumentoExp

TrVersionDefProcedimiento

trewa.bd.trapi.trapiui.tpo.TrVersionDefProcedimiento

Clase que representa una versión de procedimiento asociada a un tipo de expediente determinado.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **VERSION**	String	Descripción de la versión
→ **FECHAVIGOR**	Timestamp	Fecha desde la cual está en vigor la versión del flujo o más concretamente la normativa a la que está asociada
→ **DEFPROC**	TrDefProcedimiento	Definición de procedimiento que tiene asociado la versión
→ **TIPOEXP**	TrTipoExpediente	Tipo de expediente al que está asociado la versión
→ **STMA**	TrSistema	Sistema

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_VERSION**	CADENA(50)	Descripción de la versión
→ **CAMPO_FECHAVIGOR**	DATE	Fecha desde la cual está en vigor la versión del flujo o más concretamente la normativa a la que está asociada
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición de procedimiento que tiene asociado la versión
→ **CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura de la definición de procedimiento asociado

→	**CAMPO_DESCDEFPROC**	CADENA(50)	Descripción de la definición de procedimiento asociado
→	**CAMPO_REFTIPOEXP**	NUMÉRICO	Identificador del tipo de expediente al que está asociada la versión
→	**CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema de la versión
→	**CAMPO_DESCAMPDEFPROC**	CADENA(250)	Descripción ampliada del procedimiento
→	**CAMPO_CODWANDA**	CADENA(15)	Código w@ndA
→	**CAMPO_VIGENTEDEFPROC**	CADENA(1)	Indica si el procedimiento está vigente → "**S**" – Sí → "**N**" – No
→	**CAMPO_ABREVIATURATIPOEXP**	CADENA(10)	Abreviatura del tipo de expediente
→	**CAMPO_DESCTIPOEXP**	CADENA(50)	Descripción del tipo de expediente
→	**CAMPO_VIGENTETIPOEXP**	CADENA(1)	Indica si el tipo de expediente está vigente → "**S**" – Sí → "**N**" – No
→	**CAMPO_INFORMAR**	CADENA(1)	Indica si se informa al bus del procedimiento y de los expedientes que se tramiten

Métodos para aplicación de los filtros → `obtenerVersionesDefProcedimiento`

Servlets de subida y descarga de documentos

Para la subida y descarga de documentos, junto con la TrAPIUI, se aportan dos servlets que interactúan con la misma.

Para que los servlets de subida y descarga de documentos funcionen, se debe crear con antelación un objeto de la clase TrAPIUI y ponerlo a nivel de la sesión en la que estamos trabajando (**apiUI**).

Servlet DescargaDocumento

El Servlet DescargaDocumento inicia la descarga de un documento haciendo uso de métodos del TrAPIUI.

Los parámetros del servlet se indican en la siguiente tabla:

Parámetro	Ámbito	Ob.	Descripción
docExp	Request	Sí	Identificador del documento del expediente a descargar
multiple	Request	No	Indica si el documento es múltiple (“S”) o no (“N”). El valor por defecto es “N”.
apiUI	Session	Sí	Interfaz TrAPIUI

Ejemplo de llamada al servlet

→ Llamada desde una URL

```
http://servidorWeb/aplicacionJSP/trewa/DescargaDoc?docExp=123
```

→ Llamada desde un javascript

```
location.href='trewa/DescargaDoc?docExp=123'
```

Servlet SubidaDocumento

El Servlet SubidaDocumento sube un documento a la base de datos haciendo uso de métodos del TrAPIUI.

Los parámetros del servlet se indican en la siguiente tabla:

Parámetro	Ámbito	Ob.	Descripción
doc	Request	Sí	Identificador del documento a subir
apiUI	Session	Sí	Interfaz TrAPIUI

Ejemplo de llamada al servlet

→ Llamada desde el action del form en JSP

```
<FORM ENCTYPE="multipart/form-data" ACTION="trewa/SubidaDoc?doc=1246" METHOD="POST">
```

Métodos de la TrAPIUI

En este apartado se describe el conjunto de métodos existente en la TrAPIUI, los cuáles se han clasificado acorde a la tipología y misión de cada uno, agrupando aquellos que están relacionados según la funcionalidad que ofrecen.

Todos los métodos son accesibles a través de la interfaz **TrAPIUI**

APIs sobre procedimientos definidos

Obtención de procedimientos definidos

```
TrDefProcedimiento[ ] obtenerDefProcedimientosDefinidos(
                TpoPK idSistema,
                ClausulaWhere where,
                ClausulaOrderBy orderBy ) throws TrException
```

Devuelve para un sistema dado, los datos de todos los procedimientos definidos en dicho sistema. Si no se indica sistema, devuelve todos los procedimientos definidos en TREW@.

Entradas

TpoPK idSistema	*Identificador del sistema* Si el identificador es 'null', el método devolverá todos los procedimientos definidos existentes.
ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrDefProcedimiento. Los filtros que usen *TrDefProcedimiento.CAMPO_REFSTMA* no tendrán efecto si el método recibe un identificador del sistema distinto de 'null'.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrDefProcedimiento.

Salida

TrDefProcedimiento [ ]	Array de objetos TrDefProcedimiento.

Obtención de tipos de expediente

```
TrTipoExpediente[ ] obtenerTiposExpediente(
        TpoPK idStma,
        ClausulaWhere where,
        ClausulaOrderBy orderBy ) throws TrException
```

Devuelve los tipos de expediente definidos en TREW@ para un sistema que se le pasa como parámetro (como clave primaria del sistema "idStma"). Si no se indica sistema, devuelve todos los tipos de expediente definidos en TREW@.

Entradas

TpoPK idStma	Identificador del sistema. Si el identificador es 'null' el método devolverá todos los tipos de expediente definidos en TREW@.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoExpediente. Los filtros que usen *TrTipoExpediente.CAMPO_REFSTMA* no tendrán efecto si el método recibe un identificador de sistema distinto de 'null'.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoExpediente.

Salida

TrTipoExpediente[ ]	Array de objetos TrTipoExpediente con la información de tipos de expediente.

Es competencia del sistema que utilice TREW@, establecer posibles relaciones entre los tipos de expediente existentes en el sistema y los tipos de expediente que se definen en TREW@.

Obtención de procedimientos y versiones de procedimientos

```
TrVersionDefProcedimiento[ ] obtenerVersionesDefProcedimiento(
        TpoPK idTipoExp,
        ClausulaWhere where,
        ClausulaOrderBy orderBy ) throws TrException
```

Devuelve para un tipo de expediente dado, los datos de todas las versiones de procedimiento que tiene asociados. Si no se indica tipo de expediente, devuelve todas las versiones definidas en TREW@.

Entradas

TpoPK idTipoExp	*Identificador del tipo de expediente.* Si el identificador es 'null', el método devolverá todas las versiones definidas en TREW@.

ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrVersionDefProcedimiento. Los filtros que usen *TrVersionDefProcedimiento.CAMPO_REFTIPOEXP* no tendrán efecto si el método recibe un identificador del tipo de expediente distinto de 'null'.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrVersionDefProcedimiento.

Salida

TrVersionDefProcedimiento [ ]	Array de objetos TrVersionDefProcedimiento.

APIs sobre expedientes

Alta de expedientes

```
TpoPK crearExpediente(
        TpoPK idTipoExp,
        TpoPK idDefProc,
        TpoDate fechaAlta,
        String numExp,
        String tituloExp,
        String observaciones,
        TpoPK idUniOrg,
        TpoPK idUniOrgEnvia,
        String urlWanda
        ) throws TrException
```

Método que da de alta un nuevo expediente para tramitar en Trew@, para un tipo de expediente dado y/o una definición de procedimiento y una fecha de alta del expediente. Al expediente se le asignará la versión del procedimiento que corresponde a la fecha de alta suministrada. Se debe indicar como mínimo uno de los dos identificadores, el del tipo de expediente o el de la definición del procedimiento.

Entradas

TpoPK idTipoExp	*Identificador del tipo de expediente.* Si no se indica identificador del tipo de expediente se deberá indicar el identificador del procedimiento.
TpoPK idDefProc	*Identificador del procedimiento a seguir.* Si no se indica identificador del procedimiento se deberá indicar el identificador del tipo de expediente.

TpoDate fechaAlta	*Fecha de alta.* Entrada/Salida. Si no se indica fecha de alta, por defecto se tomará la del sistema.
String numExp	*Número del expediente para la aplicación cliente*
String tituloExp	*Título del expediente para la aplicación cliente*
String observaciones	*Observaciones para el expediente*
TpoPK idUniOrg	*Identificador de la unidad orgánica u organismo a la que pertenece el expediente*
TpoPK idUniOrgEnvia	*Identificador de la unidad orgánica u organismo que envía el expediente*
String urlWanda	*Url del expediente en w@ndA*

Salida

TpoPK	*Identificador del nuevo expediente.*

Se ofrece la posibilidad de indicar el procedimiento a seguir por el expediente (aunque no sea la que le corresponda según la fecha de vigencia de la versión) siempre que exista una versión para el tipo de expediente que siga ese procedimiento.

El método devuelve el identificador único con el que Trew@ identifica el expediente, que será imprescindible para la mayoría de funciones, es decir, los sistemas deben definir clave ajena a la entidad de expedientes de Trew@. También se devuelve la fecha con la que ha sido dado de alta el expediente en Trew@.

Además en la nueva versión permite indicar la url del expediente en w@ndA.

Obtención de datos del expediente

```
TrExpediente obtenerDatosExpediente(
                TpoPK idExpediente) throws TrException
```

Devuelve para un expediente dado, los datos del expediente guardados en TREW@. Si no se indica identificador del expediente el método lanzará una excepción.

Entradas

TpoPK idExpediente	*Identificador del expediente.* Si el identificador es 'null', se generará una excepción

Salida

TrExpediente	*Objeto TrExpediente con los datos del expediente*

Eliminación de expedientes

```
void eliminarExpediente(
            TpoPK idExpediente ) throws TrException
```

Permite eliminar un expediente que se indica como parámetro, en Trew@. Está orientada a usuarios administradores.

Entradas

TpoPK idExpediente	*Identificador del expediente.* Si no se indica identificador del tipo de expediente el método devuelve una excepción.

Modificación del procedimiento seguido

```
void modificarDefProcedimientoExpediente(
            TpoPK idExpediente,
            TpoPK idTipoExp,
            TpoPK idDefProc,
            TpoDate fecha,
            String observaciones) throws TrException
```

Permite modificar, durante la tramitación de un expediente, el procedimiento que está siguiendo, es decir, permite cambiar el tipo de expediente y en consecuencia el procedimiento según la versión. También permite indicar el motivo del cambio de procedimiento para que así conste en la "historia de procedimientos" del expediente mediante el parámetro "observaciones".

Entradas

TpoPK idExpediente	*Identificador del expediente.* Si no se indica identificador del expediente se generará una excepción.
TpoPK idTipoExp	*Identificador del tipo de expediente.* Si no se indica identificador del tipo de expediente se generará una excepción.
TpoPK idDefProc	*Identificador del procedimiento a seguir.* Si no se indica identificador del procedimiento se generará una excepción.

TpoDate fecha	*Fecha de modificación.* Entrada/Salida. Si no se indica fecha de modificación, por defecto se tomará la del sistema.
String observaciones	*Observaciones para anotar el motivo de cambio en la "historia de flujos".*

Obtención de procedimiento(s) seguido(s) por un expediente

```
TrCambioProcedimientoExpediente[ ] obtenerDatosExpediente(
        TpoPK idExpediente,
        ClausulaWhere where,
        ClausulaOrderBy orderBy ) throws TrException
```

Devuelve para un expediente dado, los datos guardados en TREW@ asociados al expediente y al procedimiento que sigue o ha seguido.

Entradas

TpoPK idExpediente	*Identificador del expediente.* Si el identificador es 'null', se generará una excepción
ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrCambioProcedimientoExpediente.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrCambioProcedimientoExpediente.

Salida

TrCambioProcedimientoExpediente [ ]	*Array de objetos TrCambioProcedimientoExpediente.*

Estos datos pueden ser usados por un sistema para relacionar "físicamente" el expediente del sistema en cuestión con el expediente de TREW@.

Para obtener más datos del tipo de expediente o del procedimiento seguido, se deben utilizar las funciones "*obtenerTiposExpediente*" y "*obtenerVersionesProcedimiento*" descritas anteriormente.

(w) Modificación de expediente

```
void modificarDatosExpediente (
        TpoPK idExpediente,
        String numExp,
        String tituloExp,
        String observaciones,
        TpoPK idUniOrg,
        TpoPK idUniOrgEnvia,
```

```
                String urlWanda) throws TrException
```

Permite modificar los datos de un expediente. Si el identificador del expediente es 'null' o no existe se generará una excepción. Además si el expediente está archivado no se permitirá modificar sus datos, generando una excepción.

A este método se le deben pasar todos los parámetros aunque sólo se quiera modificar uno de ellos.

Si se define un componente w@rdA en el sistema establecido en el TrAPIUI y el documento del expediente está guardado en el mismo w@rdA se actualizan sus datos.

Además en la nueva versión permite indicar la url del expediente en w@ndA.

Entradas

TpoPK IdExpediente	*Identificador del expediente.* Si el identificador es 'null' se generará una excepción.
String numExp	*Número del expediente para la aplicación cliente*
String tituloExp	*Título del expediente para la aplicación cliente*
String observaciones	*Observaciones para el expediente*
TpoPK IdUniOrg	*Identificador de la unidad orgánica u organismo al que pertenece el expediente*
TpoPK idUniOrgEnvia	*Identificador de la unidad orgánica u organismo que envía el expediente.*
String urlWanda	*Url del expediente en w@ndA*

(w) Archivo de un expediente

```
void archivarExpediente(
                TpoPK idExpediente,
                TpoDate fecha) throws TrException
```

Permite archivar un expediente siempre que no esté archivado. Para ello se comprueba que el identificador del expediente sea válido y no sea un expediente ya archivado, generando una excepción en caso contrario. Además archiva todos los documentos archivables del expediente.

Entradas

TpoPK IdExpediente	*Identificador del expediente.* Si el identificador es 'null' se generará una excepción.
TpoDate fecha	*Fecha de archivo del expediente* Si no se indica fecha de archivo, por defecto se tomará la del sistema.

Anular el archivo de un expediente

```
void anularArchivoExpediente (
        TpoPK idExpediente) throws TrException
```

Anula el archivo de un expediente siempre que esté archivado. Para ello se comprueba que el identificador del expediente sea válido y no sea un expediente que no está archivado, generando una excepción en caso contrario. Además anula la fecha de archivo de todos los documentos archivables del expediente.

Entradas

TpoPK IdExpediente	*Identificador del expediente.* Si el identificador es 'null' se generará una excepción.

Obtención de relaciones entre expedientes

```
TrRelacionExpediente[ ] obtenerRelacionesExpediente(
        TpoPK idExpediente,
        ClausulaWhere where,
        ClausulaOrderBy orderBy) throws TrException
```

Permite obtener las relaciones del expediente que se pasa por parámetros con otros expedientes. Devuelve tanto las relaciones en las que el expediente sea maestro como detalle de otros.

Entradas

TpoPK IdExpediente	*Identificador del expediente.* Si el identificador es 'null' se generará una excepción.
ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrCambioProcedimientoExpediente.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrCambioProcedimientoExpediente.

Salida

TrRelacionExpediente [ ]	*Array de objetos TrRelacionExpediente.*

Creación de relaciones entre expedientes

```
TpoPK insertarRelacionExpediente (
                TpoPK idExpediente,
                TrRelacionExpediente relacionExp) throws TrException
```

Permite crear una nueva relación entre expedientes. Si se indica el REFCOMPONENTE la relación se entiende que es con un expediente externo, siempre que el componente no sea el definido en la constante MOTOR_TRAMITA del sistema por defecto.

Entradas

TpoPK IdExpediente	*Identificador del expediente.* Si el identificador es 'null' se generará una excepción.
TrRelacionExpediente relacionExp	*Datos de la relación entre expedientes*

Salida

TpoPK	*Identificador de la nueva relación entre expedientes*

Modificación de relaciones entre expedientes

```
void modificarRelacionExpediente (
                TpoPK idExpediente,
                TrRelacionExpediente relacionExp) throws TrException
```

Permite modificar una relación entre expedientes. Si se indica el REFCOMPONENTE la relación se entiende que es con un expediente externo, siempre que el componente no sea el definido en la constante MOTOR_TRAMITA del sistema por defecto.

En el objeto relacionExp se deben pasar todos los datos, no solo los que se quieran modificar ya que el api modifica todos y pondria a 'null' los que no se rellenen.

Entradas

TpoPK IdExpediente	*Identificador del expediente.* Si el identificador es 'null' se generará una excepción.
TrRelacionExpediente relacionExp	*Datos de la relación entre expedientes a modificar*

Eliminación de relaciones entre expedientes

```
void eliminarRelacionesExpediente (
        TpoPK idExpediente,
        TpoPK idRelacion) throws TrException
```

Permite eliminar una relación entre expedientes. Si no se indica *idRelacion* se eliminarán todas las relaciones del expediente.

Entradas

TpoPK IdExpediente	*Identificador del expediente.* Si el identificador es 'null' se generará una excepción.
TpoPK idRelacion	*Identificador de la relación entre expedientes* Si el identificador es 'null' se eliminarán todas las relaciones del expediente.

Obtención de usuarios asignados a un expediente

```
TrUsuarioAsignado[ ] obtenerUsuariosAsignadosExpediente(
        TpoPK idExpediente,
        ClausulaWhere where,
        ClausulaOrderBy orderBy) throws TrException
```

Devuelve los usuarios asignados al expediente y datos de la propia asignación.

Entradas

TpoPK IdExpediente	*Identificador del expediente.* Si el identificador es 'null' se generará una excepción.
ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrCambioProcedimientoExpediente.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrCambioProcedimientoExpediente.

Salida

TrUsuarioAsignado[ ]	*Array de objetos TrUsuarioAsignado*

Creación de usuarios asignados a un expediente

```
void insertarUsuarioAsignadoExpediente (
        TpoPK idExpediente,
        TrUsuarioAsignado usuarioAsig) throws TrException
```

Permite asignar un usuario a un expediente. Para ello se comprueba que el expediente

existe y no esté archivado, además de que exista el usuario.

Entradas

TpoPK IdExpediente	*Identificador del expediente.* Si el identificador es 'null' se generará una excepción.
TrUsuarioAsignado usuarioAsig	*Usuario asignado a insertar*

Modificación de usuarios asignados a un expediente

```
void modificarUsuarioAsignadoExpediente (
                TpoPK idExpediente,
                TrUsuarioAsignado usuarioAsig) throws TrException
```

Permite modificar un usuario asignado a un expediente. Para ello se comprueba que el expediente existe y no esté archivado, además de que exista el usuario y que estuviese asignado al expediente.

Entradas

TpoPK IdExpediente	*Identificador del expediente.* Si el identificador es 'null' se generará una excepción.
TrUsuarioAsignado usuarioAsig	*Usuario asignado a modificar*

Eliminación de usuarios asignados a un expediente

```
void eliminarUsuarioAsignadosExpediente (
                TpoPK idExpediente,
                String usuario) throws TrException
```

Permite eliminar un usuario asignado a un expediente. Si el parámetro usuario es 'null' se eliminan todos los usuarios asignados al expediente.

Entradas

TpoPK IdExpediente	*Identificador del expediente.* Si el identificador es 'null' se generará una excepción.
String usuario	*Usuario* Si el identificador es 'null' se eliminan todos los usuarios asignados al expediente.

Obtención de expedientes

```
TrExpediente[ ] obtenerExpedientes(
                TpoPK idExpediente,
```

```
                                    ClausulaWhere where,
                                    ClausulaOrderBy orderBy) throws TrException;
```

Permite obtener los datos de un expediente. Si se le pasa el parámetro idExpediente a null se obtienen todos los expedientes.

Entradas

TpoPK idExpediente	*Identificador del expediente.* Si el identificador es 'null' se obtienen todos los expedientes,
ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrExpediente.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrExpediente.

Salida

TrExpediente [ ]	*Array de objetos TrExpediente.*

Obtención de expedientes que se encuentran en una fase

```
TrExpediente[ ] obtenerExpedientesEnFase(
                    TpoPK idStma,
                    TpoPK idFase )  throws TrException
```

Permite al usuario obtener una lista de expedientes que se encuentran en una fase determinada y para un sistema dado.

Entradas

TpoPK idStma	*Identificador del sistema.* Si el identificador es 'null' se generará una excepción.
TpoPK idFase	*Identificador de la fase.* Si el identificador es 'null' se generará una excepción.

Salida

TrExpediente [ ]	*Array de objetos TrExpediente.* Este método devuelve sólo inicializado el atributo *REFEXP*.

Obtención de expedientes pendientes para un usuario

```
TrExpediente[ ] obtenerExpedientesPendientes(
                    TpoPK idStma ) throws TrException
```

Devuelve para un sistema dado, el conjunto de expedientes sobre los que el usuario tiene algún permiso para realizar tareas o cambios de fase (tramitar, generar algún documento, etc).

Entradas

TpoPK idStma	*Identificador del sistema.* Si el identificador es 'null' se generará una excepción.

Salida

TrExpediente [ ]	*Array de objetos TrExpediente.* Este método devuelve sólo inicializado el atributo *REFEXP*.

Obtención de expedientes reservados por un usuario

```
TrExpediente[ ] obtenerExpedientesReservados(
                TpoPK idStma ) throws TrException
```

Devuelve para un sistema dado, el conjunto de expedientes que el usuario tiene reservados, ya sea el expediente completo o alguna fase del expediente.

Entradas

TpoPK idStma	*Identificador del sistema.* Si el identificador es 'null' se generará una excepción.

Salida

TrExpediente [ ]	*Array de objetos TrExpediente.* Este método devuelve sólo inicializado el atributo *REFEXP*.

Exploración de expedientes ordenados

```
TrExplorador[ ] explorarExpedientes(
                TpoPK idSistema,
                TpoPK idTipoExp,
                TpoPK idDefProc,
                TpoDate fechaAltaIni,
                TpoDate fechaAltaFin,
                String soloVigentes,
                String ordenarPor,
                String pendientes,
                String caducados,
                String reservados ) throws TrException
```

Permite obtener un conjunto de expedientes, dependiendo de los filtros que se indiquen como parámetros.

Entradas

TpoPK idSistema	*Identificador del sistema sobre el que se van a obtener los expedientes.* Si no se indica, de devolverán expedientes de todos los sistemas que cumplan el resto de filtros.
TpoPK idTipoExp	*Identificador del tipo de expediente.* Permite obtener sólo los expedientes de un tipo concreto
TpoPK idDefProc	*Identificador del procedimiento.* Permite obtener sólo los expedientes de un procedimiento concreto
TpoDate fechaAltaIni	*Fecha de alta inicial* Permite obtener sólo aquellos expedientes que fueron dados de alta después de la fecha indicada
TpoDate fechaAltaFin	*Fecha de alta fin* Permite obtener sólo aquellos expedientes que fueron dados de alta antes de la fecha indicada
String soloVigentes	*Permite obtener sólo los expedientes que son de un tipo de expediente vigente en el motor.* Por defecto 'N' → "**S**" – Sí → "**N**" – No
String ordenarPor	*Permite ordenar el resultado.* Por defecto 'T'. → "**T**" – Sistema, Tipo de expediente, Procedimiento, Metafase, Fase, Año, Expediente. → "**E**" – Sistema, Metafase, Fase, Tipo de expediente, Procedimiento, Año, Expediente. → "**TEX**" – Sistema, Tipo de expediente, Metafase, Fase, Año, Expediente. → "**TEV**" – Sistema, Procedimiento, Metafase, Fase, Año, Expediente. → "**EEX**" – Sistema, Metafase, Fase, Tipo de expediente, Año, Expediente. → "**EEV**" – Sistema, Metafase, Fase, Procedimiento, Año, Expediente. → "**TO**" – Sistema, Tipo de expediente, Procedimiento, Orden metafase, Metafase, Orden fase, Fase, Año, Expediente. → "**EO**" – Sistema, Orden metafase, Metafase, Orden fase, Fase, Tipo de expediente, Procedimiento, Año, Expediente.
String pendientes	*Permite obtener sólo los expedientes sobre los que el usuario tiene algún permiso para realizar tareas o cambios de fase.* Por defecto 'N'. → "**S**" – Sí → "**N**" – No
String caducados	*Permite obtener sólo los expedientes que se encuentran caducados.* Por defecto 'N'. → "**S**" – Sí → "**N**" – No
String reservados	*Permite obtener sólo los expedientes que se encuentran reservados por el usuario.* Por defecto 'N'. → "**S**" – Sí → "**N**" – No

Salida

TrExplorador [ ]	*Array de objetos TrExplorador.*

APIs sobre tramitación (transiciones y fases)

Obtención de la(s) fase(s) actual(es) de un expediente

```
TrFaseActualExpediente[ ] obtenerFaseActualExpediente(
                TpoPK idExpediente,
                ClausulaWhere where,
                ClausulaOrderBy orderBy ) throws TrException
```

Dado un expediente, devuelve los datos de las fases o situaciones en la que se encuentra actualmente el expediente.

Entradas

TpoPK idExpediente	*Identificador del expediente.* Si el identificador es 'null', el método devolverá todas las versiones definidas en TREW@.
ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrFaseActualExpediente.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrFaseActualExpediente.

Salida

TrFaseActualExpediente [ ]	*Array de objetos TrFaseActualExpediente.*

En este método hay que tener en cuenta que los filtros y ordenaciones sobre los datos se hacen por niveles (en caso de módulos reutilizables), es decir, en caso de que un determinado filtro no se cumpla para un nivel, hace que no se devuelva dicho nivel ni los que dependen de él.

Obtención de transiciones posibles desde una fase

```
TrTransicion[ ] obtenerTransicionesPermitidas(
                TpoPK idFase,
                TpoPK idDefProc,
                TpoPK idExpediente,
                TpoDate fecha,
                String condVisual,
                boolean soloUsuario,
                ClausulaWhere where,
                ClausulaOrderBy orderBy ) throws TrException
```

Permite a un usuario obtener para una fase y un procedimiento dado, el conjunto de transiciones posibles que se pueden realizar según el perfil del usuario. Permite filtrar las transiciones que tengan condiciones de visualización.

Entradas

TpoPK idFase	*Identificador de la fase.* Si el identificador es 'null', el método devolverá las transiciones de entrada que dan comienzo a la tramitación.
TpoPK idDefProc	*Identificador de la definición de procedimiento.* Si el identificador es 'null', o no válido, se generará una excepción
TpoPK idExpediente	*Identificador del expediente.* Opcional
TpoDate fecha	*Fecha.* *Si no se indica fecha, tomará por defecto la del sistema.*
String condVisual	*Indica si se quieren obtener las transiciones en las que se cumplan las condiciones de visualización o no.* *→ 'S' – Sólo las que cumplan las condiciones (Por defecto)* *→ 'N' – Sólo las que no cumplan las condiciones* *→ 'T' – Todas*
boolean soloUsuario	*Indica si se filtra por el usuario y su perfil. Por defecto true*
ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrTransicion.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrTransicion.

Salida

TrTransicion [ ]	Array de objetos TrTransicion que representan las transiciones posibles que se pueden realizar.

En este método hay que tener en cuenta que los filtros y ordenaciones sobre los datos se hacen por niveles (en caso de módulos reutilizables), es decir, en caso de que un determinado filtro no se cumpla para un nivel, hace que no se devuelva dicho nivel ni los que dependen de él.

Obtención de eventos posibles

```
TrTransicion[ ] obtenerEventosPosibles(
                    TpoPK idDefProc,
                    TpoPK idExpediente,
                    TpoDate fecha,
                    String condVisual,
                    boolean soloUsuario,
                    ClausulaWhere where,
                    ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener los posibles eventos para un expediente a una fecha dada (si no se indica se toma la fecha del sistema). Esto permite definir eventos que dependen de una fecha, de esta forma se permiten definir condiciones con un parámetro fecha.

Si las condiciones definidas no necesitan los parámetros "idExpediente " y "fecha", éstos se pueden pasar nulos.

Entradas

TpoPK idDefProc	*Identificador de la definición de procedimiento.* Si el identificador es 'null', o no válido, se generará una excepción.
TpoPK idExpediente	*Identificador del expediente.* Opcional
TpoDate fecha	*Fecha.* *Si no se indica fecha, tomará por defecto la del sistema.*
String condVisual	*Indica si se quieren obtener las transiciones en las que se cumplan las condiciones de visualización o no.* *→ 'S' – Sólo las que cumplan las condiciones (Por defecto)* *→ 'N' – Sólo las que no cumplan las condiciones* *→ 'T' – Todas*
boolean soloUsuario	*Indica si se filtra por el usuario y su perfil. Por defecto true*
ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrTransicion.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrTransicion.

Salida

TrTransicion [ ]	Array de objetos TrTransicion que representan los eventos posibles que se pueden realizar.

Tramitación de un expediente

```
TrMensajeCondicionAccion[ ] tramitarExpediente(
        TpoPK idExpediente,
        TpoPK idFase,
        TpoPK idTransicion,
        TpoPK idDefProc,
        TpoDate fecha,
        TpoDate fechaLimite,
        String bloqueo,
        String observaciones,
        boolean comprobarTerminadas ) throws TrException
```

Método que realiza en un procedimiento determinado para el expediente que se indica como parámetro, la transición dada, desde una fase dada, para un expediente también dado.

Entradas

TpoPK idExpediente	*Identificador del expediente.*
TpoPK idFase	*Identificador de la fase.*
TpoPK idTransicion	*Identificador del expediente.*
TpoPK idDefProc	*Identificador de la definición de procedimiento.*
TpoDate fecha	*Fecha.* Si no se indica fecha, por defecto se tomará la del sistema.
TpoDate fechaLimite	*Fecha límite de estancia en la fase.*
String bloqueo	*Indica si el usuario quiere bloquear la fase ('**F**') o el expediente ('**E**').*
String observaciones	*Observaciones de tramitación a la fase.* Esta cadena no puede tener una longitud superior a 250 caracteres.

boolean comprobarTerminadas	*Indica si se comprueban que las tareas estén terminadas.* Por defecto true.

Salida

TrMensajeCondicionAccion [ ]	*Array de objetos TrMensajeCondicionAccion.*

Devuelve un array de mensajes (avisos, errores) derivados de tramitar el expediente y comprobar sus condiciones o realizar las acciones asociadas a la transición.

Si existen condiciones obligatorias definidas sobre la transición y alguna no se cumple, no se dejará tramitar a no ser que el usuario tenga permisos de administración.

Reservar un expediente o alguna de sus fases

```
void reservarExpediente(
        TpoPK idExpediente,
        TpoPK idFase)  throws TrException
```

Permite bloquear un expediente o una fase concreta del expediente. Si no se indica fase, la reserva es al nivel de expediente y todas las situaciones quedan bloqueadas por el usuario. Si la fase representa un procedimiento reutilizable, un bloqueo de fase implica el bloqueo de todas las situaciones en las que se encuentre el expediente dentro del procedimiento reutilizable.

Los usuarios “administradores” pueden bloquear aunque existan bloqueos de otros usuarios.

Entradas

TpoPK idExpediente	*Identificador del expediente.* Si el identificador es ‘null’, o no válido, se generará una excepción.
TpoPK idFase	*Identificador de la fase.* *Opcional*

Eliminar la reserva de un expediente o de alguna de sus fases

```
void eliminarReservaExpediente(
        TpoPK idExpediente,
        TpoPK idFase) throws TrException
```

Método complementario del anterior. Permite eliminar las reservas sobre un expediente de un sistema dado. Si se indica la fase el desbloqueo sólo es a nivel de la fase.

Los usuarios “administradores” pueden desbloquear aunque existan bloqueos de otros usuarios.

Entradas

TpoPK idExpediente	*Identificador del expediente.* Si el identificador es 'null', o no válido, se generará una excepción.
TpoPK idFase	*Identificador de la fase.* Opcional

Evolución en fases de un expediente

```
TrEvolucionExpediente[ ] obtenerEvolucionExpediente(
                TpoPK idExpediente,
                ClausulaWhere where,
                ClausulaOrderBy orderBy ) throws TrException
```

Devuelve, para un expediente, los datos acerca de la evolución del expediente dentro del procedimiento, es decir, la historia de situaciones por donde ha pasado el expediente.

Entradas

TpoPK idExpediente	*Identificador del expediente.* Si el identificador es 'null', o no válido, se generará una excepción.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrEvolucionExpediente.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrEvolucionExpediente.

Salida

TrEvolucionExpediente [ ]	Array de objetos TrEvolucionExpediente recogiendo la historia de situaciones por las que ha pasado un expediente.

Deshacer el último paso de un expediente

```
TrMensajeCondicionAccion[ ] deshacerTramitacion(
                TpoPK idExpXFase ) throws TrException
```

Permite al usuario deshacer el paso dado en la tramitación de un expediente siempre que dicho paso corresponda a una situación actual del expediente, y sólo si el usuario tiene los permisos suficientes y además la transición no es un evento de salida, tipo 'ES' (abandona la fase actual). Devuelve un conjunto de mensajes o avisos derivados de las condiciones y acciones al deshacer la transición:

Entradas

TpoPK idExpXFase	*Identificador del expediente en fase* Si el identificador es 'null', o no válido, se generará una excepción.

Salida

TrMensajeCondicionAccion [ ]	Array de objetos TrMensajeCondicionAccion.

Modificar los datos de una fase del expediente

void modificarDatosFaseActual(
TrFaseActualExpediente datoEvolucion) throws TrException

Permite modificar los datos de un paso concreto en la evolución de un expediente. Sólo se pemite modificar los datos de aquellas situaciones en las que se encuentre el expediente, que hayan sido realizadas por el usuario que intenta modificarlos y siempre que no estén reservadas por otro usuario.

Entradas

TrFaseActualExpediente datoEvolucion	*Situación del expediente, a modificar.* Se debe aportar los atributos FECHAENTRADA / FECHALIMITE / OBSERVACIONES / REFEXPXFAS para la modificación aunque sólo uno de ellos fuese el cambiara.

Nota: La actualización se hace a nivel de todo el registro, por lo que no sólo se indican los campos que cambian sino también los que no cambian. Es decir, aunque por ejemplo, sólo queremos modificar la fecha de salida debemos indicar los demás datos que componen el registro, si ponemos null TREW@ intentará poner nulos siempre que sea posible.

Condiciones asociadas a una transición

```
TrCondicionTransicion[ ] obtenerCondicionesTransicion(
                TpoPK idDefProc,
                TpoPK idTransicion,
                ClausulaWhere where,
                ClausulaOrderBy orderBy ) throws TrException
```

Permite al usuario obtener los datos del conjunto de condiciones (válidas) asociadas a una transición en un procedimiento dados.

Entradas

TpoPK idDefProc	*Identificador de la definición del procedimiento.* *Si el identificador es 'null', o no válido, se generará una excepción.*
TpoPK idTransicion	*Identificador de la transición.* *Si el identificador es 'null', o no válido, se generará una excepción.*
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrCondicionTransicion.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrCondicionTransicion.

Salida

TrCondicionTransicion [ ]	*Array de objetos TrCondicionTransicion.*

Acciones asociadas a una transición

```
TrAccionTransicion[ ] obtenerAccionesTransicion(
                TpoPK idDefProc,
                TpoPK  idTransicion,
                ClausulaWhere  where,
                ClausulaOrderBy  orderBy )  throws TrException
```

Permite al usuario obtener los datos del conjunto de acciones (válidas) asociadas a una transición y en un procedimiento dados.

Entradas

TpoPK idDefProc	*Identificador de la definición del procedimiento.* *Si el identificador es 'null', o no válido, se generará una excepción.*
TpoPK idTransicion	*Identificador de la transición.* *Si el identificador es 'null', o no válido, se generará una excepción.*
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrAccionTransicion.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrAccionTransicion.

Salida

TrAccionTransicion [ ]	*Array de objetos TrAccionTransicion.*

Evaluar condiciones de una transición

```
TrMensajeCondicionAccion[ ] evaluarCondicionesTransicion(
        TpoPK idExpediente,
        TpoPK idTransicion,
        TpoPK idDefProc,
        String comprobar,
        TpoBoolean resultado,
        TpoDate fecha ) throws TrException
```

Permite al usuario evaluar el conjunto de condiciones asociadas a una transición para un expediente dado en un procedimiento también dado.

Entradas

TpoPK idExpediente	*Identificador del expediente.* *Opcional.*
TpoPK idTransicion	*Identificador de la transición.* *Si el identificador es 'null', o no válido, se generará una excepción.*
TpoPK idDefProc	*Identificador de la definición de procedimiento.* *Si el identificador es 'null', o no válido, se generará una excepción.*
String comprobar	Usado para indicar cuáles son las condiciones que se quieren evaluar: → "**T**" – Evaluar las condiciones definidas para tramitar. → "**D**" – Evaluar las condiciones definidas para deshacer. → "**A**" – Evaluar ambos tipos de condiciones. → "**V**" – Evaluar las condiciones definidas para visualización.
TpoBoolean resultado	Indica si alguna de las condiciones es obligatoria y no se cumple. → '**false'** – Alguna da las condiciones es obligatoria y no se cumple. → '**true'** – No existen condiciones obligatorias que no se cumplan. Entrada/Salida.
TpoDate fecha	Fecha Si no se indica fecha, tomará por defecto la del sistema.

Salida

TrMensajeCondicionAccion [ ]	Array de objetos TrMensajeCondicionAccion con los mensajes resultado de evaluar las condiciones.

Obtención de los tipos de actos de un procedimiento

```
TrTipoActo[ ] obtenerTiposActoAdmDefProcedimiento(
                TpoPK idDefProc,
                ClausulaWhere where,
                ClausulaOrderBy orderBy ) throws TrException
```

Permite al usuario obtener los datos del conjunto de tipos de acto administrativos que están representados mediantes transiciones en un procedimiento concreto.

Entradas

TpoPK idDefProc	*Identificador de la definición del procedimiento.* Si el identificador es 'null', o no válido, se generará una excepción.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoActo.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrTipoActo.

Salida

TrTipoActo [ ]	*Array de objetos* TrTipoActo *con la información referente a los tipos de actos administrativos que están representados mediante transiciones.*

Fecha de un acto para un expediente

```
TpoDate obtenerFechaActoAdmExpediente(
                TpoPK idExpediente,
                TpoPK idTipoActo ) throws TrException
```

Permite al usuario obtener la fecha en la que se produjo un acto administrativo para un expediente (la más reciente).

Entradas

TpoPK idExpediente	*Identificador del expediente.* *Si el identificador es 'null', o no válido, se generará una excepción.*
TpoPK idTipoActo	*Identificador del tipo de acto.*

Salida

TrDate	*Fecha en la que se produjo el acto administrativo.*

Conjunto de fases y transiciones posibles en un procedimiento

```
TrTransicionDefProcedimiento[ ] obtenerTransicionesDefProcedimiento(
                TpoPK idDefProc,
                ClausulaWhere where,
                ClausulaOrderBy orderBy ) throws TrException
```

Permite a un usuario obtener para un procedimiento dado, el conjunto de transiciones posibles que se pueden realizar y las fases en las que derivan sin tener en cuenta el perfil del usuario.

Entradas

TpoPK idDefProc	*Identificador de la definición del procedimiento.* Si el identificador es 'null', o no válido, se generará una excepción.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTransicionDefProcedimiento.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrTransicionDefProcedimiento.

Salida

TrTransicionDefProcedimiento [ ]	*Array de objetos* TrTransicionDefProcedimiento.

Enviar expediente

```
void enviarExpedienteA(
        TpoPK idExpediente,
        TrEnviarA[ ] datosTransicion,
        String bloqueo,
        String usuario,
        TrFaseActualExpediente[ ] faseActualExp ) throws TrException
```

Permite enviar el expediente a una situación concreta. Se debe indicar los datos de la transición que se desea realizar mediante un array de objetos TrEnviarA que se pasa como parámetro.

Entradas

TpoPK idExpediente	*Identificador del expediente.*
TrEnviarA[ ] datosTransicion	*Array de objetos TrEnviarA.*
String bloqueo	Indica si se permite la reserva del expediente o de la fase. → "**E**" – Permite la reserva del expediente → "**F**" – Permite la reserva de la fase
String usuario	Indica el usuario que reserva.
TrFaseActualExpediente[ ] faseActualExp	Array con fases del expediente desde las cuales se quiere hacer "enviar a ..." Es necesario inicializar correctamente los atributos *FASE.REFFASE* Y *REFFASEPADRE* en TrFaseActualExpediente. El resto de atributos son ignorados.

A través del parámetro *faseActualExp* se indica la fase del expediente desde la cual se quiere hacer "enviar a...", de esta forma se puede anotar la fecha de salida desde esta/s fase/s (ya que el expediente se puede encontrar en varias a la vez). Si no se indica nada la nueva situación es una más para el expediente, es decir, el expediente queda en la situación que estuviera más la nueva situación a la que enviamos.

Datos de una transición

```
TrTransicion[ ] obtenerDatosTransicion(
        TpoPK idTransicion,
        ClausulaWhere where,
        ClausulaOrderBy orderBy ) throws TrException
```

Devuelve los datos de una transición pasada como parámetro. Si la transición no existiese se generaría una excepción.

Entradas

TpoPK idTransicion	*Identificador de la transición* Si el identificador es 'null', o no válido, se generará una excepción.

ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTransicion. En este api no se tiene en cuenta el CAMPO_REFTRANSICION ya que se recibe el parámetro idTransición.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrTransicion.

Salida

TrTransicion [ ]	*Array de objetos TrTransicion.*

Fases finales de una transición

```
TrFase[ ] obtenerFasesFinTransicion(
        TpoPK idTransicion,
        TpoPK idDefProc,
        ClausulaWhere where,
        ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener la fase fin en la que termina la transición. En caso de una transición de división devolverá más de una fase final.

Se comprueba que la transición exista, además de que el procedimiento exista y no esté bloqueado, generando una excepción en caso contrario.

Entradas

TpoPK idTransicion	*Identificador de la transición* Si el identificador es 'null', o no válido, se generará una excepción.
TpoPK idDefProc	*Identificador de la definición del procedimiento* Si el identificador es 'null', o no válido, se generará una excepción.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTransicion.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrTransicion.

Salida

TrFase [ ]	*Array de objetos TrFase.*

Condiciones, acciones y avisos

```
TrCondicionAccionAviso[ ] obtenerCondicionesAccionesAvisos(
        TpoPK idDefProc,
        TpoPK idTranDocuOtTa,
        String tranDocuOtTa,
        String tipo,
        ClausulaWhere where,
        ClausulaOrderBy orderBy ) throws TrException
```

Devuelve todas las condiciones, acciones y avisos del procedimiento para el tipo

indicado en el parámetro tranDocuOtTa. El valor de este parámetro puede ser “T” para las condiciones, acciones y avisos al tramitar, “D” de los documentos y “O” de otras tareas. Además permite filtrar por el parámetro tipo que puede ser “C” condición, “A” acción y “W” aviso.

Entradas

TpoPK idDefProc	*Identificador de la definición del procedimiento* Si el identificador es ‘null’, o no válido, se generará una excepción.
TpoPK idTranDocuOtTa	*Identificador de la transición, documento u otra tarea.*
String tranDocuOtTa	*Indica qué condiciones, acciones y avisos se quieren obtener.* *→ ‘**T**’ – Condiciones, acciones y avisos de la transición* *→ ‘**D**’ – Condiciones, acciones y avisos del documento* *→ ‘**O**’ – Condiciones, acciones y avisos de otras tareas*
String tipo	*Indica el tipo de datos que se quieren obtener* *→ ‘**C**’ –Condiciones* *→ ‘**A**’ – Acciones* *→ ‘**W**’ – Avisos* Si se le pasa null se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTransicion.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrTransicion.

Salida

TrCondicionAccionAviso[ ]	*Array de objetos TrCondicionAccionAviso.*

APIs sobre documentos (tareas I)

Documentos permitidos en una fase

```
TrDocumentoPermitido[ ] obtenerDocumentosPermitidos(
                    TpoPK idFase,
                    TpoPK idDefProc,
                    TpoPK idExpediente,
                    TpoDate fecha,
                    String condVisual,
                    boolean soloUsuario,
                    ClausulaWhere where,
                    ClausulaOrderBy orderBy ) throws TrException
```

Devuelve al usuario el conjunto de datos sobre los documentos que puede generar, incorporar, editar, etc., en una fase dada en un procedimiento también dado.

Además permite indicar si se obtienen los que cumplan las condiciones de visualización o no y si se filtra por el usuario del api y su perfil.

Entradas

TpoPK idFase	*Identificador de la fase.* *Si el identificador es 'null', o no válido, se generará una excepción.*
TpoPK idDefProc	*Identificador de la definición del procedimiento.* *Si el identificador es 'null', o no válido, se generará una excepción.*
TpoPK idExpediente	*Identificador del expediente.* *Opcional*
TpoDate fecha	*Fecha* Opcional. Si no se indica fecha, por defecto se tomará la del sistema.
String condVisual	*Indica que documentos permitidos se obtienen* *→ '**S**' – Los que cumplan las condiciones* *→ '**N**' – Los que no cumplan las condiciones* *→ '**T**' – Todos*
boolean soloUsuario	*Indica si se filtra por el usuario establecido en el api y su perfil*
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrDocumentoPermitido.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrDocumentoPermitido.

Salida

TrDocumentoPermitido [ ]	*Array de objetos* TrDocumentoPermitido.

(w) Anotar un documento como generado

```
TrMensajeCondicionAccion[ ] generarDocumento(
        TpoPK idExpediente,
        TpoPK idDocPer,
        TpoPK idDefProc,
        TpoDate fecha,
        TpoDate fechaLimite,
        String observaciones,
        TrValorParametro[ ] datosParametros,
        TpoPK idDocExpte ) throws TrException
```

Permite al usuario generar un documento dado para un expediente en un procedimiento también suministrado. El método devuelve un conjunto de mensajes/avisos derivados de las condiciones establecidas para la generación del documento. Además se puede indicar la fecha límite para el documento del expediente.

Si se ha definido un componente w@rdA en el sistema establecido en el TrAPIUI se guardará el documento generado en w@rdA.

Entradas

TpoPK idExpediente	*Identificador del expediente.* *Si el identificador es 'null', o no válido, se generará una excepción.*
TpoPK idDocPer	*Identificador del documento permitido* Opcional.
TpoPK idDefProc	*Identificador de la definición de procedimiento.* *Si el identificador es 'null', o no válido, se generará una excepción.*
TpoDate fecha	*Fecha en la cual el documento queda anotado para el expediente como "generado".* Si no se indica fecha, por defecto se tomará la del sistema.
TpoDate fechaLimite	*Fecha límite para el documento del expediente*
String observaciones	*Observaciones de tramitación a la fase.* Esta cadena no puede tener una longitud superior a 250 caracteres.
TrValorParametro[ ] datosParametros	*Conjunto de parámetros a utilizar para la sustitución de variables contenidas en los párrafos que forman el documento.*
TpoPK idDocExpte	*Identificador único que TREW@ le da al documento generado para ese expediente.* *Entrada/Salida.*

Salida

TrMensajeCondicionAccion [ ]	Array de objetos TrMensajeCondicionAccion.

Si existen condiciones obligatorias definidas sobre el documento y alguna no se cumple, no se generará a no ser que el usuario tenga permisos de administración.

Obtener URL del documento generado

```
String obtenerURLDocumentoGenerado(
        TpoPK idDocExp,
        String multiple) throws TrException
```

Devuelve la URL del documento generado.

Entrada

TpoPK idDocExp	Identificador del documento.
String multiple	Indica si el documento es múltiple o no.

Salida

String	URL del documento generado.

(w) Anotar un documento como incorporado

```
TrMensajeCondicionAccion[ ] incorporarDocumento(
                TpoPK idExpediente,
                TpoPK idDocPer,
                TpoPK idDefProc,
                TpoDate fecha,
                String presentado,
                String correcto,
                TpoDate fechaLimite,
                String observaciones,
                TpoPK idDocExpte ) throws TrException
```

Permite al usuario incorporar un documento dado para un expediente en un procedimiento también suministrado. El método devuelve un conjunto de mensajes/avisos derivados de las condiciones establecidas para la incorporación del documento.

Si se ha definido un componente w@rdA en el sistema establecido en el TrAPIUI también se guardará el documento incorporado en w@rdA.

Entradas

TpoPK idExpediente	*Identificador del expediente.* Si el identificador es 'null', o no válido, se generará una excepción.
TpoPK idDocPer	*Identificador del documento permitido.* Si el identificador es 'null' se generará una excepción.
TpoPK idDefProc	*Identificador de la definición de procedimiento.* *Si el identificador es 'null', o no válido, se generará una excepción.*
TpoDate fecha	*Fecha de incorporación del documento.* *Es la fecha en la cual el documento queda anotado para el expediente como "incorporado".* Si no se indica fecha, por defecto se tomará la del sistema.
String presentado	Indica si el documento se ha presentado o no. → "**S**" – El documento se ha presentado → "**N**" – El documento no se ha presentado
String correcto	Indica si el documento que se ha presentado es correcto o no. → "**S**" – El documento es correcto → "**N**" – El documento no es correcto
TpoDate fechaLimite	Fecha límite de presentación o de correción del documento.
String observaciones	Observaciones para anotar a la hora de incorporar el documento.
TpoPK idDocExpte	*Identificador único que TREW@ le da al documento incorporado para ese expediente.* *Entrada/Salida.*

Salida

TrMensajeCondicionAccion [ ]	Array de objetos TrMensajeCondicionAccion.

(w) Incorporar documento no definido

```
void incorporarDocumentoNoDefinido(
        TpoPK idExpediente,
        TpoPK idTipDoc,
        TpoPK idFase,
        TpoDate fecha,
        String presentado,
        String correcto,
        TpoDate fechaLimite,
        String observaciones,
        TpoPK idDocExpte ) throws TrException
```

Permite incorporar un documento dado para un expediente no recogido en la definición del procedimiento.

Si se ha definido un componente w@rdA en el sistema establecido en el TrAPIUI también se guardará el documento incorporado en w@rdA.

Entradas

TpoPK idExpediente	*Identificador del expediente.* *Si el identificador es 'null', o no válido, se generará una excepción.*
TpoPK idTipDoc	*Identificador del tipo de documento que se quiere incorporar.* Si el identificador es 'null', o no válido, se generará una excepción.
TpoPK idFase	*Identificador de la fase del expediente a la que se quiere asociar el documento incorporado.* *Debe ser una de las fases en las que se encuentre el expediente.* *Si el identificador es 'null', o no válido, se generará una excepción.*
TpoDate fecha	*Fecha de incorporación del documento.* *Es la fecha en la cual el documento queda anotado para el expediente como "incorporado".* Si no se indica fecha, por defecto se tomará la del sistema.
String presentado	Indica si el documento se ha presentado o no. → "**S**" – El documento se ha presentado → "**N**" – El documento no se ha presentado
String correcto	Indica si el documento que se ha presentado es correcto o no. → "**S**" – El documento es correcto → "**N**" – El documento no es correcto
TpoDate fechaLimite	Fecha límite de presentación o de correción del documento.
String observaciones	Observaciones para anotar a la hora de incorporar el documento.
TpoPK idDocExpte	*Identificador único que TREW@ le da al documento incorporado para ese expediente.* *Entrada/Salida.*

(w) Incorporar físicamente un documento

```
void adjuntarFicheroDocumento(
                TpoPK idDocExp,
                InputStream fichero,
                String nombreFichero,
                String formato,
                long size) throws TrException
```

Permite añadir un fichero físico al documento "incorporado" al expediente que se indica como parámetro.

Si se ha definido un componente w@rdA en el sistema establecido en el TrAPIUI y el documento ya estaba recogido en el mismo componente w@rdA, se creará el anexo en w@rdA y se asociará al documento.

Entradas

TpoPK idDocExp	*Identificador del documento incorporado al expediente. Si el identificador es 'null', o no válido, se generará una excepción.*
InputStream fichero	*Fichero físico que se va a adjuntar al documento.*
String nombreFichero	*Nombre del fichero.*
String formato	*Tipo "mime" del fichero.*
long size	*Tamaño del fichero*

```
void adjuntarFicheroDocumento(
                TpoPK idDocExp,
                InputStream fichero,
                String nombreFichero,
                String formato,
                String idWarda,
                String idWardaAnx,
                TpoPK idComp) throws TrException
```

Permite añadir un fichero físico al documento "incorporado" al expediente que se indica como parámetro. Además permite actualizar los identificadores del documento en w@rdA y su anexo, así como el identificador del componente, siempre que se pasen los tres como parámetros.

Si se ha definido un componente w@rdA en el sistema establecido en el TrAPIUI y el documento ya estaba recogido en el mismo componente w@rdA, se creará el anexo en w@rdA y se asociará al documento. Si el documento no estaba recogido en w@rdA pero se indican los parámetros *idWarda, idWardaAnx y idComp* se actualizarán en Trew@.

Entradas

TpoPK idDocExp	*Identificador del documento incorporado al expediente.* *Si el identificador es 'null', o no válido, se generará una excepción.*
InputStream fichero	*Fichero físico que se va a adjuntar al documento.*
String nombreFichero	*Nombre del fichero.*
String formato	*Tipo "mime" del fichero.*
String idWarda	*Identificador del documento en w@rdA*
String idWardaAnx	*Identificador del anexo del documento en w@rdA*
TpoPK idComp	*Identificador dl componente que guarda el documento*

Documentos del expediente

```
TrDocumentoExpediente[ ] obtenerDocumentosExpediente(
                TpoPK idExpediente,
                 boolean soloUsuario,
                ClausulaWhere where,
                ClausulaOrderBy orderBy ) throws TrException
```

Devuelve el conjunto de documentos que se han generado y/o incorporado al expediente dado. Además permite indicar si se filtra por el usuario establecido en el api y su perfil.

Entradas

TpoPK idExpediente	*Identificador del expediente* *Si el identificador es 'null', o no válido, se generará una excepción.*
boolean soloUsuario	*Indica si se filtra por el usuario del api y su perfil*
ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrDocumentoExpediente.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrDocumentoExpediente.

Salida

TrDocumentoExpediente [ ]	*Array de objetos* TrDocumentoExpediente.

(w) Modificar los datos de un documento del expediente

void modificarDatosDocumento(
TrDocumentoExpediente datosDocExpte) throws TrException

Permite modificar los datos principales asociados a un documento generado o incorporado siempre que se tengan permisos para hacerlo (no esté reservada por otro usuario la fase dónde el documento esté permitido, etc.).

Si se define un componente w@rdA en el sistema establecido en el TrAPIUI y el documento está guardado en el mismo w@rdA se actualizan sus datos.

Entradas

TrDocumentoExpediente datosDocExpte	*Objeto TrDocumentoExpediente con la información asociada a un documento generado y/o incorporado a un expediente.* *Se precisa rellenar los siguientes atributos de TrDocumentoExpediente:* *→ REFDOCEXP – Identificador del documento a modificar.* *→ FECHA / PRESENTADO / CORRECTO / FECHA_LIMITE / OBSERVACIONES / C_HASH / FECHACADU / REUTILIZABLE / FIRMADIG*

(w) Eliminar un documento

```
void eliminarDocumento(
        TpoPK idDocExpte) throws TrException
```

Permite al usuario eliminar un documento de un expediente. Sólo se permite eliminar un documento generado/incorporado en la situación actual del expediente y que haya sido generado/incorporado por el mismo usuario que intenta eliminar y siempre que el documento se encuentre en realización.

Si se define un componente w@rdA en el sistema establecido en el TrAPIUI y el documento está guardado en el mismo w@rdA se elimina de éste.

Entradas

TpoPK idDocExpte	*Identificador del documento asociado al expediente.* *Si el identificador es 'null', o no válido, se generará una excepción.*

Condiciones asociadas a un documento

```
TrCondicionDocumento[ ] obtenerCondicionesDocumento(
        TpoPK idDefProc,
        TpoPK idDocPer,
        ClausulaWhere where,
        ClausulaOrderBy orderBy ) throws TrException
```

Permite al usuario obtener los datos del conjunto de condiciones (válidas) asociadas a un determinado documento en una fase y un procedimiento dados.

Entradas

TpoPK idDefProc	*Identificador de la definición del procedimiento* *Si el identificador es 'null', o no válido, se generará una excepción.*
TpoPK idDocPer	*Identificador del documento* *Si el identificador es 'null', o no válido, se generará una excepción.*
ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrCondicionDocumento.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrCondicionDocumento.

Salida

TrCondicionDocumento [ ]	*Array de objetos* TrCondicionDocumento.

Acciones asociadas a un documento

```
TrAccionDocumento[ ] obtenerAccionesDocumento(
                TpoPK idDefProc,
                TpoPK  idDocPer,
                ClausulaWhere  where,
                ClausulaOrderBy  orderBy )  throws TrException
```

Permite al usuario obtener los datos del conjunto de acciones (válidas) asociadas a un determinado documento en una fase y un procedimiento dados.

Entradas

TpoPK idDefProc	*Identificador de la definición del procedimiento* *Si el identificador es 'null', o no válido, se generará una excepción.*
TpoPK idDocPer	*Identificador del documento* *Si el identificador es 'null', o no válido, se generará una excepción.*
ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrAccionDocumento.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrAccionDocumento.

Salida

TrAccionDocumento [ ]	*Array de objetos* TrAccionDocumento.

Evaluar las condiciones asociadas a un documento

```
TrMensajeCondicionAccion[ ] evaluarCondicionesDocumento(
        TpoPK idExpediente,
        TpoPK idDocPer,
        TpoPK idDefProc,
        String comprobar,
        TpoBoolean resultado,
        TpoDate fecha ) throws TrException
```

Permite al usuario evaluar el conjunto de condiciones asociadas a un documento para un expediente dado en un procedimiento también dado.

Entradas

TpoPK idExpediente	*Identificador del expediente.* Opcional. Sólo para condiciones en las que se quiera tener en cuenta este dato.
TpoPK idDocPer	*Identificador del documento.* *Si el identificador es 'null', o no válido, se generará una excepción.*
TpoPK idDefProc	*Identificador de la definición de procedimiento.* *Si el identificador es 'null', o no válido, se generará una excepción.*
String comprobar	Indica cuáles son las condiciones que se quieren evaluar → "**G**" – Condiciones al generar el documento → "**I**" – Condiciones al incorporar el documento → "**T**" – Condiciones al generar e incorporar el documento → "**V**" – Condiciones al visualizar el documento
TpoBoolean resultado	Resultado. Entrada/Salida → "**false**" – Si alguna de las condiciones es obligatoria y no se cumple. → "**true**" – Si no existen condiciones obligatorias que no se cumplan.
TpoDate fecha	Fecha. Si no se indica fecha, tomará por defecto la del sistema.

Salida

TrMensajeCondicionAccion [ ]	Array de objetos TrMensajeCondicionAccion.

Datos de registro de un documento

```
TrRegistroDocumento obtenerDatosRegistroDocumento(
        TpoPK idDocExpte ) throws TrException
```

Devuelve los datos de registro que existen para un documento que se indica como parámetro (clave primaria del documento del expediente).

Entradas

TpoPK idDocExpte	*Identificador del documento del expediente* *Si el identificador es 'null', o no válido, se generará una excepción.*

Modificar los datos de registro de un documento

```
void modificarDatosRegistroDocumento(
        TpoPK idDocExpte,
        TrRegistroDocumento datosRegistro ) throws TrException
```

Permite modificar los datos de registro para un documento que se indica como parámetro (clave primaria del documento del expediente).

Entradas

TpoPK idDocExpte	*Identificador del documento del expediente* *Si el identificador es 'null', o no válido, se generará una excepción.*
TrRegistroDocumento datosRegistro	*Datos de registro para modificar.* Se precisa rellenar todos los atributos de TrRegistroDocumento aunque algunos no cambien.

Tipos de documentos

```
TrTipoDocumento[ ] obtenerTiposDocumento(
        TpoPK idStma,
        ClausulaWhere where,
        ClausulaOrderBy orderBy ) throws TrException
```

Devuelve el conjunto de datos sobre los tipos de documento definidos en el sistema, que se pasa como parámetro. Si no se indica sistema devuelve todos los tipos de documentos definidos en el tramitador.

Entradas

TpoPK idStma	*Identificador de la definición del procedimiento* *Opcional*
ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrTipoDocumento.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrTipoDocumento.

Salida

TrTipoDocumento [ ]	*Array de objetos* TrTipoDocumento.

(w) Modificar estado de un documento

```
void modificarEstadoDocumento(
        TpoPK idDocExpte,
        String estado,
        TpoDate fechaFirma) throws TrException
```

Permite modificar el estado del documento que se indica como parámetro. Los valores válidos para el parámetro *estado* son: “R” (en realización), “E” (pendiente de firma), “F” (firmado), "D" (descartado), "T" (terminado), cualquier otro valor indicado no tiene efecto. Si el documento está en estado “F” no se puede modificar su estado.

El parámetro *fechaFirma* permite dejar todas la firmas del documento a la fecha indicada en el caso de que el estado a modificar sea a ”F”. En este caso también se genera el documento PDF correspondiente y se guarda en la base de datos.

Si se define un componente w@rdA en el sistema establecido en el TrAPIUI y el documento está guardado en el mismo w@rdA se actualiza su estado.

Entradas

TpoPK idDocExpte	*Identificador del documento asociado al expediente* *Si el identificador es ‘null’, o no válido, se generará una excepción.*

String estado	*Estado del documento* → "**R**" – En realización → "**E**" – Pendiente de firma → "**F**" – Firmado → "**D**" – Descartado → "**T**" –Terminado *Si el documento está en estado "F", no se podrá modificar su estado.* *Si el estado indicado es el mismo del documento, no se hace nada.*
TpoDate fechaFirma	*Fecha de la firma* . *Este parámetro permite dejar todas las firmas del documento a la fecha indicada en el caso de que el estado se vaya a modificar a "F".* *Si no se indica fecha, tomará por defecto la del sistema*

Tipos de párrafos

```
TrTipoParrafo[ ] obtenerTiposParrafo(
                    ClausulaWhere where,
                    ClausulaOrderBy orderBy ) throws TrException;
```

Devuelve el conjunto de datos de los tipos de parrafos.

Entradas

ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoParrafo.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoParrafo.

Salida

TrTipoParrafo[ ]	Array de objetos TrTipoParrafo con la información de tipos de expediente.

Recuperar párrafos

```
TrParrafo[ ] obtenerParrafosDocumento(
                    TpoPK idDocExpte,
                    ClausulaWhere where,
                    ClausulaOrderBy orderBy ) throws TrException
```

Permite recuperar los párrafos de un documento generado. Por defecto ordena por el campo “orden” de los párrafos siempre y cuando no se le pase ninguna ClausulaOrderBy.

Entradas

TpoPK idDocExpte	*Identificador del documento asociado al expediente.* *Si el identificador es ‘null’, o no válido, se generará una excepción*
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrParrafo.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrParrafo. Si no se indica ordenación, la misma por defecto es por el orden de los párrafos.

Salida

TrParrafo[ ]	Array de objetos TrParrafo con la información de tipos de expediente.

Incorporar párrafos

void incorporarParrafosDocumento(
TpoPK idDocExpte,
TrParrafo[ ] datosParrafo) throws TrException

Permite añadir nuevos párrafos al documento del expediente que se pasa como parámetro. Se permite indicar si el párrafo es editable o no.

Entradas

TpoPK idDocExpte	*Identificador del documento asociado al expediente. Si el identificador es 'null', o no válido, se generará una excepción*
TrParrafo[ ] datosParrafo	Conjunto de párrafos a añadir al documento.

Eliminar párrafos

void eliminarParrafoDocumento(
TpoPK idParrafoDoc) throws TrException

Permite al usuario eliminar un párrafo de un documento siempre que tenga permisos.

Entradas

TpoPK idParrafoDoc	*Identificador del párrafo a eliminar. Si el identificador es 'null', o no válido, se generará una excepción*

Modificar párrafos

void modificarParrafoDocumento(
TrParrafo datosParrafo) throws TrException

Permite modificar los datos de un párrafo de un documento generado siempre que sea editable.
El método no modifica la condición de "Editable".

Entradas

TrParrafo datosParrafo	*Datos del párrafo a modificar.* Se ignora el atributo EDITABLE. Se debe aportar los atributos REFPARRDOC / ETIQUETA / PARRAFO / ORDEN / ALINEACION / ESTILO / ESTILOETIQ / TIPOPARRAFO.REFTIPOPARR / IMAGEN / FORMATO / NOMBREFICHERO para la modificación aunque sólo uno de ellos fuese el cambiara.

Incluir documento

```
void incluirDocumento(
        TpoPK idDocExpteOrigen,
        TpoPK idDocExpteDestino,
        long ordenParrafoInicio) throws TrException
```

Incluye los párrafos perteneciente al cuerpo del documento origen, en un documento destino a partir de un nº de orden inicial desplazando los párrafos del documento destino.

Entradas

TpoPK idDocExpteOrigen	*Identificador del documento origen. Si el identificador es 'null', o no válido, se generará una excepción.*
TpoPK idDocExpteDestino	*Identificador del documento destino. Si el identificador es 'null', o no válido, se generará una excepción.*
long ordenParrafoInicio	*Orden, en el documento destino, a partir del cual se va a insertar el documento origen.*

Actualizar documentos múltiples

```
void actualizarDocumentosMultiples(
        TpoPK idDocExpte,
        String modo) throws TrException
```

Permite actualizar el contenido de los párrafos y las firmas del documento que se pasa como parámetro al resto de documentos del mismo tipo, que han sido generados en la misma fase, mismo usuario, y que no se encuentren firmados, es decir, es una herramienta para propagar los cambios de un documento de tipo "múltiple" al resto de documentos del mismo tipo.

Entradas

TpoPK idDocExpte	*Identificador del documento asociado al expediente. Si el identificador es 'null', o no válido, se generará una excepción*
String modo	*Modo.* → "P" – Actualiza sólo el contenido de los párrafos de los documentos. → "F" – Actualiza sólo las firmas de los documentos. → "T" – Actualiza todo: párrafos y firmas.

(w) Eliminar documentos múltiples

void eliminarDocumentosMultiples(
TpoPK idDocExpte) throws TrException

Permite eliminar los documentos definidos como "múltiples", asociados a la misma fase, generados/incorporados por el mismo usuario y del mismo tipo que el pasado como parámetro siempre que no se encuentren firmados. Esta función también elimina el documento indicado (siempre en las mismas condiciones que el método *eliminarDocumento*).

Si se define un componente w@rdA en el sistema establecido en el TrAPIUI y el documento está guardado en el mismo w@rdA se elimina de éste.

Entradas

TpoPK idDocExpte	*Identificador del documento asociado al expediente a eliminar.* *Si el identificador es 'null', o no válido, se generará una excepción*

Evaluar párrafo

String sustituirVariablesParrafoDocumento(
TpoPK idSistema,
TpoPK idParrDocExp) throws TrException

Devuelve el párrafo con el valor de las variables según los parametros del documento que le pasamos.

Entradas

TpoPK idSistema	*Identificador del sistema.* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idParrDocExp	*Identificador del párrafo del documento*. Si el identificador es 'null', o no válido, se generará una excepción

Salida

String	Párrafo con las variables reemplazadas.

(w) Descarga documento

```
InputStream recuperarDocumentoExpediente(
        TpoPK idDocExpte,
        TpoString formato,
        TpoString nombreFichero ) throws TrException
```

Permite descargar el contenido del documento que se indoca como parametro. Los parámetros *formato* y *nombreFichero* son de Entrada / Salida por si se quiere obtener el valor de dichos atributos.

Si se ha definido un componente w@rdA en el sistema establecido en el TrAPIUI se obtiene el documento de w@rdA siempre y cuando el w@rdA definido sea el mismo en el que se guardó el documento y su adjunto. Si no es el mismo w@rdA se devolverá null.

Si no se ha definido ningún w@rdA en el sistema se devolverá el documento almacenado en Trew@.

Entradas

TpoPK idDocExpte	*Identificador del documento.* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoString formato	*Tipo "mime" del fichero. Si el documento se obtiene desde w@rdA este parámetro se devolverá vacío ya que en w@rdA no se recoge el formato del documento.* Entrada/Salida.
TpoString nombreFichero	*Nombre del fichero*. Entrada/Salida.

Salida

InputStream	Fichero físico asociado al documento.

Establece parámetros documento

```
void estableceParametrosDocumento(
        TpoPK idDocExpte,
        TrValorParametro[ ] datosParametros) throws TrException
```

Establece los parametros para un documento pasado como parámetro.

Entradas

TpoPK idDocExpte	*Identificador del documento.* *Si el identificador es 'null', o no válido, se generará una excepción*
TrValorParametro[ ] datosParametros	*Array de valores para parámetros.*

Fecha firma tipo documento

```
String obtenerFechaFirmaDocumento(
        TpoPK idTipoDoc) throws TrException
```

Mediante esta función se obtiene si para un tipo de documento aparece o no la fecha en el pie de firma. Devuelve "S" en caso de que sí aparezca y "N" en caso contrario

Entradas

TpoPK idTipoDoc	*Identificador del tipo de documento.* *Si el identificador es 'null', o no válido, se generará una excepción*

Salida

String	*Indica si existe o no fecha en el pie de firma.* → "**S**" – Sí existe fecha en el pie de firma → "**N**" – No existe fecha en el pie de firma

Obtener variables del documento del expediente

```
TrVariableDocExp[ ] obtenerVariablesDocumentoExp(
        TpoPK idDocExp) throws TrException
```

Devuelve el conjunto de variables asociadas al documento del expediente que se pase como parámetro.

Entradas

TpoPK idDocExp	*Identificador del documento del expediente.* *Si el identificador es 'null', o no válido, se generará una excepción*

ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrVariableDocExp.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrVariableDocExp.

Salida

TrVariableDocExp[ ]	*Array de objetos* TrVariableDocExp

Actualizar variables del documento del expediente

```
void actualizarVariablesDocumentoExp(
                    TpoPK idDocExp,
                    TrVariableDocExp[ ] variables,
                     boolean elimVar) throws TrException
```

Permite actualizar el conjunto de variables asociadas al documento que se indica en el parámetro *idDocExp*. El conjunto de variables se indica con el parámetro *variables* de los que sólo se tendrá en cuenta el atributo REFVARIABLE.

El parámetro elimVar puede tomar dos valores:

- **true:** Indica que se eliminen las variables que tuviese asociado el documento, con lo que a partir de entonces sólo tendrá asociadas las que se le pasen en el parámetro *variables.*
- **false:** Indica que no se eliminen las variables que tuviese asociado el documento, con lo que se añadirían las variables del parámetro *variables* a las ya existentes.

Entradas

TpoPK idDocExp	*Identificador del documento del expediente.* *Si el identificador es 'null', o no válido, se generará una excepción*
TrVariableDocExp[ ] variables	*Conjunto de variables para el documento.* Sólo se usa el atributo REFVARIABLE.
boolean elimVar	*Indica si se eliminan las variables que ya tenía el documento*

(w) Versionar documento

```
TpoPK versionarDocumentoExpediente(
                    TpoPK idDocExp,
                    TpoDate fechaFin) throws TrException
```

Permite versionar un documento del expediente creando una copia exacta del documento al que versiona con todos sus párrafos, variables, parámetros, firmas, interesados en el documento, notificaciones.

El documento al que versiona se le cambia el estado a "V" (Versionado) y se actualiza su fecha de finalización, y al nuevo documento se le asigna un número de versión posterior al que versiona.

Si se ha definido un componente w@rdA en el sistema establecido en el TrAPIUI se creará el nuevo documento versionado. Si el documento al que versiona estaba guardado en el mismo componente w@rdA que está definido en el sistema se actualizará su estado.

Entradas

TpoPK idDocExp	*Identificador del documento del expediente.* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoDate fechaFin	*Fecha de finalización del documento que se versiona* Si es 'null' por defecto se coge la del sistema

Salida

TpoPK	*Identificador del nuevo documento del expediente*

Obtener interesados en un documento de un expediente

```
TrInteresadoDocumento[ ] obtenerInteresadosDocumento(
                    TpoPK idExpediente,
                    TpoPK idInteresado,
                    TpoPK idRazonInt,
                    TpoPK idRazonIntDocu,
                    TpoPK idDocExp,
                    ClausulaWhere where,
                    ClausulaOrderBy orderBy) throws TrException
```

Permite obtener los datos de los interesados en los documentos de un expediente. Si se le pasa únicamente el parámetro *idExpediente* devolverá todos los interesados en todos los documetos de ese expediente.

Entradas

TpoPK idExpediente	*Identificador del expediente* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idInteresado	*Identificador del interesado* Opcional
TpoPK idRazonInt	*Identificador de la razón de interés del interesado del expediente* Opcional
TpoPK idRazonIntDocu	*Identificador de la razón de interés del interesado en el documento.* Opcional
TpoPK idDocExp	*Identificador del documento del expediente* Opcional
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrInteresadoDocumento.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrInteresadoDocumento.

Salida

TrInteresadoDocumento[ ]	*Array de objetos* TrInteresadoDocumento

(w) Insertar interesado en un documento de un expediente

```
void insertarInteresadoDocumento(
                    TpoPK idExpediente,
                    TpoPK idInteresado,
                    TpoPK idRazonInt,
                    TpoPK idRazonIntDocu,
                    TpoPK idDocExp,
                    String observaciones) throws TrException
```

Permite insertar un nuevo interesado en el documento. Se comprueba que existan todos los parámetros y además que el expediente no esté archivado.

Si se define un componente w@rdA en el sistema establecido en el TrAPIUI y el documento al que se le asigna el interesado está guardado en el mismo w@rdA se actualizan sus datos.

Entradas

TpoPK idExpediente	*Identificador del expediente* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idInteresado	*Identificador del interesado* *Si el identificador es 'null', o no válido, se generará una excepción*

TpoPK idRazonInt	*Identificador de la razón de interés del interesado del expediente* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idRazonIntDocu	*Identificador de la razón de interés del interesado en el documento.* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idDocExp	*Identificador del documento del expediente* *Si el identificador es 'null', o no válido, se generará una excepción*
String observaciones	*Observaciones para el nuevo interesado en el documento*

(w) Modificar interesado en un documento de un expediente

```
void modificarInteresadoDocumento(
                    TpoPK idExpediente,
                    TpoPK idInteresado,
                    TpoPK idRazonInt,
                    TpoPK idRazonIntDocu,
                    TpoPK idRazonIntDocuNueva,
                    TpoPK idDocExp,
                    String observaciones) throws TrException
```

Permite modificar la razón de interés del documento, siempre que no tenga notificaciones, y las observaciones del interesado en el documento. Se comprueba que existan todos los parámetros y además que el expediente no esté archivado.

Si se define un componente w@rdA en el sistema establecido en el TrAPIUI y el documento del que se modifica el interesado está guardado en el mismo w@rdA se actualizan sus datos.

Entradas

TpoPK idExpediente	*Identificador del expediente* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idInteresado	*Identificador del interesado* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idRazonInt	*Identificador de la razón de interés del interesado del expediente* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idRazonIntDocu	*Identificador de la razón de interés del interesado en el documento.* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idRazonIntDocuNueva	*Identificador de la nueva razón de interés del interesado en el documento.* *Si el identificador es 'null no se actualiza.*
TpoPK idDocExp	*Identificador del documento del expediente* *Si el identificador es 'null', o no válido, se generará una excepción*

String observaciones	*Observaciones para el nuevo interesado en el documento*

(w) Eliminar interesado en un documento de un expediente

```
void eliminarInteresadoDocumento(
                    TpoPK idExpediente,
                    TpoPK idInteresado,
                    TpoPK idRazonInt,
                    TpoPK idRazonIntDocu
                    TpoPK idDocExp,
                    boolean elimNotif) throws TrException
```

Permite eliminar los interesados en los documentos del expediente. Se comprueba que existan todos los parámetros, excepto el identificador de la razón de interés del documento, y además que el expediente no esté archivado.

Se da la posibilidad de eliminar las notificaciones asociadas al interesado en el documento. Para ello se usa el parámetro *elimNotif* con los siguientes valores:

- **true:** Indica que se eliminen las notificaciones, en caso de que hubiese.
- **false:** Indica que no se eliminen las notificaciones, generando una excepción en caso de que hubiese.

Si se define un componente w@rdA en el sistema establecido en el TrAPIUI y el documento del que se elimina el interesado está guardado en el mismo w@rdA se elimina también de éste.

Entradas

TpoPK idExpediente	*Identificador del expediente* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idInteresado	*Identificador del interesado* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idRazonInt	*Identificador de la razón de interés del interesado del expediente* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idRazonIntDocu	*Identificador de la razón de interés del interesado en el documento.* Opcional
TpoPK idDocExp	*Identificador del documento del expediente* *Si el identificador es 'null no se actualiza.*

boolean elimNotif	*Indica si se eliminan las notificaciones del interesado en el documento.* **true:** Se eliminan las notificaciones **false:** No se eliminan las notificaciones, generando una excepción en caso de que tuviese.

Obtener notificaciones

```
TrNotificacionInteresado[ ] obtenerNotificacionesInteresado(
                    TpoPK idExpediente,
                    TpoPK idInteresado,
                    TpoPK idRazonInt,
                    TpoPK idDocExp,
                    TpoPK idRazonIntDocu,
                    ClausulaWhere where,
                    ClausulaOrderBy orderBy) throws TrException
```

Permite obtener las notificaciones de los interesados en los documentos de un expediente. Si se le pasa el únicamente el parámetro *idExpediente* devolverá todas las notificaciones de todos los interesados en todos los documetos de ese expediente.

Entradas

TpoPK idExpediente	*Identificador del expediente* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idInteresado	*Identificador del interesado* Opcional
TpoPK idRazonInt	*Identificador de la razón de interés del interesado del expediente* Opcional
TpoPK idDocExp	*Identificador del documento del expediente* Opcional
TpoPK idRazonIntDocu	*Identificador de la razón de interés del interesado en el documento.* Opcional
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrNotificacionInteresado.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrNotificacionInteresado.

Salida

TrNotificacionInteresado [ ]	*Array de objetos* TrNotificacionInteresado

Insertar notificación

```
void insertarNotificacionInteresado(
```

```
                    TrNotificacionInteresado notificacionInte) throws TrException
```

Permite insertar una nueva notificación para un interesado en un documento del expediente. Se comprueba que exista el interesado en el documento, que el documento sea notificable y que el expediente no esté archivado.

Entradas

TrNotificacionInteresado notificacionInte	*Notificación a insertar*

Modificar notificación

```
void modificarNotificacionInteresado(
                    TrNotificacionInteresado notificacionInte) throws TrException
```

Permite modificar una notificación de un interesado en un documento del expediente. Se comprueba que exista la notificación, el interesado en el documento, que el documento sea notificable y que el expediente no esté archivado.

Entradas

TrNotificacionInteresado notificacionInte	*Notificación a modificar*

Eliminar notificaciones

```
void eliminarNotificacionInteresado(
                    TpoPK idDocExp,
                    TpoPK idExpediente,
                    TpoPK idInteresado,
                    TpoPK idRazonInt,
                    TpoPK idRazonIntDocu) throws TrException
```

Permite eliminar las notificaciones de los interesados en los documentos de un expediente. Se comprueban todos los parámetros, excepto el *idRazonIntDocu* que es opcional. Además se comprueba que el documento sea notificable y que el expediente no esté archivado.

Entradas

TpoPK idDocExp	*Identificador del documento del expediente* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idExpediente	*Identificador del expediente* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idInteresado	*Identificador del interesado* *Si el identificador es 'null', o no válido, se generará una excepción*

TpoPK idRazonInt	*Identificador de la razón de interés del interesado del expediente* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idRazonIntDocu	*Identificador de la razón de interés del interesado en el documento.* Opcional

APIs sobre bloques de datos (tareas II)

Bloques de datos permitidos en una fase

```
TrBloquePermitido[ ] obtenerBloquesPermitidos(
        String bloqueLlamante,
        TpoPK idFase,
        TpoPK idDefProc,
        TpoPK idExpediente,
        TpoDate fecha,
        String condVisual,
        boolean soloUsuario,
        ClausulaWhere where,
        ClausulaOrderBy orderBy ) throws TrException
```

Devuelve al usuario el conjunto de datos sobre los bloques o módulos detalles para los que se tiene acceso desde una fase dada y para un procedimiento también dado, desde un bloque que también pasamos como parámetro (el nombre) y que representa el bloque desde el cual se puede llamar.

Además recibe los parámetros *idExpediente* y *fecha* para evaluar las condiciones. El parámetros *condVisual* permite indicar si se obtienen los bloques que cumplan las condiciones de visualización (“S”), las que no (“N”) o todas (“T”) y el parámetro soloUsuario permite filtrar por el usuario establecido en el api y su perfil.

Entradas

String bloqueLlamante	*Nombre del bloque llamante.*
TpoPK idFase	*Identificador de la fase.* *Si el identificador es ‘null’, o no válido, se generará una excepción*
TpoPK idDefProc	*Identificador de la definición del procedimiento.* *Si el identificador es ‘null’, o no válido, se generará una excepción*
TpoPK idExpediente	*Identificador del expediente* *Opcional*
TpoDate fecha	*Fecha* Si se pasa ‘null’, por defecto se coge la del sistema.
String condVisual	*Indica qué bloques se obtienen* → “**S**” – Las que cumplen las condiciones de visualización → “**N**” – Las que no cumplen las condiciones de visualización → “**T**” – Todos
boolean soloUsuario	*Indica si se filtra por el usuario establecido en el api y su perfil*
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrBloquePermitido.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrBloquePermitido.

Salida

TrBloquePermitido [ ]	Array de objetos TrBloquePermitido con la información de bloques permitidos.

Bloques definidos

```
TrBloque[ ] obtenerBloquesDefinidos(
        TpoPK idBloque,
        ClausulaWhere where,
        ClausulaOrderBy orderBy ) throws TrException
```

Devuelve los datos sobre los bloques o módulos detalles definidos en el tramitador. Si se indica el parámetro *idBloque* se devolverán los datos del bloque correspondiente, si no (es *null*) se devuelven todos.

Entradas

TpoPK idBloque	*Identificador del bloque.* *Si es null, devuelve los datos correspondientes a todos los bloques.*
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrBloque.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrBloque.

Salida

TrBloque *[ ]*	Array de objetos TrBloque con la información de bloques.

Parámetros de un bloque de datos

```
TrParametroBloque[ ] obtenerParametrosBloques(
        TpoPK idBloque,
        ClausulaWhere where,
        ClausulaOrderBy orderBy ) throws TrException
```

Devuelve al usuario el conjunto de parámetros definido para un bloque dado.

Entradas

TpoPK idBloque	*Identificador del bloque.*
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrParametroBloque.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrParametroBloque.

Salida

TrParametroBloque [ ]	*Array de objetos TrParametroBloque.*

Iniciar tarea del expediente

```
TrMensajeCondicionAccion[ ] iniciarTareaExpediente(
                                TpoPK idExpediente,
                                TpoPK idTareaFase,
                                TpoPK idDefProc,
                                String tipoTarea,
                                TpoDate fecha,
                                TpoDate fechaLimite,
                                String observaciones,
                                TpoPK idTareaExp) throws TrException
```

Permite iniciar una nueva tarea para el expediente indicado en el parámetro *idExpediente*. El tipo de la tarea puede ser: "**I**": Incorporar un documento al expediente; "**G**": Generar un documento para el expediente; "**M**": Tarea del tipo manipular datos; **O**": Otros tipos de tarea.

Entradas

TpoPK idExpediente	*Identificador del expediente.* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idTareaFase	*Identificador de la tarea en fase* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idDefProc	*Identificador de la definición del procedimiento* *Si el identificador es 'null', o no válido, se generará una excepción*
String tipoTarea	*Tipo de tarea* → "**I**" – Incorporar → "**G**" – Generar → "**M**" – Manipular datos → "**O**" – Otras
TpoDate fecha	*Fecha de inicio de la tarea* Si se pasa 'null' por defecto se coge la del sistema
TpoDate fechaLimite	*Fecha límite para realizar la tarea*
String observaciones	*Observaciones de la tarea*
TpoPK idTareaExp	*Identificador de la nueva tarea del expediente* Entrada/Salida

Salida

TrMensajeCondicionAccion [ ]	*Array de objetos TrMensajeCondicionAccion.*

Finalizar tarea del expediente

```
void finalizarTareaExpediente(
                                TpoPK idTareaExp,
                                String tipoTarea,
                                TpoDate fecha) throws TrException
```

Permite finalizar una tarea de un expediente. El tipo de la tarea puede ser: "**I**": Para un documento incorporado; "**G**": Para un documento generado; "**M**": Tarea del tipo manipular datos; **O**": Otros tipos de tarea.

Entradas

TpoPK idTareaExp	*Identificador de la tarea* *Si el identificador es 'null', o no válido, se generará una excepción*
String tipoTarea	*Tipo de tarea* → "**I**" – Incorporar → "**G**" – Generar → "**M**" – Manipular datos → "**O**" – Otras
TpoDate fecha	*Fecha fin de la tarea* Si se pasa 'null' por defecto se coge la del sistema

Descartar tarea del expediente

```
void descartarTareaExpediente(
                    TpoPK idTareaExp,
                    String tipoTarea,
                    TpoDate fecha) throws TrException
```

Permite descartar una tarea de un expediente. El tipo de la tarea puede ser: "**I**": Para un documento incorporado; "**G**": Para un documento generado; "**M**": Tarea del tipo manipular datos; **O**": Otros tipos de tarea.

Entradas

TpoPK idTareaExp	*Identificador de la tarea* *Si el identificador es 'null', o no válido, se generará una excepción*
String tipoTarea	*Tipo de tarea* → "**I**" – Incorporar → "**G**" – Generar → "**M**" – Manipular datos → "**O**" – Otras
TpoDate fecha	*Fecha de descarte de la tarea* Si se pasa 'null' por defecto se coge la del sistema

Eliminar tarea del expediente

```
void eliminarTareaExpediente(
                    TpoPK idTareaExp,
                    String tipoTarea) throws TrException
```

Permite eliminar una tarea de un expediente. El tipo de la tarea puede ser: "**I**": Para un documento incorporado; "**G**": Para un documento generado; "**M**": Tarea del tipo manipular datos; **O**": Otros tipos de tarea.

Entradas

TpoPK idTareaExp	*Identificador de la tarea* *Si el identificador es 'null', o no válido, se generará una excepción*
String tipoTarea	*Tipo de tarea* → "**I**" – Incorporar → "**G**" – Generar → "**M**" – Manipular datos → "**O**" – Otras

Modificar tarea del expediente

```
void modificarTareaExpediente(
                    TpoPK idTareaExp,
                    String tipoTarea,
                    TpoDate fechaInicio,
                    TpoDate fechaLimite,
                    String observaciones) throws TrException
```

Permite modificar la fecha de inicio, fecha limite y observaciones de una tarea de un expediente. El tipo de la tarea puede ser: "**I**": Para un documento incorporado; "**G**": Para un documento generado; "**M**": Tarea del tipo manipular datos; **O**": Otros tipos de tarea.

Entradas

TpoPK idTareaExp	*Identificador de la tarea* *Si el identificador es 'null', o no válido, se generará una excepción*
String tipoTarea	*Tipo de tarea* → "**I**" – Incorporar → "**G**" – Generar → "**M**" – Manipular datos → "**O**" – Otras
TpoDate fechaInicio	*Fecha de incio de la tarea* Si se pasa 'null' por defecto se coge la del sistema
TpoDate fechaLimite	*Fechalímite de realización de la tarea*
String observaciones	*Observaciones de la tarea*

Reanudar tarea del expediente

```
void reanudarTareaExpediente(
                    TpoPK idTareaExp,
                    String tipoTarea) throws TrException
```

Permite reanudar una tarea de un expediente. El tipo de la tarea puede ser: "**I**": Para un documento incorporado; "**G**": Para un documento generado; "**M**": Tarea del tipo manipular datos; **O**": Otros tipos de tarea.

Entradas

TpoPK idTareaExp	*Identificador de la tarea* *Si el identificador es 'null', o no válido, se generará una excepción*
String tipoTarea	*Tipo de tarea* → "**I**" – Incorporar → "**G**" – Generar → "**M**" – Manipular datos → "**O**" – Otras

Modificar estado de una tarea del expediente

```
void modificarEstadoOtrasTareas(
                    TpoPK idTarea,
                    String estado,
                    TpoDate fecha) throws TrException
```

Permite modificar el estado de la tarea que se indique en el parámetro idTarea. El parámetro estado indica el nuevo estado de la tarea y el parámetro fecha sólo tendrá en cuenta para los estados "D" y "F".

Entradas

TpoPK idTarea	*Identificador de la tarea* *Si el identificador es 'null', o no válido, se generará una excepción*
String estado	*Estado de la tarea* → "**I**" – Iniciada → "**D**" – Descartada → "**F**" – Finalizada
TpoDate fecha	*Fecha de descarte o finalización de la tarea.* Si es 'null' se coge la del sistema

Tareas permitidas

```
TrTareaPermitida[ ] obtenerTareasPermitidas(
                    TpoPK idFase,
                    TpoPK idDefProc,
                    TpoPK idExpediente,
                    TpoDate fecha,
                    String tareaLlamante,
                    String condVisual,
                    boolean soloUsuario,
                   ClausulaWhere where,
                   ClausulaOrderBy orderBy ) throws TrException
```

Devuelve un conjunto de tareas permitidas que puede realizar un expediente desde una fase y un procedimiento dados. Mediante el parámetro *condVisual* indicamos qué tareas queremos obtener: "S": las que cumplen las condiciones de visualización, "N": las que no las cumplen y "T": todas. Con el parámetro *soloUsuario* se prmite filtrar por el usuario establecido en el api y su perfil.

Entradas

TpoPK idFase	*Identificador de la fase* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idDefProc	*Identificador de la definición del procedimiento* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idExpediente	*Identificador del expediente* Opcional
TpoDate fecha	*Fecha* Si es 'null' se coge la del sistema
String tareaLlamante	*Tarea llamante*
String condVisual	*Indica qué tareas se obtienen* → "**S**" – Las tareas que cumplen las condiciones de visualización → "**N**" – Las tareas que no cumplen las condiciones de visualización → "**T**" – Todas
boolean soloUsuario	*Indica si se filtra por el usuario establecido en el api y su perfil*
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTareaPermitida.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTareaPermitida.

Salida

TrTareaPermitida [ ]	*Array de objetos TrTareaPermitida.*

Tareas del expediente

```
TrTareaExpediente[ ] obtenerTareasExpediente(
                        TpoPK idExpediente,
                        boolean soloUsuario,
                        ClausulaWhere where,
                        ClausulaOrderBy orderBy ) throws TrException
```

Devuelve un conjunto de tareas definidas para un expediente. Si se indica el parámetro *soloUsuario* a **true** devolvera solo aquellas tareas para las que el usuario tenga permisos establecidos.

Entradas

TpoPK idExpediente	*Identificador del expediente* *Si el identificador es 'null', o no válido, se generará una excepción*
boolean soloUsuario	*Indica si se filtra por el usuario establecido en el api y su perfil*
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTareaExpediente.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTareaExpediente.

Salida

TrTareaExpediente [ ]	*Array de objetos TrTareaExpediente.*

Evaluar condiciones de otras tareas

```
TrMensajeCondicionAccion[ ] evaluarCondicionesOtrasTareas(
                        TpoPK idExpediente,
                        TpoPK idTareaFase,
                        TpoPK idDefProc,
                        String comprobar,
                        TpoBoolean resultado,
                        TpoDate fecha ) throws TrException
```

Permite al usuario evaluar el conjunto de condiciones asociadas a una tarea para un expediente dado en un procedimiento también dado.

Entradas

TpoPK idExpediente	*Identificador del expediente.* *Opcional.*
TpoPK idTareaFase	*Identificador de la tarea en fase.* *Si el identificador es 'null', o no válido, se generará una excepción.*
TpoPK idDefProc	*Identificador de la definición de procedimiento.* *Si el identificador es 'null', o no válido, se generará una excepción.*

String comprobar	Usado para indicar cuáles son las condiciones que se quieren evaluar: → "**I**" – Evaluar las condiciones definidas al iniciar la tarea → "**V**" – Evaluar las condiciones definidas para visualización.
TpoBoolean resultado	Indica si alguna de las condiciones es obligatoria y no se cumple. → '**false'** – Alguna da las condiciones es obligatoria y no se cumple. → '**true'** – No existen condiciones obligatorias que no se cumplan.
TpoDate fecha	Entrada/Salida. Fecha Si no se indica fecha, tomará por defecto la del sistema.

Salida

TrMensajeCondicionAccion [ ]	Array de objetos TrMensajeCondicionAccion con los mensajes resultado de evaluar las condiciones.

APIs sobre caducidades

Caducidades activas del expediente

TrCaducidadExpediente[ ] obtenerCaducidadesExpediente (
TpoPK idExpediente,
ClausulaWhere where,
ClausulaOrderBy orderBy) *throws TrException*

Devuelve las caducidades activas para un expediente.

Entradas

TpoPK idExpediente	*Identificador del expediente.* *Si el identificador es 'null', o no válido, se generará una excepción*
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrCaducidadExpediente.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrCaducidadExpediente.

Salida

TrCaducidadExpediente [ ]	*Array de objetos* TrCaducidadExpediente.

Modificaciones en la caducidad del expediente

TrModificacionCaducidadExpediente[ ] obtenerModificacionesCaducidadExpediente(
TpoPK idCaducidadExp,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Devuelve las modificaciones realizadas sobre una caducidad concreta de un expediente.

Entradas

TpoPK idCaducidadExp	*Identificador de la caducidad del expediente.* Si el identificador es 'null', o no válido, se generará una excepción.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrModificacionCaducidadExpediente.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrModificacionCaducidadExpediente.

Salida

TrModificacionCaducidadExpediente *[ ]*	*Array de objetos* TrModificacionCaducidadExpediente*.* *→ Para el caso de "suspensiones" de tiempo, no tiene sentido devolver la información de loas atributos UNIDAD y NUMUNIDADES. En este caso sólo se devolverá FECHA (en la que se produce la suspensión) y FECHAFINAL (en la que se reanuda la cuenta de tiempo).*

Para las modificaciones tipo "S", si esta fecha final es nula indicará que sobre la caducidad se ha suspendido el plazo y aún no se ha reanudado.

Sólo podrá existir una modificación "viva" del tipo "S", es decir, sólo existirá una con fecha final nula.

Avisos establecidos en la caducidad del expediente

```
TrAvisoCaducidad[ ] obtenerAvisosCaducidadExpediente(
        TpoPK idCaducidadExp,
        ClausulaWhere where,
        ClausulaOrderBy orderBy ) throws TrException
```

Devuelve los avisos definidos sobre una caducidad de un expediente.

Entradas

TpoPK idCaducidadExp	*Identificador de la caducidad del expediente.* *Si el identificador es 'null', o no válido, se generará una excepción*
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrAvisoCaducidad.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrAvisoCaducidad.

Salida

TrAvisoCaducidad [ ]	*Array de objetos* TrAvisoCaducidad.

Ampliar o reducir el plazo para una caducidad del expediente

```
void ampliarReducirCaducidadExpediente(
        TpoPK idCaducidadExp,
        String tipo,
        String unidad,
        int numUnidades,
        TpoDate fecha) throws TrException
```

Permite modificar una caducidad del expediente, ampliando o reduciendo el plazo.

Entradas

TpoPK idCaducidadExp	*Identificador de la caducidad del expediente.* *Si el identificador es 'null', o no válido, se generará una excepción*
String tipo	*Tipo de la modificación.* → "**A**" – Ampliación → "**R**" – Reducción
String unidad	*Unidad de medida del tiempo.* → "**D**" – Días → "**M**" – Meses → "**A**" – Años
Int numUnidades	*Número de unidades que se cuentan.(nº de días, meses, años).*
TpoDate fecha	*Fecha en la que se produjo la modificación sobre la caducidad o el aviso.* Si este parámetro no se indica se toma la fecha del sistema

Suspender o reanudar el plazo para una caducidad del expediente

```
void suspenderReanudarCaducidad(
        TpoPK idCaducidadExp,
        String tipo,
        TpoDate fecha) throws TrException
```

Permite modificar una caducidad del expediente, suspendiendo o reanudando el plazo.

Entradas

TpoPK idCaducidadExp	*Identificador de la caducidad del expediente.* *Si el identificador es 'null', o no válido, se generará una excepción*
String tipo	*Tipo de la modificación.* → "**S**" – Suspender el plazo → "**R**" – Reanudar el plazo
TpoDate fecha	*Fecha en la que se produce la suspensión op reanudación.* Si no se indica, toma la fecha del sistema.

Establecer avisos sobre una caducidad del expediente

```
void establecerAvisoCaducidadExpediente(
        TpoPK idCaducidadExp,
        TpoDate fecha,
        TpoDate fechaAviso,
        TpoPK idTransicion,
        String usuario) throws TrException
```

Permite definir un aviso previo, en transición que se hace o en tiempo para la caducidad de un expediente.

Entradas

TpoPK idCaducidadExp	*Identificador de la caducidad del expediente.* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoDate fecha	*Fecha en la que se da de alta el aviso.* Si no se indica, toma la fecha del sistema.
TpoDate fechaAviso	*Fecha en la que se quiere que se avise.*
TpoPK idTransicion	*Identificador de la transición* en la que hay que avisar al usuario. Si el identificador es 'null', o no válido, se generará una excepción
String usuario	*Usuario al que hay que avisar.*

Para establecer el aviso se debe indicar *fechaAviso* o *idTransicion* y obligatoriamente *usuario*.

Si se indican tanto *fechaAviso* como *idTransicion* el aviso se entenderá como lo que se produzca primero, es decir, o bien llegue la fecha de aviso o bien en el expediente se produzca la transición.

Obtención de los expedientes caducados

```
TrExpedienteCaducado[ ] obtenerExpedientesCaducados(
                TpoPK idStma,
                TpoDate fechaReferencia,
                String tipo,
                ClausulaWhere where,
                ClausulaOrderBy orderBy ) throws TrException
```

Devuelve al usuario para un sistema dado (sino se indica sistema de devuelven todos), el conjunto de expedientes que estarían caducados a la fecha de referencia indicada (si no se indica se toma la del sistema).

El expediente puede haber caducado bien por una de las caducidades definidas en el flujo, o bien porque ha superado la fecha límite de estancia en una fase.

Entradas

TpoPK idStma	*Identificador del sistema.* *Opcional*
TpoDate fechaReferencia	*Fecha de referencia*
String tipo	*Tipo de caducidad que se quiere buscar.* → "**C**" – Caducidades definidas en el flujo → "**F**" – Caducidades de fase → "**A**" – Ambas
ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrExpedienteCaducado. → Filtros sobre *CAMPO_REFSTMA* no tendrán efecto si el método recibe un identificador de sistema distinto de 'null'. → Filtros sobre CAMPO_REFCADEXP no tendrán efecto si el tipo de filtro es "F". → Filtros sobre CAMPO_REFFASE no tendrán efecto si el tipo de filtro es "C".
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrExpedienteCaducado.

Salida

TrExpedienteCaducado [ ]	*Array de objetos* TrExpedienteCaducado*.* → Cuando el expediente ha caducado debido a una de las caducidades definidas en el flujo, se devuelve CADUCIDADEXP.REFCADEXP pero si ha caducado por haber superado el tiempo límite en una fase se devolverá REFFASE.

Caducidades definidas

```
TrCaducidad[ ] obtenerCaducidadesDefProcedimiento (
                    TpoPK idDefProc,
                    ClausulaWhere where,
                    ClausulaOrderBy orderBy ) throws TrException
```

Devuelve las caducidades definidas sobre un procedimiento concreto que se pasa como parámetro.

Entradas

TpoPK idStma	*Identificador de la definición del procedimiento.* *Si el identificador es 'null', o no válido, se generará una excepción*
ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrCaducidad.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrCaducidad.

Salida

TrCaducidad *[ ]*	*Array de objetos* TrCaducidad.

APIs sobre el sistema de intercambio de mensajes

Mensajes de un usuario

```
TrMensaje[ ] obtenerMensajesUsuario(
                ClausulaWhere where,
                ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener el conjunto de mensajes que un usuario tiene en el sistema de intercambio de mensajes.

Entradas

ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrMensaje.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrMensaje.

Salida

TrMensaje *[ ]*	*Array de objetos* TrMensaje.

Mensajes enviados

```
TrMensaje[ ] obtenerMensajesEnviados(
                ClausulaWhere where,
                ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener el conjunto de mensajes que un usuario ha enviado mediante el sistema de intercambio de mensajes.

Entradas

ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrMensaje.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrMensaje.

Salida

TrMensaje *[ ]*	*Array de objetos* TrMensaje.

Modificar el estado de un mensaje

```
void modificarEstadoMensaje(
        TpoPK idMensaje,
        String leido) throws TrException
```

Permite al usuario modificar el estado de un mensaje recibido, permite poner un mensaje no leído como leído. Un usuario "Administrador" puede modificar el estado de todos los mensajes, ya sean suyos o no.

Entradas

TpoPK idMensaje	*Identificador del mensaje.*
String leido	*Indica el estado del mensaje.* → '**S**' – El mensaje ha sido leído → '**N'** – El mensaje no ha sido leído

Enviar un mensaje a un usuario

```
void crearMensaje(
        String texto,
        String prioridad,
        String usuarioDestino,
        TpoPK idExpediente) throws TrException
```

Permite al usuario crear una mensaje en el sistema de intercambio de mensaje. Permite indicar el texto, prioridad, el usuario destino y asociar el mensaje a un expediente.

Entradas

String texto	*Texto del mensaje*
String prioridad	*Indica la prioridad del mensaje* → '**A**' – Prioridad alta → '**M'** – Prioridad media → '**B**' – Prioridad baja
String usuarioDestino	*Usuario receptor del mensaje*
TpoPK idExpediente	*Identificador del expediente asociado al mensaje.* *Si el identificador es 'null', o no válido, se generará una excepción*

Eliminar un mensaje

```
void eliminarMensaje(
                TpoPK idMensaje) throws TrException
```

Permite a un usuario la eliminación de los mensajes recibidos en el sistema de intercambio de mensajes.

Si el usuario es “administrador” puede eliminar cualquier mensaje ya sea suyo o no.

Entradas

TpoPK idMensaje	*Identificador del mensaje.* *Si el identificador es ‘null’, o no válido, se generará una excepción.*

APIs sobre firmas de documentos

Firmantes definidos

TrFirmante[ ] obtenerFirmantesDefinidos(
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener todos los firmantes definidos en el sistema.

Entradas

ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrFirmante.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrFirmante.

Salida

TrFirmante [ ]	*Array de objetos* TrFirmante.

Firmantes tipo documentos

TrFirmanteTipoDocumento[ ] obtenerFirmantesTipoDocumento(
TpoPK idTipoDoc,
TpoDate fecha) throws TrException

Permite obtener los firmantes definidos en el sistema y vigentes a la fecha que se pasa como parámetro para el tipo de documento que se indica.
Como un mismo documento puede tener distintos firmantes dependiendo de la Unidad Orgánica donde estemos tratando el documento, la lista de firmantes se devolverá en función de la Unidad Orgánica del usuario que hace la llamada al método, para ello se filtra por el tipo de Unidad Orgánica y la provincia.

Entradas

TpoPK idTipoDoc	*Identificador del tipo de documento.*
TpoDate fecha	*Fecha* Si no se indica fecha de alta, por defecto se tomará la del sistema.

Salida

TrFirmanteTipoDocumento [ ]	*Array de objetos* TrFirmanteTipoDocumento.

Disposiciones firmantes

TrTextoDisposicion[ ] obtenerDisposicionesFirmante(
TpoPK idTeDi,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener los datos de la disposición que se indica como parámetro.

Entradas

TpoPK idTeDi	*Identificador de la disposición.* *Si el identificador es "null" devolverá todas.*
ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrTextoDisposicion.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrTextoDisposicion.

Salida

TrTextoDisposicion *[ ]*	*Array de objetos* TrTextoDisposicion.

Delegar sustituir firma

```
void delegarSustituirFirmante(
        String idPtoTrab,
        TpoPK idUniOrg,
        String idPtoTrabDelegSust,
        TpoPK idUniOrgDelegSust,
        TpoPK idTeDi,
        TpoDate fecha,
        TpoDate fechaFin,
        String tipo,
        TpoPK idTipoDoc) throws TrException
```

Permite establecer delegaciones/sustituciones de firma en el sistema.

Entradas

String idPtoTrab	*Identificador del puesto de trabajo del firmante principal.*
TpoPK idUniOrg	*Identificador de la unidad orgánica del firmante principal*
String idPtoTrabDelegSust	*Identificador del puesto de trabajo del delegado/sustituto*
TpoPK idUniOrgDelegSust	*Identificador de la Unidad orgánica del delegado/sustituto*
TpoPK idTeDi	*Identificador de textos disposiciones*
TpoDate fecha	*Fecha en que comienza la sustitución.* *Si no se indica nada, por defecto toma la del sistema.*
TpoDate fechaFin	*Fecha en la que termina la sustitución.*
String tipo	*Tipo* → "**D**" – Delegación (valor por defecto) → "**S**" – Sustitución
TpoPK idTipoDoc	*Identificador del tipo de documento en el que se delega (para el caso de delegaciones).*

Revocar delegar sustituir firma

```
void revocarDelegacionSustitucionFirmante(
        String idPtoTrab,
        TpoPK idUniOrg,
        String idPtoTrabSust,
        TpoPK idUniOrgSust,
        TpoDate fecha,
        String tipo,
        TpoPK idTipoDoc) throws TrException
```

Complementaria a la anterior, permite revocar las delegaciones/sustituciones de firma.

Entradas

String idPtoTrab	*Identificador del puesto de trabajo del firmante principal.*
TpoPK idUniOrg	*Identificador de la unidad orgánica del firmante principal*
String idPtoTrabSust	*Identificador del puesto de trabajo del delegado/sustituto*
TpoPK idUniOrgSust	*Identificador de la Unidad orgánica del delegado/sustituto*
TpoDate fecha	*Fecha de revocación* *Si no se indica nada, por defecto toma la del sistema.*
String tipo	*Tipo* → "**D**" – Delegación (valor por defecto) → "**S**" – Sustitución
TpoPK idTipoDoc	*Identificador del tipo de documento en el que se delega (para el caso de delegaciones).*

Firmas documentos

TrFirmaDocumentoExpediente[ ] obtenerFirmasDocumento(
TpoPK idDocExpte,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener todos los datos de las firmas del documento que se indica como parámetro.

Entradas

TpoPK idDocExpte	*Identificador del documento.* *Si el identificador es 'null', o no válido, se generará una excepción.*
ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrFirmaDocumentoExpediente.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrFirmaDocumentoExpediente.

Salida

TrFirmaDocumentoExpediente [ ]	*Array de objetos* TrFirmaDocumentoExpediente.

Firmar documento

```
void firmarDocumento(
                    TpoPK idDocExpte,
                    TpoDate fecha,
                    String idPuestoTrab,
                    TpoPK idUniOrg,
                    String usuarioPTUO,
                    String editable,
                    String txtPie,
                    String txtFdo) throws TrException
```

Permite firmar un documento que se pasa como parámetro.

Se podrá firmar siempre que se tengan permisos para ello en el tramitador.

Entradas

TpoPK idDocExpte	*Identificador del documento.* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoDate fecha	*Fecha de la firma.* *Si no se indica nada, por defecto toma la del sistema.*
String idPuestoTrab	*Identificador del puesto de trabajo del firmante.*
TpoPK idUniOrg	*Identificador de la unidad orgánica del firmante, en caso de que el usuario que llama a la función no lo sea.*
String usuarioPTUO	*Usuario que ocupa el puesto de trabajo,*
String editable	*Indica si el pie de firma es editable o no.* → "**S**" – Sí es editable → "**N**" – No es editable
String txtPie	*Texto de pie de firma.*
String txtFdo	*Texto firmado.*

Si el usuario que llama a la función no se corresponde con el firmante, entonces se estará hablando de una "firma manuscrita" y será necesario indicar el puesto de trabajo y la unidad orgánica del firmante.

Ya que varios usuarios pueden ocupar el mismo puesto, indicando el usuario que ocupa el puesto, se dice qué usuario es realmente el que firma.

Anular firma de un documento

```
void anularFirmaDocumento(
        TpoPK idDocExpte,
        String idPuestoTrab
        TpoPK idUniOrg
        String usuarioPTUO ) throws TrException
```

Permite anular la firma de un documento que se indica como parámetro.

Sólo se permite anular la firma al usuario que la ordenó.

Entradas

TpoPK idDocExpte	*Identificador del documento.* *Si el identificador es 'null', o no válido, se generará una excepción*
String idPuestoTrab	*Identificador del puesto de trabajo del firmante.*
TpoPK idUniOrg	*Identificador de la unidad orgánica del firmante.*
String usuarioPTUO	*Usuario que ocupa el puesto de trabajo.*

(w) Incluir firma digital a un documento

```
void incluirFirmaDigital(
        TpoPK idComponente,
        TpoPK idDocExp,
        String usuarioDig,
        TpoPK idUniOrg,
        String codPtoTrab,
        TpoDate fecha,
        String codTransaccion,
        byte[ ] pkcs7) throws TrException
```

Permite incluir una firma digital al documento que se indique en el parámetro idDocExp. Si se ha definido un componente para guardar los documentos se envían los datos de la firma.

Entradas

TpoPK idComponente	*Identificador del componente* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idDocExp	*Identificador del documento.* *Si el identificador es 'null', o no válido, o el expediente está archivado, se generará una excepción*
String usuarioDig	*Identificador del usuario digital*
TpoPK idUniOrg	*Identificador de la unidad orgánica u organismo.*
String codPtoTrab	*Código del puesto de trabajo*
TpoDate fecha	*Fecha de la firma.* *Obligatoria.*
String codTransaccion	*Código de la transacción de firma*
byte[ ] pkcs7	*Recibo de la firma digital*

Eliminar firma digital

```
void eliminarFirmaDigital(
                TpoPK idDocExp,
                String usuarioDig,
                TpoPK idUniOrg,
                String codPtoTrab) throws TrException
```

Permite eliminar una firma digital de un documento.

Entradas

TpoPK idDocExp	*Identificador del documento.* *Si el identificador es 'null', o no válido, o el expediente está archivado, se generará una excepción*
String usuarioDig	*Identificador del usuario digital*
TpoPK idUniOrg	*Identificador de la unidad orgánica u organismo.*
String codPtoTrab	*Código del puesto de trabajo*

APIs sobre interesados

Obtener interesados

```
TrInteresado[ ] obtenerInteresados(
                    ClausulaWhere where,
                    ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener un conjunto de interesados.

Entradas

ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrInteresado.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrInteresado.

Salida

TrInteresado [ ]	*Array de objetos TrInteresado*. El atributo OTROSDATOS nos indica si el interesado tiene recogido otros datos – "S" – ó no los tiene - "N".

Obtener datos de contacto del interesado

```
TrDatosContacto[ ] obtenerDatosContactoInteresado(
                    TpoPK idInteresado,
                    ClausulaWhere where,
                    ClausulaOrderBy orderBy ) throws TrException
```

Devuelve los datos de contacto de un interesado.

Entradas

TpoPK idInteresado	*Identificador del interesado* *Si el identificador es 'null', o no válido se generará una excepción*
ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrInteresado.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrInteresado.

Salida

TrDatosContacto [ ]	*Array de objetos TrDatosContacto*.

Insertar interesado

```
TpoPK insertarInteresado(
                    TrInteresado interesado,
                    TrDatosContacto datosContactoDefecto) throws TrException
```

Permite insertar un nuevo interesado con sus datos de contacto por defecto. Si no se envían datos de contacto no se le asignará ninguno. Una vez insertado devuelve el identificador del nuevo interesado

Entradas

TrInteresado interesado	*Interesado a insertar*
TrDatosContacto datosContactoDefecto	*Datos de contacto por defecto para el interesado*

Salida

TpoPK	*Identificador del nuevo interesado*

Modificar interesado

```
void modificarInteresado(
                    TrInteresado interesado,
                    TrDatosContacto datosContactoDefecto) throws TrException
```

Permite modificar los datos del interesado y sus datos de contacto por defecto. Para ello se comprueba que exista el interesado (interesado.REFINTERESADO). Posteriormente se comprueba si se recibe el parámetro *datosContactoDefecto*. Si no se reciben datos de contacto se inserta sólo el interesado sin tener en cuenta el atributo interesado.REFDATOCONT, y si se recibe datos de contacto, se asignan por defecto al interesado.

Entradas

TrInteresado interesado	*Interesado a modificar*
TrDatosContacto datosContactoDefecto	*Datos de contacto por defecto para el interesado*

Eliminar interesado

```
void eliminarInteresado(
                    TpoPK idInteresado,
                    boolean elimRelaInt) throws TrException
```

Permite eliminar el interesado que se indique en el parámetro *idInteresado* así como sus relaciones, según se indique en el parámetro elimRelaInt.

Entradas

TpoPK idInteresado	*Identificador del interesado* *Si el identificador es 'null', o no válido se generará una excepción*
boolean elimRelaInt	*Indica si se eliminan las relaciones del interesado.* - ***true*** – Se eliminan las relaciones - ***false*** – *No* se eliminan las relaciones

Obtener razones de interés

```
TrRazonInteres[ ] obtenerRazonesInteres(
                    ClausulaWhere where,
                    ClausulaOrderBy orderBy) throws TrException
```

Devuelve un conjunto de razones de interés.

Entradas

ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrRazonInteres.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrRazonInteres.

Salida

TrRazonInteres [ ]	*Array de objetos TrRazonInteres*

Obtener interesados en el expediente

```
TrInteresadoExpediente[ ] obtenerInteresadosExpediente(
                                TpoPK idExpediente,
                                TpoPK idInteresado,
                                TpoPK idRazonInt,
                                ClausulaWhere where,
                                ClausulaOrderBy orderBy) throws TrException
```

Permite obtener un conjunto de interesados en un expediente que se indica mediante el parámetro *idExpediente*.

Entradas

TpoPK idExpediente	*Identificador del expediente* *Si el identificador es 'null', o no válido se generará una excepción*
TpoPK idInteresado	*Identificador del interesado* *Opcional*
TpoPK idRazonInt	*Identificador de la razón de interés* *Opcional*
ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrInteresadoExpediente.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrInteresadoExpediente.

Salida

TrInteresadoExpediente [ ]	*Array de objetos TrInteresadoExpediente*

Insertar interesado en un expediente

```
void insertarInteresadoExpediente(
                        TpoPK idExpediente,
                        TpoPK idRazonInt,
                        TrInteresado interesado,
                        TrDatosContacto datosContacto,
                        String observaciones) throws TrException
```

Permite asignar un interesado a un expediente concreto. Si no existe el interesado se insertará uno nuevo, en este caso los datos de contacto serán obligatorios. En caso de que el interesado ya existiese, y el parámetro *datosContacto* no se envíe, se tomarán por defecto los que tenga dicho interesado por defecto.

Entradas

TpoPK idExpediente	*Identificador del expediente* *Si el identificador es 'null', o no válido se generará una excepción*
TpoPK idRazonInt	*Identificador de la razón de interés* *Si el identificador es 'null', o no válido se generará una excepción*
TrInteresado interesado	*Interesado a insertar al expediente*
TrDatosContacto datosContacto	*Datos de contacto para el interesado en el expediente*
String observaciones	*Observaciones del interesado en el expediente*

Modificar interesado en un expediente

```
void modificarInteresadoExpediente(
                    TpoPK idExpediente,
                    TpoPK idInteresado,
                    TpoPK idRazonInt,
                    TrDatosContacto datosContacto,
                    TpoPK idRazonIntNueva,
                    String observaciones) throws TrException
```

Permite modificar un interesado en un expediente, así como sus datos de contacto. Además permite modificar la razón de interés del interesado en el expediente, siempre que no exista como interesado en un documento.

Entradas

TpoPK idExpediente	*Identificador del expediente* *Si el identificador es 'null', o no válido se generará una excepción*
TpoPK idInteresado	*Identificador del interesado* *Si el identificador es 'null', o no válido se generará una excepción*
TpoPK idRazonInt	*Identificador de la razón de interés* *Si el identificador es 'null', o no válido se generará una excepción*
TrDatosContacto datosContacto	*Datos de contacto para el interesado en el expediente*
TpoPK idRazonIntNueva	*Identificador de la nueva razón de interés*
String observaciones	*Observaciones del interesado en el expediente*

Eliminar interesado en un expediente

```
void eliminarInteresadoExpediente(
                    TpoPK idExpediente,
                    TpoPK idInteresado,
                    TpoPK idRazonInt,
                    boolean elimInteDocu) throws TrException
```

Permite eliminar los interesados en un expediente. Además si se indica en el parámetro *elimInteDoc* se eliminarán los registros hijos de interesados en un documento y sus notificaciones.

Entradas

TpoPK idExpediente	*Identificador del expediente* *Si el identificador es 'null', o no válido se generará una excepción*
TpoPK idInteresado	*Identificador del interesado* *Si el identificador es 'null', o no válido se generará una excepción*
TpoPK idRazonInt	*Identificador de la razón de interés* Opcional
boolean elimInteDoc	*Indica si se eliminan los interesados en el documento y sus notificaciones*

Obtener tipos de contacto

```
TrTipoContacto[ ] obtenerTiposContacto(
                        ClausulaWhere where,
                        ClausulaOrderBy orderBy) throws TrException
```

Permite obtener un conjunto de tipos de contactos.

Entradas

ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrTipoContacto.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrTipoContacto.

Salida

TrTipoContacto [ ]	*Array de objetos TrTipoContacto*

Obtener relaciones entre interesados

```
TrRelacionInteresado[ ] obtenerRelacionesInteresado(
                        TpoPK idInteresado,
                        ClausulaWhere where,
                        ClausulaOrderBy orderBy) throws TrException
```

Permite obtener un conjunto de objetos TrRelacionInteresado que recoge todas las relaciones del interesado especificado en el parámetro *idInteresado* con otros interesados (sentido de la relación AB), así como las relaciones de otros interesados con el *idInteresado* (sentido de la relación BA).

Entradas

TpoPK idInteresado	*Identificador del interesado* *Si el identificador es 'null', o no válido se generará una excepción*
ClausulaWhere where	*Filtro a aplicar.* Consultar campos posibles para TrRelacionInteresado.
ClausulaOrderBy orderBy	*Ordenación a aplicar.* Consultar campos posibles para TrRelacionInteresado.

Salida

TrRelacionInteresado [ ]	*Array de objetos TrRelacionInteresado*

Insertar relación entre interesados

```
void insertarRelacionInteresado(
                    TpoPK idInteIni,
                    TpoPK idInteIni,
                    TpoPK idTipoCont) throws TrException
```

Permite insertar una nueva relación entre interesados. Además de los identificadores de los interesados se deberá indicar el identificador del tipo de contacto existente entre ellos.

Entradas

TpoPK idInteIni	*Identificador de un interesado* *Si el identificador es 'null', o no válido se generará una excepción*
TpoPK idInteFin	*Identificador del otro interesado* *Si el identificador es 'null', o no válido se generará una excepción*
TpoPK idTipoCont	*Identificador del tipo de contacto entre los interesados* *Si el identificador es 'null', o no válido se generará una excepción*

Modificar relación entre interesados

```
void modificarRelacionInteresado(
                    TpoPK idInteIni,
                    TpoPK idInteIni,
                    TpoPK idTipoCont,
                    TpoPK idInteFinNuevo,
                    TpoPK idTipoContNuevo) throws TrException
```

Modifica una relación entre interesados. Permite modificar el interesado con el que se relaciona mediante el parámetro *idInteFinNuevo* así como el tipo de contacto entre ellos mediante el parámetro *idTipoContNuevo*.

Entradas

TpoPK idInteIni	*Identificador de un interesado* *Si el identificador es 'null', o no válido se generará una excepción*
TpoPK idInteFin	*Identificador del otro interesado* *Si el identificador es 'null', o no válido se generará una excepción*
TpoPK idTipoCont	*Identificador del tipo de contacto entre los interesados* *Si el identificador es 'null', o no válido se generará una excepción*
TpoPK idInteFinNuevo	*Identificador del otro interesado a relacionar* *Si el identificador es 'null', o no válido se generará una excepción*
TpoPK idTipoContNuevo	*Identificador del nuevo tipo de contacto entre los interesados* *Si el identificador es 'null', o no válido se generará una excepción*

Eliminar relación entre interesados

```
void eliminarRelacionInteresado(
                    TpoPK idInteIni,
                    TpoPK idInteIni,
                    TpoPK idTipoCont) throws TrException
```

Permite eliminar las relaciones entre los interesados indicando los identificadores de los interesados inicio, fin y el tipo de contacto. Además permite eliminar todas las relaciones de un interesado si sólo se indica el parámetro *idInteIni.*

Entradas

TpoPK idInteIni	*Identificador de un interesado* *Si el identificador es 'null', o no válido se generará una excepción*
TpoPK idInteFin	*Identificador del otro interesado* *Opcional*
TpoPK idTipoCont	*Identificador del tipo de contacto entre los interesados* Opcional

Otras APIs de obtención de datos definidas

Datos de una fase

```
TrFase[ ] obtenerDatosFase(
        TpoPK idFase,
        ClausulaWhere where,
        ClausulaOrderBy orderBy ) throws TrException
```

Devuelve datos relativos a la fase cuya id es pasado como parametro.

Entradas

TpoPK idFase	Identificador de la fase. Si el identificador es 'null' el método devolverá todas las fases.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrFase.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrFase.

Salida

TrFase [ ]	Array de objetos TrFase con la información de fases.

Variables sistema

```
TrVariable[ ] obtenerVariablesSistema(
        TpoPK idStma,
        ClausulaWhere where,
        ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener los valores de las variables de un sistema.

Entradas

TpoPK idStma	Identificador del sistema. Si el identificador es 'null' el método devolverá todas las variables del sistema.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrVariable.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrVariable.

Salida

TrVariable [ ]	Array de objetos TrVariable con la información de las variables del sistema

Valor de una variable

```
String obtenerValorVariable(
        TpoPK idVariable,
        TpoPK idDocExpte) throws TrException
```

Devuelve el valor de la variable utilizando los parámetros asignados al documento.

Entradas

TpoPK idVariable	*Identificador de la variable.* *Si el identificador es 'null', o no válido, se generará una excepción*
TpoPK idDocExpte	*Identificador del documento.* *Si el identificador es 'null', o no válido, se generará una excepción*

Salida

String	Valor de la variable

Relaciones establecidas

```
TrRelacionDefinida[ ] obtenerRelacionesDefinidas(
        TpoPK idStma,
        ClausulaWhere where,
        ClausulaOrderBy orderBy ) throws TrException
```

Mediante este procedimiento se obtienen las relaciones establecidas entre fases, transiciones, tipos de evolución, etc., definidas en el tramitador para un sistema que se indica como parámetro.

Entradas

TpoPK idStma	Identificador del sistema. Si el identificador es 'null' el método devolverá todas las relaciones definidas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrRelacionDefinida. Los filtros que usen TrRelacionDefinida *CAMPO_REFSTMA* no tendrán efecto si el método recibe un identificador de sistema distinto de 'null'.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrRelacionDefinida.

Salida

TrRelacionDefinida [ ]	Array de objetos TrRelacionDefinida

Tipos de identificador

TrTipoIdentificador[ ] obtenerTiposIdentificador(
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener los distintos tipos de identificador que existan.

Entradas

ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoIdentificador.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoIdentificador.

Salida

TrTipoIdentificador [ ]	Array de objetos *TrTipoIdentificador*

Tipos de organización

TrTipoOrganizacion[ ] obtenerTiposOrganizacion(
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener los distintos tipos de organización que existan.

Entradas

ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoOrganización.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoOrganización.

Salida

TrTipoOrganización [ ]	Array de objetos *TrTipoOrganización*

Componentes

TrComponente[ ] obtenerComponentes(
TpoPK idComponente,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Devuelve el componente que se indique en el parámetro *idComponente*. Si se envía 'null' se devuelven todos.

Entradas

TpoPK idComponente	Identificador del componente Si el identificador es 'null' devolverá todos los componentes
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrComponente.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrComponente.

Salida

TrComponente [ ]	Array de objetos *TrComponente*

Datos componentes

TrDatoComponente[ ] obtenerDatosComponente(
TpoPK idDatoComponente,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener los datos de los componentes que se indique en el parámetro *idDatoComponente*. Si se envía 'null' se devuelven todos.

Entradas

TpoPK idDatoComponente	Identificador del dato del componente Si el identificador es 'null' devolverá todos
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrDatoComponente.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrDatoComponente.

Salida

TrDatoComponente [ ]	Array de objetos *TrDatoComponente*

Sistemas

TrSistema[ ] obtenerSistemas(

```
                TpoPK idSistema,
                ClausulaWhere where,
                ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener los datos del sistema que se indique en el parámetro *idSistema*. Si se envía 'null' se devuelven todos.

Entradas

TpoPK idSistema	Identificador del sistema Si el identificador es 'null' devolverá todos
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrSistema.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrSistema.

Salida

TrSistema [ ]	Array de objetos *TrSistema*

Organismos

```
TrOrganismo[ ] obtenerOrganismos(
                TpoPK idOrganismo,
                ClausulaWhere where,
                ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener los datos del organismo que se indique en el parámetro *idOrganismo*. Si se envía 'null' se devuelven todos.

Entradas

TpoPK idOrganismo	Identificador del organismo Si el identificador es 'null' devolverá todos
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrOrganismo.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrOrganismo.

Salida

TrOrganismo [ ]	Array de objetos *TrOrganismo*

Tipos de organismos

```
TrTipoOrganismo[ ] obtenerTiposOrganismo(
        TpoPK idTipoOrg,
        ClausulaWhere where,
        ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener los datos de los tipos de organismo que se indique en el parámetro *idTipoOrg*. Si se envía 'null' se devuelven todos.

Entradas

TpoPK idTipoOrg	Identificador del tipo de organismo Si el identificador es 'null' devolverá todos
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoOrganismo.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoOrganismo.

Salida

TrTipoOrganismo [ ]	Array de objetos *TrTipoOrganismo*

Puestos de trabajo

```
TrPuestoTrabajo[ ] obtenerPuestosTrabajo(
        String codPtoTrab,
        ClausulaWhere where,
        ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener los datos de los tipos de organismo que se indique en el parámetro *idTipoOrg*. Si se envía 'null' se devuelven todos.

Entradas

String idPtoTrab	Identificador del puesto de trabajo Si el identificador es 'null' devolverá todos
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrPuestoTrabajo.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrPuestoTrabajo.

Salida

TrPuestoTrabajo [ ]	Array de objetos *TrPuestoTrabajo*

Puestos de trabajo por organismos

TrPtoTrabOrganismo[ ] obtenerPuestosTrabajoOrganismo(
String idPtoTrab,
TpoPK idOrganismo,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener los datos de los puestos de trabajo por organismos que se indiquen por parámetros. Si se pasan a 'null' se devolverán todos.

Entradas

String idPtoTrab	Identificador del puesto de trabajo Si el identificador es 'null' devolverá todos
TpoPK idOrganismo	Identificador del organismo Si el identificador es 'null' devolverá todos
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrPtoTrabOrganismo.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrPtoTrabOrganismo.

Salida

TrPtoTrabOrganismo [ ]	Array de objetos *TrPtoTrabOrganismo*

Empleados

TrEmpleado[ ] obtenerEmpleados(
String usuario,
TpoPK idOrganismo,
String idPtoTrab,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener los datos del empleado que se indique por parámetros. Si se pasan a 'null' se devolverán todos.

Entradas

String usuario	Identificador del usuario Si el identificador es 'null' devolverá todos
TpoPK idOrganismo	Identificador del organismo Si el identificador es 'null' devolverá todos
String idPtoTrab	Identificador del puesto de trabajo Si el identificador es 'null' devolverá todos
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrEmpleado.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrEmpleado.

Salida

TrEmpleado [ ]	Array de objetos *TrEmpleado*

Municipios

```
TrMunicipio[ ] obtenerMunicipios(
                String idMunicipio,
                String idProvincia,
                ClausulaWhere where,
                ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener los datos del municipio que se indique por parámetros. Si se pasan a 'null' se devolverán todos.

Entradas

String idMunicipio	Identificador del municipio Si el identificador es 'null' devolverá todos
String idProvincia	Identificador de la provincia Si el identificador es 'null' devolverá todos
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrMunicipio.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrMunicipio.

Salida

TrMunicipio [ ]	Array de objetos *TrMunicipio*

Provincias

```
TrProvincia[ ] obtenerProvincias(
                String idProvincia,
                ClausulaWhere where,
                ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener los datos de la provincia que se indique por parámetro. Si se pasa 'null' se devolverán todas.

Entradas

String idProvincia	Identificador de la provincia Si el identificador es 'null' devolverá todos
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrProvincia.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrProvincia.

Salida

TrProvincia [ ]	Array de objetos *TrProvincia*

Países

```
TrPais[ ] obtenerPaises(
        String idPais,
        ClausulaWhere where,
        ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener los datos del país que se indique por parámetro. Si se pasa 'null' se devolverán todos.

Entradas

String idPais	Identificador del país Si el identificador es 'null' devolverá todos
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrPais.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrPais.

Salida

TrPais [ ]	Array de objetos *TrPais*

Tipos de vía

```
TrTipoVia[ ] obtenerTiposVia(
        String idTipoVia,
        ClausulaWhere where,
        ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener los datos de los tipos de vía que se indique en el parámetro *idTipoVia*. Si se pasan a 'null' se devolverán todos.

Entradas

String idPtoTrab	Identificador del tipo de vía Si el identificador es 'null' devolverá todos
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoVia.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoVia.

Salida

TrTipoVia [ ]	Array de objetos *TrTipoVia*

Usuarios

```
TrUsuario[ ] obtenerUsuarios(
                    String idUsuario,
                    ClausulaWhere where,
                    ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener los datos de los usuarios que se indique en el parámetro *idUsuario*. Si se pasan a 'null' se devolverán todos.

Entradas

String idUsuario	Identificador del usuario Si el identificador es 'null' devolverá todos
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrUsuario.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrUsuario.

Salida

TrUsuario [ ]	Array de objetos *TrUsuario*

Perfiles del usuario

```
TrPerfilUsuario[ ] obtenerPerfilesUsuario(
                    String usuario,
                    ClausulaWhere where,
                    ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener los distintos perfiles de usuario que tiene asociado el usuario que se especifica por parámetros.

Entradas

String usuario	Identificador del usuario del que se quieren obtener sus perfiles. Si el identificador es 'null' o no válido se generará una excepción.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrPerfilUsuario.
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrPerfilUsuario.

Salida

TrPerfilUsuario [ ]	Array de objetos *TrPerfilUsuario*

Otros datos de interesados, expedientes y procedimientos

String obtenerOtrosDatos(
TpoPK id,
String tipo) throws TrException

Permite obtener otros datos de los interesados, expedientes o procedimientos según se indique en el parámetro *tipo.*

Entradas

TpoPK id	Identificador del interesado, expediente o procedimiento. Si el identificador es 'null' o no válido se generará una excepción.
String tipo	Indica el tipo del que se quieren obtener otros datos. → "**P**" – Procedimiento → "**I**" – Interesado → "**E**" – Expediente Si el tipo es 'null' o no válido se generará una excepción.

Salida

String	Otros datos del interesado, expediente o procedimiento.

void actualizarOtrosDatos(
TpoPK id,
String tipo,
String contenido) throws TrException

Permite actualizar otros datos de los interesados, expedientes o procedimientos según se indique en el parámetro *tipo.*

Entradas

TpoPK id	Identificador del interesado, expediente o procedimiento. Si el identificador es 'null' o no válido se generará una excepción.
String tipo	Indica el tipo del que se quieren obtener otros datos. → "**P**" – Procedimiento → "**I**" – Interesado → "**E**" – Expediente Si el tipo es 'null' o no válido se generará una excepción.
String contenido	Otros datos a actualizar en el interesadom expediente o procedimiento.

Otras APIs

Establecer el usuario

void establecerUsuarioSistema(
String usuario) throws TrException

Permite establecer el usuario.

Entradas

String usuario *Usuario*

Los usuarios deben tener el rol o perfil de usuario TR_R_USUARIO o el TR_R_ADMINISTRADOR.

Obtener usuario establecido

String obtenerUsuarioSistema(TpoString rol)

Permite obtener el usuario establecido en el TrAPIUI y su rol.

Entradas/Salidas

TpoString rol *Rol del usuario establecido en el api*
Parámetro de entrada/salida

Salida

String usuario *Usuario establecido en el api*

Obtener mensaje de error

String obtenerMensajeError(long numError) throws TrException;

Permite obtener el mensaje de error asociado al número de error que se indica en el parámetro numError.

Entradas

long numError *Número del error*

Salidas

String *Cadena con el mensaje de error*

Establece parámetros

```
void establecerConfiguracionSistema(
        String report,
        String conexion,
        TpoPK idStma) throws TrException
```

Este procedimiento permite parametrizar las URL devueltas por el tramitador para abrir los documentos almacenados. Los valores indicados en los parámetros deben estar definidos antes como constantes en el tramitador.

Entradas

String report	*Cadena de llamada para el report server.*
String conexion	*Cadena de conexión que se suministra al report server.*
TpoPK idStma	*Identificador del sistema*

Establecer conexión fija

```
synchronized void establecerConexionFija( boolean fija )
```

Permite establecer la conexión como fija, es decir una vez que se crea no se cierra hasta que no se ejecute el método cerrarSesion. Este método sólo tiene efecto cuando la TrAPIUI se ha creado con un perfil de conexión que sea un datasource.

Si se está usando un perfil de conexión que es un datasource y el conexionFija es false se devuelve la conexión al finalizar la ejecución de un método, en caso contrario no se devuelve.

Entrada

boolean fija	Indica si la conexión es fija para la TrAPIUI o se devuelve al pool de conexiones al finalizar la ejecución de un método. → "**true**" – La conexión es fija y no se devuelve al pool. Por defecto toma este valor. → "**false**" – Se devuelve la conexión al pool al terminar la ejecución de cada método, haciendose commit al final de cada método.

Obtener estado conexión fija

synchronized boolean obtenerEstadoConexionFija( )

Devuelve el valor actual del atributo conexionFija. El estado de la conexión fija sólo tiene efecto cuando la TrAPIUI se ha creado usando un perfil de conexión que es un datasource. Si el valor devuelto es true indica que la conexión se fija y no se devuelve al pool hasta cerrar la sesión, en caso contrario si el valor es false indica que no es fija y que se devuelve la conexión al pool al finalizar la ejecución de cada método.

Salida

boolean	Valor del atributo conexionFija → "**true**" – La conexión es fija y no se devuelve al pool. Por defecto toma este valor. → "**false**" – Se devuelve la conexión al pool al terminar la ejecución de cada método, haciendose commit al final de cada método.

Cierre de la sesión

void cerrarSesion( )

Cierra la sesión actual de trabajo, haciendo por defecto un commit si el autoCommit está a true, si no se realiza un rollback.

void cerrarSesion(boolean commit)

Cierra la sesión actual de trabajo. Realiza un commit o un rollback según se indique en el parámetro *commit*.

Entrada

boolean commit	→ "**true**" – Se realiza un commit para guardar los cambios. → "**false**" – Se realiza un rollback para no guardar los cambios, desde el último commit.

AutoCommit

void setAutoCommit(boolean value)

Establece el valor del autoCommit.

Entrada

boolean value	→ "**true**" – Se realiza un commit cada vez que se ejecute un api. → "**false**" – No se realiza commit cuando se ejecute un api, el usuario debe hacer uso del método commit() para guardar los cambios.

boolean getAutoCommit()

Devuelve el estado actual del autoCommit.

Salida

boolean	→ "**true**" – Se realiza un commit cada vez que se ejecute un api. → "**false**" – No se realiza commit cuando se ejecute un api, el usuario debe hacer uso del método commit() para guardar los cambios.

Commit y Rollback

boolean commit()

Realiza un commit sobre la base de datos guardando los últimos cambios.

Salida

boolean → "**true**" – El commit se ha realizado correctamente
→ "**false**" – El commit no se ha realizado correctamente

boolean rollback()

Realiza un rollback sobre la base de datos deshaciendo los últimos cambios.

Salida

boolean → "**true**" – El rollback se ha realizado correctamente
→ "**false**" – El rollback no se ha realizado correctamente

Comprobación de conexión

boolean hayConexion( )

Indica si existe o no una conexión válida a la base de datos.

Salida

boolean → "**true**" – Existe una conexión válida a la base de datos.
→ "**false**" – No existe conexión válida a la base de datos.

Comprobación del formato de fecha

String obtenerFormatoFecha( )

Devuelve el formato de fecha establecido en el TrAPIUI.

Salida

String Formato de fecha establecido.

APIs exclusivas para Port@firmas

El conjunto de apis exclusivas de port@firmas sólo funcionarán si se ha definido el componente de port@firmas en el sistema establecido en el TrAPIUI o en el sistema por defecto.

Actualiza entregas

void actualizaEntregas(TpoPK idDocExp) throws TrException

Método que comprueba las entregas en port@firmas y actualiza la información en trew@. Si el estado es devuelto lanza una exception con el código –20420 en cuyo mensaje se indica el usuario que la la ha devuelto. Otras excepciones que puede lanzar este método son si la petición no existe (-20419) y si el documento no ha sido enviado a port@firmas (-20418).

Entrada

TpoPK idDocExp — Identificador del documento del expediente a comprobar

Enviar un documento a port@firmas

```
void enviaDocumentoExpediente(
                    TpoPK[ ] arrayDocExp,
                    String aplicacion,
                    String tipoDocu,
                    boolean notificaEmail,
                    boolean notificaAviso,
                    String[ ] estadosNotificacion,
                    TpoDate fechaInicio,
                    TpoDate fechaCaducidad,
                    String remitenteNombre,
                    String remitenteEmail,
                    String referencia,
                    String asunto,
                    BigDecimal prioridad,
                    String texto,
                    TrAccionDocumentoPortafirmas[ ] accionesPf) throws TrException
```

Método que envía uno o varios documentos a port@firmas en una misma petición. Para ello todos los documentos deben tener los mismos firmantes, tomándose como referencia el primer documento del array. Permite indicar un array de acciones mediante el parámetro accionesPf para un cambio de estados del documento en port@firmas.

Entradas

TpoPK[ ] arrayDocExp	Identificadores de los documentos del expediente a enviar a port@firmas.
String aplicacion	Aplicación con la que se registra en portafirmas la petición
String tipoDocu	Tipo de documento (opcional). Si no se indica se envía la etiqueta del tipo de documento en Trew@.
boolean notificaEmail	Indica si se notifica por email (opcional)
boolean notificaAviso	Indica si se notifica por Avisador (opcional)
String[ ] estadosNotificacion	Estados en los que se quiere recibir notificaciones
TpoDate fechaInicio	Fecha de inicio de la firma
TpoDate fechaCaducidad	Fecha máxima de firma del documento
String remitenteNombre	Nombre del remitente de la petición
String remitenteEmail	email del remitente del petición
String referencia	Referencia o número de expediente
String asunto	Asunto
BigDecimal prioridad	Prioridad de la petición , (1 a 5), 0 o nulo valor por defecto
String texto	Texto adjunto a la petición
TrAccionDocumentoPortafirmas[ ] accionesPf	Acciones para un cambio de estados del documento en port@firmas

Estados de la petición

String[ ] obtenerEstadosPortafirmas() throws TrException

Método que devuelve todos los estados posibles de una petición a port@firmas

Salida

String[]	Array de objetos String´s con los estados posibles de una transacción

Tipos de documento

String[ ] obtenerTiposDocumentoPortafirmas() throws TrException

Método que devuelve la lista de tipos de documentos dados de alta en port@firmas

Salida

String[] Array de objetos String´s con los tipos de documentos admitidos

APIs exclusivas para @visador

El conjunto de apis exclusivas de @visador sólo funcionarán si se ha definido el componente de @visador en el sistema establecido en el TrAPIUI o en el sistema por defecto.

Crear aviso

```
void crearAviso(
        String asunto,
        String texto,
        String prioridad,
        int periodicidad,
        int repeticiones,
        String[ ] usuariosDestino,
        String[ ] gruposDestino,
        String[ ] emailsDestino) throws TrException;
```

Método que crea un aviso con los datos proporcionados

Entrada

String asunto	Título del aviso
String texto	Texto del aviso
String prioridad	Prioridad, (1 a 5), 0 o nulo valor por defecto
int periodicidad	Periodicidad, >0 repeticiones cada X segundos
int repeticiones	Repeticiones del aviso
String[ ] usuariosDestino	Lista de usuarios de aplicación destinatarios
String[ ] gruposDestino	Lista de grupos a los que están subscritos los usuarios destinatarios
String[ ] emailsDestino	Direcciones de correo electrónico

Crear mensaje aviso

```
void crearMensajeAviso(
        String asunto,
        String texto,
        String prioridad,
        int periodicidad,
        int repeticiones,
        String usuarioDestino,
        TpoPK idExpediente) throws TrException;
```

Método que genera un aviso mediante @visador y vincula este a un expediente

Entrada

String asunto	Título del aviso (solo aviso)
String texto	Contenido del mensaje/aviso
String prioridad	Prioridad del mensaje/aviso, (1 a 5), 0 o nulo valor por defecto
int periodicidad	Periodicidad del aviso (solo aviso)
int repeticiones	Veces que se remite el aviso (solo aviso)
String] usuarioDestino	Usuario destinatario del mensaje/aviso
TpoPK] idExpediente	Identificador del expediente al que se añade el mensaje

APIs exclusivas para Notific@dor

El conjunto de apis exclusivas de notific@dor sólo funcionarán si se ha definido el componente de notific@dor en el sistema establecido en el TrAPIUI o en el sistema por defecto.

Notificar documentos interesados

```
void notificaDocumentosInteresados(
                    TrInteresadoDocumento[ ] documentosInteresados,
                    String asunto,
                    String texto) throws TrException;
```

Método que para la lista de documentos por interesados, envía notificaciones a dichos usuarios con los documentos como adjuntos y con el asunto y texto pasado por parámetro.

Entrada

TrInteresadoDocumento[ ] documentosInteresados	Documentos y destinatarios
String asunto	Título de la notificación
String texto	Texto de la notificación

Revisar notificaciones pendientes

```
void revisaNotificacionesPendientes() throws TrException;
```

Método que revisa el estado de todas las notificaciones que estén pendientes.

Revisar notificaciones pendientes

```
void revisaNotificacionesPendientes(TpoPK idExpediente,
                                    TpoPK idDocExp,
                                    TpoPK idInteresado) throws TrException;
```

Método que revisa el estado de las notificaciones que estén pendientes, para el expediente, documento e interesado indicados.

Entrada

TpoPK idExpediente	Identificador del expediente. Opcional.
TpoPK idDocExp	Identificador del documento del expediente. Opcional.
TpoPK idInteresado	Identificador del interesado. Opcional.

Descripción de la TrAPIADM

Descripción de clases involucradas

Interfaz J-TrAPIADM

La interfaz J-TrAPIADM define todos los métodos de administración del motor de tramitación TREW@. En la librería este acceso se realiza a través de la implementación de la interfaz **TrAPIADM** que se describe en este apartado.

TrAPIADM

trewa.bd.trapi.trapiadm.TrAPIADM

Interfaz de administración del motor de tramitación TREW@. Todos los métodos quedan descritos en el apartado "***Métodos del TrAPIADM***".

Para poder trabajar con la TrAPIADM es preciso crear una instancia de la misma a través de la clase Factoría: ***TrAPIADMFactory***, descrita en el apartado siguiente.

Clase Factoría

La clase Factoría se encarga de crear una instancia correcta de la J-TrAPIADM de forma que los métodos de esta última aparezcan disponibles para las aplicaciones.

TrAPIADMFactory

trewa.bd.trapi.trapiadm.TrAPIADMFactory

Clase que permite la correcta creación de una instancia de la TrAPIADM.

Métodos para la creación de instancias del TrAPIADM

→ `TrAPIADM crearAPIADM ()`

Devuelve una instancia del TrAPIADM. La información para el establecimiento de la conexión la obtiene a través del perfil “default”.

Salidas	*TrAPIADM*	Instancia del TrAPIADM o “null” si no se pudo crear.

→ `TrAPIADM crearAPIADM(String usuario, String clave)`

Devuelve una instancia del TrAPADM. La información para el establecimiento de la conexión la obtiene a través del perfil “default” pero en este caso hace uso del usuario y password que recibe como parámetros.

Entradas	*String usuario*	Usuario
	String clave	Clave de usuario
Salidas	*TrAPIADM*	Instancia del TrAPIADM o “null” si no se pudo crear.

→ `TrAPIADM crearAPIADM(String strPerfil)`

Devuelve una instancia del TrAPIADM. La información para el establecimiento de la conexión la obtiene a través del perfil especificado en la llamada como parámetro.

Entradas	*String strPerfil*	*Nombre del perfil para el establecimiento de la conexión.*
Salidas	*TrAPIADM*	*Instancia del TrAPIADM o “null” si no se pudo crear.*

→ `TrAPIADM crearAPIADM(String strPerfil, String usuario, String clave)`

Devuelve una instancia del TrAPIADM. La información para el establecimiento de la conexión la obtiene a través del perfil especificado en la llamada como parámetro, pero en este caso hace uso del usuario y password que recibe como parámetros.

Entradas	*String strPerfil*	*Nombre del perfil para el establecimiento de la conexión.*
	String usuario	*Usuario*
	String clave	*Clave de usuario*
Salida	*TrAPIADM*	*Instancia del TrAPIADM o “null” si no se pudo crear.*

Ejemplo de uso

En el siguiente ejemplo se muestra cómo obtener una instancia del TrAPIADM a través de la clase TrAPIADMFactory:

```
TrAPIADM apiADM = null;

apiADM = TrAPIADMFactory.crearAPIADM( “perfilTrewa”);
if ( apiADM != null ){
       // Ya se puede acceder a los métodos del TrAPIADM
       pK = new TpoPK( BigDecimal.valueOf(39) );
       try{
           TrDefProcedimiento[] def = apiADM.obtenerDefProcedimiento(null,null,null);
       ...
      }
     catch(TrException ex)
     {
         ex.printStackTrace();
     }
}
```

Clases de entidad

Estas clases son el mecanismo de intercambio de información entre las aplicaciones y el motor Trew@ a través de las J-TrAPIs.

Las clases de entidad asociadas al TrAPIADM se encuentran en el paquete **trewa.bd.trapi.tpo**. Estas clases de entidad permiten realizar todo el mantenimiento del motor de tramitación Trew@ mediante el uso de métodos de inserción, modificación, eliminación y obtención.

TrAccion

trewa.bd.trapi.tpo.TrAccion

Clase que representa la información de una acción.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFACCION**	TpoPK	Identificador de la acción.
→ **NOMBRE**	String	Nombre con el que se denomina la acción.
→ **DESCRIPCION**	String	Texto que describe la acción.
→ **PARAMREFEXP**	String	Indica si se le pasa el id del expediente a la acción. → "**S**" – Sí → "**N**" – No
→ **PARAMREFTRAN**	String	Indica si se le pasa el id de la transicion a la acción. → "**S**" – Sí → "**N**" – No
→ **PARAMREFDOCPER**	String	Indica si se le pasa el id del documento permitido a la acción. → "**S**" – Sí → "**N**" – No
→ **PARAMREFEXPXFAS**	String	Indica si se le pasa el id del paso en la evolución actual a la acción. → "**S**" – Sí → "**N**" – No
→ **PARAMREFDEFPROC**	String	Indica si se le pasa el id de la definición del procedimiento a la acción. → "**S**" – Sí → "**N**" – No
→ **PARAMFECHA**	String	Indica si se le paso la fecha de tramitación/generación/incorporación

		indicada a la acción. → "**S**" – Sí → "**N**" – No
→ **PARAMUSUARIO**	String	Indica si se le pasa el usuario a la acción. → "**S**" – Sí → "**N**" – No
→ **PARAMREFFASE**	String	Indica si se le pasa el identificador de la fase a la acción. → "**S**" – Sí → "**N**" – No
→ **PARAMREFTIPODOC**	String	Indica si se le pasa el identificador del tipo de documento a la acción. → "**S**" – Sí → "**N**" – No
→ **IMPLEMENTACION**	String	Indica el tipo de implementación asociada a la acción. → "**F**" – Función en el servidor → "**J**" – Java
→ **PAQUETE**	String	Indica el paquete que contiene la función a ejecutar si se trata de una función en el servidor, o el nombre del paquete y clase al que pertenece la función java a ejecutar.
→ **NOMBFUNCION**	String	Indica el nombre de la función a ejecutar.
→ **STMA**	TrSistema	Sistema de la acción.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFACCION**	NUMÉRICO	Identificador de la acción
→ **CAMPO_NOMBRE**	CADENA(50)	Nombre de la acción
→ **CAMPO_DESCRIPCION**	CADENA(250)	Descripción de la acción
→ **CAMPO_PARAMREFEXP**	CADENA(1)	Indica si se le pasa el id del expediente a la acción. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMREFTRAN**	CADENA(1)	Indica si se le pasa el id de la transicion a la acción. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMREFDOCPER**	CADENA(1)	Indica si se le pasa el id del documento permitido a la acción. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMREFEXPXFAS**	CADENA(1)	Indica si se le pasa el id del paso en la evolución actual a la acción. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMREFDEFPROC**	CADENA(1)	Indica si se le pasa el id de la definición del procedimiento a la acción. → "**S**" – Sí → "**N**" – No

→ **CAMPO_PARAMFECHA**	CADENA(1)	Indica si se le paso la fecha de tramitación/generación/incorporación indicada a la acción. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMUSUARIO**	CADENA(1)	Indica si se le pasa el usuario a la acción. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMREFFASE**	CADENA(1)	Indica si se le pasa el identificador de la fase a la acción. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMREFTIPODOC**	CADENA(1)	Indica si se le pasa el identificador del tipo de documento a la acción. → "**S**" – Sí → "**N**" – No
→ **CAMPO_IMPLEMENTACION**	CADENA(1)	Indica el tipo de implementación asociada a la acción. → "**F**" – Función en el servidor → "**J**" – Java
→ **CAMPO_PAQUETE**	CADENA(100)	Indica el paquete que contiene la función a ejecutar si se trata de una función en el servidor, o el nombre del paquete y clase al que pertenece la función java a ejecutar.
→ **CAMPO_NOMBFUNCION**	CADENA(50)	Indica el nombre de la función a ejecutar
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema

Métodos para aplicación de los filtros → obtenerAccion

TrAccionBloquePermitido

trewa.bd.trapi.tpo.TrAccionBloquePermitido

Representa la información de una acción asociada a una tarea en fase.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **ACCION**	TrAccion	Acción
→ **BLOQUEPER**	TrBloquePermitido	Bloque permitido.
→ **DEFPROC**	TrDefProcedimiento	Definición del procedimiento.
→ **MENSAJEOK**	String	Mensaje a mostrar cuando se cumple la acción.
→ **MENSAJENOOK**	String	Mensaje a mostrar cuando la acción no se cumple.
→ **VALIDA**	String	Indica si la acción asociada a la tarea es válida. → "**S**" – Sí

		→ "**N**" – No
→ **MOSTRARMSJ**	String	Mensaje a mostrar. → "**O**" – Mensaje OK → "**N**" – Mensaje No OK → "**A**" – Ambos → "**I**" – Ninguno
→ **COMPROBAR**	String	Indica si se realiza la acción. → "**I**" – La acción se produce al Incorporar → "**V**" – La acción se produce al Visualizar

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFACCION**	NUMÉRICO	Identificador de la acción
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento
→ **CAMPO_REFBLOQUEPER**	NUMÉRICO	Identificador del bloque permitido o tarea en fase.
→ **CAMPO_REFFASE**	NUMÉRICO	Identificador de la fase
→ **CAMPO_REFBLOQUEINI**	NUMÉRICO	Identificador del bloque inicial o tarea llamante
→ **CAMPO_REFBLOQUEFIN**	NUMÉRICO	Identificador del bloque final o tarea
→ **CAMPO_VALIDA**	CADENA(1)	Indica si la acción asociada a la tarea es válida. → "**S**" – Sí → "**N**" – No
→ **CAMPO_COMPROBAR**	CADENA(1)	Indica si se realiza la acción. → "**I**" – La acción se produce al Incorporar → "**V**" – La acción se produce al Visualizar
→ **CAMPO_MOSTRARMSJ**	CADENA(1)	Mensaje a mostrar. → "**O**" – Mensaje OK → "**N**" – Mensaje No OK → "**A**" – Ambos → "**I**" – Ninguno
→ **CAMPO_MENSAJEOK**	CADENA(200)	Mensaje a mostrar cuando la acción se cumple.
→ **CAMPO_MENSAJENOOK**	CADENA(200)	Mensaje a mostrar cuando la acción no se cumple.

Métodos para aplicación de los filtros → obtenerAccionBloquePermitido

TrAccionDocumentoPermitido

trewa.bd.trapi.tpo.TrAccionDocumentoPermitido

Clase que representa la información de una acción asociada a un documento permitido.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **ACCION**	TrAccion	Acción
→ **TIPODOC**	TrTipoDocumento	Tipo de documento.
→ **DEFPROC**	TrDefProcedimiento	Definición del procedimiento.
FASE	TrFase	Fase.
→ **MENSAJEOK**	String	Mensaje a mostrar cuando se cumple la acción.
→ **MENSAJENOOK**	String	Mensaje a mostrar cuando la acción no se cumple.
→ **VALIDA**	String	Indica si la acción asociada a la tarea es válida. → "**S**" – Sí → "**N**" – No
→ **MOSTRARMSJ**	String	Mensaje a mostrar. → "**O**" – Mensaje OK → "**N**" – Mensaje No OK → "**A**" – Ambos → "**I**" – Ninguno
→ **COMPROBAR**	String	Indica si se realiza la acción. → "**G**" – La acción se produce al Generar el documento → "**I**" – La acción se produce al Incorporar el documento → "**T**" – La acción se produce al Generar e Incorporar el documento → "**V**" – La acción se produce al Visualizar el documento

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFACCION**	NUMÉRICO	Identificador de la acción
→ **CAMPO_REFTIPODOC**	NUMÉRICO	Identificador del tipo de documento
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento
→ **CAMPO_REFFASE**	NUMÉRICO	Identificador de la fase
→ **CAMPO_MENSAJEOK**	CADENA(200)	Mensaje a mostrar cuando se cumple la acción.
→ **CAMPO_MENSAJENOOK**	CADENA(200)	Mensaje a mostrar cuando no se cumple la acción.
→ **CAMPO_VALIDA**	CADENA(1)	Indica si la acción asociada a la tarea es válida. → "**S**" – Sí → "**N**" – No
→ **CAMPO_MOSTRARMSJ**	CADENA(1)	Mensaje a mostrar. → "**O**" – Mensaje OK → "**N**" – Mensaje No OK → "**A**" – Ambos → "**I**" – Ninguno
→ **CAMPO_COMPROBAR**	CADENA(1)	Indica si se realiza la acción.

→ "**G**" – La acción se produce al Generar el documento
→ "**I**" – La acción se produce al Incorporar el documento
→ "**T**" – La acción se produce al Generar e Incorporar el documento
→ "**V**" – La acción se produce al Visualizar el documento

Métodos para aplicación de los filtros → `obtenerAccionDocumentoPermitido`

TrAccionTransicion

trewa.bd.trapi.tpo.TrAccionTransicion

Clase que representa la información de una acción asociada a una transición.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **ACCION**	`TrAccion`	Acción
→ **TRANSICION**	`TrTransicion`	Transición.
→ **DEFPROC**	`TrDefProcedimiento`	Definición del procedimiento.
→ **MENSAJEOK**	`String`	Mensaje a mostrar cuando se cumple la acción.
→ **MENSAJENOOK**	`String`	Mensaje a mostrar cuando la acción no se cumple.
→ **VALIDA**	`String`	Indica si la acción asociada a la transición es válida. → "**S**" – Sí → "**N**" – No
→ **MOSTRARMSJ**	`String`	Mensaje a mostrar. → "**O**" – Mensaje OK → "**N**" – Mensaje No OK → "**A**" – Ambos → "**I**" – Ninguno
→ **COMPROBAR**	`String`	Indica si se realiza la acción. → "**T**" – La acción se produce al Tramitar → "**D**" – La acción se produce al Deshacer → "**A**" – La acción se produce al Tramitar y Deshacer → "**V**" – La acción se produce al Visualizar

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFACCION**	`NUMÉRICO`	Identificador de la acción
→ **CAMPO_REFTRANSICION**	`NUMÉRICO`	Identificador de la transición
→ **CAMPO_REFDEFPROC**	`NUMÉRICO`	Identificador de la definición del procedimiento.

→ **CAMPO_VALIDA**	CADENA(1)	Indica si la acción asociada a la transición es válida. → "**S**" – Sí → "**N**" – No
→ **CAMPO_MOSTRARMSJ**	CADENA(1)	Mensaje a mostrar. → "**O**" – Mensaje OK → "**N**" – Mensaje No OK → "**A**" – Ambos → "**I**" – Ninguno
→ **CAMPO_COMPROBAR**	CADENA(1)	Indica si se realiza la acción. → "**T**" – La acción se produce al Tramitar → "**D**" – La acción se produce al Deshacer → "**A**" – La acción se produce al Tramitar y Deshacer → "**V**" – La acción se produce al Visualizar
→ **CAMPO_MENSAJEOK**	CADENA(200)	Mensaje a mostrar cuando se cumple la acción.
→ **CAMPO_MENSAJENOOK**	CADENA(200)	Mensaje a mostrar cuando la acción no se cumple.

Métodos para aplicación de los filtros → obtenerAccionTransicion

TrAmbitoLey

trewa.bd.trapi.tpo.TrAmbitoLey

Clase que representa los diferentes ámbitos de la ley para las normativas.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFAMBITOLEY**	TpoPK	Identificador del ámbito de ley.
→ **ABREVIATURA**	String	Abreviatura del ámbito de ley.
→ **DESCRIPCION**	String	Descripción del ámbito de ley.
→ **OBSOLETO**	String	Indica si el ámbito de ley está obsoleto. → "**S**" – Sí → "**N**" – No
→ **CODWANDA**	String	Identificador del componente en w@ndA.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFAMBITOLEY**	NUMÉRICO	Identificador del ámbito de ley.
→ **CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura del ámbito de ley.
→ **CAMPO_DESCRIPCION**	CADENA(100)	Descripción del ámbito de ley.

→ **CAMPO_OBSOLETO**	CADENA(1)	Indica si el ámbito de ley está obsoleto. → "**S**" – Sí → "**N**" – No
→ **CAMPO_CODWANDA**	CADENA(50)	Identificador del componente en w@ndA.

Métodos para aplicación de los filtros → obtenerAmbitoLey

TrAviso

trewa.bd.trapi.tpo.TrAviso

Clase que representa la información de un aviso.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFAVISO**	TpoPK	Identificador del aviso.
→ **NOMBRE**	String	Nombre con el que se denomina al aviso.
→ **DESCRIPCION**	String	Texto que describe el aviso.
→ **STMA**	TrSistema	Sistema.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFAVISO**	NUMÉRICO	Identificador del aviso.
→ **CAMPO_NOMBRE**	CADENA(50)	Nombre con el que se denomina al aviso.
→ **CAMPO_DESCRIPCION**	CADENA(250)	Texto que describe el aviso.
→ **CAMPO_REFSTMA**	NUMÉRICO	Identifocador del sistema

Métodos para aplicación de los filtros → obtenerAviso

TrAvisoBloquePermitido

trewa.bd.trapi.tpo.TrAvisoBloquePermitido

Clase que representa la información de un aviso asociado a una tarea en fase.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **AVISO**	TrAviso	Aviso.
→ **BLOQUEPER**	TrBloquePermitido	Bloque permitido.
→ **DEFPROC**	TrDefProcedimiento	Definición del procedimiento.
→ **MENSAJEOK**	String	Mensaje a mostrar.
→ **VALIDA**	String	Indica si el aviso asociado a la tarea es válido. → "**S**" – Sí → "**N**" – No
→ **COMPROBAR**	String	Indica si se realiza el aviso. → "**I**" – El aviso se produce al Incorporar → "**V**" – El aviso se produce al Visualizar

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFAVISO**	NUMÉRICO	Identificador del aviso
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento.
→ **CAMPO_REFBLOQUEPER**	NUMÉRICO	Identificador del bloque permitido o tarea en fase
→ **CAMPO_REFFASE**	NUMÉRICO	Identificador de la fase
→ **CAMPO_REFBLOQUEINI**	NUMÉRICO	Identificador del bloque inicial o tarea llamante
→ **CAMPO_REFBLOQUEFIN**	NUMÉRICO	Identificador del bloque fin o tarea.
→ **CAMPO_VALIDA**	CADENA(1)	Indica si el aviso asociado a la tarea es válido. → "**S**" – Sí → "**N**" – No
→ **CAMPO_COMPROBAR**	CADENA(1)	Indica si se realiza el aviso. → "**I**" – El aviso se produce al Incorporar → "**V**" – El aviso se produce al Visualizar
→ **CAMPO_MENSAJEOK**	CADENA(200)	Mensaje a mostrar.

Métodos para aplicación de los filtros → obtenerAvisoBloquePermitido

TrAvisoDocumentoPermitido

trewa.bd.trapi.tpo.TrAvisoDocumentoPermitido

Clase que representa la información de un aviso asociado a un documento permitido.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción

→ **AVISO**	`TrAviso`	Aviso.
→ **TIPODOC**	`TrTipoDocumento`	Tipo de documento.
→ **DEFPROC**	`TrDefProcedimiento`	Definición del procedimiento.
→ **FASE**	`TrFase`	Fase.
→ **MENSAJEOK**	`String`	Mensaje a mostrar para el aviso.
→ **VALIDA**	`String`	Indica si el aviso asociado a la tarea es válido. → "**S**" – Sí → "**N**" – No
→ **COMPROBAR**	`String`	Indica si se realiza el aviso. → "**G**" – El aviso se produce al Generar el documento → "**I**" – El aviso se produce al Incorporar el documento → "**T**" – El aviso se produce al Generar e Incorporar el documento → "**V**" - El aviso se produce al Visualizar el documento

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFAVISO**	`NUMÉRICO`	Identificador del aviso.
→ **CAMPO_REFTIPODOC**	`NUMÉRICO`	Identificador del tipo de documento.
→ **CAMPO_REFDEFPROC**	`NUMÉRICO`	Identificador de la definición del procedimiento.
→ **CAMPO_REFFASE**	`NUMÉRICO`	Identificador de la fase.
→ **CAMPO_MENSAJEOK**	`CADENA(200)`	Mensaje a mostrar para el aviso.
→ **CAMPO_VALIDA**	`CADENA(1)`	Indica si el aviso asociado a la tarea es válido. → "**S**" – Sí → "**N**" – No
→ **CAMPO_COMPROBAR**	`CADENA(1)`	Indica si se realiza el aviso. → "**G**" – El aviso se produce al Generar el documento → "**I**" – El aviso se produce al Incorporar el documento → "**T**" – El aviso se produce al Generar e Incorporar el documento → "**V**" - El aviso se produce al Visualizar el documento

Métodos para aplicación de los filtros → `obtenerAvisoDocumentoPermitido`

TrAvisoTransicion

trewa.bd.trapi.tpo.TrAvisoTransicion

Clase que representa la información de un aviso asociado a una transición.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **AVISO**	TrAviso	Aviso.
→ **TRANSICION**	TrTransicion	Transición.
→ **DEFPROC**	TrDefProcedimiento	Definición del procedimiento.
→ **MENSAJEOK**	String	Mensaje a mostrar para el aviso.
→ **VALIDA**	String	Indica si el aviso asociado a la transición es válido. → "**S**" – Sí → "**N**" – No
→ **COMPROBAR**	String	Indica si se realiza el aviso. → "**T**" – El aviso se produce al Tramitar → "**D**" – El aviso se produce al Deshacer → "**A**" – El aviso se produce al Tramitar y Deshacer → "**V**" – El aviso se produce al Visualizar

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFAVISO**	NUMÉRICO	Identificador del aviso
→ **CAMPO_REFTRANSICION**	NUMÉRICO	Identificador de la transición
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento
→ **CAMPO_MENSAJEOK**	CADENA(200)	Mensaje a mostrar para el aviso.
→ **CAMPO_VALIDA**	CADENA(1)	Indica si el aviso asociado a la transición es válido. → "**S**" – Sí → "**N**" – No
→ **CAMPO_COMPROBAR**	CADENA(1)	Indica si se realiza el aviso. → "**T**" – El aviso se produce al Tramitar → "**D**" – El aviso se produce al Deshacer → "**A**" – El aviso se produce al Tramitar y Deshacer → "**V**" – El aviso se produce al Visualizar

Métodos para aplicación de los filtros → obtenerAvisoTransicion

TrBloque

trewa.bd.trapi.tpo.TrBloque

Clase que representa la información referente a un bloque.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFBLOQUE**	TpoPK	Identificador del bloque.
→ **NOMBRE**	String	Nombre del bloque.
→ **DESCRIPCION**	String	Descripción del bloque.
→ **TIPO**	String	Tipo de bloque →"**W**" - Web →"**R**" – Informe →"**F**" – Pantalla →"**O**" – Otro
→ **INFORMAR**	String	Indica si se informa al bus de esta tarea. → "**S**" – Sí → "**N**" – No
→ **STMA**	TrSistema	Sistema.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFBLOQUE**	NUMÉRICO	Identificador del bloque
→ **CAMPO_NOMBRE**	CADENA(50)	Nombre del bloque
→ **CAMPO_DESCRIPCION**	CADENA(250)	Descripción del bloque
→ **CAMPO_TIPO**	CADENA(1)	Tipo de bloque →"**W**" - Web →"**R**" – Informe →"**F**" – Pantalla →"**O**" – Otro
→ **CAMPO_INFORMAR**	CADENA(1)	Indica si se informa al bus de esta tarea. → "**S**" – Sí → "**N**" – No
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema

Métodos para aplicación de los filtros → obtenerBloque

TrBloquePermitido

trewa.bd.trapi.tpo.TrBloquePermitido

Clase que representa la información referente a las tareas de tipo "MANIPULAR_DATOS" definidas en una fase de un procedimiento.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFBLOQUEPER**	TpoPK	Identificador del bloque permitido o tarea en fase.
→ **ETIQUETA**	String	Etiqueta de la tarea en fase
→ **DESCRIPCION**	String	Descripción de la tarea en fase
→ **VALIDO**	String	Indica si es una tarea válida → "**S**" – Sí → "**N**" – No
→ **ORDEN**	Integer	Indica el orden de la tarea en la fase
→ **FASE**	TrFase	Fase
→ **BLOQUEINI**	TrBloque	Bloque inicial o tarea llamante
→ **BLOQUEFIN**	TrBloque	Bloque final o tarea
→ **OBLIGATORIO**	String	Indica si es una tarea obligatoria → "**S**" – Sí → "**N**" – No
→ **INFORMAR**	String	Indica si se informa al bus de la tarea → "**S**" – Sí → "**N**" – No
→ **ETIQLARGA**	String	Etiqueta larga para la tarea

TrBloquePermitidoDefProc

trewa.bd.trapi.tpo.TrBloquePermitidoDefProc

Clase que representa la información referente a las tareas de tipo "MANIPULAR_DATOS" asociadas a una definición de procedimiento.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **BLOQUEPER**	TrBloquePermitido	Bloque permitido.
→ **REFDEFPROC**	TpoPK	Identificador de la definición del procedimiento.
→ **UNIDAD**	String	Unidad de medida para el plazo máximo de realización. → "**D**" – Días → "**M**" – Meses → "**A**" – Años
→ **NUMUNIDADES**	Integer	Indica el nº de unidades del plazo máximo de realización: nº de días, nº de meses, nº de años.
→ **DESCFECHALIM**	String	Descripción del significado de la fecha límite de realización de la tarea.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento
→ **CAMPO_UNIDAD**	CADENA(1)	Unidad de medida para el plazo máximo de realización. → "**D**" – Días → "**M**" – Meses → "**A**" – Años
→ **CAMPO_NUMUNIDADES**	NUMÉRICO	Indica el nº de unidades del plazo máximo de realización: nº de días, nº de meses, nº de años.
→ **CAMPO_DESCFECHALIM**	CADENA(100)	Descripción del significado de la fecha límite de realización de la tarea.

Métodos para aplicación de los filtros → obtenerBloquePermitidoDefProc

TrBloquePermitidoPerfil

trewa.bd.trapi.tpo.TrBloquePermitidoPerfil

Clase que representa la información referente a un bloque asociado a un perfil de usuario.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **PERFILUSU**	TrPerfilUsuario	Perfil de usuario.
→ **REFDEFPROC**	TpoPK	Identificador de la definición del procedimiento.
→ **REFFASE**	TpoPK	Identificador de la fase.
→ **REFBLOQUEINI**	TpoPK	Identificador del bloque inicial.
→ **REFBLOQUEFIN**	TpoPK	Identificador del bloque final.
→ **REFBLOQUEPER**	TpoPK	Identificador del bloque permitido.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFPERFILUSU**	NUMÉRICO	Identificador del perfil de usuario.
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento.
→ **CAMPO_REFFASE**	NUMÉRICO	Identificador dela fase.
→ **CAMPO_REFBLOQUEINI**	NUMÉRICO	Identificador del bloque inicial o tarea llamante.

→ **CAMPO_REFBLOQUEFIN**	NUMÉRICO	Identificador del bloque final o tarea.
→ **CAMPO_REFBLOQUEPER**	NUMÉRICO	Identificador del bloque permitido o tarea en fase.

Métodos para aplicación de los filtros → obtenerBloquePermitidoPerfil

TrCaducidad

trewa.bd.trapi.tpo.TrCaducidad

Clase que representa la información referente a caducidades.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFCADUCIDAD**	TpoPK	Identificador de la caducidad.
→ **ABREVIATURA**	String	Abreviatura de la caducidad.
→ **DESCRIPCION**	String	Descripción de la caducidad.
→ **UNIDAD**	String	Unidad de medida para el plazo máximo de realización. → "**D**" – Días → "**M**" – Meses → "**A**" – Años
→ **NUMUNIDADES**	Integer	Indica el nº de unidades del plazo máximo de realización: nº de días, nº de meses, nº de años.
→ **TIPO**	String	Indica el tipo de caducidad. → "**P**" – Plazo de ejecición → "**O**" – Otro
→ **VIGENTE**	String	Indica si la caducidad está vigente. → "**S**" – Sí → "**N**" – No
→ **TRANSICION**	TrTransicion	Transición.
→ **DEFPROC**	TrDefProcedimiento	Definición del procedimiento.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFCADUCIDAD**	NUMÉRICO	Identificador de la caducidad
→ **CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura de la caducidad
→ **CAMPO_DESCRIPCION**	CADENA(50)	Descripción de la caducidad
→ **CAMPO_UNIDAD**	CADENA(1)	Unidad de medida para el plazo máximo de realización. → "**D**" – Días → "**M**" – Meses

		→ "**A**" – Años
→ **CAMPO_UNIDADES**	NUMÉRICO	Indica el nº de unidades del plazo máximo de realización: nº de días, nº de meses, nº de años.
→ **CAMPO_TIPO**	CADENA(1)	Indica el tipo de caducidad. → "**P**" – Plazo de ejecición → "**O**" – Otro
→ **CAMPO_VIGENTE**	CADENA(1)	Indica si la caducidad está vigente. → "**S**" – Sí → "**N**" – No
→ **CAMPO_REFTRANSICION**	NUMÉRICO	Identificador de la transición
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento.

Métodos para aplicación de los filtros → obtenerCaducidad

TrComponente

trewa.bd.trapi.tpo.TrComponente

Clase que representa los componentes que pueden intervenir en el sistema w@ndA.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFCOMPONENTE**	TpoPK	Identificador global del componente.
→ **NOMBRE**	String	Nombre del componente.
→ **DESCRIPCION**	String	Descripción del componente.
→ **DIRECCIONIP**	String	Dirección IP del componente.
→ **TIPOCOMPONENTE**	TrTipoComponente	Tipo de componente.
→ **USUARIO**	String	Usuario para acceder al componente.
→ **PASSWORD**	String	Password del usuario para acceder al componente.
→ **CODWANDA**	Long	Identificador del componente en w@ndA.
→ **REFORGANISMO**	TpoPK	Identificador del organismo donde se encuentra el componente

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFCOMPONENTE**	NUMÉRICO	Identificador global del componente.
→ **CAMPO_NOMBRE**	CADENA(50)	Nombre del componente.
→ **CAMPO_DESCRIPCION**	CADENA(250)	Descripción del componente.
→ **CAMPO_DIRECCIONIP**	CADENA(25)	Dirección IP del componente.

→ **CAMPO_REFTIPOCOMP**	NUMÉRICO	Identificador del tipo de componente
→ **CAMPO_USUARIO**	CADENA(30)	Usuario para acceder al componente.
→ **CAMPO_PASSWORD**	CADENA(30)	Password del usuario para acceder al componente.
→ **CAMPO_CODWANDA**	NUMÉRICO	Identificador del componente en w@ndA.
→ **CAMPO_REFORGANISMO**	NUMÉRICO	Identificador del organismo donde se encuentra el componente

Métodos para aplicación de los filtros → obtenerComponente

TrCondicionAccion

trewa.bd.trapi.tpo.TrCondicionAccion

Clase que representa la información referente a una condición o acción

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFCONDACC**	TpoPK	Identificador de la condición o acción
→ **NOMBRE**	String	Nombre con el que se denomina la condición o acción
→ **DESCRIPCION**	String	Descripción de la condición o acción
→ **PARAMREFEXP**	String	Indica si se le pasa el id del expediente a la condición o acción. → "**S**" – Sí → "**N**" – No
→ **PARAMREFTRAN**	String	Indica si se le pasa el id de la transicio a la condición o acción. → "**S**" – Sí → "**N**" – No
→ **PARAMREFDOCPER**	String	Indica si se le pasa el id del paso en la evolución actual a la condición o acción. → "**S**" – Sí → "**N**" – No
→ **PARAMREFEXPXFAS**	String	Indica si se le pasa el id del paso en la evolución actual a la condición o acción. → "**S**" – Sí → "**N**" – No
→ **PARAMREFDEFPROC**	String	Indica si se le pasa el id de la definición del procedimiento a la condición o acción. → "**S**" – Sí → "**N**" – No
→ **PARAMFECHA**	String	Indica si se le paso la fecha de tramitación/generación/incorporación indicada a la condición o acción.

		→ "**S**" – Sí → "**N**" – No
→ **PARAMUSUARIO**	String	Indica si se le pasa el usuario a la condición o acción. → "**S**" – Sí → "**N**" – No
→ **PARAMREFFASE**	String	Indica si se le pasa el identificador de la fase a la condición o acción. → "**S**" – Sí → "**N**" – No
→ **PARAMREFTIPODOC**	String	Indica si se le pasa el identificador del tipo de documento a la condición o acción. → "**S**" – Sí → "**N**" – No
→ **IMPLEMENTACION**	String	Indica el tipo de implementación asociada a la condición o acción. → "**F**" – Función en el servidos → "**J**" – Java
→ **PAQUETE**	String	Indica el paquete que contiene la función a ejecutar si se trata de una función en el servidor, o el nombre del paquete y clase al que pertenece la función java a ejecutar.
→ **NOMBFUNCION**	String	Indica el nombre de la función a ejecutar.
→ **COMPLEJA**	String	Indica si la condición es compleja. → "**S**" – Sí → "**N**" – No
→ **EXPRESION**	String	Contiene la expresion a evaluar en el caso de expresiones complejas.
→ **STMA**	TrSistema	Sistema de la condición o acción.
→ **TIPO**	String	Indica el tipo → "**C**" – Condición → "**A**" – Acción → "**W**" – Aviso

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFCONDACC**	NUMÉRICO	Identificador de la condición o acción.
→ **CAMPO_NOMBRE**	CADENA(50)	Nombre con el que se denomina la condición o acción.
→ **CAMPO_DESCRIPCION**	CADENA(250)	Descripción de la condición o acción.
→ **CAMPO_PARAMREFEXP**	CADENA(1)	Indica si se le pasa el id del expediente a la condición o acción. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMREFTRAN**	CADENA(1)	Indica si se le pasa el id de la transicio a la condición o acción. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMREFDOCPER**	CADENA(1)	Indica si se le pasa el id del documento permitido a la condición o

		acción. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMREFEXPXFAS**	CADENA(1)	Indica si se le pasa el id del paso en la evolución actual a la condición o acción. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMREFDEFPROC**	CADENA(1)	Indica si se le pasa el id de la definición del procedimiento a la condición o acción. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMFECHA**	CADENA(1)	Indica si se le paso la fecha de tramitación/generación/incorporación indicada a la condición o acción. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMUSUARIO**	CADENA(1)	Indica si se le pasa el usuario a la condición o acción. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMREFFASE**	CADENA(1)	Indica si se le pasa el identificador de la fase a la condición o acción. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMREFTIPODOC**	CADENA(1)	Indica si se le pasa el identificador del tipo de documento a la condición o acción. → "**S**" – Sí → "**N**" – No
→ **CAMPO_IMPLEMENTACION**	CADENA(1)	Indica el tipo de implementación asociada a la condición o acción. → "**F**" – Función en el servidos → "**J**" – Java
→ **CAMPO_PAQUETE**	CADENA(100)	Indica el paquete que contiene la función a ejecutar si se trata de una función en el servidor, o el nombre del paquete y clase al que pertenece la función java a ejecutar.
→ **CAMPO_NOMBFUNCION**	CADENA(50)	Indica el nombre de la función a ejecutar.
→ **CAMPO_COMPLEJA**	CADENA(1)	Indica si la condición es compleja. → "**S**" – Sí → "**N**" – No
→ **CAMPO_EXPRESION**	CADENA(4000)	Contiene la expresion a evaluar en el caso de expresiones complejas.
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema.
→ **CAMPO_TIPO**	CADENA(1)	Indica el tipo → "**C**" – Condición → "**A**" – Acción → "**W**" – Aviso

Métodos para aplicación de los filtros → obtenerCondAccAvi

TrCondAccionBloquePermitido

trewa.bd.trapi.tpo. TrCondAccionBloquePermitido

Clase que representa la información de una condición o acción asociada a una tarea en fase.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **CONDACC**	`TrCondicionAccion`	Condición o acción.
→ **BLOQUEPER**	`TrBloquePermitido`	Bloque permitido.
→ **DEFPROC**	`TrDefProcedimiento`	Definición del procedimiento.
→ **MENSAJEOK**	`String`	Mensaje a mostrar cuando se cumple la condición o acción.
→ **MENSAJENOOK**	`String`	Mensaje a mostrar cuando la condición o acción no se cumple.
→ **OBLIGATORIA**	`String`	Indica si la condición o acción se debe cumplir obligatoriamente o no. → "**S**" – Sí → "**N**" – No
→ **VALIDA**	`String`	Indica si la condición o acción asociada a la tarea es válida. → "**S**" – Sí → "**N**" – No
→ **MOSTRARMSJ**	`String`	Mensaje a mostrar. → "**O**" – Mensaje OK → "**N**" – Mensaje No OK → "**A**" – Ambos → "**I**" – Ninguno
→ **COMPROBAR**	`String`	Indica si se realiza la condición o acción. → "**I**" – La condición se produce al Incorporar → "**V**" – La condición se produce al Visualizar

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFCONDADD**	`NUMÉRICO`	Identificador de la condición o acción
→ **CAMPO_REFDEFPROC**	`NUMÉRICO`	Identificador de la definición del procedimiento
→ **CAMPO_REFBLOQUEPER**	`NUMÉRICO`	Identificador del bloque permitido o tarea en fase
→ **CAMPO_REFFASE**	`NUMÉRICO`	Identificador de la fase.

→ **CAMPO_REFBLOQUEINI**	NUMÉRICO	Identificador del bloque inicial o tarea llamante
→ **CAMPO_REFBLOQUEFIN**	NUMÉRICO	Identificador del bloque final o tarea
→ **CAMPO_OBLIGATORIA**	CADENA(1)	Indica si la condición o acción se debe cumplir obligatoriamente o no. → "**S**" – Sí → "**N**" – No
→ **CAMPO_VALIDA**	CADENA(1)	Indica si la condición o acción asociada a la tarea es válida. → "**S**" – Sí → "**N**" – No
→ **CAMPO_COMPROBAR**	CADENA(1)	Indica si se realiza la condición o acción. → "**I**" – La condición o acción se produce al Incorporar → "**V**" – La condición o acción se produce al Visualizar
→ **CAMPO_MOSTRARMSJ**	CADENA(1)	Mensaje a mostrar. → "**O**" – Mensaje OK → "**N**" – Mensaje No OK → "**A**" – Ambos → "**I**" – Ninguno
→ **CAMPO_MENSAJEOK**	CADENA(200)	Mensaje a mostrar cuando se cumple la condición o acción.
→ **CAMPO_MENSAJENOOK**	CADENA(200)	Mensaje a mostrar cuando no se cumple la condición o acción.

Métodos para aplicación de los filtros → obtenerCondAccAviBloquePermitido

TrCondAccionDocumentoPermitido

trewa.bd.trapi.tpo. TrCondAccionDocumentoPermitido

Clase que representa la información de una condición o acción asociada a un documento permitido.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **CONDACC**	TrCondicionAccion	Condición o acción.
→ **TIPODOC**	TrTipoDocumento	Tipo de documento.
→ **DEFPROC**	TrDefProcedimiento	Definición del procedimiento.
FASE	TrFase	Fase.
→ **MENSAJEOK**	String	Mensaje a mostrar cuando se cumple la condición o acción.
→ **MENSAJENOOK**	String	Mensaje a mostrar cuando la condición o acción no se cumple.
→ **OBLIGATORIA**	String	Indica si la condición o acción se debe cumplir obligatoriamente o no. → "**S**" – Sí → "**N**" – No
→ **VALIDA**	String	Indica si la condición o acción asociada a la tarea es válida.

		→ "**S**" – Sí → "**N**" – No
→ **MOSTRARMSJ**	String	Mensaje a mostrar. → "**O**" – Mensaje OK → "**N**" – Mensaje No OK → "**A**" – Ambos → "**I**" – Ninguno
→ **COMPROBAR**	String	Indica si se realiza la condición o acción. → "**G**" – La condición o acción se produce al Generar el documento → "**I**" – La condición o acción se produce al Incorporar el documento → "**T**" – La condición o acción se produce al Generar e Incorporar el documento → "**V**" – La condición o acción se produce al Visualizar el documento

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFCONDACC**	NUMÉRICO	Identificador de la condición o acción.
→ **CAMPO_REFTIPODOC**	NUMÉRICO	Identificador del tipo de documento.
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento.
→ **CAMPO_REFFASE**	NUMÉRICO	Identificador de la fase.
→ **CAMPO_MENSAJEOK**	CADENA(200)	Mensaje a mostrar cuando se cumple la condición o acción.
→ **CAMPO_MENSAJENOOK**	CADENA(200)	Mensaje a mostrar cuando la condición o acción no se cumple.
→ **CAMPO_OBLIGATORIA**	CADENA(1)	Indica si la condición o acción se debe cumplir obligatoriamente o no. → "**S**" – Sí → "**N**" – No
→ **CAMPO_VALIDA**	CADENA(1)	Indica si la condición o acción asociada a la tarea es válida. → "**S**" – Sí → "**N**" – No
→ **CAMPO_MOSTRARMSJ**	CADENA(1)	Mensaje a mostrar. → "**O**" – Mensaje OK → "**N**" – Mensaje No OK → "**A**" – Ambos → "**I**" – Ninguno
→ **CAMPO_COMPROBAR**	CADENA(1)	Indica si se realiza la condición o acción. → "**G**" – La condición o acción se produce al Generar el documento → "**I**" – La condición o acción se produce al Incorporar el documento → "**T**" – La condición o acción se produce al Generar e Incorporar el documento

→ "**V**" – La condición o acción se produce al Visualizar el documento

Métodos para aplicación de los filtros → obtenerCondAccAviDocumentoPermitido

TrCondAccionTransicion

trewa.bd.trapi.tpo. TrCondAccionTransicion

Clase que representa la información de una condición o acción de una transición.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **CONDACC**	TrCondicionAccion	Condición o acción.
→ **TRANSICION**	TrTransicion	Transición.
→ **DEFPROC**	TrDefProcedimiento	Definición del procedimiento.
→ **MENSAJEOK**	String	Mensaje a mostrar cuando se cumple la condición o acción.
→ **MENSAJENOOK**	String	Mensaje a mostrar cuando la condición o acción no se cumple.
→ **OBLIGATORIA**	String	Indica si la condición o acción se debe cumplir obligatoriamente o no. → "**S**" – Sí → "**N**" – No
→ **VALIDA**	String	Indica si la condición o acción asociada a la tarea es válida. → "**S**" – Sí → "**N**" – No
→ **MOSTRARMSJ**	String	Mensaje a mostrar. → "**O**" – Mensaje OK → "**N**" – Mensaje No OK → "**A**" – Ambos → "**I**" – Ninguno
→ **COMPROBAR**	String	Indica si se realiza la condición o acción. → "**T**" – La condición o acción se produce al Tramitar → "**D**" – La condición o acción se produce al Deshacer → "**A**" – La condición o acción se produce al Tramitar y Deshacer → "**V**" – La condición o acción se produce al Visualizar

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFCONDACC**	NUMÉRICO	Identificador de la condición o acción.
→ **CAMPO_REFTRANSICION**	NUMÉRICO	Identificador de la transición.
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento.
→ **CAMPO_OBLIGATORIA**	CADENA(1)	Indica si la condición o acción se debe cumplir obligatoriamente o no. → "**S**" – Sí → "**N**" – No
→ **CAMPO_VALIDA**	CADENA(1)	Indica si la condición o acción asociada a la tarea es válida. → "**S**" – Sí → "**N**" – No
→ **CAMPO_MOSTRARMSJ**	CADENA(1)	Mensaje a mostrar. → "**O**" – Mensaje OK → "**N**" – Mensaje No OK → "**A**" – Ambos → "**I**" – Ninguno
→ **CAMPO_COMPROBAR**	CADENA(1)	Indica si se realiza la condición o acción. → "**T**" – La condición o acción se produce al Tramitar → "**D**" – La condición o acción se produce al Deshacer → "**A**" – La condición o acción se produce al Tramitar y Deshacer → "**V**" – La condición o acción se produce al Visualizar
→ **CAMPO_MENSAJEOK**	CADENA(200)	Mensaje a mostrar cuando se cumple la condición.
→ **CAMPO_MENSAJENOOK**	CADENA(200)	Mensaje a mostrar cuando no se cumple la condición o acción.

Métodos para aplicación de los filtros → obtenerCondAccAviTransicion

TrCondicion

trewa.bd.trapi.tpo.TrCondicion

Clase que representa la información referente a una condición.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFCONDICION**	TpoPK	Identificador de la condición.
→ **NOMBRE**	String	Nombre con el que se denomina la condición.
→ **DESCRIPCION**	String	Descripción de la condición.
→ **PARAMREFEXP**	String	Indica si se le pasa el id del expediente a la condición. → "**S**" – Sí

		→ "**N**" – No
→ **PARAMREFTRAN**	`String`	Indica si se le pasa el id de la transicio a la condición. → "**S**" – Sí → "**N**" – No
→ **PARAMREFDOCPER**	`String`	Indica si se le pasa el id del paso en la evolución actual a la condición. → "**S**" – Sí → "**N**" – No
→ **PARAMREFEXPXFAS**	`String`	Indica si se le pasa el id del paso en la evolución actual a la condición. → "**S**" – Sí → "**N**" – No
→ **PARAMREFDEFPROC**	`String`	Indica si se le pasa el id de la definición del procedimiento a la condición. → "**S**" – Sí → "**N**" – No
→ **PARAMFECHA**	`String`	Indica si se le paso la fecha de tramitación/generación/incorporación indicada a la condición. → "**S**" – Sí → "**N**" – No
→ **PARAMUSUARIO**	`String`	Indica si se le pasa el usuario a la condición. → "**S**" – Sí → "**N**" – No
→ **PARAMREFFASE**	`String`	Indica si se le pasa el identificador de la fase a la condición. → "**S**" – Sí → "**N**" – No
→ **PARAMREFTIPODOC**	`String`	Indica si se le pasa el identificador del tipo de documento a la condición. → "**S**" – Sí → "**N**" – No
→ **IMPLEMENTACION**	`String`	Indica el tipo de implementación asociada a la condición. → "**F**" – Función en el servidos → "**J**" – Java
→ **PAQUETE**	`String`	Indica el paquete que contiene la función a ejecutar si se trata de una función en el servidor, o el nombre del paquete y clase al que pertenece la función java a ejecutar.
→ **NOMBFUNCION**	`String`	Indica el nombre de la función a ejecutar.
→ **COMPLEJA**	`String`	Indica si la condición es compleja. → "**S**" – Sí → "**N**" – No
→ **EXPRESION**	`String`	Contiene la expresion a evaluar en el caso de expresiones complejas.
→ **STMA**	`TrSistema`	Sistema de la condición.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFCONDICION**	NUMÉRICO	Identificador de la condición.
→ **CAMPO_NOMBRE**	CADENA(50)	Nombre con el que se denomina la condición.
→ **CAMPO_DESCRIPCION**	CADENA(250)	Descripción de la condición.
→ **CAMPO_PARAMREFEXP**	CADENA(1)	Indica si se le pasa el id del expediente a la condición. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMREFTRAN**	CADENA(1)	Indica si se le pasa el id de la transicio a la condición. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMREFDOCPER**	CADENA(1)	Indica si se le pasa el id del documento permitido a la condición. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMREFEXPXFAS**	CADENA(1)	Indica si se le pasa el id del paso en la evolución actual a la condición. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMREFDEFPROC**	CADENA(1)	Indica si se le pasa el id de la definición del procedimiento a la condición. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMFECHA**	CADENA(1)	Indica si se le paso la fecha de tramitación/generación/incorporación indicada a la condición. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMUSUARIO**	CADENA(1)	Indica si se le pasa el usuario a la condición. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMREFFASE**	CADENA(1)	Indica si se le pasa el identificador de la fase a la condición. → "**S**" – Sí → "**N**" – No
→ **CAMPO_PARAMREFTIPODOC**	CADENA(1)	Indica si se le pasa el identificador del tipo de documento a la condición. → "**S**" – Sí → "**N**" – No
→ **CAMPO_IMPLEMENTACION**	CADENA(1)	Indica el tipo de implementación asociada a la condición. → "**F**" – Función en el servidos → "**J**" – Java
→ **CAMPO_PAQUETE**	CADENA(100)	Indica el paquete que contiene la función a ejecutar si se trata de una función en el servidor, o el nombre del paquete y clase al que pertenece la función java a ejecutar.

→ **CAMPO_NOMBFUNCION**	CADENA(50)	Indica el nombre de la función a ejecutar.
→ **CAMPO_COMPLEJA**	CADENA(1)	Indica si la condición es compleja. → "**S**" – Sí → "**N**" – No
→ **CAMPO_EXPRESION**	CADENA(4000)	Contiene la expresion a evaluar en el caso de expresiones complejas.
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema.

Métodos para aplicación de los filtros → obtenerCondicion

TrCondicionBloquePermitido

trewa.bd.trapi.tpo.TrCondicionBloquePermitido

Clase que representa la información de una condición asociada a una tarea en fase.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **CONDICION**	TrCondicion	Condición.
→ **BLOQUEPER**	TrBloquePermitido	Bloque permitido.
→ **DEFPROC**	TrDefProcedimiento	Definición del procedimiento.
→ **MENSAJEOK**	String	Mensaje a mostrar cuando se cumple la condición.
→ **MENSAJENOOK**	String	Mensaje a mostrar cuando la condición no se cumple.
→ **OBLIGATORIA**	String	Indica si la condición se debe cumplir obligatoriamente o no. → "**S**" – Sí → "**N**" – No
→ **VALIDA**	String	Indica si la condición asociada a la tarea es válida. → "**S**" – Sí → "**N**" – No
→ **MOSTRARMSJ**	String	Mensaje a mostrar. → "**O**" – Mensaje OK → "**N**" – Mensaje No OK → "**A**" – Ambos → "**I**" – Ninguno
→ **COMPROBAR**	String	Indica si se realiza la condición. → "**I**" – La condición se produce al Incorporar → "**V**" – La condición se produce al Visualizar

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFCONDICION**	NUMÉRICO	Identificador de la condición
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento
→ **CAMPO_REFBLOQUEPER**	NUMÉRICO	Identificador del bloque permitido o tarea en fase
→ **CAMPO_REFFASE**	NUMÉRICO	Identificador de la fase.
→ **CAMPO_REFBLOQUEINI**	NUMÉRICO	Identificador del bloque inicial o tarea llamante
→ **CAMPO_REFBLOQUEFIN**	NUMÉRICO	Identificador del bloque final o tarea
→ **CAMPO_OBLIGATORIA**	CADENA(1)	Indica si la condición se debe cumplir obligatoriamente o no. → "**S**" – Sí → "**N**" – No
→ **CAMPO_VALIDA**	CADENA(1)	Indica si la condición asociada a la tarea es válida. → "**S**" – Sí → "**N**" – No
→ **CAMPO_COMPROBAR**	CADENA(1)	Indica si se realiza la condición. → "**I**" – La condición se produce al Incorporar → "**V**" – La condición se produce al Visualizar
→ **CAMPO_MOSTRARMSJ**	CADENA(1)	Mensaje a mostrar. → "**O**" – Mensaje OK → "**N**" – Mensaje No OK → "**A**" – Ambos → "**I**" – Ninguno
→ **CAMPO_MENSAJEOK**	CADENA(200)	Mensaje a mostrar cuando se cumple la condición.
→ **CAMPO_MENSAJENOOK**	CADENA(200)	Mensaje a mostrar cuando no se cumple la condición.

Métodos para aplicación de los filtros → obtenerCondicionBloquePermitido

TrCondicionDocumentoPermitido

trewa.bd.trapi.tpo.TrCondicionDocumentoPermitido

Clase que representa la información de una condición asociada a un documento permitido.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **CONDICION**	TrCondicion	Condición.
→ **TIPODOC**	TrTipoDocumento	Tipo de documento.
→ **DEFPROC**	TrDefProcedimiento	Definición del procedimiento.
FASE	TrFase	Fase.

→ **MENSAJEOK**	String	Mensaje a mostrar cuando se cumple la condición.
→ **MENSAJENOOK**	String	Mensaje a mostrar cuando la condición no se cumple.
→ **OBLIGATORIA**	String	Indica si la condición se debe cumplir obligatoriamente o no. → "**S**" – Sí → "**N**" – No
→ **VALIDA**	String	Indica si la condición asociada a la tarea es válida. → "**S**" – Sí → "**N**" – No
→ **MOSTRARMSJ**	String	Mensaje a mostrar. → "**O**" – Mensaje OK → "**N**" – Mensaje No OK → "**A**" – Ambos → "**I**" – Ninguno
→ **COMPROBAR**	String	Indica si se realiza la condición. → "**G**" – La condición se produce al Generar el documento → "**I**" – La condición se produce al Incorporar el documento → "**T**" – La condición se produce al Generar e Incorporar el documento → "**V**" – La condición se produce al Visualizar el documento

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFCONDICION**	NUMÉRICO	Identificador de la condición.
→ **CAMPO_REFTIPODOC**	NUMÉRICO	Identificador del tipo de documento.
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento.
→ **CAMPO_REFFASE**	NUMÉRICO	Identificador de la fase.
→ **CAMPO_MENSAJEOK**	CADENA(200)	Mensaje a mostrar cuando se cumple la condición.
→ **CAMPO_MENSAJENOOK**	CADENA(200)	Mensaje a mostrar cuando la condición no se cumple.
→ **CAMPO_OBLIGATORIA**	CADENA(1)	Indica si la condición se debe cumplir obligatoriamente o no. → "**S**" – Sí → "**N**" – No
→ **CAMPO_VALIDA**	CADENA(1)	Indica si la condición asociada a la tarea es válida. → "**S**" – Sí → "**N**" – No
→ **CAMPO_MOSTRARMSJ**	CADENA(1)	Mensaje a mostrar. → "**O**" – Mensaje OK → "**N**" – Mensaje No OK → "**A**" – Ambos → "**I**" – Ninguno
→ **CAMPO_COMPROBAR**	CADENA(1)	Indica si se realiza la condición. → "**G**" – La condición se produce al Generar el documento → "**I**" – La condición se produce al Incorporar el documento

→ "**T**" – La condición se produce al Generar e Incorporar el documento
→ "**V**" – La condición se produce al Visualizar el documento

Métodos para aplicación de los filtros → `obtenerCondicionDocumentoPermitido`

TrCondicionTransicion

trewa.bd.trapi.tpo.TrCondicionTransicion

Clase que representa la información de una condición de una transición.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **CONDICION**	`TrCondicion`	Condición.
→ **TRANSICION**	`TrTransicion`	Transición.
→ **DEFPROC**	`TrDefProcedimiento`	Definición del procedimiento.
→ **MENSAJEOK**	`String`	Mensaje a mostrar cuando se cumple la condición.
→ **MENSAJENOOK**	`String`	Mensaje a mostrar cuando la condiciónno se cumple.
→ **OBLIGATORIA**	`String`	Indica si la condiciónse debe cumplir obligatoriamente o no. → "**S**" – Sí → "**N**" – No
→ **VALIDA**	`String`	Indica si la condiciónasociada a la tarea es válida. → "**S**" – Sí → "**N**" – No
→ **MOSTRARMSJ**	`String`	Mensaje a mostrar. → "**O**" – Mensaje OK → "**N**" – Mensaje No OK → "**A**" – Ambos → "**I**" – Ninguno
→ **COMPROBAR**	`String`	Indica si se realiza la condición. → "**T**" – La condición se produce al Tramitar → "**D**" – La condición se produce al Deshacer → "**A**" – La condición se produce al Tramitar y Deshacer → "**V**" – La condición se produce al Visualizar

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFCONDICION**	`NUMÉRICO`	Identificador de la condición.

→ **CAMPO_REFTRANSICION**	NUMÉRICO	Identificador de la transición.
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento.
→ **CAMPO_OBLIGATORIA**	CADENA(1)	Indica si la condiciónse debe cumplir obligatoriamente o no. → "**S**" – Sí → "**N**" – No
→ **CAMPO_VALIDA**	CADENA(1)	Indica si la condiciónasociada a la tarea es válida. → "**S**" – Sí → "**N**" – No
→ **CAMPO_MOSTRARMSJ**	CADENA(1)	Mensaje a mostrar. → "**O**" – Mensaje OK → "**N**" – Mensaje No OK → "**A**" – Ambos → "**I**" – Ninguno
→ **CAMPO_COMPROBAR**	CADENA(1)	Indica si se realiza la condición. → "**T**" – La condición se produce al Tramitar → "**D**" – La condición se produce al Deshacer → "**A**" – La condición se produce al Tramitar y Deshacer → "**V**" – La condición se produce al Visualizar
→ **CAMPO_MENSAJEOK**	CADENA(200)	Mensaje a mostrar cuando se cumple la condición.
→ **CAMPO_MENSAJENOOK**	CADENA(200)	Mensaje a mostrar cuando no se cumple la condición.

Métodos para aplicación de los filtros → obtenerCondicionTransicion

TrConstante

trewa.bd.trapi.tpo.TrConstante

Clase que recoge la información de las constantes del motor de tramitación.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **CODCONSTANTE**	String	Código de la constante.
→ **DESCRIPCION**	String	Descripción de la constante.
→ **VALOR**	String	Valor de la constante.
→ **STMA**	TrSistema	Sistema.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_CODIGOCTE**	CADENA(20)	Código de la constante.

→ **CAMPO_DESCRIPCION**	CADENA(250)	Descripción de la constante.
→ **CAMPO_VALOR**	CADENA(500)	Valor de la constante.
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema.

Métodos para aplicación de los filtros → obtenerConstante

TrConstanteGn

trewa.bd.trapi.tpo.TrConstanteGn

Clase que recoge la información de las constantes generales.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **CODCONSTANTE**	String	Código de la constante general.
→ **DESCRIPCION**	String	Descripción de la constante general.
→ **VALOR**	String	Valor de la constante general.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_CODCONSTANTE**	CADENA(10)	Código de la constante general.
→ **CAMPO_DESCRIPCION**	CADENA(64)	Descripción de la constante general.
→ **CAMPO_VALOR**	CADENA(80)	Valor de la constante general.

Métodos para aplicación de los filtros → obtenerConstanteGeneral

TrDatoComponente

trewa.bd.trapi.tpo.TrDatoComponente

Clase que recoge otros datos necesarios para el componente (atributo-valor).

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFDATOCOMP**	TpoPK	Identificador del dato del componente.
→ **COMPONENTE**	TrComponente	Componente.

→ **ATRIBUTO**	`String`	Nombre del atributo del componente.
→ **VALOR**	`String`	Valor del atributo del componente.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFDATOCOMP**	`NUMÉRICO`	Identificador del dato del componente.
→ **CAMPO_REFCOMP**	`NUMÉRICO`	Identificador del componente.
→ **CAMPO_ATRIBUTO**	`CADENA(32)`	Nombre del atributo del componente.
→ **CAMPO_VALOR**	`CADENA(255)`	Valor del atributo del componente.

Métodos para aplicación de los filtros → `obtenerDatoComponente`

TrDefProcedimiento

trewa.bd.trapi.tpo.TrDefProcedimiento

Clase que representa la información de una definición de un procedimiento.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFDEFPROC**	`TpoPK`	Identificador de la definición del procedimiento.
→ **ABREVIATURA**	`String`	Abreviatura de la definición del procedimiento.
→ **DESCRIPCION**	`String`	Descripción de la definición del procedimiento.
→ **STMA**	`TrSistema`	Sistema de la definición del procedimiento.
→ **INFORMAR**	`String`	Indica si se tendrá que informar al bus de este procedimiento y de los expedientes que se tramiten. → "**S**" – Sí → "**N**" – No
→ **BLOQUEADO**	`String`	Indica si el procedimiento está bloqueado. → "**S**" – Sí → "**N**" – No
→ **USUARIOBLQ**	`String`	Usuario que tiene bloqueado el procedimiento
→ **VIGENTE**	`String`	Indica si el procedimiento está vigente. → "**S**" – Sí → "**N**" – No
→ **CATEGORIA**	`String`	Indica si es Familia, subfamilia o procedimiento. → "**F**" – Familia → "**S**" – Subfamilia → "**P**" – Procedimiento

→ **DESCRIPCIONAMP**	String	Descripción ampliada del procedimiento.
→ **REFDEFPROCPADRE**	TpoPK	Identificador de la familia/subfamilia.
→ **ORGANISMO**	TrOrganismo	Organismo a la que pertenece.
→ **ORGCOMPETENTE**	TrOrganismo	Organismo competente.
→ **ORGRESUELVE**	TrOrganismo	Organismo que resuelve.
→ **ORGTRAMITA**	TrOrganismo	Organismo que tramita.
→ **CODWANDA**	String	Valor en w@ndA del procedimiento
→ **COMENTARIOS**	String	Comentarios al procedimiento/familia/subfamilia

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento.
→ **CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura de la definición del procedimiento.
→ **CAMPO_DESCRIPCION**	CADENA(50)	Descripción de la definición del procedimiento.
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema de la definición del procedimiento.
→ **CAMPO_INFORMAR**	CADENA(1)	Indica si se tendrá que informar al bus de este procedimiento y de los expedientes que se tramiten. → "**S**" – Sí → "**N**" – No
→ **CAMPO_USUARIOBLQ**	CADENA(10)	Indica el usuario que tiene bloqueado el procedimiento
→ **CAMPO_VIGENTE**	CADENA(1)	Indica si el procedimiento está vigente. → "**S**" – Sí → "**N**" – No
→ **CAMPO_CATEGORIA**	CADENA(1)	Indica si es Familia, subfamilia o procedimiento. → "**F**" – Familia → "**S**" – Subfamilia → "**P**" – Procedimiento
→ **CAMPO_DESCRIPCIONAMP**	CADENA(250)	Descripción ampliada del procedimiento.
→ **CAMPO_REFDEFPROCPADRE**	NUMÉRICO	Identificador de la familia/subfamilia.
→ **CAMPO_REFORGANISMO**	NUMÉRICO	Organismo al que pertenece.
→ **CAMPO_REFORGCOMP**	NUMÉRICO	Organismo competente.
→ **CAMPO_REFORGRES**	NUMÉRICO	Organismo que resuelve.
→ **CAMPO_REFORGTRAM**	NUMÉRICO	Organismo que tramita.
→ **CAMPO_CODWANDA**	CADENA(15)	Valor en w@ndA del procedimiento

Métodos para aplicación de los filtros → obtenerDefProcedimiento

TrDefProcedimientoGr

trewa.bd.trapi.tpo.TrDefProcedimientoGr

Clase que representa la información de una definición de un procedimiento gráfico.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFDEFPROCGR**	TpoPK	Identificador de la definición gráfica del procedimiento.
→ **DESCDIAGRAMA**	String	Descripción de la definición gráfica del procedimiento.
→ **DEFPROC**	TrDefProcedimiento	Definición del procedimiento.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFDEFPROCGR**	NUMÉRICO	Identificador de la definición gráfica del procedimiento.
→ **CAMPO_DESCDIAGRAMA**	CADENA(250)	Descripción de la definición gráfica del procedimiento.
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento.

Métodos para aplicación de los filtros → obtenerDefProcedimientoGr

TrDocumentoDelegado

trewa.bd.trapi.tpo.TrDocumentoDelegado

Clase que representa los documentos para los que los delegados de firma tienen competencia para firmar.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFDOCDEL**	TpoPK	Identificador del documento delegado.
→ **FECHAINIVIG**	Timestamp	Fecha de inicio de vigencia para la delegación del documento.
→ **FECHAFINVIG**	Timestamp	Fecha fin de vigencia para la delegación del documento.
→ **FIRMDEF**	TrFirmanteDefinido	Firmante definido.
→ **TIPODOC**	TrTipoDocumento	Tipo de documento.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFDOCDEL**	NUMÉRICO	Identificador del documento delegado.

→ **CAMPO_FECHAINIVIG**	DATE	Fecha de inicio de vigencia para la delegación del documento.
→ **CAMPO_FECHAFINVIG**	DATE	Fecha fin de vigencia para la delegación del documento.
→ **CAMPO_REFFIRMDEF**	NUMÉRICO	Identificador del firmante definido.
→ **CAMPO_REFTIPODOC**	NUMÉRICO	Identificador del tipo de documento.

Métodos para aplicación de los filtros → obtenerDocumentoDelegado

TrDocumentoPermitido

trewa.bd.trapi.tpo.TrDocumentoPermitido

Clase que representa los documentos permitidos en una determinada fase.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFDOCPER**	TpoPK	Identificador del documento permitido.
→ **OBLIGATORIO**	String	Indica si el documento permitido es obligatorio. → "**S**" – Sí → "**N**" – No
→ **VALIDO**	String	Indica si el documento permitido es válido
→ **ORDEN**	Integer	Indica el orden del documento permitido en la fase.
→ **TIPODOC**	TrTipoDocumento	Tipo de documento
→ **FASE**	TrFase	Fase.
→ **INFORMAR**	String	Indica si se informa al bus del documento → "**S**" – Sí → "**N**" – No
→ **ETIQUETA**	String	Texto con el que se etiqueta al documento permitido en la fase
→ **DESCRIPCION**	String	Descripción de lo que representa el documento permitido en la fase
→ **ETIQLARGA**	String	Etiqueta larga para el documento permitido

TrDocumentoPermitidoDefProc

trewa.bd.trapi.tpo.TrDocumentoPermitidoDefProc

Clase que representa los documentos permitidos en una determinada fase que se asocian a un procedimiento.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **DOCPER**	TrDocumentoPermitido	Documento permitido.
→ **REFDEFPROC**	TpoPK	Identificador de la definción del procedimiento.
→ **UNIDAD**	String	Unidad de medida para el plazo máximo de realización. → "**D**" – Días → "**M**" – Meses → "**A"** – Años
→ **NUMUNIDADES**	Integer	Indica el nº de unidades del plazo máximo de realización: nº de días, nº de meses, nº de años.
→ **DESCFECHALIM**	String	Descripción del significado de la fecha límite de realización de la tarea.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definción del procedimiento.
→ **CAMPO_UNIDAD**	CADENA(1)	Unidad de medida para el plazo máximo de realización. → "**D**" – Días → "**M**" – Meses → "**A"** – Años
→ **CAMPO_NUMUNIDADES**	NUMÉRICO	Indica el nº de unidades del plazo máximo de realización: nº de días, nº de meses, nº de años.
→ **CAMPO_DESCFECHALIM**	CADENA(100)	Descripción del significado de la fecha límite de realización de la tarea.

Métodos para aplicación de los filtros → obtenerDocumentoPermitidoDefProc

TrDocumentoPermitidoPerfil

trewa.bd.trapi.tpo.TrDocumentoPermitidoPerfil

Clase que representa los documentos permitidos por procedimiento y perfil de usuario.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **PERMISO**	String	Permiso para el perfil. Sus valores pueden ser:

		→ "**G**" – Generar → "**I**" – Incorporar → "**F**" – Firmar → "**E**" – Editar → "**T**" – Todo
→ **PERFILUSU**	TrPerfilUsuario	Perfil de usuario.
→ **REFTIPODOC**	TpoPK	Identificador del tipo de documento.
→ **REFFASE**	TpoPK	Identificador de la fase.
→ **REFDEFPROC**	TpoPK	Identificador de la definición del procedimiento.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_PERMISO**	CADENA(1)	Permiso para el perfil. Sus valores pueden ser: → "**G**" – Generar → "**I**" – Incorporar → "**F**" – Firmar → "**E**" – Editar → "**T**" – Todo
→ **CAMPO_REFPERFILUSU**	NUMÉRICO	Identificador del perfil de usuario
→ **CAMPO_REFTIPODOC**	NUMÉRICO	Identificador del tipo de documento
→ **CAMPO_REFFASE**	NUMÉRICO	Identificador de la fase
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento.

Métodos para aplicación de los filtros → obtenerDocumentoPermitidoPerfil

TrEmpleado

trewa.bd.trapi.tpo.TrEmpleado

Clase que representa los usuarios que ocupan los puestos de trabajo en los organismos definidos.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **USUARIO**	TrUsuario	Usuario.
→ **ORGANISMO**	TrOrganismo	Organismo.
→ **PTOTRABAJO**	TrPuestoTrabajo	Puesto de trabajo.
→ **TIPO**	String	Indica el tipo de empleado.

		→ "**F**" – Funcionario → "**L**" – Laboral
→ **FECHANOMBRAMIENTO**	`Timestamp`	Fecha de ocupación del puesto por el usuario.
→ **FECHACESE**	`Timestamp`	Fecha de baja del usuario en el puesto.
→ **TRATAMIENTO**	`String`	Texto de tratamiento para los documentos (D., Dª, Ilmo., etc.)
→ **TXTREFERENCIA**	`String`	Texto de referencia para el usuario.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_USUARIO**	`CADENA(10)`	Código del usuario.
→ **CAMPO_REFORGANISMO**	`NUMÉRICO`	Identificador del organismo.
→ **CAMPO_CODPTOTRAB**	`CADENA(10)`	Código del puesto de trabajo.
→ **CAMPO_TIPO**	`CADENA(1)`	Indica el tipo de empleado. → "**F**" – Funcionario → "**L**" – Laboral
→ **CAMPO_FECHANOMBRAMIENTO**	`DATE`	Fecha de ocupación del puesto por el usuario.
→ **CAMPO_FECHACESE**	`DATE`	Fecha de baja del usuario en el puesto
→ **CAMPO_TRATAMIENTO**	`CADENA(64)`	Texto de tratamiento para los documentos (D., Dª, Ilmo., etc.)
→ **CAMPO_TXTREFERENCIA**	`CADENA(10)`	Texto de referencia para el usuario.

Métodos para aplicación de los filtros → `obtenerEmpleado`

TrError

trewa.bd.trapi.tpo.TrError

Clase que representa los errores definidos para el tramitador.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFERROR**	`TpoPK`	Identificador del error.
→ **MENSAJE**	`String`	Mensaje del error.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFERROR**	`NUMÉRICO`	Identificador del error.

→ **CAMPO_MENSAJE**	CADENA(250)	Mensaje del error

Métodos para aplicación de los filtros → obtenerError

TrExtremoTransicionGr

trewa.bd.trapi.tpo.TrExtremoTransicionGr

Clase que representa la información gráfica de los distintos extremos de transición que pueden darse.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFEXTREMOGR**	TpoPK	Identificador del extremo gráfico.
→ **XIZQ**	Integer	Componente X de la coordenada superior izquierda del gráfico.q
→ **YARR**	Integer	Componente Y de la coordenada superior izquierda del gráfico.
→ **ANCHO**	Integer	Ancho del gráfico.
→ **ALTO**	Integer	Alto del gráfico.
→ **DEFPROCGR**	TrDefProcedimientoGr	Definición del procedimiento gráfico.
→ **FASE**	TrFase	Fase.
→ **TIPO**	String	Indica el tipo de extremo de la transición. → "**FA**" – Fase normal → "**I**" – Fase inicial → "**D**" – División → "**U**" – Unión → "**F**" – Fase final → "**ES**" – Evento salida, abandona la fase actual. → "**EN**" – Evento que no abandona la fase actual.
→ **METAFASEGR**	TrMetafaseGr	Metafase gráfica.
→ **COLORFONDO**	Long	Color de fondo del extremo de la transición.
→ **COLORTEXTO**	Long	Color del texto del extremo de la transición.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFEXTREMOGR**	NUMÉRICO	Identificador del extremo gráfico.
→ **CAMPO_XIZQ**	NUMÉRICO	Componente X de la coordenada superior izquierda del gráfico.
→ **CAMPO_YARR**	NUMÉRICO	Componente Y de la coordenada superior izquierda del gráfico.

→ **CAMPO_ANCHO**	NUMÉRICO	Ancho del gráfico.
→ **CAMPO_ALTO**	NUMÉRICO	Alto del gráfico.
→ **CAMPO_REFDEFPROCGR**	NUMÉRICO	Identificador de la definición del procedimiento gráfico.
→ **CAMPO_REFFASE**	NUMÉRICO	Identificador de la fase.
→ **CAMPO_TIPO**	CADENA(2)	Indica el tipo de extremo de la transición. → "**FA**" – Fase normal → "**I**" – Fase inicial → "**D**" – División → "**U**" – Unión → "**F**" – Fase final → "**ES**" – Evento salida, abandona la fase actual. → "**EN**" – Evento que no abandona la fase actual.
→ **CAMPO_REFMETAFASEGR**	NUMÉRICO	Identificador de la metafase gráfica.
→ **CAMPO_COLORFONDO**	NUMÉRICO	Color de fondo del extremo de la transición.
→ **CAMPO_COLORTEXTO**	NUMÉRICO	Color del texto del extremo de la transición.

Métodos para aplicación de los filtros → obtenerExtremoTransicionGr

TrFase

trewa.bd.trapi.tpo.TrFase

Clase que representa la información de una fase.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFFASE**	TpoPK	Identificador de la fase.
→ **NOMBRE**	String	Nombre de la fase.
→ **DESCRIPCION**	String	Descripcion de la fase.
→ **ORDEN**	Integer	Orden no riguroso de la fase dentro de la metafase.
→ **DEFPROC**	TrDefProcedimientoq	Definición del procedimiento.
→ **METAFASE**	TrMetafase	Metafase a la que pertenece la fase.
→ **STMA**	TrSistema	Sistema al que está asociada la fase.
→ **INFORMAR**	String	Indica si se informa al bus de la entrada en la fase. → "**S**" – Sí → "**N**" – No
→ **TEXTOAUXILIAR**	String	Texto auxiliar para la fase

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFFASE**	NUMÉRICO	Identificador de la fase.
→ **CAMPO_NOMBRE**	CADENA(50)	Nombre de la fase.
→ **CAMPO_DESCRIPCION**	CADENA(250)	Descripción de la fase.
→ **CAMPO_ORDEN**	NUMÉRICO	Orden no riguroso de la fase dentro de la metafase.
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento.
→ **CAMPO_REFMETAFASE**	NUMÉRICO	Identificador de la metafase a la que pertenece la fase.
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema al que está asociada la fase.
→ **CAMPO_INFORMAR**	CADENA(1)	Indica si se informa al bus de la entrada en la fase. → "**S**" – Sí → "**N**" – No
→ **CAMPO_TEXTOAUXILIAR**	CADENA(500)	Texto auxiliar para la fase.

Métodos para aplicación de los filtros → obtenerFase

TrFirma

trewa.bd.trapi.tpo.TrFirma

Clase que representa la firma de los usuarios firmantes.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFFIRMA**	TpoPK	Identificador de la firma.
→ **FIRMA**	byte[]	Gráfico de la firma.
→ **USUARIO**	String	Usuario que firma.
→ **FORMATO**	String	Tipo de formato de la imagen (tipo mime).
→ **NOMBREFICHERO**	String	Nombre del fichero de la imagen.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFFIRMA**	NUMÉRICO	Identificador de la firma.
→ **CAMPO_USUARIO**	CADENA(10)	Usuario que firma.
→ **CAMPO_FORMATO**	CADENA(128)	Tipo de formato de la imagen (tipo mime).
→ **CAMPO_NOMBREFICHERO**	CADENA(64)	Nombre del fichero de la imagen.

Métodos para aplicación de los filtros → obtenerFirma

TrFirmanteDefinido

trewa.bd.trapi.tpo.TrFirmanteDefinido

Clase que representa los firmantes principales, sustitutos y delegados definidos en el sistema.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFFIRMDEF**	TpoPK	Identificador del firmante definido.
→ **TIPO**	String	Tipo de firmante. → "**P**" – Principal → "**D**" – Delegado → "**S**" – Sustituto
→ **FECHAINIVIG**	Timestamp	Fecha de inicio de vigencia del firmante definido.
→ **FECHAFINVIG**	Timestamp	Fecha de fin de vigencia del firmante definido.
→ **TEXTODISPOSICION**	TrTextoDisposicionFirma	Texto de la disposición.
→ **REFDELEGSUST**	TpoPK	Identificador del delegado o sustituto.
→ **PTOTRABAJO**	TrPuestoTrabajo	Puesto de trabajo.
→ **ORGANISMO**	TrOrganismo	Organismo.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFFIRMDEF**	NUMÉRICO	Identificador del firmante definido.
→ **CAMPO_TIPO**	CADENA(1)	Tipo de firmante. → "**P**" – Principal → "**D**" – Delegado → "**S**" – Sustituto
→ **CAMPO_FECHAINIVIGE**	DATE	Fecha de inicio de vigencia del firmante definido.
→ **CAMPO_FECHAFINVIG**	DATE	Fecha de fin de vigencia del firmante definido.
→ **CAMPO_TEXTODISPOSICION**	CADENA(10)	Texto de la disposición.
→ **CAMPO_REFDELEGSUST**	NUMÉRICO	Identificador del del delegado o sustituto.
→ **CAMPO_CODPTOTRAB**	CADENA(10)	Código del puesto de trabajo.
→ **CAMPO_REFORGANISMO**	NUMÉRICO	Identificador del organismo.

Métodos para aplicación de los filtros → obtenerFirmanteDefinido

TrFirmaTipoDocumento

trewa.bd.trapi.tpo.TrFirmaTipoDocumento

Clase que representa las firmas que puede tener un tipo de documento.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **TIPODOC**	`TrTipoDocumento`	Tipo de documento.
→ **FIRMDEF**	`TrFirmanteDefinido`	Firmante definido.
→ **ORDEN**	`Integer`	Orden de la firma.
→ **EDITABLE**	`String`	Indica si el pie de firma que corresponde con la firma es editable o no. → "**S**" – Sí → "**N**" – No
→ **ETIQUETA**	`String`	Etiqueta del pie de firma.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFTIPODOC**	`NUMÉRICO`	Identificador del tipo de documento.
→ **CAMPO_REFFIRMDEF**	`NUMÉRICO`	Identificador del firmante definido.
→ **CAMPO_ORDEN**	`NUMÉRICO`	Orden de la firma.
→ **CAMPO_EDITABLE**	`CADENA(1)`	Indica si el pie de firma que corresponde con la firma es editable o no. → "**S**" – Sí → "**N**" – No
→ **CAMPO_ETIQUETA**	`CADENA(20)`	Etiqueta del pie de firma.

Métodos para aplicación de los filtros → `obtenerFirmaTipoDocumento`

TrIndicacionFicha

trewa.bd.trapi.tpo.TrIndicacionFicha

Clase que representa datos variables sobre la ficha descriptiva del procedimiento.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción

→ **REFINDFICHA**	TpoPK	Identificador de la indicación de la ficha del procedimiento.
→ **DEFPROC**	TrDefProcedimiento	Definición del procedimiento.
→ **TIPOINDICA**	TrTipoIndicacion	Tipo de indicación de la ficha.
→ **DESCRIPCION**	String	Descripción de la indicación de la ficha del procedimiento.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFINDFICHA**	NUMÉRICO	Identificador de la indicación de la ficha del procedimiento.
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento.
→ **CAMPO_REFTIPOINDICA**	NUMÉRICO	Identificador del tipo de indicación de la ficha.
→ **CAMPO_DESCRIPCION**	CADENA(300)	Descripción de la indicación de la ficha del procedimiento.

Métodos para aplicación de los filtros → obtenerIndicacionFicha

TrLimiteCaducidad

trewa.bd.trapi.tpo.TrLimiteCaducidad

Clase que representa el límite para una caducidad.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **TRANSICION**	TrTransicion	Transición.
→ **CADUCIDAD**	TrCaducidad	Caducidad.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFTRANSICION**	NUMÉRICO	Identificador de la transición.
→ **CAMPO_REFCADUCIDAD**	NUMÉRICO	Identificador de la caducidad.

Métodos para aplicación de los filtros → obtenerLimiteCaducidad

TrMetafase

trewa.bd.trapi.tpo.TrMetafase

Clase que representa una agrupación de fases según el significado en la tramitación.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFMETAFASE**	TpoPK	Identificador de la metafase.
→ **NOMBRE**	String	Nombre de la metafase.
→ **DESCRIPCION**	String	Descripción de la metafase.
→ **ORDEN**	Integer	Orden no riguroso de la metafase dentro del procedimiento en el que aparecen.
→ **INFORMAR**	String	Indica si se informa al bus de la entrada en la metafase. → "**S**" – Sí → "**N**" – No
→ **STMA**	TrSistema	Sistema.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFMETAFASE**	NUMÉRICO	Identificador de la metafase.
→ **CAMPO_NOMBRE**	CADENA(50)	Nombre de la metafase.
→ **CAMPO_DESCRIPCION**	CADENA(250)	Descripción de la metafase.
→ **CAMPO_ORDEN**	NUMÉRICO	Orden no riguroso de la metafase dentro del procedimiento en el que aparecen.
→ **CAMPO_INFORMAR**	CADENA(1)	Indica si se informa al bus de la entrada en la metafase. → "**S**" – Sí → "**N**" – No
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema

Métodos para aplicación de los filtros → obtenerMetafase

TrMetafaseGr

trewa.bd.trapi.tpo.TrMetafaseGr

Clase que representa la información gráfica de una metafase.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFMETAFASEGR**	TpoPK	Identificador de la metafase gráfica.
→ **XIZQ**	Integer	Componente X de la coordenada superior izquierda del

		gráfico.q
→ **YARR**	`Integer`	Componente Y de la coordenada superior izquierda del gráfico.
→ **ANCHO**	`Integer`	Ancho del gráfico.
→ **ALTO**	`Integer`	Alto del gráfico.
→ **DEFPROCGR**	`TrDefProcedimientoGr`	Definición del procedimiento gráfico.
→ **METAFASE**	`TrMetafase`	Metafase.
→ **COLORFONDO**	`Long`	Color de fondo del gráfico de metafase.
→ **COLORTEXTO**	`Long`	Color del texto del gráfico de metafase.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFMETAFASEGR**	`NUMÉRICO`	Identificador de la metafase gráfica.
→ **CAMPO_XIZQ**	`NUMÉRICO`	Componente X de la coordenada superior izquierda del gráfico
→ **CAMPO_YARR**	`NUMÉRICO`	Componente Y de la coordenada superior izquierda del gráfico
→ **CAMPO_ANCHO**	`NUMÉRICO`	Ancho del gráfico
→ **CAMPO_ALTO**	`NUMÉRICO`	Alto del gráfico
→ **CAMPO_REFDEFPROCGR**	`NUMÉRICO`	Identificador de la definición del procedimiento gráfico.
→ **CAMPO_REFMETAFASE**	`NUMÉRICO`	Identificador de la metafase.
→ **CAMPO_COLORFONDO**	`NUMÉRICO`	Color de fondo del gráfico de metafase.
→ **CAMPO_COLORTEXTO**	`NUMÉRICO`	Color del texto del gráfico de metafase.

Métodos para aplicación de los filtros → `obtenerMetafaseGr`

TrMunicipio

trewa.bd.trapi.tpo.TrMunicipio

Clase que representa la información de un municipio de una provincia.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **CODMUNICIPIO**	`String`	Código del municipio.
→ **PROVINCIA**	`TrProvincia`	Provincia a la que pertenece el municipio.
→ **NOMBRE**	`String`	Nombre del municipio.
→ **DESCRIPCION**	`String`	Descripción del municipio.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_CODMUNICIPIO**	CADENA(3)	Código del municipio.
→ **CAMPO_CODPROVINCIA**	CADENA(2)	Código de la provincia a la que pertenece el municipio.
→ **CAMPO_NOMBRE**	CADENA(32)	Nombre del municipio.
→ **CAMPO_DESCRIPCION**	CADENA(64)	Descripción del municipio.

Métodos para aplicación de los filtros → obtenerMunicipio

TrNodoTransicionGr

trewa.bd.trapi.tpo.TrNodoTransicionGr

Clase que representa a información gráfica de los nodos de una transición.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFNODOTRANGR**	TpoPK	Identificador del nodo de la transición gráfica.
→ **X**	Integer	Componente X de la coordenada del nodo.
→ **Y**	Integer	Componente Y de la coordenada del nodo.
→ **ORDEN**	Integer	Orden del nodo en un gráfico de transición.
→ **TRANSGR**	TrTransicionGr	Transición gráfica.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFNODOTRANGR**	NUMÉRICO	Identificador del nodo de la transición gráfica.
→ **CAMPO_X**	NUMÉRICO	Componente X de la coordenada del nodo
→ **CAMPO_Y**	NUMÉRICO	Componente Y de la coordenada del nodo
→ **CAMPO_ORDEN**	NUMÉRICO	Orden del nodo en un gráfico de transición
→ **CAMPO_REFTRANGR**	NUMÉRICO	Identificador de la transición gráfica.

Métodos para aplicación de los filtros → obtenerNodoTransicionGr

TrNormativa

trewa.bd.trapi.tpo.TrNormativa

Clase que representa las distintas normativas que rigen los procedimientos.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFNORMATIVA**	TpoPK	Identificador de la normativa.
→ **TIPONORMA**	TrTipoNormativa	Tipo de normativa.
→ **TIPOPUBLI**	TrTipoPublicacion	Tipo de publicación.
→ **DESCNORMATIVA**	String	Descripción de la normativa.
→ **FECHAVIGOR**	Timestamp	Fecha de entrada en vigor de la normativa.
→ **TITULO**	String	Título de la normativa.
→ **NUMERO**	Long	Número de la normativa.
→ **FECHAPUBLI**	Timestamp	Fecha de publicación de la normativa.
→ **NUMPUBLI**	Long	Número de publicación de la normativa.
→ **ANYO**	Integer	Año de la normativa.
→ **AMBITOLEY**	TrAmbitoLey	Ámbito de la ley de la normativa.
→ **CODWANDA**	String	Valor en w@ndA de la normativa

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFNORMATIVA**	NUMÉRICO	Identificador de la normativa.
→ **CAMPO_REFTIPONORMA**	NUMÉRICO	Identificador del tipo de normativa.
→ **CAMPO_REFTIPOPUBLI**	NUMÉRICO	Identificador del tipo de publicación.
→ **CAMPO_DESCNORMATIVA**	CADENA(250)	Descripción de la normativa.
→ **CAMPO_FECHAVIGOR**	DATE	Fecha de entrada en vigor de la normativa.
→ **CAMPO_TITULO**	CADENA(50)	Título de la normativa
→ **CAMPO_NUMERO**	NUMÉRICO	Número de la normativa.
→ **CAMPO_FECHAPUBLI**	DATE	Fecha de publicación de la normativa.
→ **CAMPO_NUMPUBLI**	NUMÉRICO	Número de publicación de la normativa.
→ **CAMPO_CODWANDA**	CADENA(15)	Valor en w@ndA de la normativa
→ **CAMPO_ANYO**	NUMÉRICO	Año de la normativa.
→ **CAMPO_REFAMBITOLEY**	NUMÉRICO	Identificador del ámbito de la ley de la normativa.

Métodos para aplicación de los filtros → obtenerNormativa

TrNormativaDefProcedimiento

trewa.bd.trapi.tpo.TrNormativaDefProcedimiento

Clase que representa las distintas normativas por las que se rige un procedimiento.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFDEFPROC**	TpoPK	Identificador de la definición del procedimiento.
→ **NORMATIVA**	TrNormativa	Normativa.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento.
→ **CAMPO_REFNORMATIVA**	NUMÉRICO	Identificador de la normativa.
→ **CAMPO_REFTIPONORMA**	NUMÉRICO	Identificador del tipo de normativa.
→ **CAMPO_REFTIPOPUBLI**	NUMÉRICO	Identificador del tipo de publicación.
→ **CAMPO_DESCNORMATIVA**	CADENA(250)	Descripción de la normativa.
→ **CAMPO_FECHAVIGOR**	DATE	Fecha de entrada en vigor de la normativa.
→ **CAMPO_TITULO**	CADENA(50)	Título de la normativa
→ **CAMPO_NUMERO**	NUMÉRICO	Número de la normativa.
→ **CAMPO_FECHAPUBLI**	DATE	Fecha de publicación de la normativa.
→ **CAMPO_NUMPUBLI**	NUMÉRICO	Número de publicación de la normativa.
→ **CAMPO_CODWANDA**	CADENA(15)	Valor en w@ndA de la normativa
→ **CAMPO_ANYO**	NUMÉRICO	Año de la normativa.
→ **CAMPO_REFAMBITOLEY**	NUMÉRICO	Identificador del ámbito de la ley de la normativa.

Métodos para aplicación de los filtros → obtenerNormativaDefProcedimiento

TrOrganismo

trewa.bd.trapi.tpo.TrOrganismo

Clase que representa la información de los distintos organismos o unidades orgánicas.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFORGANISMO**	TpoPK	Identificador del organismo.
→ **CODMUNICIPIO**	String	Código del municipio.
→ **CODPROVINCIA**	String	Código de la provincia.
→ **TIPOVIA**	String	Tipo de vía.
→ **CODORG**	String	Código del organismo.
→ **NOMBRE**	String	Nombre del organismo.
→ **DESCRIPCION**	String	Descripción del organismo.

→ **TIPO**	String	Tipo de organismo. → "**DC**" – Delegación Provincial → "**SC**" – Servicios Centrales → "**CE**" – Centro Adscrito → "**EE**" – Empresa Externa
→ **NOMBREVIA**	String	Nombre de la vía pública del domicilio.
→ **NUMERO**	Integer	Número del domicilio.
→ **LETRA**	String	Letra del domicilio.
→ **ESCALERA**	String	Escalera del domicilio.
→ **PISO**	Integer	Piso del domicilio.
→ **PUERTA**	Stringq	Puerta del domicilio.
→ **CODPOSTAL**	Integer	Código postal del domicilio.
→ **TELEFONO**	String	Número(s) de teléfono de contacto.
→ **TLFMOVIL**	String	Número(s) de teléfono móvil.
→ **FAX**	String	Número(s) de Fax.
→ **EMAIL**	String	Correo electrónico.
→ **CODCIWA**	String	Código de identificación w@ndA.
→ **IDARIES**	String	Identificador de aries.
→ **IDENTIFICADOR**	String	Número de identificación (CIF).
→ **DIGCONTROL**	String	Dígito de control para el número de identificación.
→ **FECHAINIVIG**	Timestamp	Fecha de inicio de vigencia del organismo.
→ **FECHAFINVIG**	Timestamp	Fecha de fin de vigencia del organismo
→ **REFORGPADRE**	TpoPK	Identificador del organismo superior al que pertenece.
→ **TIPOORG**	TrTipoOrganismo	Tipo de organismo.
→ **TIPOORGZ**	TrTipoOrganizacion	Tipo de organización.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFORGANISMO**	NUMÉRICO	Identificador del organismo.
→ **CAMPO_CODMUNICIPIO**	CADENA(3)	Código del municipio.
→ **CAMPO_CODPROVINCIA**	CADENA(2)	Código de la provincia.
→ **CAMPO_TIPOVIA**	CADENA(10)	Código del tipo de vía.
→ **CAMPO_CODORG**	CADENA(16)	Código del organismo
→ **CAMPO_NOMBRE**	CADENA(32)	Nombre del organismo.
→ **CAMPO_DESCRIPCION**	CADENA(200)	Descripción del organismo.
→ **CAMPO_TIPO**	CADENA(2)	Tipo de organismo. → "**DC**" – Delegación Provincial → "**SC**" – Servicios Centrales → "**CE**" – Centro Adscrito → "**EE**" – Empresa Externa
→ **CAMPO_NOMBREVIA**	CADENA(64)	Nombre de la vía pública del domicilio.
→ **CAMPO_NUMERO**	NUMÉRICO	Número del domicilio.

→ **CAMPO_LETRA**	CADENA(2)	Letra del domicilio.
→ **CAMPO_ESCALERA**	CADENA(2)	Escalera del domicilio.
→ **CAMPO_PISO**	NUMÉRICO	Piso del domicilio.
→ **CAMPO_PUERTA**	CADENA(2)	Puerta del domicilio.
→ **CAMPO_CODPOSTAL**	NUMÉRICO	Código postal del domicilio.
→ **CAMPO_TELEFONO**	CADENA(25)	Número(s) de teléfono de contacto.
→ **CAMPO_TLFMOVIL**	CADENA(25)	Número(s) de teléfono móvil.
→ **CAMPO_FAX**	CADENA(25)	Número(s) de Fax.
→ **CAMPO_EMAIL**	CADENA(64)	Correo electrónico.
→ **CAMPO_CIWA**	CADENA(15)	Código de identificación w@ndA.
→ **CAMPO_IDARIES**	CADENA(5)	Identificador de aries.
→ **CAMPO_IDENTIFICADOR**	CADENA(15)	Número de identificación (CIF).
→ **CAMPO_DIGCONTROL**	CADENA(1)	Dígito de control para el número de identificación.
→ **CAMPO_FECHAINIVIG**	DATE	Fecha de inicio de vigencia del organismo.
→ **CAMPO_FECHAFINVIG**	DATE	Fecha de fin de vigencia del organismo
→ **CAMPO_REFORGPADRE**	NUMÉRICO	Identificador del organismo superior al que pertenece.
→ **CAMPO_REFTIPOORG**	NUMÉRICO	Identificador del tipo de organismo.
→ **CAMPO_REFTIPOORGZ**	NUMÉRICO	Identificador del tipo de organización.

Métodos para aplicación de los filtros → obtenerOrganismo

TrPais

trewa.bd.trapi.tpo.TrPais

Clase que representa la información de un país.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **CODPAIS**	String	Código del país.
→ **NOMBRE**	String	Nombre del país.
→ **DESCRIPCION**	String	Descripción del país.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_CODPAIS**	CADENA(3)	Código del país.
→ **CAMPO_NOMBRE**	CADENA(32)	Nombre del país.
→ **CAMPO_DESCRIPCION**	CADENA(64)	Descripción del país.

Métodos para aplicación de los filtros → obtenerPais

TrParametro

trewa.bd.trapi.tpo.TrParametro

Clase que representa los parámetros definidos en el sistema.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFPARAM**	TpoPK	Identificador del parámetro.
→ **NOMBRE**	String	Nombre del parámetro.
→ **DESCRIPCION**	String	Descripción del parámetro.
→ **TIPO**	String	Tipo del parámetro. → "**N**" – Numérico → "**C**" – Cadena → "**F**" – Fecha
→ **TAMANIO**	Integer	Tamaño del parámetro.
→ **STMA**	TrSistema	Sistema.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFPARAM**	NUMÉRICO	Identificador del parámetro.
→ **CAMPO_NOMBRE**	CADENA(20)	Nombre del parámetro.
→ **CAMPO_DESCRIPCION**	CADENA(100)	Descripción del parámetro.
→ **CAMPO_TIPO**	CADENA(1)	Tipo del parámetro. → "**N**" – Numérico → "**C**" – Cadena → "**F**" – Fecha
→ **CAMPO_TAMANIO**	NUMÉRICO	Tamaño del parámetro.
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema

Métodos para aplicación de los filtros → obtenerParametro

TrParametroBloque

trewa.bd.trapi.tpo.TrParametroBloque

Clase que representa un parámetro que se define para un bloque.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **PARAMETRO**	`TrParametro`	Parámetro.
→ **BLOQUE**	`TrBloque`	Bloque.
→ **ORDEN**	`Integer`	Orden del parámetro.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFPARAM**	`NUMÉRICO`	Identificador del parámetro.
→ **CAMPO_REFBLOQUE**	`NUMÉRICO`	Identificador del bloque.
→ **CAMPO_ORDEN**	`NUMÉRICO`	Orden del parámetro.

Métodos para aplicación de los filtros → `obtenerParametroBloque`

TrParametroVariable

trewa.bd.trapi.tpo.TrParametroVariable

Clase que representa un parámetro que se define para una variable.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **PARAMETRO**	`TrParametro`	Parámetro.
→ **VARIABLE**	`TrVariable`	Variable.
→ **ORDEN**	`Integer`	Orden del parámetro.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFPARAM**	`NUMÉRICO`	Identificador del parámetro.
→ **CAMPO_REFVARIABLE**	`NUMÉRICO`	Identificador de la variable.
→ **CAMPO_ORDEN**	`NUMÉRICO`	Orden del parámetro.

Métodos para aplicación de los filtros → `obtenerParametroVariable`

TrParrafoTipoDocumento

trewa.bd.trapi.tpo.TrParrafoTipoDocumento

Clase que representa un párrafo para un tipo de documento.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFPARRTIPODOC**	TpoPK	Identificador del párrafo del tipo de documento.
→ **PARRAFO**	String	Texto del párrafo.
→ **ORDEN**	Integer	Orden del párrafo dentro del documento.
→ **ALINEACION**	String	Alineación del párrafo. → "**I**" – Izquierda → "**C**" – Centrado → "**D**" – Derecha → "**J**" – Justificado
→ **ETIQUETA**	String	Etiqueta del párrafo.
→ **EDITABLE**	String	Indica si el párrafo es editable. → "**S**" – Sí → "**N**" – No
→ **ESTILO**	String	Estilo del texto del párrafo. → "**NORMAL**" – Normal → "**NEGRIT**" – Negrita → "**CURSIV**" – Cursiva → "**SUBRAY**" – Subrayado → "**NE-CU**" – Negrita y Cursiva → "**NE-SU**" – Negrita y Subrayado → "**CU-SU**" – Cursiva y Subrayado → "**NE-CU-SU**" – Negrita, Cursiva y Subrayado
→ **ESTILOETIQ**	String	Estilo de la etiqueta del párrafo. → "**NORMAL**" – Normal → "**NEGRIT**" – Negrita → "**CURSIV**" – Cursiva → "**SUBRAY**" – Subrayado → "**NE-CU**" – Negrita y Cursiva → "**NE-SU**" – Negrita y Subrayado → "**CU-SU**" – Cursiva y Subrayado → "**NE-CU-SU**" – Negrita, Cursiva y Subrayado
→ **TIPODOC**	TrTipoDocumento	Tipo de documento.
→ **TIPOPARR**	TrTipoParrafo	Tipo de párrafo.
→ **FUSIONAR**	String	Indica si se fusionan o no variables al generar.

		→ "**S**" – Sí → "**N**" – No
→ **IMAGEN**	`byte[]`	Imagen del párrafo.
→ **FORMATO**	`String`	Tipo de formato de la imagen (tipo mime).
→ **NOMBREFICHERO**	`String`	Nombre del fichero del documento físico.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFPARRTIPODOC**	`NUMÉRICO`	Identificador del párrafo del tipo de documento.
→ **CAMPO_PARRAFO**	`CADENA`	Texto del párrafo
→ **CAMPO_ORDEN**	`NUMÉRICO`	Orden del párrafo dentro del documento.
→ **CAMPO_ALINEACION**	`CADENA(1)`	Alineación del párrafo. → "**I**" – Izquierda → "**C**" – Centrado → "**D**" – Derecha → "**J**" – Justificado
→ **CAMPO_ETIQUETA**	`CADENA(15)`	Etiqueta del párrafo.
→ **CAMPO_EDITABLE**	`CADENA(1)`	Indica si el párrafo es editable. → "**S**" – Sí → "**N**" – No
→ **CAMPO_ESTILO**	`CADENA(8)`	Estilo del texto del párrafo. → "**NORMAL**" – Normal → "**NEGRIT**" – Negrita → "**CURSIV**" – Cursiva → "**SUBRAY**" – Subrayado → "**NE-CU**" – Negrita y Cursiva → "**NE-SU**" – Negrita y Subrayado → "**CU-SU**" – Cursiva y Subrayado → "**NE-CU-SU**" – Negrita, Cursiva y Subrayado
→ **CAMPO_ESTILOETIQ**	`CADENA(8)`	Estilo de la etiqueta del párrafo. → "**NORMAL**" – Normal → "**NEGRIT**" – Negrita → "**CURSIV**" – Cursiva → "**SUBRAY**" – Subrayado → "**NE-CU**" – Negrita y Cursiva → "**NE-SU**" – Negrita y Subrayado → "**CU-SU**" – Cursiva y Subrayado → "**NE-CU-SU**" – Negrita, Cursiva y Subrayado
→ **CAMPO_REFTIPODOC**	`NUMÉRICO`	Identificador el tipo de documento.
→ **CAMPO_REFTIPOPARR**	`NUMÉRICO`	Identificador del tipo de párrafo.
→ **CAMPO_FUSIONAR**	`CADENA(1)`	Indica si se fusionan o no variables al generar. → "**S**" – Sí

		→ "**N**" – No
→ **CAMPO_FORMATO**	CADENA(128)	Tipo de formato de la imagen (tipo mime)
→ **CAMPO_NOMBREFICHERO**	CADENA(64)	Nombre del fichero del documento físico

Métodos para aplicación de los filtros → obtenerParrafoTipoDocumento

TrPerfilUsuario

trewa.bd.trapi.tpo.TrPerfilUsuario

Clase que representa los perfiles de usuario que pueden existir.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFPERFILUSU**	TpoPK	Identificador del perfil de usuario.
→ **NOMBRE**	String	Nombre del perfil de usuario.
→ **DESCRIPCION**	String	Descripción del perfil de usuario.
→ **STMA**	String	Sistema del perfil de usuario.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFPERFILUSU**	NUMÉRICO	Identificador del perfil de usuario.
→ **CAMPO_NOMBRE**	CADENA(25)	Nombre del perfil de usuario.
→ **CAMPO_DESCRIPCION**	CADENA(250)	Descripción del perfil de usuario.
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema del perfil de usuario.

Métodos para aplicación de los filtros → obtenerPerfilUsuario

TrPlantilla

trewa.bd.trapi.tpo.TrPlantilla

Clase que representa las plantillas usadas en la generación de documentos.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción

→ **REFPLANTILLA**	TpoPK	Identificador de la plantilla.
→ **NOMBRE**	String	Nombre de la plantilla.
→ **DESCRIPCION**	String	Descripción de la plantilla.
→ **NOMBINFORME**	String	Nombre del informa de la plantilla.
→ **STMA**	TrSistema	Sistema de la plantilla.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFPLANTILLA**	NUMÉRICO	Identificador de la plantilla
→ **CAMPO_NOMBRE**	CADENA(15)	Nombre de la plantilla
→ **CAMPO_DESCRIPCION**	CADENA(250)	Descripción de la plantilla
→ **CAMPO_NOMBINFORME**	CADENA(15)	Nombre del informe de la plantilla
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema de la plantilla

Métodos para aplicación de los filtros → obtenerPlantilla

TrPlantillaProcedimiento

trewa.bd.trapi.tpo.TrPlantillaProcedimiento

Clase que representa las posibles plantillas del procedimiento necesarias para comenzar un expediente o que servirán a lo largo de la vida de su tramitación.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFPLANTPROC**	TpoPK	Identificador de la plantilla del procedimiento.
→ **REFDEFPROC**	TpoPK	Identificador de la definción del procedimiento.
→ **NOMBRE**	String	Nombre de la plantilla del procedimiento.
→ **DESCRIPCION**	String	Descripción de la plantilla del procedimiento.
→ **FICHERO**	byte[]	Fichero de la plantilla del procedimiento.
→ **NOMBREFICHERO**	String	Nombre del fichero físico de la plantilla.
→ **FORMATO**	String	Tipo de formato del fichero (tipo mime).

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFPLANTPROC**	NUMÉRICO	Identificador de la plantilla del procedimiento.
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definción del procedimiento.

→	**CAMPO_NOMBRE**	CADENA(100)	Nombre de la plantilla del procedimiento.
→	**CAMPO_DESCRIPCION**	CADENA(250)	Descripción de la plantilla del procedimiento.
→	**CAMPO_NOMBREFICHERO**	CADENA(64)	Nombre del fichero físico de la plantilla.
→	**CAMPO_FORMATO**	CADENA(128)	Tipo de formato del fichero (tipo mime).

Métodos para aplicación de los filtros → obtenerPlantillaProcedimiento

TrProvincia

trewa.bd.trapi.tpo.TrProvincia

Clase que representa los datos de una provincia.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**CODPROVINCIA**	String	Código de la provincia.
→	**NOMBRE**	String	Nombre de la provincia.
→	**DESCRIPCION**	String	Descripción de la provincia.

Campos definidos para filtrados y ordenación

	Campo	Tipo	Descripción y restricciones de uso
→	**CAMPO_CODPROVINCIA**	CADENA(2)	Código de la provincia.
→	**CAMPO_NOMBRE**	CADENA(32)	Nombre de la provincia.
→	**CAMPO_DESCRIPCION**	CADENA(64)	Descripción de la provincia.

Métodos para aplicación de los filtros → obtenerProvincia

TrPtoTrabOrganismo

trewa.bd.trapi.tpo.TrPtoTrabOrganismo

Clase que representa los distintos puestos de trabajo existentes en cada organismo.

Atributos accesibles mediante métodos get/set

	Atributo	Tipo	Descripción
→	**ORGANISMO**	TrOrganismo	Organismo.

→ **PUESTOTRABAJO** TrPuestoTrabajo Puesto de trabajo.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_CODPTOTRAB**	CADENA(10)	Código del puesto de trabajo.
→ **CAMPO_NOMBREPTOTRAB**	CADENA(32)	Nombre del puesto de trabajo.
→ **CAMPO_DESCRIPCIONPTOTRAB**	CADENA(64)	Descripción del puesto de trabajo.
→ **CAMPO_REFORGANISMO**	NUMÉRICO	Identificador del organismo.
→ **CAMPO_CODMUNICIPIO**	CADENA(3)	Código del municipio.
→ **CAMPO_CODPROVINCIA**	CADENA(2)	Código de la provincia.
→ **CAMPO_TIPOVIA**	CADENA(10)	Código del tipo de vía.
→ **CAMPO_CODORG**	CADENA(16)	Código del organismo
→ **CAMPO_NOMBRE**	CADENA(32)	Nombre del organismo.
→ **CAMPO_DESCRIPCION**	CADENA(200)	Descripción del organismo.
→ **CAMPO_TIPO**	CADENA(2)	Tipo de organismo. → "**DC**" – Delegación Provincial → "**SC**" – Servicios Centrales → "**CE**" – Centro Adscrito → "**EE**" – Empresa Externa
→ **CAMPO_NOMBREVIA**	CADENA(64)	Nombre de la vía pública del domicilio.
→ **CAMPO_NUMERO**	NUMÉRICO	Número del domicilio.
→ **CAMPO_LETRA**	CADENA(2)	Letra del domicilio.
→ **CAMPO_ESCALERA**	CADENA(2)	Escalera del domicilio.
→ **CAMPO_PISO**	NUMÉRICO	Piso del domicilio.
→ **CAMPO_PUERTA**	CADENA(2)	Puerta del domicilio.
→ **CAMPO_CODPOSTAL**	NUMÉRICO	Código postal del domicilio.
→ **CAMPO_TELEFONO**	CADENA(25)	Número(s) de teléfono de contacto.
→ **CAMPO_TLFMOVIL**	CADENA(25)	Número(s) de teléfono móvil.
→ **CAMPO_FAX**	CADENA(25)	Número(s) de Fax.
→ **CAMPO_EMAIL**	CADENA(64)	Correo electrónico.
→ **CAMPO_CIWA**	CADENA(15)	Código de identificación w@ndA.
→ **CAMPO_IDARIES**	CADENA(5)	Identificador de aries.
→ **CAMPO_IDENTIFICADOR**	CADENA(15)	Número de identificación (CIF).
→ **CAMPO_DIGCONTROL**	CADENA(1)	Dígito de control para el número de identificación.
→ **CAMPO_FECHAINIVIG**	DATE	Fecha de inicio de vigencia del organismo.
→ **CAMPO_FECHAFINVIG**	DATE	Fecha de fin de vigencia del organismo
→ **CAMPO_REFORGPADRE**	NUMÉRICO	Identificador del organismo superior al que pertenece.
→ **CAMPO_REFTIPOORG**	NUMÉRICO	Identificador del tipo de organismo.
→ **CAMPO_REFTIPOORGZ**	NUMÉRICO	Identificador del tipo de organización.

Métodos para aplicación de los filtros → obtenerPtoTrabOrganismo

TrPuestoTrabajo

trewa.bd.trapi.tpo.TrPuestoTrabajo

Clase que representa los distintos puesto de trabajo.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **CODPTOTRAB**	String	Código del puesto de trabajo.
→ **NOMBRE**	String	Nombre del puesto de trabajo.
→ **DESCRIPCION**	String	Descripción del puesto de trabajo.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_CODPTOTRAB**	CADENA(10)	Código del puesto de trabajo.
→ **CAMPO_NOMBRE**	CADENA(32)	Nombre del puesto de trabajo.
→ **CAMPO_DESCRIPCION**	CADENA(64)	Descripción del puesto de trabajo.

Métodos para aplicación de los filtros → obtenerPuestoTrabajo

TrRazonInteres

trewa.bd.trapi.tpo.TrRazonInteres

Clase que representa las razones de interés de los interesados.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFRAZONINT**	TpoPK	Identificador de la razón de interés.
→ **ABREVIATURA**	String	Abreviatura de la razón de interés.
→ **DESCRIPCION**	String	Descripción de la razón de interés.
→ **OBSOLETO**	String	Indica si la razón de interés está obsoleta. → "**S**" – Sí → "**N**" – No
→ **CODWANDA**	String	Valor en w@ndA de la razón de interés

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFRAZONINT**	NUMÉRICO	Identificador de la razón de interés.
→ **CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura de la razón de interés.
→ **CAMPO_DESCRIPCION**	CADENA(100)	Descripción de la razón de interés.
→ **CAMPO_OBSOLETO**	CADENA(1)	Indica si la razón de interés está obsoleta. → "**S**" – Sí → "**N**" – No
→ **CAMPO_CODWANDA**	CADENA(10)	Valor en w@ndA de la razón de interés

Métodos para aplicación de los filtros → obtenerRazonInteres

TrRelacion

trewa.bd.trapi.tpo.TrRelacion

Clase que representa las relaciones que se establecen entre fases, transiciones y tipos de documentos.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFRELACION**	TpoPK	Identificador de la relación
→ **FASEA**	TrFase	Fase "A".
→ **FASEB**	TrFase	Fase "B".
→ **TRANSICION**	TrTransicion	Transición.
→ **TIPODOC**	TrTipoDocumento	Tipo de documento.
→ **TIPORELA**	TrTipoRelacion	Tipo de relación.
→ **DEFPROC**	TrDefProcedimiento	Definición del procedimiento.
→ **DESCRIPCION**	String	Descripción de la relación.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFRELACION**	NUMÉRICO	Identificador de la relación.
→ **CAMPO_REFFASEA**	NUMÉRICO	Identificador de la fase "A".
→ **CAMPO_REFFASEB**	NUMÉRICO	Identificador de la fase "B".
→ **CAMPO_REFTRANSICION**	NUMÉRICO	Identificador de la transición.
→ **CAMPO_REFTIPODOC**	NUMÉRICO	Identificador del tipo de documento.

→ **CAMPO_REFTIPORELA**	NUMÉRICO	Identificador del tipo de relación.
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Definición del procedimiento.
→ **CAMPO_DESCRIPCION**	CADENA(200)	Descripción de la relación.

Métodos para aplicación de los filtros → obtenerRelacion

TrSistema

trewa.bd.trapi.tpo.TrSistema

Clase que representa la información de un sistema.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFSTMA**	TpoPK	Identificador del sistema.
→ **CODSTMA**	String	Código del sistema.
→ **DESCRIPCION**	String	Descripción del sistema.
→ **CONEXIONBUS**	String	Indica si se tiene conexión al bus. → "**S**" – Sí → "**N**" – No
→ **COMPONENTEINF**	TrComponente	Componente a informar.
→ **COMPONENTEAVI**	TrComponente	Componente del avisador.
→ **COMPONENTEGUAR**	TrComponente	Componente que guarda los documentos.
→ **COMPONENTEARCH**	TrComponente	Componente que archiva.
→ **COMPONENTEREG**	TrComponente	Componente que registra los documentos.
→ **COMPONENTEFIRMA**	TrComponente	Componente que firma los documentos.
→ **COMPONENTENOTI**	TrComponente	Componente que notifica los documentos.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema
→ **CAMPO_CODSTMA**	CADENA(10)	Código del sistema
→ **CAMPO_DESCRIPCION**	CADENA(128)	Descripción del sistema
→ **CAMPO_CONEXIONBUS**	CADENA(1)	Indica si se tiene conexión al bus. → "**S**" – Sí → "**N**" – No
→ **CAMPO_REFCOMPINF**	NUMÉRICO	Identificador del componente a informar.

→ **CAMPO_REFCOMPAVI**	NUMÉRICO	Identificador del componente del avisador.
→ **CAMPO_REFCOMPGUAR**	NUMÉRICO	Identificador del componente que guarda los documentos.
→ **CAMPO_REFCOMPARCH**	NUMÉRICO	Identificador del componente que archiva.
→ **CAMPO_REFCOMPREG**	NUMÉRICO	Identificador del componente que registra los documentos.
→ **CAMPO_REFCOMPFIRMA**	NUMÉRICO	Identificador del componente que firma los documentos.
→ **CAMPO_REFCOMPNOTI**	NUMÉRICO	Identificador del componente que firma los documentos.

Métodos para aplicación de los filtros → obtenerSistema

TrTextoDisposicionFirma

trewa.bd.trapi.tpo.TrTextoDisposicionFirma

Clase que representa los textos para las firmas.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFTEXTODISP**	TpoPK	Identificador del texto de la disposición.
→ **DESCRIPCION**	String	Descripción de la disposición.
→ **TIPO**	String	Tipo de disposición → "**D**" – Decreto → "**T**" – Otros → "**O**" – Orden

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFTEXTODISP**	NUMÉRICO	Identificador del texto de la disposición.
→ **CAMPO_DESCRIPCION**	CADENA(100)	Descripción de la disposición.
→ **CAMPO_TIPO**	CADENA(1)	Tipo de disposición → "**D**" – Decreto → "**T**" – Otros → "**O**" – Orden

Métodos para aplicación de los filtros → obtenerTextoDisposicionFirma

TrTipoActo

trewa.bd.trapi.tpo.TrTipoActo

Clase que representa los posibles tipos de actos administrativos que pueden existir en un procedimiento administrativo.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFTIPOACTO**	TpoPK	Identificador del tipo de acto.
→ **ABREVIATURA**	String	Abreviatura del tipo de acto.
→ **DESCRIPCION**	String	Descripción del tipo de acto.
→ **STMA**	TrSistema	Sistema del tipo de acto.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFTIPOACTO**	NUMÉRICO	Identificador del tipo de acto.
→ **CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura del tipo de acto.
→ **CAMPO_DESCRIPCION**	CADENA(200)	Descripción del tipo de acto.
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema del tipo de acto.

Métodos para aplicación de los filtros → obtenerTipoActo

TrTipoComponente

trewa.bd.trapi.tpo.TrTipoComponente

Clase que representa los distintos tipos de componentes que pueden exisitir en el ámbito w@ndA.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFTIPOCOMP**	TpoPK	Identificador del tipo de componente.
→ **ABREVIATURA**	String	Abreviatura del tipo de componente.
→ **DESCRIPCION**	String	Descripción del tipo de componente.
→ **CODWANDA**	Integer	Valor en w@ndA del tipo de componente.
→ **OBSOLETO**	String	Indica si el tipo de componente está obsoleto. → "**S**" – Sí → "**N**" – No

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFTIPOCOMP**	NUMÉRICO	Identificador del tipo de componente
→ **CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura del tipo de componente
→ **CAMPO_DESCRIPCION**	CADENA(50)	Descripción del tipo de componente
→ **CAMPO_CODWANDA**	NUMÉRICO	Valor en w@ndA del tipo de componente
→ **CAMPO_OBSOLETO**	CADENA(1)	Indica si el tipo de componente está obsoleto. → "**S**" – Sí → "**N**" – No

Métodos para aplicación de los filtros → obtenerTipoComponente

TrTipoContacto

trewa.bd.trapi.tpo.TrTipoContacto

Clase que representa los distintos tipos de contacto que pueden existir entre ciudadanos.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFTIPOCONT**	TpoPK	Identificador del tipo de contacto.
→ **ABREVIATURA**	String	Abreviatura del tipo de contacto.
→ **DESCRIPCION**	String	Descripción del tipo de contacto.
→ **CODWANDA**	String	Valor en w@ndA del tipo de contacto
→ **OBSOLETO**	String	Indica si el tipo de contacto está obsoleto. → "**S**" – Sí → "**N**" – No

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFTIPOCONT**	NUMÉRICO	Identificador del tipo de contacto
→ **CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura del tipo de contacto
→ **CAMPO_DESCRIPCION**	CADENA(50)	Descripción del tipo de contacto
→ **CAMPO_CODWANDA**	NUMÉRICO	Valor en w@ndA del tipo de contacto
→ **CAMPO_OBSOLETO**	CADENA(1)	Indica si el tipo de contacto está obsoleto. → "**S**" – Sí → "**N**" – No

Métodos para aplicación de los filtros → obtenerTipoContacto

TrTipoDocumento

trewa.bd.trapi.tpo.TrTipoDocumento

Clase que representa los tipos de documentos que se pueden dar en cualquier procedimiento.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFTIPODOC**	TpoPK	Identificador del tipo de documento.
→ **ETIQUETA**	String	Etiqueta del tipo de documento.
→ **NOMBRE**	String	Nombre del tipo de documento.
→ **DESCRIPCION**	String	Descripción del tipo de documento.
→ **ENTRADASALIDA**	String	Indica si el documento es de entrada o salida. → "**E**" – Entrada → "**S**" – Salida → "**ES**" – Entrada/Salida
→ **INCGEN**	String	Indica si el documento es generado o incorporado. → "**I**" – Incorporado → "**G**" – Generado
→ **MULTIPLE**	String	Indica si el tipo de documento representa a documentos múltiples. → "**S**" – Sí → "**N**" – No
→ **TEXTOAUXILIAR**	String	Texto auxiliar.
→ **FECHAFIRMA**	String	Indica si se muestra la fecha en el pie de firma. → "**S**" – Sí → "**N**" – No
→ **STMA**	TrSistema	Sistema.
→ **PLANTILLA**	TrPlantilla	Plantilla.
→ **INFORMAR**	String	Indica si se informa al bus de los documento de este tipo. → "**S**" – Sí → "**N**" – No
→ **ARCHIVABLE**	String	Indica si se permite archivar los documentos de este tipo. → "**S**" – Sí → "**N**" – No
→ **VERSIONABLE**	String	Indica si se versionan los documentos de este tipo. → "**S**" – Sí

		→ "**N**" – No
→ **REUTILIZABLE**	String	Indica si se permite reutilizar en otros expedientes los documentos de este tipo. → "**S**" – Sí → "**N**" – No
→ **MODOGEN**	String	Modo de generación. → "**R**" – Report Server Oracle → "**P**" – Generación PDF → "**O**" – Office Bean
→ **FUSIONAR**	String	Indica si se fusionan las variables al generar. → "**S**" – Sí → "**N**" – No
→ **FIRMADIGI**	String	Indica si se puede firmar digitalmente. → "**S**" – Sí → "**N**" – No
→ **TIPOFIRMA**	String	Tipo de firma para el documento. → "**C**" – Cascada → "**P**" – Paralelo
→ **PLANTILLAOFFICE**	byte[]	Plantilla binaria para Open Office.
→ **FORMATO**	String	Tipo de formato de la plantilla (tipo mime).
→ **NOMBREFICHERO**	String	Nombre del fichero de la plantilla.
→ **REGISTRABLE**	String	Indica si se puede registrar el documento. → "**S**" – Sí → "**N**" – No
→ **NOTIFICABLE**	String	Indica si se notifica. → "**S**" – Sí → "**N**" – No
→ **OBSOLETO**	String	Indica si el documento está obsoleto. → "**S**" – Sí → "**N**" – No
→ **CODWANDA**	String	Valor en w@ndA del tipo de documento.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFTIPODOC**	NUMÉRICO	Identificador del tipo de documento.
→ **CAMPO_ETIQUETA**	CADENA(11)	Etiqueta del tipo de documento.
→ **CAMPO_NOMBRE**	CADENA(50)	Nombre del tipo de documento.
→ **CAMPO_DESCRIPCION**	CADENA(250)	Descripción del tipo de documento.
→ **CAMPO_ENTRADASALIDA**	CADENA(2)	Indica si el documento es de entrada o salida. → "**E**" – Entrada → "**S**" – Salida → "**ES**" – Entrada/Salida

→ **CAMPO_INCGEN**	CADENA(1)	Indica si el documento es generado o incorporado. → "**I**" – Incorporado → "**G**" – Generado
→ **CAMPO_MULTIPLE**	CADENA(1)	Indica si el tipo de documento representa a documentos múltiples. → "**S**" – Sí → "**N**" – No
→ **CAMPO_TEXTOAUXILIAR**	CADENA(50)	Texto auxiliar.
→ **CAMPO_FECHAFIRMA**	CADENA(1)	Indica si se muestra la fecha en el pie de firma. → "**S**" – Sí → "**N**" – No
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema
→ **CAMPO_REFPLANTILLA**	NUMÉRICO	Identificador de la plantilla
→ **CAMPO_INFORMAR**	CADENA(1)	Indica si se informa al bus de los documento de este tipo. → "**S**" – Sí → "**N**" – No
→ **CAMPO_ARCHIVABLE**	CADENA(1)	Indica si se permite archivar los documentos de este tipo. → "**S**" – Sí → "**N**" – No
→ **CAMPO_VERSIONABLE**	CADENA(1)	Indica si se versionan los documentos de este tipo. → "**S**" – Sí → "**N**" – No
→ **CAMPO_REUTILIZABLE**	CADENA(1)	Indica si se permite reutilizar en otros expedientes los documentos de este tipo. → "**S**" – Sí → "**N**" – No
→ **CAMPO_MODOGEN**	CADENA(1)	Modo de generación. → "**R**" – Report Server Oracle → "**P**" – Generación PDF → "**O**" – Office Bean
→ **CAMPO_FUSIONAR**	CADENA(1)	Indica si se fusionan las variables al generar. → "**S**" – Sí → "**N**" – No
→ **CAMPO_FIRMADIGI**	CADENA(1)	Indica si se puede firmar digitalmente. → "**S**" – Sí → "**N**" – No
→ **CAMPO_TIPOFIRMA**	CADENA(1)	Tipo de firma para el documento. → "**C**" – Cascada → "**P**" – Paralelo
→ **CAMPO_FORMATO**	CADENA(128)	Tipo de formato de la plantilla (tipo mime).
→ **CAMPO_NOMBREFICHERO**	CADENA(64)	Nombre del fichero de la plantilla.
→ **CAMPO_REGISTRABLE**	CADENA(1)	Indica si se puede registrar el documento. → "**S**" – Sí → "**N**" – No
→ **CAMPO_NOTIFICABLE**	CADENA(1)	Indica si se notifica.

		→ "**S**" – Sí → "**N**" – No
→ **CAMPO_OBSOLETO**	CADENA(1)	Indica si el documento está obsoleto. → "**S**" – Sí → "**N**" – No
→ **CAMPO_CODWANDA**	CADENA(30)	Valor en w@ndA del tipo de documento.

Métodos para aplicación de los filtros → obtenerTipoDocumento

TrTipoExpediente

trewa.bd.trapi.tpo.TrTipoExpediente

Clase que representa un tipo de expediente.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFTIPOEXP**	TpoPK	Identificador del tipo de expediente.
→ **ABREVIATURA**	String	Abreviatura del tipo de expediente.
→ **DESCRIPCION**	String	Descripción del tipo de expediente.
→ **VIGENTE**	String	Indica si el tipo de expediente está vigente. → "**S**" – Sí → "**N**" – No
→ **STMA**	TrSistema	Sistema del tipo de expediente.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFTIPOEXP**	NUMÉRICO	Identificador del tipo de expediente.
→ **CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura del tipo de expediente.
→ **CAMPO_DESCRIPCION**	CADENA(50)	Descripción del tipo de expediente.
→ **CAMPO_VIGENTE**	CADENA(1)	Indica si el tipo de expediente está vigente. → "**S**" – Sí → "**N**" – No
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema del tipo de expediente.

Métodos para aplicación de los filtros → obtenerTipoExpediente

TrTipoIdentificador

trewa.bd.trapi.tpo.TrTipoIdentificador

Clase que representa los distintos tipos de identificadores (DNI, NIF...).

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **CODTIPOIDENT**	String	Código del tipo de identificador.
→ **DESCRIPCION**	String	Descripción del tipo de identificador.
→ **TIPOIDENT**	String	Indica si se trata de una persona física o jurídica. → "**F**" – Física → "**J**" – Jurídica
→ **CODWANDA**	String	Valor en w@ndA del tipo de identificador
→ **OBSOLETO**	String	Indica si el tipo de identificador está obsoleto. → "**S**" – Sí → "**N**" – No

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_CODTIPOIDENT**	NUMÉRICO	Código del tipo de identificador.
→ **CAMPO_DESCRIPCION**	CADENA(10)	Descripción del tipo de identificador.
→ **CAMPO_TIPOIDENT**	CADENA(1)	Indica si se trata de una persona física o jurídica. → "**F**" – Física → "**J**" – Jurídica
→ **CAMPO_CODWANDA**	CADENA(30)	Valor en w@ndA del tipo de identificador
→ **CAMPO_OBSOLETO**	CADENA(1)	Indica si el tipo de identificador está obsoleto. → "**S**" – Sí → "**N**" – No

Métodos para aplicación de los filtros → obtenerTipoIdentificador

TrTipoIndicacion

trewa.bd.trapi.tpo.TrTipoIndicacion

Clase que representa los tipos de datos variables sobre la ficha descriptiva del procedimiento.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFTIPOIND**	TpoPK	Identificador del tipo de indicación de la ficha.
→ **ABREVIATURA**	String	Abreviatura del tipo de indicación de la ficha descriptiva del procedimiento.
→ **DESCRIPCION**	String	Descripción del tipo de indicación de la ficha descriptiva del procedimiento.
→ **CODWANDA**	String	Valor en w@ndA del tipo de indicación de la ficha descriptiva del procedimiento.
→ **OBSOLETO**	String	Indica si el tipo de indicación está obsoleto. → "**S**" – Sí → "**N**" – No

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFTIPOIND**	NUMÉRICO	Identificador del tipo de indicación de la ficha.
→ **CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura del tipo de indicación de la ficha descriptiva del procedimiento.
→ **CAMPO_DESCRIPCION**	CADENA(100)	Descripción del tipo de indicación de la ficha descriptiva del procedimiento.
→ **CAMPO_CODWANDA**	CADENA(30)	Valor en w@ndA del tipo de indicación de la ficha descriptiva del procedimiento.
→ **CAMPO_OBSOLETO**	CADENA(1)	Indica si el tipo de indicación está obsoleto. → "**S**" – Sí → "**N**" – No

Métodos para aplicación de los filtros → obtenerTipoIndicacion

TrTipoNormativa

trewa.bd.trapi.tpo.TrTipoNormativa

Clase que representa la información de los distintos tipos de normativas.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFTIPONORM**	TpoPK	Identificador del tipo de normativa.
→ **ABREVIATURA**	String	Abreviatura del tipo de normativa.

→ **DESCRIPCION**	String	Descripción del tipo de normativa.
→ **CODWANDA**	String	Valor en w@ndA del tipo de normativa.
→ **OBSOLETO**	String	Indica si el tipo de normativa está obsoleto. → "**S**" – Sí → "**N**" – No

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFTIPONORM**	NUMÉRICO	Identificador del tipo de normativa.
→ **CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura del tipo de normativa.
→ **CAMPO_DESCRIPCION**	CADENA(100)	Descripción del tipo de normativa.
→ **CAMPO_CODWANDA**	CADENA(30)	Valor en w@ndA del tipo de normativa.
→ **CAMPO_OBSOLETO**	CADENA(1)	Indica si el tipo de normativa está obsoleto. → "**S**" – Sí → "**N**" – No

Métodos para aplicación de los filtros → obtenerTipoNormativa

TrTipoOrganismo

trewa.bd.trapi.tpo.TrTipoOrganismo

Clase que representa los distintos tipos de organismos.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFTIPOORG**	TpoPK	Identificador del tipo de organismo.
→ **ABREVIATURA**	String	Abreviatura del tipo de organismo.
→ **DESCRIPCION**	String	Descripción del tipo de organismo.
→ **CODWANDA**	String	Valor en w@ndA del tipo de organismo
→ **OBSOLETO**	String	Indica si el tipo de organismo está obsoleto. → "**S**" – Sí → "**N**" – No

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFTIPOORG**	NUMÉRICO	Identificador del tipo de organismo.

→ **CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura del tipo de organismo.
→ **CAMPO_DESCRIPCION**	CADENA(100)	Descripción del tipo de organismo.
→ **CAMPO_CODWANDA**	CADENA(1)	Valor en w@ndA del tipo de organismo.
→ **CAMPO_OBSOLETO**	CADENA(1)	Indica si el tipo de organismoestá obsoleto. → "**S**" – Sí → "**N**" – No

Métodos para aplicación de los filtros → obtenerTipoOrganismo

TrTipoOrganizacion

trewa.bd.trapi.tpo.TrTipoOrganizacion

Clase que representa los distintos tipos de organismos.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **CODTIPOORGZ**	String	Código del tipo de organización.
→ **DESCRIPCION**	String	Descripción del tipo de organización.
→ **VIGENTE**	String	Indica si el tipo de organización está vigente. → "**S**" – Sí → "**N**" – No

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_CODTIPOORGZ**	CADENA(1)	Código del tipo de organización.
→ **CAMPO_DESCRIPCION**	CADENA(100)	Descripción del tipo de organización.
→ **CAMPO_VIGENTE**	CADENA(1)	Indica si el tipo de organización está vigente. → "**S**" – Sí → "**N**" – No

Métodos para aplicación de los filtros → obtenerTipoOrganizacion

TrTipoParrafo

trewa.bd.trapi.tpo.TrTipoParrafo

Clase que representa un tipo de párrafo para los documentos.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFTIPOPARR**	TpoPK	Identificador del tipo de párrafo.
→ **ABREVIATURA**	String	Abreviatura del tipo de párrafo.
→ **DESCRIPCION**	String	Descripción del tipo de párrafo.
→ **PARRAFO**	String	Contenido del tipo de párrafo.
→ **ALINEACION**	String	Alineación del tipo de párrafo. → "**I**" – Izquierda → "**C**" – Centrado → "**D**" – Derecha → "**J**" – Justificado
→ **ETIQUETA**	String	Etiqueta del tipo de párrafo.
→ **ESTILO**	String	Estilo del tipo del párrafo. → "**NORMAL**" – Normal → "**NEGRIT**" – Negrita → "**CURSIV**" – Cursiva → "**SUBRAY**" – Subrayado → "**NE-CU**" – Negrita y Cursiva → "**NE-SU**" – Negrita y Subrayado → "**CU-SU**" – Cursiva y Subrayado → "**NE-CU-SU**" – Negrita, Cursiva y Subrayado
→ **ESTILOETIQ**	String	Estilo de la etiqueta del tipo de párrafo. → "**NORMAL**" – Normal → "**NEGRIT**" – Negrita → "**CURSIV**" – Cursiva → "**SUBRAY**" – Subrayado → "**NE-CU**" – Negrita y Cursiva → "**NE-SU**" – Negrita y Subrayado → "**CU-SU**" – Cursiva y Subrayado → "**NE-CU-SU**" – Negrita, Cursiva y Subrayado
→ **UBICACIÓN**	String	Ubicación del tipo de párrafo. → "**A**" – Cabecera → "**C**" – Cuerpo → "**P**" – Pie → "**O**" – Otro
→ **EDITABLE**	String	Indica si el tipo de párrafo es editable. → "**S**" – Sí → "**N**" – No
→ **IMAGEN**	byte[]	Imagen del tipo de párrafo
→ **STMA**	TrSistema	Sistema.
→ **NOMBREFICHERO**	String	Nombre del fichero de la imagen física.
→ **FORMATO**	String	Tipo de formato de la imagen (tipo mime).

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFTIPOPARR**	NUMÉRICO	Identificador del tipo de párrafo.
→ **CAMPO_ABREVIATURA**	CADENA(15)	Abreviatura del tipo de párrafo.
→ **CAMPO_DESCRIPCION**	CADENA(250)	Descripción del tipo de párrafo.
→ **CAMPO_PARRAFO**	CADENA(4000)	Contenido del tipo de párrafo.
→ **CAMPO_ALINEACION**	CADENA(1)	Alineación del tipo de párrafo. → "**I**" – Izquierda → "**C**" – Centrado → "**D**" – Derecha → "**J**" – Justificado
→ **CAMPO_ETIQUETA**	CADENA(15)	Etiqueta del tipo de párrafo.
→ **CAMPO_ESTILO**	CADENA(8)	Estilo del tipo del párrafo. → "**NORMAL**" – Normal → "**NEGRIT**" – Negrita → "**CURSIV**" – Cursiva → "**SUBRAY**" – Subrayado → "**NE-CU**" – Negrita y Cursiva → "**NE-SU**" – Negrita y Subrayado → "**CU-SU**" – Cursiva y Subrayado → "**NE-CU-SU**" – Negrita, Cursiva y Subrayado
→ **CAMPO_ESTILOETIQ**	CADENA(8)	Estilo de la etiqueta del tipo de párrafo. → "**NORMAL**" – Normal → "**NEGRIT**" – Negrita → "**CURSIV**" – Cursiva → "**SUBRAY**" – Subrayado → "**NE-CU**" – Negrita y Cursiva → "**NE-SU**" – Negrita y Subrayado → "**CU-SU**" – Cursiva y Subrayado → "**NE-CU-SU**" – Negrita, Cursiva y Subrayado
→ **CAMPO_UBICACION**	CADENA(1)	Ubicación del tipo de párrafo. → "**A**" – Cabecera → "**C**" – Cuerpo → "**P**" – Pie → "**O**" – Otro
→ **CAMPO_EDITABLE**	CADENA(1)	Indica si el tipo de párrafo es editable. → "**S**" – Sí → "**N**" – No
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema.
→ **CAMPO_NOMBREFICHERO**	CADENA(64)	Nombre del fichero de la imagen física.
→ **CAMPO_FORMATO**	CADENA(128)	Tipo de formato de la imagen (tipo mime).

Métodos para aplicación de los filtros → obtenerTipoParrafo

TrTipoPublicacion

trewa.bd.trapi.tpo.TrTipoPublicacion

Clase que representa la información de los distintos tipos de publicación de las normativas.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFTIPOPUB**	TpoPK	Identificador del tipo de publicación de la normativa
→ **ABREVIATURA**	String	Abreviatura del tipo de publicación.
→ **DESCRIPCION**	String	Descripción del tipo de publicación.
→ **CODWANDA**	Integer	Valor en w@ndA del tipo de publicación
→ **OBSOLETO**	String	Indica si el tipo de publicación está obsoleto. → "**S**" – Sí → "**N**" – No

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFTIPOPUB**	NUMÉRICO	Identificador del tipo de publicación de la normativa
→ **CAMPO_ABREVIATURA**	CADENA(10)	Abreviatura del tipo de publicación.
→ **CAMPO_DESCRIPCION**	CADENA(100)	Descripción del tipo de publicación.
→ **CAMPO_CODWANDA**	NUMÉRICO	Valor en w@ndA del tipo de publicación
→ **CAMPO_OBSOLETO**	CADENA(50)	Indica si el tipo de publicación está obsoleto. → "**S**" – Sí → "**N**" – No

Métodos para aplicación de los filtros → obtenerTipoPublicacion

TrTipoRelacion

trewa.bd.trapi.tpo.TrTipoRelacion

Clase que representa las posibles relaciones a establecer.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFTIPORELA**	TpoPK	Identificador del tipo de relación.
→ **NOMBRE**	String	Nombre del tipo de relación.q
→ **DESCRIPCION**	String	Descripción del tipo de relación.
→ **STMA**	TrSistema	Sistema del tipo de relación.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFTIPORELA**	NUMÉRICO	Identificador del tipo de relación.
→ **CAMPO_NOMBRE**	CADENA(25)	Nombre del tipo de relación.q
→ **CAMPO_DESCRIPCION**	CADENA(250)	Descripción del tipo de relación.
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema del tipo de relación.

Métodos para aplicación de los filtros → obtenerTipoRelacion

TrTipoVia

trewa.bd.trapi.tpo.TrTipoVia

Clase que representa los distintos tipos de vía.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **CODTIPOVIA**	String	Código del tipo de vía.
→ **DESCRIPCION**	String	Descripción del tipo de vía.
→ **CODWANDA**	String	Valor en w@ndA del tipo de vía
→ **OBSOLETO**	String	Indica si el tipo de vía está obsoleto. → "**S**" – Sí → "**N**" – No

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_CODTIPOVIA**	CADENA(10)	Código del tipo de vía.
→ **CAMPO_DESCRIPCION**	CADENA(32)	Descripción del tipo de vía.

→ **CAMPO_CODWANDA**	CADENA(15)	Valor en w@ndA del tipo de vía
→ **CAMPO_OBSOLETO**	CADENA(1)	Indica si el tipo de vía está obsoleto. → "**S**" – Sí → "**N**" – No

Métodos para aplicación de los filtros → obtenerTipoVia

TrTransicion

trewa.bd.trapi.tpo.TrTransicion

Clase que representa una transición

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFTRANSICION**	TpoPK	Identificador de la transición
→ **ETIQUETA**	String	Etiqueta de la transición
→ **ORDEN**	Integer	Orden entre las posibles transiciones que se pueden dar desde una determinada fase.
→ **DESCRIPCION**	String	Descripción de la transición.
→ **DESCFECHA**	String	Descripción de lo que significa la fecha de realización de la transición.
→ **TIPO**	String	Indica el tipo de la transición. → "**FA**" – Fase normal → "**I**" – Fase inicial → "**D**" – División → "**U**" – Unión → "**F**" – Fase final → "**ES**" – Evento salida, abandona la fase actual. → "**EN**" – Evento que no abandona la fase actual.
→ **VALIDA**	String	Indica si la transición es válida → "**S**" – Sí → "**N**" – No
→ **FASEINI**	TrFase	Fase inicial
→ **FASEFIN**	TrFase	Fase final
→ **REFTRANSPADRE**	TpoPK	Identificador de la transición padre
→ **TIPOACTO**	TrTipoActo	Tipo de acto.
→ **INFORMAR**	String	Indica si se informa al bus → "**S**" – Sí → "**N**" – No
→ **ETIQLARGA**	String	Etiqueta larga de la transición

TrTransicionDefProcedimiento

trewa.bd.trapi.tpo.TrTransicionDefProcedimiento

Clase que representa una transición asociada a un procedimiento.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFDEFPROC**	TpoPK	Identificador de la definición de procedimiento.
→ **TRANSICION**	TrTransicion	Transición.
→ **NUMMAX**	Integer	Número máximo de veces por la que se puede pasar por una transición en un procedimiento.
→ **UNIDAD**	String	Unidad de medida. → "**D**" – Días → "**M**" – Meses → "**A**" – Años
→ **NUMUNIDADES**	String	Número unidades: días, meses o años.
→ **DESCFECHALIM**	String	Descripción del significado de la fecha límite de estancia en la fase.
→ **REFTRANPROV**	TpoPK	Identificador de la transición provocada por la fecha límite

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición de procedimiento
→ **CAMPO_REFTRANSICION**	NUMÉRICO	Identificador de la transición
→ **CAMPO_NUMMAX**	NUMÉRICO	Número máximo de veces por la que se puede pasar por una transición en un procedimiento
→ **CAMPO_UNIDAD**	CADENA(1)	Unidad de medida. → "**D**" – Días → "**M**" – Meses → "**A**" – Años
→ **CAMPO_NUMUNIDADES**	NUMÉRICO	Número unidades: días, meses o años.
→ **CAMPO_DESCFECHALIM**	CADENA(50)	Descripción del significado de la fecha límite de estancia en la fase
→ **CAMPO_REFTRANPROV**	NUMÉRICO	Identificador de la transición provocada por la fecha límite
→ **CAMPO_ETIQUETA**	CADENA(11)	Etiqueta de la transición
→ **CAMPO_ORDEN**	NUMÉRICO	Orden entre las posibles transiciones que se pueden dar desde una determinada fase.
→ **CAMPO_DESCTRANSICION**	CADENA(50)	Descripción de la transición.

→ **CAMPO_DESCFECHA**	CADENA(50)	Descripción de lo que significa la fecha de realización de la transición
→ **CAMPO_TIPO**	CADENA(2)	Tipo de la transición. → "**N**" – Normal → "**D**" – División → "**U**" – Unión → "**ES**" – Evento de salida, abandona la fase actual. → "**EN**" – Evento que no provoca la salida de la fase actual.
→ **CAMPO_VALIDA**	CADENA(1)	Indica si la transición es válida o no. → "**S**" – Sí es válida → "**N**" – No es válida.
→ **CAMPO_REFFASEINI**	NUMÉRICO	Identificador de la fase inicial de la transición.
→ **CAMPO_REFSTMAFASEINI**	NUMÉRICO	Identificador del sistema de la fase inicial de la transición.
→ **CAMPO_NOMBREFASEINI**	CADENA(50)	Nombre de la fase inicial de la transición.
→ **CAMPO_REFFASEFIN**	NUMÉRICO	Identificador de la fase final de la transición.
→ **CAMPO_REFSTMAFASEFIN**	NUMÉRICO	Identificador del sistema de la fase final de la trasición.
→ **CAMPO_NOMBREFASEFIN**	CADENA(50)	Nombre de la fase final de la transición.
→ **CAMPO_REFTRANXTRAN**	NUMÉRICO	Identificador de la transición que se divide.
→ **CAMPO_REFTIPOACTO**	NUMÉRICO	Identificador del tipo de acto.
→ **CAMPO_INFORMAR**	CADENA(1)	Indica si se informa al bus del cambio de fase. → "**S**" – Sí. → "**N**" – No.
→ **CAMPO_ETIQLARGA**	CADENA(25)	Etiqueta larga de la transición.

Métodos para aplicación de los filtros → obtenerTransicionDefProcedimiento

TrTransicionGr

trewa.bd.trapi.tpo.TrTransicionGr

Clase que representa la información de una transición gráfica.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFTRANSGR**	TpoPK	Identificador de la transición gráfica.
→ **EXTGRINI**	TrExtremoTransicionGr	Extremo gráfico inicial.
→ **EXTGRFIN**	TrExtremoTransicionGr	Extremo gráfico final.
→ **TRANSICION**	TrTransicion	Transición.
→ **DEFPROCGR**	TrDefProcedimientoGr	Definición de procedimiento.
→ **COLORLINEA**	Long	Color de línea de la transición.
→ **COLORTEXTO**	Long	Color del texto de la etiqueta de la transición.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFTRANSGR**	NUMÉRICO	Identificador de la transición gráfica.
→ **CAMPO_REFEXTGRINI**	NUMÉRICO	Identificador del extremo gráfico inicial.
→ **CAMPO_REFEXTGRFIN**	NUMÉRICO	Identificador del extremo gráfico final.
→ **CAMPO_REFTRANSICION**	NUMÉRICO	Identificador de la transición.
→ **CAMPO_REFDEFPROCGR**	NUMÉRICO	Identificador de la definición del procedimiento gráfico.
→ **CAMPO_COLORLINEA**	NUMÉRICO	Color de línea de la transición.
→ **CAMPO_COLORTEXTO**	NUMÉRICO	Color del texto de la etiqueta de la transición.

Métodos para aplicación de los filtros → obtenerTransicionGr

TrTransicionPerfil

trewa.bd.trapi.tpo.TrTransicionPerfil

Clase que representa un perfil de usuario que tiene permisos para realizar una determinada transición.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **TRANDEFPROC**	TrTransicionDefProcedimiento	Transición asociada a una definición de procedimiento.
→ **REFPERFILUSU**	TpoPK	Identificador del perfil de usuario.
→ **PERMISO**	String	Tipo de permiso. → "**T**" – Tramitar → "**D**" – Deshacer → "**A**" – Ambos

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición de procedimiento.
→ **CAMPO_REFTRANSICION**	NUMÉRICO	Identificador de la transición.
→ **CAMPO_REFPERFILUSU**	NUMÉRICO	Identificador del perfil de usuario.
→ **CAMPO_PERMISO**	CADENA(1)	Tipo de permiso. → "**T**" – Tramitar → "**D**" – Deshacer

→ "**A**" – Ambos

Métodos para aplicación de los filtros → `obtenerTransicionPerfil`

TrUsuario

trewa.bd.trapi.tpo.TrUsuario

Clase que representa la información de una transición gráfica.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **CODUSUARIO**	`String`	Código del usuario.
→ **TIPOIDENT**	`String`	Tipo de identificador. → "**D**" – DNI / NIF → "**P**" – Pasaporte → "**N**" – NIE → "**C**" – CIF → "**O**" – Otros
→ **IDENTIFICADOR**	`String`	Código de identificación.
→ **NOMBRE**	`String`	Nombre del usuario.
→ **APELLIDO1**	`String`	Primer apellido del usuario.
→ **APELLIDO2**	`String`	Extremo gráfico inicial.
→ **SEXO**	`String`	Sexo del usuario. → "**F**" – Femenino → "**M**" – Masculino
→ **EMAIL**	`String`	Dirección de correo electrónico del usuario.
→ **CLAVE**	`String`	Clave del usuario (típicamente encriptada).
→ **FECHAALTA**	`Timestamp`	Fecha de alta en el sistema.
→ **FECHABAJA**	`Timestamp`	Fecha de baja en el sistema.
→ **ANAGRAMAFISCAL**	`String`	Anagrama fiscal del usuario

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_CODUSUARIO**	`CADENA(10)`	Código del usuario.
→ **CAMPO_TIPOIDENT**	`CADENA(1)`	Tipo de identificador. → "**D**" – DNI / NIF → "**P**" – Pasaporte → "**N**" – NIE

		→ "**C**" – CIF
		→ "**O**" – Otros
→ **CAMPO_IDENTIFICADOR**	CADENA(15)	Código de identificación.
→ **CAMPO_NOMBRE**	CADENA(32)	Nombre del usuario.
→ **CAMPO_APELLIDO1**	CADENA(32)	Primer apellido del usuario.
→ **CAMPO_APELLIDO2**	CADENA(32)	Extremo gráfico inicial.
→ **CAMPO_SEXO**	CADENA(1)	Sexo del usuario.
		→ "**F**" – Femenino
		→ "**M**" – Masculino
→ **CAMPO_EMAIL**	CADENA(64)	Dirección de correo electrónico del usuario.
→ **CAMPO_CLAVE**	CADENA(32)	Clave del usuario (típicamente encriptada).
→ **CAMPO_FECHAALTA**	DATE	Fecha de alta en el sistema.
→ **CAMPO_FECHABAJA**	DATE	Fecha de baja en el sistema.
→ **CAMPO_ANAGRAMAFISCAL**	VARCHAR2(50)	Anagrama fiscal del usuario

Métodos para aplicación de los filtros → obtenerUsuario

TrUsuarioPerfilUsuario

trewa.bd.trapi.tpo.TrUsuarioPerfilUsuario

Clase que representa un perfil de usuario asignado a un usuario.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **PERFILUSU**	TrPerfilUsuario	Perfil de usuario.
→ **USUARIO**	String	Código del usuario.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFPERFILUSU**	NUMÉRICO	Identificador del perfil de usuario.
→ **CAMPO_USUARIO**	CADENA(10)	Código del usuario.

Métodos para aplicación de los filtros → obtenerUsuarioPerfilUsuario

TrVariable

trewa.bd.trapi.tpo.TrVariable

Clase que representa una variable definida en el tramitador para los documentos.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFVARIABLE**	TpoPK	Identificador de la variable.
→ **NOMBRE**	String	Nombre de la variable.
→ **DESCRIPCION**	String	Descripción de la variable.
→ **FUNCION**	String	Nombre de la función que calcula la variable.
→ **STMA**	TrSistema	Sistema.
→ **TIPOACTO**	TrTipoActo	Tipo de acto.
→ **PAQUETE**	String	Nombre del paquete que contiene la función almacenada o método java.
→ **IMPLEMENTACION**	String	Indica el tipo de implementación. → "**F**" – Función PL/SQL → "**J**" – Java

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFVARIABLE**	NUMÉRICO	Identificador de la variable.
→ **CAMPO_NOMBRE**	CADENA(25)	Nombre de la variable.
→ **CAMPO_DESCRIPCION**	CADENA(250)	Descripción de la variable.
→ **CAMPO_FUNCION**	CADENA(50)	Nombre de la función que calcula la variable.
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema.
→ **CAMPO_REFTIPOACTO**	NUMÉRICO	Identificador del tipo de acto.
→ **CAMPO_PAQUETE**	CADENA(100)	Nombre del paquete que contiene la función almacenada o método java.
→ **CAMPO_IMPLEMENTACION**	CADENA(1)	Indica el tipo de implementación. → "**F**" – Función PL/SQL → "**J**" – Java

Métodos para aplicación de los filtros → obtenerVariable

TrVariableTipoDocumento

trewa.bd.trapi.tpo.TrVariableTipoDocumento

Clase que representa las variables que en principio se definen para un tipo de documento. Sólo válido para plantillas de Open Office.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **REFTIPODOC**	TpoPK	Identificador del tipo de documento.
→ **VARIABLE**	TrVariable	Variable.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFTIPODOC**	NUMÉRICO	Identificador del tipo de documento.
→ **CAMPO_REFVARIABLE**	NUMÉRICO	Identificador de la variable.
→ **CAMPO_NOMBRE**	CADENA(25)	Nombre de la variable.
→ **CAMPO_DESCRIPCION**	CADENA(250)	Descripción de la variable.
→ **CAMPO_FUNCION**	CADENA(50)	Nombre de la función que calcula la variable.
→ **CAMPO_REFSTMA**	NUMÉRICO	Identificador del sistema.
→ **CAMPO_REFTIPOACTO**	NUMÉRICO	Identificador del tipo de acto.
→ **CAMPO_PAQUETE**	CADENA(100)	Nombre del paquete que contiene la función almacenada o método java.
→ **CAMPO_IMPLEMENTACION**	CADENA(1)	Indica el tipo de implementación. → "**F**" – Función PL/SQL → "**J**" – Java

Métodos para aplicación de los filtros → obtenerVariableTipoDocumento

TrVersionDefProcedimiento

trewa.bd.trapi.tpo.TrVersionDefProcedimiento

Clase que representa una versión de una definición de procedimiento.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **DEFPROC**	TrDefProcedimiento	Definición del procedimiento.
→ **TIPOEXP**	TrTipoExpediente	Tipo de expediente.
→ **FECHAVIGOR**	Timestamp	Fecha en la que entra en vigor la definición del procedimiento.
→ **DESCRIPCION**	String	Descripción de la versión de la definición del procedimiento.

Campos definidos para filtrados y ordenación

Campo	Tipo	Descripción y restricciones de uso
→ **CAMPO_REFDEFPROC**	NUMÉRICO	Identificador de la definición del procedimiento
→ **CAMPO_REFTIPOEXP**	NUMÉRICO	Identificador del tipo de expediente
→ **CAMPO_FECHAVIGOR**	DATE	Fecha en la que entra en vigor la definición del procedimiento.
→ **CAMPO_DESCRIPCION**	CADENA(50)	Descripción de la versión de la definición del procedimiento.

Métodos para aplicación de los filtros → obtenerVersionDefProcedimiento

Métodos de la TrAPIADM

Mantenimiento de metafases

Insertar metafase

TpoPK insertarMetafase(TrMetafase metafase) throws TrException

Inserta una nueva metafase. Si todo es correcto devuelve el identificador de la nueva metafase, si algo falla lanza una excepción.

Entradas

TrMetafase metafase	Metafase a insertar

Salida

TpoPK	Identificador de la nueva metafase, cero si no se ha insertado

Modificar metafase

int modificarMetafase(TrMetafase metafase) throws TrException

Modifica una metafase existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrMetafase metafase	Metafase a modificar

Salida

int	Número de filas modificadas

Eliminar metafase

int eliminarMetafase(TpoPK idMetafase) throws TrException

Elimina una metafase existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idMetafase	Identificador de la metafase a eliminar

Salida

int	Número de filas eliminadas

Obtener metafase

TrMetafase[ ] obtenerMetafase(TpoPK idMetafase, ClausulaWhere where, ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias metafases. Si se pasa el parámetro *idMetafase* se devuelve los datos de esa metafase, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idMetafase	Identificador de la metafase a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrMetafase
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrMetafase

Salida

TrMetafase[ ]	Array de objetos TrMetafase

Mantenimiento de definiciones de procedimientos

Insertar definición de procedimiento

TpoPK insertarDefProcedimiento(TrDefProcedimiento defProc) throws TrException

Inserta una nueva definición de procedimiento. Si todo es correcto devuelve el identificador de la nueva definición de procedimiento, si algo falla lanza una excepción.

Entradas

TrDefProcedimiento defProc	Definición del procedimiento a insertar

Salida

TpoPK	Identificador de la nueva definición de procedimiento, cero si no se ha insertado

Modificar definición de procedimiento

int modificarDefProcedimiento(TrDefProcedimiento defProc) throws TrException

Modifica una definición de procedimiento existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrDefProcedimiento defProc	Definición de procedimiento a modificar

Salida

int	Número de filas modificadas

Eliminar definición de procedimiento

int eliminarDefProcedimiento(TpoPK idDefProc) throws TrException

Elimina una definición de procedimiento existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idDefProc	Identificador de la definición de procedimiento a eliminar

Salida

int	Número de filas eliminadas

Obtener definición de procedimiento

TrDefProcedimiento[ ] obtenerDefProcedimiento(TpoPK idDefProc,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias definiciones de procedimientos. Si se pasa el parámetro *idDefProc* se devuelve los datos de esa definición de procedimiento, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idDefProc	Identificador de la definición de procedimiento a obtener. Si se pasa *null* se obtienen todas.

ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrDefProcedimiento
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrDefProcedimiento

Salida

TrDefProcedimiento[ ]	Array de objetos TrDefProcedimiento

Bloquear definición de procedimiento

void bloquearDefProcedimiento(TpoPK idDefProc) throws TrException

Bloquea la definición del procedimiento que se indique en el parámetro *idDefProc*, siempre que no esté bloqueada por otro usuario.

Entradas

TpoPK idDefProc	Identificador de la definición del procedimiento

Desbloquear definición de procedimiento

void desbloquearDefProcedimiento(TpoPK idDefProc) throws TrException

Desbloquea la definición del procedimiento que se indique en el parámetro *idDefProc*, siempre que esté bloqueada por el mismo usuario.

Entradas

TpoPK idDefProc	Identificador de la definición del procedimiento

Procedimiento bloqueado por un usuario

String procedimientoBloqueadoPor(TpoPK idDefProc,
TpoString nombre) throws TrException;

Permite obtener el usuario que tiene bloqueado el procedimiento.

Entradas

TpoPK idDefProc	Identificador de la definición del procedimiento

Entrada / Salida

TpoString nombre	Nombre y apellidos del usuario

Salida

String	Identificador del usuario que tiene bloqueado el procedimiento.

Obtener otros datos

String obtenerOtrosDatosDefProcedimiento(TpoPK idDefProc) throws TrException

Permite obtener otros datos de la definición del procedimiento

Entradas

TpoPK idDefProc	Identificador de la definición del procedimiento

Salida

String	Otros datos del procedimiento (típicamente xml).

Actualizar otros datos

void actualizarOtrosDatosDefProcedimiento(TpoPK idDefProc,
String contenido) throws TrException

Permite actualizar otros datos de la definición del procedimiento

Entradas

TpoPK idDefProc	Identificador de la definición del procedimiento.
String contenido	Contenido para otros datos del procedimiento.

Mantenimiento de fases

Insertar fase

TpoPK insertarFase(TrFase fase) throws TrException

Inserta una nueva fase. Si todo es correcto devuelve el identificador de la nueva fase, si algo falla lanza una excepción.

Entradas

TrFase fase	Fase a insertar

Salida

TpoPK	Identificador de la nueva fase, cero si no se ha insertado

Modificar fase

int modificarFase(TrFase fase) throws TrException

Modifica una fase existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrFase fase	Fase a modificar

Salida

int	Número de filas modificadas

Eliminar fase

int eliminarFase(TpoPK idFase) throws TrException

Elimina una fase existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idFase	Identificador de la fase a eliminar

Salida

int	Número de filas eliminadas

Obtener fase

TrFase [ ] obtenerFase(TpoPK idFase,

ClausulaWhere where, ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias fases. Si se pasa el parámetro *idFase* se devuelve los datos de esa fase, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idFase	Identificador de la fase a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrFase
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrFase

Salida

TrFase[ ]	Array de objetos TrFase

Mantenimiento de transiciones

Insertar transición

```
void insertarTransicionDefProcedimiento(
                                    TrTransicionDefProcedimiento transicionDefProc,
                                    TpoPK idTransicion,
                                    TpoPK idDefProc) throws TrException
```

Inserta una nueva transición asociada a un procedimiento. Si la transición ya existía, simplemente se asocia al procedimiento. Si todo es correcto devuelve el identificador de la nueva transición asociada al procedimiento, si algo falla lanza una excepción.

Entradas

TrTransicionDefProcedimiento transicionDefProc	Transición asociada al procedimiento a insertar

Entrada / Salida

TpoPK idTransición	Identificador de la transición asociada al procedimiento, cero si no se ha insertado.
TpoPK idDefProc	Identificador del procedimiento al que se ha asociado la transición , cero si no se ha insertado.

Modificar transición

```
int modificarTransicionDefProcedimiento(
                    TrTransicionDefProcedimiento transicionDefProc ) throws TrException
```

Modifica una transición asociada a un procedimiento existente. Si se modifican los datos de la transición estos afectan a todos los procedimientos. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

*TrTransicionDefProcedimiento*transicionDefProc	transición asociada a un procedimiento a modificar

Salida

int	Número de filas modificadas

Eliminar transición

```
int eliminarTransicionDefProcedimiento( TpoPK idTransicion,
                                        TpoPK idDefProc) throws TrException
```

Elimina una transición asociada a un procedimiento existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTransicion	Identificador de la transición
TpoPK idDefProc	Identificador del procedimiento

Salida

int	Número de filas eliminadas

Obtener transición

```
TrTransicionDefProcedimiento[ ] obtenerTransicionDefProcedimiento(
                                    TpoPK idTransicion,
                                    TpoPK idDefProc,
                                    ClausulaWhere where,
                                    ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener una o varias transiciones. Si se pasa el parámetro *idTransición* se devuelve los datos de esa transición asociada al procedimiento que se indique en el parámetro *idDefProc*, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idTransición	Identificador de la transición. Si se pasa *null* se obtienen todas.
TpoPK idDefProc	Identificador del procedimiento. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTransicionDefProcedimiento
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTransicionDefProcedimiento

Salida

TrTransicionDefProcedimiento[ ]	Array de objetos TrTransicionDefProcedimiento

Ajustar transición

```
int ajustarTransicionExpedientesFase(
                                    TpoPK idTransicion,
                                    TpoPK idTransicionAjuste) throws TrException
```

Permite ajustar la transición de los expedientes en fase. Este método es útil cuando se invalida una transición y se crea una nueva y queremos que los expedientes en fase ya existentes usen la nueva transición.

Entradas

TpoPK idTransicion	Identificador de la transición. Si se pasa *null* o una transición inválida generará una excepción.
TpoPK idTransicionAjuste	Identificador de la nueva transición. Si se pasa *null* o una transición inválida generará una excepción.

Salida

int	Número de filas ajustadas

Mantenimiento de perfiles de usuario

Insertar perfil de usuario

TpoPK insertarPerfilUsuario(TrPerfilUsuario perfilUsuario) throws TrException

Inserta un nuevo perfil de usuario. Si todo es correcto devuelve el identificador del nuevo perfil de usuario, si algo falla lanza una excepción.

Entradas

TrPerfilUsuario perfilUsuario	Perfil de usuario a insertar

Salida

TpoPK	Identificador del nuevo perfil de usuario, cero si no se ha insertado

Modificar perfil de usuario

int modificarPerfilUsuario(TrPerfilUsuario perfilUsuario) throws TrException

Modifica un perfil de usuario existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrPerfilUsuario perfilUsuario	Perfil de usuario a modificar

Salida

int	Número de filas modificadas

Eliminar perfil de usuario

int eliminarPerfilUsuario(TpoPK idPerfilUsu) throws TrException

Elimina un perfil de usuario existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idPerfilUsu	Identificador del perfil de usuario a eliminar

Salida

int	Número de filas eliminadas

Obtener perfil de usuario

TrPerfilUsuario [ ] obtenerPerfilUsuario(TpoPK idPerfilUsu,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios perfiles de usuario. Si se pasa el parámetro *idPerfilUsu* se devuelve los datos de ese perfil de usuario, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idPerfilUsu	Identificador del perfil de usuario a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrPerfilUsuario
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrPerfilUsuario

Salida

TrPerfilUsuario[ ]	Array de objetos TrPerfilUsuario

Mantenimiento de tipos de expediente

Insertar tipo de expediente

TpoPK insertarTipoExpediente(TrTipoExpediente tipoExp) throws TrException

Inserta un nuev tipo de expediente. Si todo es correcto devuelve el identificador del nuevo tipo de expediente, si algo falla lanza una excepción.

Entradas

TrTipoExpediente tipoExp	Tipo de expediente a insertar

Salida

TpoPK	Identificador del nuevo tipo de expediente, cero si no se ha insertado

Modificar tipo de expediente

int modificarTipoExpediente(TrTipoExpediente tipoExp) throws TrException

Modifica un tipo de expediente existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrTipoExpediente tipoExp	Tipo de expediente a modificar

Salida

int	Número de filas modificadas

Eliminar tipo de expediente

int eliminarTipoExpediente(TpoPK idTipoExp) throws TrException

Elimina un tipo de expediente existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoExp	Identificador del tipo de expediente a eliminar

Salida

int	Número de filas eliminadas

Obtener tipo de expediente

TrTipoExpediente [ ] obtenerTipoExpediente(TpoPK idTipoExp,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios tipos de expediente. Si se pasa el parámetro *idTipoExp* se devuelve los datos de ese tipo de expediente, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idTipoExp	Identificador del tipo de expediente a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoExpediente

ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoExpediente

Salida

TrTipoExpediente[ ]	Array de objetos TrTipoExpediente

Mantenimiento de versiones de procedimientos

Insertar versión de procedimiento

```
void insertarVersionDefProcedimiento(
                    TrVersionDefProcedimiento versionDefProc,
                    TpoPK idDefProc,
                    TpoPK idTipoExp) throws TrException
```

Inserta una nueva versión de un procedimiento asociado a un tipo de expediente. Si todo es correcto devuelve el identificador del procedimiento y del tipo de expediente, si algo falla lanza una excepción.

Entradas

TrVersionDefProcedimiento versionDefProc	Versión del procedimiento a insertar

Entrada / Salida

TpoPK idDefProc	Identificador del procedimiento, cero si no se ha insertado la versión.
TpoPK idTipoExp	Identificador del tipo de expediente, cero si no se ha insertado la versión.

Modificar versión de procedimiento

```
int modificarVersionDefProcedimiento(
                    TrVersionDefProcedimiento versionDefProc ) throws TrException
```

Modifica una versión de procedimiento existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrVersionDefProcedimiento versionDefProc	Versión del procedimiento a modificar

Salida

int	Número de filas modificadas

Eliminar versión de procedimiento

```
int eliminarVersionDefProcedimiento( TpoPK idDefProc,
                                     TpoPK idTipoExp ) throws TrException
```

Elimina una versión de procedimiento existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idDefProc	Identificador del procedimiento
TpoPK idTipoExp	Identificador del tipo de expediente

Salida

int	Número de filas eliminadas

Obtener versión de procedimiento

```
TrVersionDefProcedimiento[ ] obtenerVersionDefProcedimiento(
                                     TpoPK idDefProc,
                                     TpoPK idTipoExp,
                                     ClausulaWhere where,
                                     ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener una o varias versiones de procedimiento. Si se pasa el parámetro idDefProc se devuelven las versiones de ese procedimiento y si se para el parámetro *idTipoExp* se devuelven las asociadas a ese tipo de expediente, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idDefProc	Identificador del procedimiento. Si se pasa *null* se obtienen todas las versiones asociadas a ese procedimiento.
TpoPK idTipoExp	Identificador del tipo de expediente. Si se pasa *null* se obtienen todas las versiones asociadas a ese tipo de expediente.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrVersionDefProcedimiento
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrVersionDefProcedimiento

Salida

TrVersionDefProcedimiento *[ ]* Array de objetos TrVersionDefProcedimiento

Mantenimiento de extremos de transiciones gráficas

Insertar extremo de transición gráfica

TpoPK insertarExtremoTransicionGr(TrExtremoTransicionGr extremoTransGr) throws TrException

Inserta un nuevo extremo de una transición gráfica. Si todo es correcto devuelve el identificador del nuevo extremo de la transición gráfica, si algo falla lanza una excepción.

Entradas

TrExtremoTransicionGr extremoTransGr	Extremo de una transición gráfica a insertar

Salida

TpoPK	Identificador del nuevo extremo de una transición gráfica, cero si no se ha insertado

Modificar extremo de transición gráfica

int modificarExtremoTransicionGr(TrExtremoTransicionGr extremoTransGr) throws TrException

Modifica un extremo de una transición gráfica existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrExtremoTransicionGr extremoTransGr	Extremo de una transición gráfica a modificar

Salida

int	Número de filas modificadas

Eliminar extremo de transición gráfica

int eliminarExtremoTransicionGr(TpoPK idExtremoTransGr) throws TrException

Elimina un extremo de una transición gráfica existente. Si todo es correcto devuelve el

número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idExtremoTransGr	Identificador del extremo de una transición gráfica a eliminar

Salida

int	Número de filas eliminadas

Obtener extremo de transición gráfica

```
TrExtremoTransicionGr [ ] obtenerExtremoTransicionGr(
                                TpoPK idExtremoTransGr,
                                ClausulaWhere where,
                                ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener uno o varios extremos de una o varias transiciones gráficas. Si se pasa el parámetro *idExtremoTransGr* se devuelve los datos de ese extremo de la transición gráfica, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idExtremoTransGr	Identificador del extremo de la transición gráfica a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrExtremoTransicionGr
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrExtremoTransicionGr

Salida

TrExtremoTransicionGr[ ]	Array de objetos TrExtremoTransicionGr

Mantenimiento de definición de procedimientos gráficos

Insertar definición de procedimiento gráfico

```
TpoPK insertarDefProcedimientoGr( TrDefProcedimientoGr defProcGr ) throws TrException
```

Inserta una nueva definición de procedimiento gráfico. Si todo es correcto devuelve el identificador de la nueva definición de procedimiento gráfico, si algo falla lanza una excepción.

Entradas

TrDefProcedimientoGr defProcGr	Definición de procedimiento gráfico a insertar

Salida

TpoPK	Identificador de la nueva definición de procedimiento gráfico, cero si no se ha insertado

Modificar definición de procedimiento gráfico

int modificarDefProcedimientoGr(TrDefProcedimientoGr defProcGr) throws TrException

Modifica una definición de procedimiento gráfico existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrDefProcedimientoGr defProcGr	Definición de procedimiento gráfico a modificar

Salida

int	Número de filas modificadas

Eliminar definición de procedimiento gráfico

int eliminarDefProcedimientoGr(TpoPK idDefProcGr) throws TrException

Elimina una definición de procedimiento gráfico existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idDefProcGr	Identificador de la definición de procedimiento gráfico a eliminar

Salida

int	Número de filas eliminadas

Obtener definición de procedimiento gráfico

TrDefProcedimientoGr [ ] obtenerDefProcedimientoGr(TpoPK idDefProcGr,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias definiciones de procedimientos gráficos. Si se pasa el parámetro *idDefProcGr* se devuelve los datos de esa definición de procedimiento gráfico, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idDefProcGr	Identificador de la definición de procedimiento gráfico a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrDefProcedimientoGr
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrDefProcedimientoGr

Salida

TrDefProcedimientoGr[ ]	Array de objetos TrDefProcedimientoGr

Mantenimiento de metafases gráficas

Insertar metafase gráfica

TpoPK insertarMetafaseGr(TrMetafaseGr metafaseGr) throws TrException

Inserta una nueva metafase gráfica. Si todo es correcto devuelve el identificador de la nueva metafase gráfica, si algo falla lanza una excepción.

Entradas

TrMetafaseGr metafaseGr	Metafase gráfica a insertar

Salida

TpoPK	Identificador de la nueva metafase gráfica, cero si no se ha insertado

Modificar metafase gráfica

int modificarMetafaseGr(TrMetafaseGr metafaseGr) throws TrException

Modifica una metafase gráfica existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrMetafaseGr metafaseGr	Metafase gráfica a modificar

Salida

int	Número de filas modificadas

Eliminar metafase gráfica

int eliminarMetafaseGr(TpoPK idMetafaseGr) throws TrException

Elimina una metafase gráfica existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idMetafaseGr	Identificador de la metafase gráfica a eliminar

Salida

int	Número de filas eliminadas

Obtener metafase gráfica

TrMetafaseGr [ ] obtenerMetafaseGr(TpoPK idMetafaseGr,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias metafases gráficas. Si se pasa el parámetro *idMetafaseGr* se devuelve los datos de esa metafase gráfica, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idMetafaseGr	Identificador de la metafase gráfica a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrMetafaseGr
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrMetafaseGr

Salida

TrMetafaseGr[ ]	Array de objetos TrMetafaseGr

Mantenimiento de nodos de transición gráfica

Insertar nodo de transición gráfica

TpoPK insertarNodoTransicionGr(TrNodoTransicionGr nodoTransGr) throws TrException

Inserta un nuevo nodo de transición gráfica. Si todo es correcto devuelve el identificador del nuevo nodo de transición gráfica, si algo falla lanza una excepción.

Entradas

TrNodoTransicionGr nodoTransGr	Nodo de transición gráfica a insertar

Salida

TpoPK	Identificador del nuevo nodo de transición gráfica, cero si no se ha insertado

Modificar nodo de transición gráfica

int modificarNodoTransicionGr(TrNodoTransicionGr nodoTransGr) throws TrException

Modifica un nodo de transición gráfica existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrNodoTransicionGr nodoTransGr	Nodo de transición gráfica a modificar

Salida

int	Número de filas modificadas

Eliminar nodo de transición gráfica

int eliminarNodoTransicionGr(TpoPK idNodoTransGr) throws TrException

Elimina un nodo de transición gráfica existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idNodoTransGr	Identificador del nodo de transición gráfica a eliminar

Salida

int	Número de filas eliminadas

Obtener nodo de transición gráfica

TrNodoTransicionGr [ ] obtenerNodoTransicionGr(TpoPK idNodoTransGr,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios nodos de las transiciones gráficas. Si se pasa el parámetro *idNodoTransGr* se devuelve los datos de ese nodo de transición gráfica, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idNodoTransGr	Identificador del nodo de transición gráfica a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrNodoTransicionGr
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrNodoTransicionGr

Salida

TrNodoTransicionGr[ ]	Array de objetos TrNodoTransicionGr

Mantenimiento de transiciones gráficas

Insertar transición gráfica

TpoPK insertarTransicionGr(TrTransicionGr transicionGr) throws TrException

Inserta una nueva transición gráfica. Si todo es correcto devuelve el identificador de la nueva transición gráfica, si algo falla lanza una excepción.

Entradas

TrTransicionGr transicionGr	Transición gráfica a insertar

Salida

TpoPK	Identificador de la nueva transición gráfica, cero si no se ha insertado

Modificar transición gráfica

int modificarTransicionGr(TrTransicionGr transicionGr) throws TrException

Modifica una transición gráfica existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrTransicionGr transicionGr	transición gráfica a modificar

Salida

int	Número de filas modificadas

Eliminar transición gráfica

int eliminarTransicionGr(TpoPK idTransicionGr) throws TrException

Elimina una transición gráfica existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTransicionGr	Identificador de la transición gráfica a eliminar

Salida

int	Número de filas eliminadas

Obtener transición gráfica

TrTransicionGr [ ] obtenerTransicionGr(TpoPK idTransicionGr,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias transiciones gráficas. Si se pasa el parámetro *idTransicionGr* se devuelve los datos de esa transición gráfica, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idTransicionGr	Identificador de la transición gráfica a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTransicionGr

	ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTransicionGr

Salida

TrTransicionGr[ ]	Array de objetos TrTransicionGr

Mantenimiento de tareas (Manipulación datos y otras)

Insertar tarea

TpoPK insertarBloque(TrBloque bloque) throws TrException

Inserta una nueva tarea o bloque. Si todo es correcto devuelve el identificador de la nueva tarea o bloque, si algo falla lanza una excepción.

Entradas

TrBloque bloque	Tarea o bloque a insertar

Salida

TpoPK	Identificador de la nueva tarea o bloque, cero si no se ha insertado

Modificar tarea

int modificarBloque(TrBloque bloque) throws TrException

Modifica una tarea o bloque existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrBloque bloque	Tarea o bloque a modificar

Salida

int	Número de filas modificadas

Eliminar tarea

int eliminarBloque(TpoPK idBloque) throws TrException

Elimina una tarea o bloque existente. Si todo es correcto devuelve el número de filas

eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idBloque	Identificador de la tarea o bloque a eliminar

Salida

int	Número de filas eliminadas

Obtener tarea

```
TrBloque [ ] obtenerBloque( TpoPK idBloque,
                            ClausulaWhere where,
                            ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener una o varias tareas o bloques. Si se pasa el parámetro *idBloque* se devuelve los datos de esa tarea o bloque, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idBloque	Identificador de la tarea o bloque a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrBloque
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrBloque

Salida

TrBloque[ ]	Array de objetos TrBloque

Mantenimiento de tareas en fase (Manipulación datos y otras)

Insertar tarea en fase

```
void insertarBloquePermitidoDefProc(TrBloquePermitidoDefProc bloquePer,
                                    TpoPK idBloquePer,
                                    TpoPK idDefProc ) throws TrException
```

Inserta una nueva tarea en fase o bloque permitido. Si el bloque permitido ya existía se asocia al procedimiento. Si todo es correcto devuelve el identificador de la nueva tarea en fase o bloque permitido, si algo falla lanza una excepción.

Entradas

TrBloquePermitidoDefProc bloquePer	tarea en fase o bloque permitido a insertar

Entrada / Salida

TpoPK idBloquePer	Identificador de la nueva tarea en fase o bloque permitido, cero si no se ha insertado
TpoPK idDefProc	Identificador de la definición del procedimiento, cero si no se ha insertado

Modificar tarea en fase

int modificarBloquePermitidoDefProc(TrBloquePermitidoDefProc bloquePer) throws TrException

Modifica una tarea en fase o bloque permitido existente. Si el bloque permitido está asociada a varios procedimientos los cambios afectan a todos ellos. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrBloquePermitidoDefProc bloquePer	Tarea en fase o bloque permitido a modificar

Salida

int	Número de filas modificadas

Eliminar tarea en fase

int eliminarBloquePermitidoDefProc (
TpoPK idBloquePer, TpoPK idDefProc) throws TrException

Elimina una tarea en fase o bloque permitido existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idBloquePer	Identificador de la tarea en fase o bloque permitido a eliminar
TpoPK idDefProc	Identificador de la definición del procedimiento.

Salida

int	Número de filas eliminadas

Obtener tarea en fase

```
TrBloquePermitidoDefProc [ ] obtenerBloquePermitidoDefProc(
                                    TpoPK idBloquePer,
                                    TpoPK idDefProc,
                                    ClausulaWhere where,
                                    ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener una o varias tareas en fase o bloques permitidos. Si se pasa el parámetro idBloquePer se devuelve los datos de esa tarea en fase o bloque permitido, y si se pasa el parámetro idDefProc se devuelven los que estén asociados a ese procedimiento, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idBloquePer	Identificador de la tarea en fase o bloque permitido a obtener. Si se pasa *null* se obtienen todas.
TpoPK idDefProc	Identificador de la definición de procedimiento. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrBloquePermitidoDefProc
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrBloquePermitidoDefProc

Salida

TrBloquePermitidoDefProc[ ]	Array de objetos TrBloquePermitidoDefProc

Mantenimiento de caducidades

Insertar caducidad

```
TpoPK insertarCaducidad( TrCaducidad caducidad ) throws TrException
```

Inserta una nueva caducidad. Si todo es correcto devuelve el identificador de la nueva caducidad, si algo falla lanza una excepción.

Entradas

TrCaducidad caducidad	Caducidad a insertar

Salida

TpoPK	Identificador de la nueva caducidad, cero si no se ha insertado

Modificar caducidad

int modificarCaducidad(TrCaducidad caducidad) throws TrException

Modifica una caducidad existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrCaducidad caducidad	Caducidad a modificar

Salida

int	Número de filas modificadas

Eliminar caducidad

int eliminarCaducidad(TpoPK idCaducidad) throws TrException

Elimina una caducidad existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idCaducidad	Identificador de la caducidad a eliminar

Salida

int	Número de filas eliminadas

Obtener caducidad

TrCaducidad [ ] obtenerCaducidad(TpoPK idCaducidad,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias caducidades. Si se pasa el parámetro *idCaducidad* se devuelve los datos de esa caducidad, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idCaducidad	Identificador de la caducidad a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrCaducidad
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrCaducidad

Salida

TrCaducidad[ ]	Array de objetos TrCaducidad

Mantenimiento de tareas (Manipulación de escritos)

Insertar tipo de documento

TpoPK insertarTipoDocumento(TrTipoDocumento tipoDoc) throws TrException

Inserta una nueva tarea o tipo de documento. Si todo es correcto devuelve el identificador de la nueva tarea o tipo de documento, si algo falla lanza una excepción.

Entradas

TrTipoDocumento tipoDoc	Tarea o tipo de documento a insertar

Salida

TpoPK	Identificador de la nueva tarea o tipo de documento, cero si no se ha insertado

Modificar tipo de documento

int modificarTipoDocumento(TrTipoDocumento tipoDoc) throws TrException

Modifica una tarea o tipo de documento existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrTipoDocumento tipoDoc	Tarea o tipo de documento a modificar

Salida

int	Número de filas modificadas

Eliminar tipo de documento

int eliminarTipoDocumento(TpoPK idTipoDoc) throws TrException

Elimina una tarea o tipo de documento existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoDoc	Identificador de la tarea o tipo de documento a eliminar

Salida

int	Número de filas eliminadas

Obtener tipo de documento

TrTipoDocumento [ ] obtenerTipoDocumento(TpoPK idTipoDoc,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias tareas o tipos de documentos Si se pasa el parámetro *idTipoDoc* se devuelve los datos de esa tarea o tipo de documento, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idTipoDoc	Identificador de la tarea o tipo de documento a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoDocumento
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoDocumento

Salida

TrTipoDocumento[ ]	Array de objetos TrTipoDocumento

Mantenimiento de tareas en fase (Manipulación de escritos)

Insertar tarea en fase

void insertarDocumentoPermitidoDefProc(
TrDocumentoPermitidoDefProc docPermitido,

TpoPK idDocumentoPer, TpoPK idDefProc) throws TrException

Inserta una nueva tarea en fase o documento permitido. Si el documento permitido ya existía se asocia al procedimiento. Si todo es correcto devuelve el identificador de la nueva tarea en fase o documento permitido, si algo falla lanza una excepción.

Entradas

TrDocumentoPermitidoDefProc docPermitido	Tarea en fase o documento permitido a insertar

Entrada / Salida

TpoPK idDocumentoPer	Identificador de la nueva tarea en fase o documento permitido, cero si no se ha insertado
TpoPK idDefProc	Identificador de la definición del procedimiento, cero si no se ha insertado

Modificar tarea en fase

int modificarDocumentoPermitidoDefProc(TrDocumentoPermitidoDefProc docPermitido) throws TrException

Modifica una tarea en fase o documento permitido existente. Si el documento permitido está asociada a varios procedimientos los cambios afectan a todos ellos. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrDocumentoPermitidoDefProc docPermitido	Tarea en fase o documento permitido a modificar

Salida

int	Número de filas modificadas

Eliminar tarea en fase

int eliminarDocumentoPermitidoDefProc (TpoPK idDocumentoPer, TpoPK idDefProc) throws TrException

Elimina una tarea en fase o documento permitido existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idDocumentoPer	Identificador de la tarea en fase o documento permitido a eliminar

TpoPK idDefProc	Identificador de la definición del procedimiento.

Salida

int	Número de filas eliminadas

Obtener tarea en fase

```
TrDocumentoPermitidoDefProc [ ] obtenerDocumentoPermitidoDefProc(
                                    TpoPK idDocumentoPer,
                                    TpoPK idDefProc,
                                    ClausulaWhere where,
                                    ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener una o varias tareas en fase o documentos permitidos. Si se pasa el parámetro idDocumentoPer se devuelve los datos de esa tarea en fase o documento permitido, y si se pasa el parámetro idDefProc se devuelven los que estén asociados a ese procedimiento, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idDocumentoPer	Identificador de la tarea en fase o documento permitido a obtener. Si se pasa *null* se obtienen todas.
TpoPK idDefProc	Identificador de la definición de procedimiento. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrDocumentoPermitidoDefProc
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrDocumentoPermitidoDefProc

Salida

TrDocumentoPermitidoDefProc[ ]	Array de objetos TrDocumentoPermitidoDefProc

Mantenimiento de condiciones

Insertar condición

```
TpoPK insertarCondicion( TrCondicion condicion ) throws TrException
```

Inserta una nueva condición. Si todo es correcto devuelve el identificador de la nueva condición, si algo falla lanza una excepción.

Entradas

TrCondicion condicion	Condición a insertar

Salida

TpoPK	Identificador de la nueva condición, cero si no se ha insertado

Modificar condición

```
int modificarCondicion(  TrCondicion  condicion ) throws TrException
```

Modifica una condición existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrCondicion condicion	Condición a modificar

Salida

int	Número de filas modificadas

Eliminar condición

```
int eliminarCondicion( TpoPK idCondicion) throws TrException
```

Elimina una condición existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idCondicion	Identificador de la condición a eliminar

Salida

int	Número de filas eliminadas

Obtener condición

```
TrCondicion [ ] obtenerCondicion( TpoPK idCondicion,
                                  ClausulaWhere where,
                                  ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener una o varias condiciones. Si se pasa el parámetro *idCondicion* se devuelve los datos de esa condición, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idCondicion	Identificador de la condición a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrCondicion
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrCondicion

Salida

TrCondicion[ ]	Array de objetos TrCondicion

Mantenimiento de acciones

Insertar acción

TpoPK insertarAccion(TrAccion accion) throws TrException

Inserta una nueva acción. Si todo es correcto devuelve el identificador de la nueva Acción, si algo falla lanza una excepción.

Entradas

TrAccion accion	Acción a insertar

Salida

TpoPK	Identificador de la nueva acción, cero si no se ha insertado

Modificar acción

int modificarAccion(TrAccion accion) throws TrException

Modifica una acción existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrAccion accion	Acción a modificar

Salida

int	Número de filas modificadas

Eliminar acción

int eliminarAccion(TpoPK idAccion) throws TrException

Elimina una acción existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idAccion	Identificador de la acción a eliminar

Salida

int	Número de filas eliminadas

Obtener acción

TrAccion [ ] obtenerAccion(TpoPK idAccion,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias acciones. Si se pasa el parámetro *idAccion* se devuelve los datos de esa acción, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idAccion	Identificador de la acción a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrAccion
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrAccion

Salida

TrAccion[ ]	Array de objetos TrAccion

Mantenimiento de avisos

Insertar aviso

TpoPK insertarAviso(TrAviso aviso) throws TrException

Inserta un nuevo aviso. Si todo es correcto devuelve el identificador de la nueva aviso, si algo falla lanza una excepción.

Entradas

TrAviso aviso	Aviso a insertar

Salida

TpoPK	Identificador de la nueva aviso, cero si no se ha insertado

Modificar aviso

int modificarAviso(TrAviso aviso) throws TrException

Modifica un aviso existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrAviso aviso	Aviso a modificar

Salida

int	Número de filas modificadas

Eliminar aviso

int eliminarAviso(TpoPK idAviso) throws TrException

Elimina un aviso existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idAviso	Identificador del aviso a eliminar

Salida

int	Número de filas eliminadas

Obtener aviso

```
TrAviso [ ] obtenerAviso( TpoPK idAviso,
                          ClausulaWhere where,
                          ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener uno o varios avisos. Si se pasa el parámetro *idAviso* se devuelve los datos de ese aviso, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idAviso	Identificador del aviso a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrAviso
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrAviso

Salida

TrAviso[ ]	Array de objetos TrAviso

Mantenimiento de condiciones de una transición

Insertar condición de una transición

```
void insertarCondicionTransicion(TrCondicionTransicion condicionTrans,
                                 TpoPK idCondicion,
                                 TpoPK idTransicion,
                                 TpoPK idDefProc) throws TrException
```

Permite asociar una condición a una transición. Si todo es correcto devuelve el identificador de la condición asociada a la transición, si algo falla lanza una excepción.

Entradas

TrCondicionTransicion condicionTrans	Condición asociada a una transición a insertar

Entrada / Salida

TpoPK idCondicion	Identificador de la condición asociada a la transición, cero si no se ha insertado
TpoPK idTransicion	Identificador de la transición, cero si no se ha insertado

TpoPK idDefProc	Identificador de la definición de procedimiento, cero si no se ha insertado

Modificar condición de una transición

int modificarCondicionTransicion(TrCondicionTransicion condicionTrans) throws TrException

Modifica una condición asociada a una transición existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrCondicionTransicion condicionTrans	Condición asociada a una transición a modificar

Salida

int	Número de filas modificadas

Eliminar condición de una transición

int eliminarCondicionTransicion(TpoPK idCondicion,
TpoPK idTransicion,
TpoPK idDefProc) throws TrException

Elimina la asociación entre una condición y una transición existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idCondicion	Identificador de la condición asociada a una transición a eliminar
TpoPK idTransicion	Identificador de la transición
TpoPK idDefProc	Identificador de la definición de procedimiento

Salida

int	Número de filas eliminadas

Obtener condición de una transición

TrCondicionTransicion[ ] obtenerCondicionTransicion(TpoPK idCondicion,

```
                                    TpoPK idTransicion,
                                    TpoPK idDefProc,
                                    ClausulaWhere where,
                                    ClausulaOrderBy orderBy) throws TrException
```

Permite obtener una o varias condiciones asociadas a una transición. Si se pasa el parámetro *idCondicion* se devuelve los datos de esa condición asociada a la transición, si se pasa el parámetro *idTransicion* se filtra por la transición indicada, igualmente pasa con el parámetro *idDefProc* para la definición del procedimiento,si se pasa *null*, se devuelven todas.

Entradas

TpoPK idCondicion	Identificador de la condición. Si se pasa *null* se obtienen todas.
TpoPK idTransicion	Identificador de la transición. Si se pasa *null* se obtienen todas.
TpoPK idDefProc	Identificador de la definición de procedimiento. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrCondicionTransicion
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrCondicionTransicion

Salida

TrCondicionTransicion[ ]	Array de objetos TrCondicionTransicion

Mantenimiento de acciones de una transición

Insertar acción de una transición

```
void insertarAccionTransicion( TrAccionTransicion accionTrans,
                                    TpoPK idAccion,
                                    TpoPK idTransicion,
                                    TpoPK idDefProc) throws TrException
```

Permite asociar una acción a una transición. Si todo es correcto devuelve el identificador de la acción asociada a la transición, si algo falla lanza una excepción.

Entradas

TrAccionTransicion accionTrans	Acción asociada a una transición a insertar

Entrada / Salida

TpoPK idAccion	Identificador de la acción asociada a la transición, cero si no se ha insertado
TpoPK idTransicion	Identificador de la transición, cero si no se ha insertado
TpoPK idDefProc	Identificador de la definición de procedimiento, cero si no se ha insertado

Modificar acción de una transición

```
int modificarAccionTransicion(
                    TrAccionTransicion  accionTrans ) throws TrException
```

Modifica una acción asociada a una transición existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrAccionTransicion accionTrans	Acción asociada a una transición a modificar

Salida

int	Número de filas modificadas

Eliminar acción de una transición

```
int eliminarAccionTransicion( TpoPK idAccion,
                              TpoPK idTransicion,
                              TpoPK idDefProc) throws TrException
```

Elimina la asociación entre una acción y una transición existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idAccion	Identificador de la acción asociada a una transición a eliminar
TpoPK idTransicion	Identificador de la transición
TpoPK idDefProc	Identificador de la definición de procedimiento

Salida

int	Número de filas eliminadas

Obtener acción de una transición

```
TrAccionTransicion[ ] obtenerAccionTransicion( TpoPK idAccion,
                                               TpoPK idTransicion,
                                               TpoPK idDefProc,
                                               ClausulaWhere where,
                                               ClausulaOrderBy orderBy) throws TrException
```

Permite obtener una o varias acciones asociadas a una transición. Si se pasa el parámetro *idAccion* se devuelve los datos de esa acción asociada a la transición, si se pasa el parámetro *idTransicion* se filtra por la transición indicada, igualmente pasa con el parámetro *idDefProc* para la definición del procedimiento,si se pasa *null*, se devuelven todas.

Entradas

TpoPK idAccion	Identificador de la acción. Si se pasa *null* se obtienen todas.
TpoPK idTransicion	Identificador de la transición. Si se pasa *null* se obtienen todas.
TpoPK idDefProc	Identificador de la definición de procedimiento. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrAccionTransicion
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrAccionTransicion

Salida

TrAccionTransicion[ ]	Array de objetos TrAccionTransicion

Mantenimiento de avisos de una transición

Insertar aviso de una transición

```
void insertarAvisoTransicion( TrAvisoTransicion avisoTrans,
                              TpoPK idAviso,
                              TpoPK idTransicion,
                              TpoPK idDefProc) throws TrException
```

Permite asociar un aviso a una transición. Si todo es correcto devuelve el identificador del aviso asociada a la transición, si algo falla lanza una excepción.

Entradas

TrAvisoTransicion avisoTrans	Aviso asociada a una transición a insertar

Entrada / Salida

TpoPK idAviso	Identificador del aviso asociada a la transición, cero si no se ha insertado
TpoPK idTransicion	Identificador de la transición, cero si no se ha insertado
TpoPK idDefProc	Identificador de la definición de procedimiento, cero si no se ha insertado

Modificar aviso de una transición

```
int modificarAvisoTransicion(
                    TrAvisoTransicion  avisoTrans ) throws TrException
```

Modifica un aviso asociado a una transición existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrAvisoTransicion avisoTrans	Aviso asociada a una transición a modificar

Salida

int	Número de filas modificadas

Eliminar aviso de una transición

```
int eliminarAvisoTransicion( TpoPK idAviso,
                             TpoPK idTransicion,
                             TpoPK idDefProc) throws TrException
```

Elimina la asociación entre un aviso y una transición existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idAviso	Identificador del aviso asociada a una transición a eliminar
TpoPK idTransicion	Identificador de la transición

TpoPK idDefProc	Identificador de la definición de procedimiento

Salida

int	Número de filas eliminadas

Obtener aviso de una transición

```
TrAvisoTransicion[ ] obtenerAvisoTransicion( TpoPK idAviso,
                                             TpoPK idTransicion,
                                             TpoPK idDefProc,
                                             ClausulaWhere where,
                                             ClausulaOrderBy orderBy) throws TrException
```

Permite obtener uno o varios avisos asociadas a una transición. Si se pasa el parámetro *idAviso* se devuelve los datos de ese aviso asociada a la transición, si se pasa el parámetro *idTransicion* se filtra por la transición indicada, igualmente pasa con el parámetro *idDefProc* para la definición del procedimiento,si se pasa *null*, se devuelven todas.

Entradas

TpoPK idAviso	Identificador del aviso. Si se pasa *null* se obtienen todas.
TpoPK idTransicion	Identificador de la transición. Si se pasa *null* se obtienen todas.
TpoPK idDefProc	Identificador de la definición de procedimiento. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrAvisoTransicion
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrAvisoTransicion

Salida

TrAvisoTransicion[ ]	Array de objetos TrAvisoTransicion

Mantenimiento de condiciones de documentos

Insertar condición de un documento

void insertarCondicionDocumentoPermitido (
TrCondicionDocumentoPermitido condicionDocPer,
TpoPK idTipoDoc, TpoPK idDefProc,
TpoPK idFase, TpoPK idCondicion) throws TrException

Asocia una condición a un documento. Si todo es correcto devuelve los identificadores del tipo de documento, del procedimiento, de la fase y de la condición, si algo falla lanza una excepción.

Entradas

TrCondicionDocumentoPermitido condicionDocPer	Condición del documento a insertar

Entrada / Salida

TpoPK idTipoDoc	Identificador del tipo de documento o tarea, cero si no se ha insertado
TpoPK idDefProc	Identificador de la definición del procedimiento, cero si no se ha insertado
TpoPK idFase	Identificador de la fase, cero si no se ha insertado
TpoPK idCondicion	Identificador de la condición, cero si no se ha insertado

Modificar condición de un documento

int modificarCondicionDocumentoPermitido(
TrCondicionDocumentoPermitido condicionDocPer) throws TrException

Modifica una condición de un documento existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrCondicionDocumentoPermitido condicionDocPer	Condición del documento a modificar

Salida

int	Número de filas modificadas

Eliminar condición de un documento

int eliminarCondicionDocumentoPermitido(TpoPK idTipoDoc,

```
                                        TpoPK idDefProc,
                                        TpoPK idFase,
                                        TpoPK idCondicion ) throws TrException
```

Elimina una condición de un documento existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoDoc	Identificador del tipo de documento o tarea
TpoPK idDefProc	Identificador de la definición de procedimiento
TpoPK idFase	Identificador de la fase
TpoPK idCondicion	Identificador de la condición

Salida

int	Número de filas eliminadas

Obtener condición de un documento

```
TrCondicionDocumentoPermitido[ ] obtenerCondicionDocumentoPermitido(
                                        TpoPK idTipoDoc,
                                        TpoPK idDefProc,
                                        TpoPK idFase,
                                        TpoPK idCondicion,
                                        ClausulaWhere where,
                                        ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener una o varias condiciones de los documentos. Si no se pasa ningún parámetro se devuelven todas.

Entradas

TpoPK idTipoDoc	Identificador del tipo de documento o tarea
TpoPK idDefProc	Identificador de la definición de procedimiento
TpoPK idFase	Identificador de la fase

TpoPK idCondicion	Identificador de la condición
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para *TrCondicionDocumentoPermitido*
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para *TrCondicionDocumentoPermitido*

Salida

TrCondicionDocumentoPermitido[ ]	Array de objetos *TrCondicionDocumentoPermitido*

Mantenimiento de acciones de documentos

Insertar acción de un documento

```
void insertarAccionDocumentoPermitido (
                    TrAccionDocumentoPermitido accionDocPer,
                    TpoPK idTipoDoc, TpoPK idDefProc,
                    TpoPK idFase, TpoPK idAccion) throws TrException
```

Asocia una acción a un documento. Si todo es correcto devuelve los identificadores del tipo de documento, del procedimiento, de la fase y de la acción, si algo falla lanza una excepción.

Entradas

TrAccionDocumentoPermitido accionDocPer	Acción del documento a insertar

Entrada / Salida

TpoPK idTipoDoc	Identificador del tipo de documento o tarea, cero si no se ha insertado
TpoPK idDefProc	Identificador de la definición del procedimiento, cero si no se ha insertado
TpoPK idFase	Identificador de la fase, cero si no se ha insertado
TpoPK idAccion	Identificador de la acción, cero si no se ha insertado

Modificar acción de un documento

```
int modificarAccionDocumentoPermitido(
                    TrAccionDocumentoPermitido accionDocPer ) throws TrException
```

Modifica una acción de un documento existente. Si todo es correcto devuelve el

número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrAccionDocumentoPermitido accionDocPer	Acción del documento a modificar

Salida

int	Número de filas modificadas

Eliminar acción de un documento

```
int eliminarAccionDocumentoPermitido( TpoPK idTipoDoc,
                                      TpoPK idDefProc,
                                      TpoPK idFase,
                                      TpoPK idAccion ) throws TrException
```

Elimina una acción de un documento existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoDoc	Identificador del tipo de documento o tarea
TpoPK idDefProc	Identificador de la definición de procedimiento
TpoPK idFase	Identificador de la fase
TpoPK idAccion	Identificador de la acción

Salida

int	Número de filas eliminadas

Obtener acción de un documento

```
TrAccionDocumentoPermitido[ ] obtenerAccionDocumentoPermitido(
                                      TpoPK idTipoDoc,
                                      TpoPK idDefProc,
                                      TpoPK idFase,
                                      TpoPK idAccion,
                                      ClausulaWhere where,
```

ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias acciones de los documentos. Si no se pasa ningún parámetro se devuelven todas.

Entradas

TpoPK idTipoDoc	Identificador del tipo de documento o tarea
TpoPK idDefProc	Identificador de la definición de procedimiento
TpoPK idFase	Identificador de la fase
TpoPK idAccion	Identificador de la acción
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para *TrAccionDocumentoPermitido*
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para *TrAccionDocumentoPermitido*

Salida

TrAccionDocumentoPermitido[ ]	Array de objetos *TrAccionDocumentoPermitido*

Mantenimiento de avisos de documentos

Insertar aviso de un documento

void insertarAvisoDocumentoPermitido (
TrAvisoDocumentoPermitido avisoDocPer,
TpoPK idTipoDoc, TpoPK idDefProc,
TpoPK idFase, TpoPK idAviso) throws TrException

Asocia un aviso a un documento. Si todo es correcto devuelve los identificadores del tipo de documento, del procedimiento, de la fase y del aviso, si algo falla lanza una excepción.

Entradas

TrAvisoDocumentoPermitido avisoDocPer	Aviso del documento a insertar

Entrada / Salida

TpoPK idTipoDoc	Identificador del tipo de documento o tarea, cero si no se ha insertado

TpoPK idDefProc	Identificador de la definición del procedimiento, cero si no se ha insertado
TpoPK idFase	Identificador de la fase, cero si no se ha insertado
TpoPK idAviso	Identificador del aviso, cero si no se ha insertado

Modificar aviso de un documento

```
int modificarAvisoDocumentoPermitido(
        TrAvisoDocumentoPermitido avisoDocPer ) throws TrException
```

Modifica un aviso de un documento existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrAvisoDocumentoPermitido avisoDocPer	Aviso del documento a modificar

Salida

int	Número de filas modificadas

Eliminar aviso de un documento

```
int eliminarAvisoDocumentoPermitido( TpoPK idTipoDoc,
                                     TpoPK idDefProc,
                                     TpoPK idFase,
                                     TpoPK idAviso ) throws TrException
```

Elimina un aviso de un documento existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoDoc	Identificador del tipo de documento o tarea
TpoPK idDefProc	Identificador de la definición de procedimiento
TpoPK idFase	Identificador de la fase
TpoPK idAviso	Identificador del aviso

Salida

int	Número de filas eliminadas

Obtener aviso de un documento

```
TrAvisoDocumentoPermitido[ ] obtenerAvisoDocumentoPermitido(
                                TpoPK idTipoDoc,
                                TpoPK idDefProc,
                                TpoPK idFase,
                                TpoPK idAviso,
                                ClausulaWhere where,
                                ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener uno o varios avisos de los documentos. Si no se pasa ningún parámetro se devuelven todos.

Entradas

TpoPK idTipoDoc	Identificador del tipo de documento o tarea
TpoPK idDefProc	Identificador de la definición de procedimiento
TpoPK idFase	Identificador de la fase
TpoPK idAviso	Identificador del aviso
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para *TrAvisoDocumentoPermitido*
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para *TrAvisoDocumentoPermitido*

Salida

TrAvisoDocumentoPermitido[ ]	Array de objetos *TrAvisoDocumentoPermitido*

Mantenimiento de condiciones de otras tareas

Insertar condición de otras tareas

```
void insertarCondicionBloquePermitido(
                                TrCondicionBloquePermitido condicionBloquePer,
                                TpoPK idCondicion,
```

```
                                    TpoPK idDefProc,
                                    TpoPK idBloquePer ) throws TrException
```

Asocia una condición a una tarea en fase o bloque permitido. Si todo es correcto devuelve los identificadores de la condición, de la definición del procedimiento y del bloque permitido, si algo falla lanza una excepción.

Entradas

TrCondicionBloquePermitido condicionBloquePer	Condición del bloque permitido a insertar

Entrada / Salida

TpoPK idCondicion	Identificador de la condición, cero si no se ha insertado
TpoPK idDefProc	Identificador de la definición del procedimiento, cero si no se ha insertado
TpoPK idBloquePer	Identificador de la tarea en fase o bloque permitido, cero si no se ha insertado

Modificar condición de otras tareas

```
int modificarCondicionBloquePermitido(
                    TrCondicionBloquePermitido condicionBloquePer) throws TrException
```

Modifica una condición de un bloque permitido existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrCondicionBloquePermitido condicionBloquePer	Condición del bloque permitido a modificar

Salida

int	Número de filas modificadas

Eliminar condición de otras tareas

```
int eliminarCondicionBloquePermitido( TpoPK idCondicion,
                                      TpoPK idDefProc,
                                      TpoPK idBloquePer) throws TrException
```

Elimina una condición asociada a un bloque permitido existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idCondicion	Identificador de la condición
TpoPK idDefProc	Identificador de la definición del procedimiento
TpoPK idBloquePer	Identificador del bloque permitido o tarea en fase

Salida

int	Número de filas eliminadas

Obtener condición de otras tareas

```
TrCondicionBloquePermitido[ ] obtenerCondicionBloquePermitido(
                                TpoPK idCondicion,
                                TpoPK idDefProc,
                                TpoPK idBloquePer,
                                ClausulaWhere where,
                                ClausulaOrderBy orderBy) throws TrException
```

Permite obtener una o varias condiciones asociadas a un bloque permitido. Si no se pasa ningún parámetro se devuelve todas.

Entradas

TpoPK idCondicion	Identificador de la condición
TpoPK idDefProc	Identificador de la definición del procedimiento
TpoPK idBloquePer	Identificador del bloque permitido o tarea en fase
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrCondicionBloquePermitido
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrCondicionBloquePermitido

Salida

TrCondicionBloquePermitido[ ]	Array de objetos TrCondicionBloquePermitido

Mantenimiento de acciones de otras tareas

Insertar acción de otras tareas

```
void insertarAccionBloquePermitido(
                                    TrAccionBloquePermitido accionBloquePer,
                                    TpoPK idAccion,
                                    TpoPK idDefProc,
                                    TpoPK idBloquePer ) throws TrException
```

Asocia una acción a una tarea en fase o bloque permitido. Si todo es correcto devuelve los identificadores de la acción, de la definición del procedimiento y del bloque permitido, si algo falla lanza una excepción.

Entradas

TrAccionBloquePermitido accionBloquePer	Acción del bloque permitido a insertar

Entrada / Salida

TpoPK idAccion	Identificador de la acción, cero si no se ha insertado
TpoPK idDefProc	Identificador de la definición del procedimiento, cero si no se ha insertado
TpoPK idBloquePer	Identificador de la tarea en fase o bloque permitido, cero si no se ha insertado

Modificar acción de otras tareas

```
int modificarAccionBloquePermitido(
                    TrAccionBloquePermitido accionBloquePer) throws TrException
```

Modifica una acción de un bloque permitido existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrAccionBloquePermitido accionBloquePer	Acción del bloque permitido a modificar

Salida

int	Número de filas modificadas

Eliminar acción de otras tareas

```
int eliminarAccionBloquePermitido( TpoPK idAccion,
```

TpoPK idDefProc, TpoPK idBloquePer) throws TrException

Elimina una acción asociada a un bloque permitido existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idAccion	Identificador de la acción
TpoPK idDefProc	Identificador de la definición del procedimiento
TpoPK idBloquePer	Identificador del bloque permitido o tarea en fase

Salida

int	Número de filas eliminadas

Obtener acción de otras tareas

TrAccionBloquePermitido[ ] obtenerAccionBloquePermitido(TpoPK idAccion, TpoPK idDefProc, TpoPK idBloquePer, ClausulaWhere where, ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias acciones asociadas a un bloque permitido. Si no se pasa ningún parámetro se devuelve todas.

Entradas

TpoPK idAccion	Identificador de la acción
TpoPK idDefProc	Identificador de la definición del procedimiento
TpoPK idBloquePer	Identificador del bloque permitido o tarea en fase
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrAccionBloquePermitido
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrAccionBloquePermitido

Salida

TrAccionBloquePermitido[ ]	Array de objetos TrAccionBloquePermitido

Mantenimiento de avisos de otras tareas

Insertar aviso de otras tareas

```
void insertarAvisoBloquePermitido(
                    TrAvisoBloquePermitido avisoBloquePer,
                    TpoPK idAviso,
                    TpoPK idDefProc,
                    TpoPK idBloquePer ) throws TrException
```

Asocia un aviso a una tarea en fase o bloque permitido. Si todo es correcto devuelve los identificadores del aviso, de la definición del procedimiento y del bloque permitido, si algo falla lanza una excepción.

Entradas

TrAvisoBloquePermitido avisoBloquePer	Aviso del bloque permitido a insertar

Entrada / Salida

TpoPK idAviso	Identificador del aviso, cero si no se ha insertado
TpoPK idDefProc	Identificador de la definición del procedimiento, cero si no se ha insertado
TpoPK idBloquePer	Identificador de la tarea en fase o bloque permitido, cero si no se ha insertado

Modificar aviso de otras tareas

```
int modificarAvisoBloquePermitido(
                    TrAvisoBloquePermitido avisoBloquePer) throws TrException
```

Modifica un aviso de un bloque permitido existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrAvisoBloquePermitido avisoBloquePer	Aviso del bloque permitido a modificar

Salida

int	Número de filas modificadas

Eliminar aviso de otras tareas

```
int eliminarAvisoBloquePermitido( TpoPK idAviso,
                                  TpoPK idDefProc,
                                  TpoPK idBloquePer) throws TrException
```

Elimina un aviso asociado a un bloque permitido existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idAviso	Identificador del aviso
TpoPK idDefProc	Identificador de la definición del procedimiento
TpoPK idBloquePer	Identificador del bloque permitido o tarea en fase

Salida

int	Número de filas eliminadas

Obtener aviso de otras tareas

```
TrAvisoBloquePermitido[ ] obtenerAvisoBloquePermitido(
                                  TpoPK idAviso,
                                  TpoPK idDefProc,
                                  TpoPK idBloquePer,
                                  ClausulaWhere where,
                                  ClausulaOrderBy orderBy) throws TrException
```

Permite obtener uno o varios avisos asociados a un bloque permitido. Si no se pasa ningún parámetro se devuelve todos.

Entradas

TpoPK idAviso	Identificador del aviso
TpoPK idDefProc	Identificador de la definición del procedimiento
TpoPK idBloquePer	Identificador del bloque permitido o tarea en fase

ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrAvisoBloquePermitido
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrAvisoBloquePermitido

Salida

TrAvisoBloquePermitido[ ]	Array de objetos TrAvisoBloquePermitido

Mantenimiento de datos de un componente

Insertar datos de un componente

```
TpoPK insertarDatoComponente(
                    TrDatoComponente datoComponente ) throws TrException
```

Inserta un nuevo dato de un componente. Si todo es correcto devuelve el identificador de la nueva dato de un componente, si algo falla lanza una excepción.

Entradas

TrDatoComponente datoComponente	Dato de un componente a insertar

Salida

TpoPK	Identificador del nuevo dato de un componente, cero si no se ha insertado

Modificar datos de un componente

```
int modificarDatoComponente(
                    TrDatoComponente  datoComponente ) throws TrException
```

Modifica una dato de un componente existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrDatoComponente datoComponente	Dato de un componente a modificar

Salida

int	Número de filas modificadas

Eliminar datos de un componente

int eliminarDatoComponente(TpoPK idDatoComponente) throws TrException

Elimina un dato de un componente existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idDatoComponente	Identificador del dato de un componente a eliminar

Salida

int	Número de filas eliminadas

Obtener datos de un componente

TrDatoComponente [ ] obtenerDatoComponente(TpoPK idDatoComponente,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios datos de un componente. Si se pasa el parámetro *idDatoComponente* se devuelve los datos de esa dato de un componente, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idDatoComponente	Identificador del dato de un componente a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrDatoComponente
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrDatoComponente

Salida

TrDatoComponente[ ]	Array de objetos TrDatoComponente

Mantenimiento de componentes

Insertar componente

TpoPK insertarComponente(TrComponente componente) throws TrException

Inserta un nuevo componente. Si todo es correcto devuelve el identificador de la nueva

componente, si algo falla lanza una excepción.

Entradas

TrComponente componente	Componente a insertar

Salida

TpoPK	Identificador del nuevo componente, cero si no se ha insertado

Modificar componente

int modificarComponente(TrComponente componente) throws TrException

Modifica una componente existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrComponente componente	Componente a modificar

Salida

int	Número de filas modificadas

Eliminar componente

int eliminarComponente(TpoPK idComponente) throws TrException

Elimina un componente existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idComponente	Identificador del componente a eliminar

Salida

int	Número de filas eliminadas

Obtener componente

TrComponente [ ] obtenerComponente(TpoPK idComponente,
ClausulaWhere where,

ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios componentes. Si se pasa el parámetro *idComponente* se devuelve los datos de ese componente, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idComponente	Identificador del componente a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrComponente
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrComponente

Salida

TrComponente[ ]	Array de objetos TrComponente

Mantenimiento de constantes del sistema

Insertar constante

String insertarConstante(TrConstante constante) throws TrException

Inserta una nueva constante. Si todo es correcto devuelve el identificador de la nueva constante, si algo falla lanza una excepción.

Entradas

TrConstante constante	Constante a insertar

Salida

TpoPK	Identificador de la nueva constante, null si no se ha insertado

Modificar constante

int modificarConstante(TrConstante constante) throws TrException

Modifica una constante existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrConstante constante	constante a modificar

Salida

int	Número de filas modificadas

Eliminar constante

int eliminarConstante(String idConstante) throws TrException

Elimina una constante existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idConstante	Identificador de la constante a eliminar

Salida

int	Número de filas eliminadas

Obtener constante

TrConstante [ ] obtenerConstante(TpoPK idConstante,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias constantes. Si se pasa el parámetro *idConstante* se devuelve los datos de esa constante, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idConstante	Identificador de la constante a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrConstante
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrConstante

Salida

TrConstante[ ]	Array de objetos TrConstante

Mantenimiento de tipos de actos

Insertar tipo de acto

TpoPK insertarTipoActo(TrTipoActo tipoActo) throws TrException

Inserta un nuevo tipo de acto. Si todo es correcto devuelve el identificador del nuevo tipo de acto, si algo falla lanza una excepción.

Entradas

TrTipoActo tipoActo	Tipo de acto a insertar

Salida

TpoPK	Identificador del nueva tipo de acto, cero si no se ha insertado

Modificar tipo de acto

int modificarTipoActo(TrTipoActo tipoActo) throws TrException

Modifica un tipo de acto existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrTipoActo tipoActo	Tipo de acto a modificar

Salida

int	Número de filas modificadas

Eliminar tipo de acto

int eliminarTipoActo(TpoPK idTipoActo) throws TrException

Elimina un tipo de acto existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoActo	Identificador del tipo de acto a eliminar

Salida

int	Número de filas eliminadas

Obtener tipo de acto

```
TrTipoActo [ ] obtenerTipoActo( TpoPK idTipoActo,
                                ClausulaWhere where,
                                ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener uno o varios tipos de actos. Si se pasa el parámetro *idTipoActo* se devuelve los datos de ese tipo de acto, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idTipoActo	Identificador del tipo de acto a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoActo
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoActo

Salida

TrTipoActo[ ]	Array de objetos TrTipoActo

Mantenimiento de perfiles de usuario de una transición

Insertar perfil de usuario de una transición

```
void insertarTransicionPerfil ( TrTransicionPerfil transicionPerfil,
                                TpoPK idDefProc,
                                TpoPK idTransicion,
                                TpoPK idPerfilUsu) throws TrException
```

Permite asociar un perfil de usuario a una transición. Si todo es correcto devuelve los identificadores de la transición, de la definición del procedimiento y del perfil de usuario, si algo falla lanza una excepción.

Entradas

TrTransicionPerfil transicionPerfil	Perfil y transición a asociar

Entrada / Salida

TpoPK idDefProc	Identificador de la definición del procedimiento, cero si no se ha insertado
TpoPK idTransicion	Identificador de la transición, cero si no se ha insertado

TpoPK idPerfilUsu	Identificador del perfil de usuario, cero si no se ha insertado

Modificar perfil de usuario de una transición

```
int modificarTransicionPerfil( TrTransicionPerfil transicionPerfil) throws TrException
```

Modifica una relación entre una transición y un perfil de usuario existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrTransicionPerfil transicionPerfil	Perfil y transición a modificar

Salida

int	Número de filas modificadas

Eliminar perfil de usuario de una transición

```
int eliminarTransicionPerfil( TpoPK idDefProc,
                              TpoPK idTransicion,
                              TpoPK idPerfilUsu ) throws TrException
```

Elimina una relación entre una transición y un perfil de usuario existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idDefProc	Identificador de la definición del procedimiento
TpoPK idTransicion	Identificador de la transición
TpoPK idPerfilUsu	Identificador del perfil de usuario

Salida

int	Número de filas eliminadas

Obtener perfil de usuario de una transición

```
TrTransicionPerfil[ ] obtenerTransicionPerfil( TpoPK idDefProc,
                                               TpoPK idTransicion,
```

```
                                    TpoPK idPerfilUsu,
                                    ClausulaWhere where,
                                     ClausulaOrderBy orderBy) throws TrException
```

Permite obtener una o varias relaciones entre transiciones y perfiles de usuario. Si no se pasa ningún parámetro se devuelve todas y si se pasa alguno se filtra por él.

Entradas

TpoPK idDefProc	Identificador de la definición de procedimiento. Si se pasa *null* se obtienen todas.
TpoPK idTransicion	Identificador de la transición. Si se pasa *null* se obtienen todas.
TpoPK idPerfilUsu	Identificador del perfil de usuario. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTransicionPerfil
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTransicionPerfil

Salida

TrTransicionPerfil[ ]	Array de objetos TrTransicionPerfil

Mantenimiento de límites de caducidades

Insertar límite de caducidad

```
void insertarLimiteCaducidad ( TrLimiteCaducidad limiteCaducidad,
                               TpoPK idCaducidad,
                               TpoPK idTransicion) throws TrException;
```

Inserta una nuevo límite para una caducidad. Si todo es correcto devuelve el identificador de la caducidad y de la transición, si algo falla lanza una excepción.

Entradas

TrLimiteCaducidad limiteCaducidad	Límite para una caducidad a insertar

Entrada / Salida

TpoPK idCaducidad	Identificador de la caducidad, cero si no se ha insertado
TpoPK idTransicion	Identificador de la transición, cero si no se ha insertado

Eliminar límite de caducidad

```
int eliminarLimiteCaducidad( TpoPK idCaducidad,
                             TpoPK idTransicion) throws TrException
```

Elimina un límite para una caducidad existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idCaducidad	Identificador de la caducidad
TpoPK idTransicion	Identificador de la transición

Salida

int	Número de filas eliminadas

Obtener límite de caducidad

```
TrLimiteCaducidad[ ] obtenerLimiteCaducidad( TpoPK idCaducidad,
                                             TpoPK idTransicion,
                                             ClausulaWhere where,
                                             ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener uno o varios límites para una caducidad. Si no se pasa ningún parámetros se devuelve todos los límites de caducidad de todas las transiciones y si se pasa algún parámetro se filtra el resultado por ese identificador.

Entradas

TpoPK idCaducidad	Identificador de la caducidad. Si se pasa *null* se obtienen todas.
TpoPK idTransicion	Identificador de la transición. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrLimiteCaducidad
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrLimiteCaducidad

Salida

TrLimiteCaducidad[ ]	Array de objetos TrLimiteCaducidad

Mantenimiento de documentos delegados

Insertar documento delegado

TpoPK insertarDocumentoDelegado(TrDocumentoDelegado docDelegado) throws TrException

Inserta un nuevo documento delegado. Si todo es correcto devuelve el identificador del nuevo documento delegado, si algo falla lanza una excepción.

Entradas

TrDocumentoDelegado docDelegado	Documento delegado a insertar

Salida

TpoPK	Identificador del nuevo documento delegado, cero si no se ha insertado

Modificar documento delegado

int modificarDocumentoDelegado(TrDocumentoDelegado docDelegado) throws TrException

Modifica un documento delegado existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrDocumentoDelegado docDelegado	Documento delegado a modificar

Salida

int	Número de filas modificadas

Eliminar documento delegado

int eliminarDocumentoDelegado(TpoPK idDocDelegado) throws TrException

Elimina un documento delegado existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idDocDelegado	Identificador del documento delegado a eliminar

Salida

int	Número de filas eliminadas

Obtener documento delegado

TrDocumentoDelegado [ ] obtenerDocumentoDelegado(TpoPK idDocDelegado,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios documentos delegados. Si se pasa el parámetro *idDocDelegado* se devuelve los datos de esa documento delegado, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idDocDelegado	Identificador del documento delegado a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrDocumentoDelegado
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrDocumentoDelegado

Salida

TrDocumentoDelegado[ ]	Array de objetos TrDocumentoDelegado

Mantenimiento de ficha del procedimiento

Insertar ficha del procedimiento

TpoPK insertarIndicacionFicha(TrIndicacionFicha indicacionFicha) throws TrException

Inserta una nueva ficha para el procedimiento. Si todo es correcto devuelve el identificador de la nueva ficha para el procedimiento, si algo falla lanza una excepción.

Entradas

TrIndicacionFicha indicacionFicha	Ficha para el procedimiento a insertar

Salida

TpoPK	Identificador de la nueva ficha para el procedimiento, cero si no se ha insertado

Modificar ficha del procedimiento

int modificarIndicacionFicha(TrIndicacionFicha indicacionFicha) throws TrException

Modifica una ficha para el procedimiento existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrIndicacionFicha indicacionFicha	Ficha para el procedimiento a modificar

Salida

int	Número de filas modificadas

Eliminar ficha del procedimiento

int eliminarIndicacionFicha(TpoPK idIndFicha) throws TrException

Elimina una ficha para el procedimiento existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idIndFicha	Identificador de la ficha para el procedimiento a eliminar

Salida

int	Número de filas eliminadas

Obtener ficha del procedimiento

TrIndicacionFicha [ ] obtenerIndicacionFicha(TpoPK idIndFicha,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias fichas del procedimiento. Si se pasa el parámetro *idIndFicha* se devuelve los datos de esa ficha para el procedimiento, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idIndFicha	Identificador de la ficha para el procedimiento a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrIndicacionFicha
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrIndicacionFicha

Salida

TrIndicacionFicha[ ]	Array de objetos TrIndicacionFicha

Mantenimiento de normativas

Insertar normativa

TpoPK insertarNormativa(TrNormativa normativa) throws TrException

Inserta una nueva normativa. Si todo es correcto devuelve el identificador de la nueva normativa, si algo falla lanza una excepción.

Entradas

TrNormativa normativa	Normativa a insertar

Salida

TpoPK	Identificador de la nueva normativa, cero si no se ha insertado

Modificar normativa

int modificarNormativa(TrNormativa normativa) throws TrException

Modifica una normativa existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrNormativa normativa	normativa a modificar

Salida

int	Número de filas modificadas

Eliminar normativa

int eliminarNormativa(TpoPK idNormativa) throws TrException

Elimina una normativa existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idNormativa	Identificador de la normativa a eliminar

Salida

int	Número de filas eliminadas

Obtener normativa

TrNormativa [ ] obtenerNormativa(TpoPK idNormativa,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias normativas. Si se pasa el parámetro *idNormativa* se devuelve los datos de esa normativa, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idNormativa	Identificador de la normativa a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrNormativa
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrNormativa

Salida

TrNormativa[ ]	Array de objetos TrNormativa

Mantenimiento de normativas del procedimiento

Insertar normativa del procedimiento

void insertarNormativaDefProcedimiento(

TrNormativaDefProcedimiento normativaDefProc, TpoPK idDefProc, TpoPK idNormativa) throws TrException

Permite asociar una normativa a un procedimiento. Si todo es correcto los identificadores de la normativa y de la definición del procedimiento, si algo falla lanza una excepción.

Entradas

TrNormativaDefProcedimiento normativaDefProc	Nomativa y procedimiento a relacionar

Entrada / Salida

TpoPK idDefProc	Identificador de la definición del procedimiento, cero si no se ha insertado
TpoPK idNormativa	Identificador de la normativa, cero si no se ha insertado

Eliminar normativa del procedimiento

int eliminarNormativaDefProcedimiento(TpoPK idDefProc, TpoPK idNormativa) throws TrException

Elimina una normativa asociada a un procedimiento existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idDefProc	Identificador de la definición del procedimiento
TpoPK idNormativa	Identificador de la normativa

Salida

int	Número de filas eliminadas

Eliminar normativas del procedimiento

int eliminarNormativaDefProcedimiento(TpoPK idDefProc) throws TrException

Elimina todas las normativas asociadas a un procedimiento existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idDefProc	Identificador de la definición del procedimiento

Salida

int	Número de filas eliminadas

Obtener normativa del procedimiento

```
TrNormativaDefProcedimiento[ ] obtenerNormativaDefProcedimiento(
                                    TpoPK idDefProc,
                                    TpoPK idNormativa,
                                    ClausulaWhere where,
                                    ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener una o varias normativas asociadas a algún procedimiento. Si no se pasa ningún parámetro se obtienen todas las normativas de todos los procedimientos y si se pasa alguno de los identificadores se filtrará por él.

Entradas

TpoPK idDefProc	Identificador de la definición del procedimiento. Si se pasa *null* se obtienen todas.
TpoPK idNormativa	Identificador de la normativa. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrNormativaDefProcedimiento
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrNormativaDefProcedimiento

Salida

TrNormativaDefProcedimiento[ ]	Array de objetos TrNormativaDefProcedimiento

Mantenimiento de organismos

Insertar organismo

```
TpoPK insertarOrganismo(TrOrganismo organismo ) throws TrException
```

Inserta un nuevo organismo. Si todo es correcto devuelve el identificador del nuevo organismo, si algo falla lanza una excepción.

Entradas

TrOrganismo organismo	Organismo a insertar

Salida

TpoPK	Identificador del nuevo organismo, cero si no se ha insertado

Modificar organismo

```
int modificarOrganismo(  TrOrganismo  organismo ) throws TrException
```

Modifica un organismo existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrOrganismo organismo	Organismo a modificar

Salida

int	Número de filas modificadas

Eliminar organismo

```
int eliminarOrganismo( TpoPK idOrganismo) throws TrException
```

Elimina un organismo existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idOrganismo	Identificador del organismo a eliminar

Salida

int	Número de filas eliminadas

Obtener organismo

```
TrOrganismo [ ] obtenerOrganismo( TpoPK idOrganismo,
                                  ClausulaWhere where,
                                  ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener uno o varios organismos. Si se pasa el parámetro *idOrganismo* se

devuelve los datos de ese organismo, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idOrganismo	Identificador del organismo a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrOrganismo
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrOrganismo

Salida

TrOrganismo[ ]	Array de objetos TrOrganismo

Mantenimiento de plantillas del procedimiento

Insertar plantilla del procedimiento

TpoPK insertarPlantillaProcedimiento(TrPlantillaProcedimiento plantillaProc) throws TrException

Inserta una nueva plantilla del procedimiento. Si todo es correcto devuelve el identificador de la nueva plantilla del procedimiento, si algo falla lanza una excepción.

Entradas

TrPlantillaProcedimiento plantillaProc	Plantilla del procedimiento a insertar

Salida

TpoPK	Identificador de la nueva plantilla del procedimiento, cero si no se ha insertado

Modificar plantilla del procedimiento

int modificarPlantillaProcedimiento(TrPlantillaProcedimiento plantillaProc) throws TrException

Modifica una plantilla del procedimiento existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrPlantillaProcedimiento plantillaProc	plantilla del procedimiento a modificar

Salida

int	Número de filas modificadas

Eliminar plantilla del procedimiento

int eliminarPlantillaProcedimiento(TpoPK idPlantillaProc) throws TrException

Elimina una plantilla del procedimiento existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idPlantillaProc	Identificador de la plantilla del procedimiento a eliminar

Salida

int	Número de filas eliminadas

Obtener plantilla del procedimiento

TrPlantillaProcedimiento [ ] obtenerPlantillaProcedimiento(TpoPK idPlantillaProc,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias plantillas del procedimiento. Si se pasa el parámetro *idPlantillaProc* se devuelve los datos de esa plantilla del procedimiento, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idPlantillaProc	Identificador de la plantilla del procedimiento a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrPlantillaProcedimiento
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrPlantillaProcedimiento

Salida

TrPlantillaProcedimiento[ ]	Array de objetos TrPlantillaProcedimiento

Mantenimiento de errores

Insertar error

TpoPK insertarError(TrError error) throws TrException

Inserta un nuevo error. Si todo es correcto devuelve el identificador de la nueva error, si algo falla lanza una excepción.

Entradas

TrError error	Error a insertar

Salida

TpoPK	Identificador del nuevo error, cero si no se ha insertado

Modificar error

int modificarError(TrError error) throws TrException

Modifica un error existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrError error	Error a modificar

Salida

int	Número de filas modificadas

Eliminar error

int eliminarError(TpoPK idError) throws TrException

Elimina un error existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idError	Identificador del error a eliminar

Salida

int	Número de filas eliminadas

Obtener error

TrError [ ] obtenerError(TpoPK idError, ClausulaWhere where, ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios errores Si se pasa el parámetro *idError* se devuelve los datos de ese error, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idError	Identificador del error a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrError
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrError

Salida

TrError[ ]	Array de objetos TrError

Mantenimiento de firmantes definidos

Insertar firmante definido

TpoPK insertarFirmanteDefinido(TrFirmanteDefinido firmanteDefinido) throws TrException

Inserta un nuevo firmante definido. Si todo es correcto devuelve el identificador de la nueva firmante definido, si algo falla lanza una excepción.

Entradas

TrFirmanteDefinido firmanteDefinido	Firmante definido a insertar

Salida

TpoPK	Identificador del nuevo firmante definido, cero si no se ha insertado

Modificar firmante definido

int modificarFirmanteDefinido(TrFirmanteDefinido firmanteDefinido) throws TrException

Modifica un firmante definido existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrFirmanteDefinido firmanteDefinido	Firmante definido a modificar

Salida

int	Número de filas modificadas

Eliminar firmante definido

int eliminarFirmanteDefinido(TpoPK idFirmanteDef) throws TrException

Elimina un firmante definido existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idFirmanteDef	Identificador del firmante definido a eliminar

Salida

int	Número de filas eliminadas

Obtener firmante definido

TrFirmanteDefinido [ ] obtenerFirmanteDefinido(TpoPK idFirmanteDef,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios firmantes definidos. Si se pasa el parámetro *idFirmanteDef* se devuelve los datos de ese firmante definido, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idFirmanteDef	Identificador del firmante definido a obtener. Si se pasa *null* se obtienen todos.

ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrFirmanteDefinido
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrFirmanteDefinido

Salida

TrFirmanteDefinido[ ]	Array de objetos TrFirmanteDefinido

Mantenimiento de firmas

Insertar firma

TpoPK insertarFirma(TrFirma firma) throws TrException

Inserta una nueva firma. Si todo es correcto devuelve el identificador de la nueva firma, si algo falla lanza una excepción.

Entradas

TrFirma firma	Firma a insertar

Salida

TpoPK	Identificador de la nueva firma, cero si no se ha insertado

Modificar firma

int modificarFirma(TrFirma firma) throws TrException

Modifica una firma existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrFirma firma	Firma a modificar

Salida

int	Número de filas modificadas

Eliminar firma

int eliminarFirma(TpoPK idFirma) throws TrException

Elimina una firma existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idFirma	Identificador de la firma a eliminar

Salida

int	Número de filas eliminadas

Obtener firma

TrFirma [ ] obtenerFirma(TpoPK idFirma,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias firmas. Si se pasa el parámetro *idFirma* se devuelve los datos de esa firma, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idFirma	Identificador de la firma a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrFirma
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrFirma

Salida

TrFirma[ ]	Array de objetos TrFirma

Mantenimiento de plantillas

Insertar plantilla

TpoPK insertarPlantilla(TrPlantilla plantilla) throws TrException

Inserta una nueva plantilla para la generación de documentos. Si todo es correcto devuelve el identificador de la nueva plantilla, si algo falla lanza una excepción.

Entradas

TrPlantilla plantilla	Plantilla a insertar

Salida

TpoPK	Identificador de la nueva plantilla, cero si no se ha insertado

Modificar plantilla

```
int modificarPlantilla(  TrPlantilla  plantilla ) throws TrException
```

Modifica una plantilla para la generación de documentos existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrPlantilla plantilla	Plantilla a modificar

Salida

int	Número de filas modificadas

Eliminar plantilla

```
int eliminarPlantilla( TpoPK idPlantilla) throws TrException
```

Elimina una plantilla para la generación de documentos existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idPlantilla	Identificador de la plantilla a eliminar

Salida

int	Número de filas eliminadas

Obtener plantilla

```
TrPlantilla [ ] obtenerPlantilla( TpoPK idPlantilla,
                                  ClausulaWhere where,
                                  ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener una o varias plantillas para la generación de documentos. Si se pasa el parámetro *idPlantilla* se devuelve los datos de esa plantilla, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idPlantilla	Identificador de la plantilla a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrPlantilla
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrPlantilla

Salida

TrPlantilla[ ]	Array de objetos TrPlantilla

Mantenimiento de relaciones

Insertar relación

TpoPK insertarRelacion(TrRelacion relacion) throws TrException

Inserta una nueva relación entre fases, transiciones, tipos de documentos, etc. Si todo es correcto devuelve el identificador de la nueva relación, si algo falla lanza una excepción.

Entradas

TrRelacion relacion	relación a insertar

Salida

TpoPK	Identificador de la nueva relación, cero si no se ha insertado

Modificar relación

int modificarRelacion(TrRelacion relacion) throws TrException

Modifica una relación existente entre fases, transiciones, tipos de documentos, etc. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrRelacion relacion	relación a modificar

Salida

int	Número de filas modificadas

Eliminar relación

int eliminarRelacion(TpoPK idRelacion) throws TrException

Elimina una relación existente entre fases, transiciones, tipos de documentos, etc. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idRelacion	Identificador de la relación a eliminar

Salida

int	Número de filas eliminadas

Obtener relación

TrRelacion [ ] obtenerRelacion(TpoPK idRelacion,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias relaciones entre fases, transiciones, tipos de documentos, etc. Si se pasa el parámetro *idRelacion* se devuelve los datos de esa relación, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idRelacion	Identificador de la relación a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrRelacion
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrRelacion

Salida

TrRelacion[ ]	Array de objetos TrRelacion

Mantenimiento de disposiciones de firma

Insertar disposición para la firma

TpoPK insertarTextoDisposicionFirma(TrTextoDisposicionFirma textoDisp) throws TrException

Inserta una nueva disposición para la firma. Si todo es correcto devuelve el identificador de la nueva disposición para la firma, si algo falla lanza una excepción.

Entradas

TrTextoDisposicionFirma textoDisp	Disposición para la firma a insertar

Salida

TpoPK	Identificador de la nueva disposición para la firma, cero si no se ha insertado

Modificar disposición para la firma

int modificarTextoDisposicionFirma(TrTextoDisposicionFirma textoDisp) throws TrException

Modifica una disposición para la firma existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrTextoDisposicionFirma textoDisp	Disposición para la firma a modificar

Salida

int	Número de filas modificadas

Eliminar disposición para la firma

int eliminarTextoDisposicionFirma(TpoPK idTextoDisp) throws TrException

Elimina una disposición para la firma existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTextoDisp	Identificador de la disposición para la firma a eliminar

Salida

int	Número de filas eliminadas

Obtener disposición para la firma

```
TrTextoDisposicionFirma [ ] obtenerTextoDisposicionFirma(
                                TpoPK idTextoDisp,
                                ClausulaWhere where,
                                ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener una o varias disposiciones para las firmas. Si se pasa el parámetro *idTextoDisp* se devuelve los datos de esa disposición para la firma, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idTextoDisp	Identificador de la disposición para la firma a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTextoDisposicionFirma
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTextoDisposicionFirma

Salida

TrTextoDisposicionFirma[ ]	Array de objetos TrTextoDisposicionFirma

Mantenimiento de parámetros

Insertar parámetro

```
TpoPK insertarParametro(TrParametro parametro ) throws TrException
```

Inserta un nuevo parámetro. Si todo es correcto devuelve el identificador del nuevo parámetro, si algo falla lanza una excepción.

Entradas

TrParametro parametro	Parámetro a insertar

Salida

TpoPK	Identificador del nuevo parámetro, cero si no se ha insertado

Modificar parámetro

```
int modificarParametro( TrParametro parametro ) throws TrException
```

Modifica un parámetro existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrParametro parametro	Parámetro a modificar

Salida

int	Número de filas modificadas

Eliminar parámetro

```
int eliminarParametro( TpoPK idParametro) throws TrException
```

Elimina un parámetro existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idParametro	Identificador del parámetro a eliminar

Salida

int	Número de filas eliminadas

Obtener parámetro

```
TrParametro [ ] obtenerParametro( TpoPK idParametro,
                                  ClausulaWhere where,
                                  ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener uno o varios parámetros. Si se pasa el parámetro *idParametro* se devuelve los datos de ese parámetro, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idParametro	Identificador del parámetro a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrParametro
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrParametro

Salida

TrParametro[ ]	Array de objetos TrParametro

Mantenimiento de párrafos

Insertar párrafo

TpoPK insertarParrafoTipoDocumento(TrParrafoTipoDocumento parrafoTipoDoc) throws TrException

Inserta un nuevo párrafo para un tipo de documento. Si todo es correcto devuelve el identificador del nuevo párrafo, si algo falla lanza una excepción.

Entradas

TrParrafoTipoDocumento parrafoTipoDoc	Párrafo a insertar

Salida

TpoPK	Identificador del nuevo párrafo, cero si no se ha insertado

Modificar párrafo

int modificarParrafoTipoDocumento(TrParrafoTipoDocumento parrafoTipoDoc) throws TrException

Modifica un párrafo para un tipo de documento existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrParrafoTipoDocumento parrafoTipoDoc	Párrafo a modificar

Salida

int	Número de filas modificadas

Eliminar párrafo

int eliminarParrafoTipoDocumento(TpoPK idParrafoTipoDoc) throws TrException

Elimina un párrafo para un tipo de documento existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idParrafoTipoDoc	Identificador del párrafo a eliminar

Salida

int	Número de filas eliminadas

Obtener párrafo

TrParrafoTipoDocumento [ ] obtenerParrafoTipoDocumento(
TpoPK idParrafoTipoDoc,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios párrafos. Si se pasa el parámetro *idParrafoTipoDoc* se devuelve los datos de ese párrafo, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idParrafoTipoDoc	Identificador del párrafo a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrParrafoTipoDocumento
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrParrafoTipoDocumento

Salida

TrParrafoTipoDocumento[ ]	Array de objetos TrParrafoTipoDocumento

Mantenimiento de tipos de párrafos

Insertar tipo de párrafo

TpoPK insertarTipoParrafo(TrTipoParrafo tipoParrafo) throws TrException

Inserta un nuevo tipo de párrafo. Si todo es correcto devuelve el identificador del nuevo tipo de párrafo, si algo falla lanza una excepción.

Entradas

TrTipoParrafo tipoParrafo	Tipo de párrafo a insertar

Salida

TpoPK	Identificador del nuevo tipo de párrafo, cero si no se ha insertado

Modificar tipo de párrafo

int modificarTipoParrafo(TrTipoParrafo tipoParrafo) throws TrException

Modifica un tipo de párrafo existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrTipoParrafo tipoParrafo	Tipo de párrafo a modificar

Salida

int	Número de filas modificadas

Eliminar tipo de párrafo

int eliminarTipoParrafo(TpoPK idTipoParrafo) throws TrException

Elimina un tipo de párrafo existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoParrafo	Identificador del tipo de párrafo a eliminar

Salida

int	Número de filas eliminadas

Obtener tipo de párrafo

```
TrTipoParrafo [ ] obtenerTipoParrafo( TpoPK idTipoParrafo,
                                      ClausulaWhere where,
                                      ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener uno o varios tipos de párrafos. Si se pasa el parámetro *idTipoParrafo* se devuelve los datos de ese tipo de párrafo, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idTipoParrafo	Identificador del tipo de párrafo a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoParrafo
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoParrafo

Salida

TrTipoParrafo[ ]	Array de objetos TrTipoParrafo

Mantenimiento de tipos de relaciones

Insertar tipo de relación

```
TpoPK insertarTipoRelacion(TrTipoRelacion tipoRelacion ) throws TrException
```

Inserta un nuevo tipo de relación. Si todo es correcto devuelve el identificador del nuevo tipo de relación, si algo falla lanza una excepción.

Entradas

TrTipoRelacion tipoRelacion	Tipo de relación a insertar

Salida

TpoPK	Identificador del nuevo tipo de relación, cero si no se ha insertado

Modificar tipo de relación

```
int modificarTipoRelacion( TrTipoRelacion tipoRelacion ) throws TrException
```

Modifica un tipo de relación existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrTipoRelacion tipoRelacion	Tipo de relación a modificar

Salida

int	Número de filas modificadas

Eliminar tipo de relación

int eliminarTipoRelacion(TpoPK idTipoRelacion) throws TrException

Elimina un tipo de relación existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoRelacion	Identificador del tipo de relación a eliminar

Salida

int	Número de filas eliminadas

Obtener tipo de relación

TrTipoRelacion [ ] obtenerTipoRelacion(TpoPK idTipoRelacion,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios tipos de relación. Si se pasa el parámetro *idTipoRelacion* se devuelve los datos de ese tipo de relación, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idTipoRelacion	Identificador del tipo de relación a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoRelacion
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoRelacion

Salida

TrTipoRelacion[ ]	Array de objetos TrTipoRelacion

Mantenimiento de variables

Insertar variable

TpoPK insertarVariable(TrVariable variable) throws TrException

Inserta una nueva variable. Si todo es correcto devuelve el identificador de la nueva variable, si algo falla lanza una excepción.

Entradas

TrVariable variable	ariable a insertar

Salida

TpoPK	Identificador de la nueva variable, cero si no se ha insertado

Modificar variable

int modificarVariable(TrVariable variable) throws TrException

Modifica una variable existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrVariable variable	Variable a modificar

Salida

int	Número de filas modificadas

Eliminar variable

int eliminarVariable(TpoPK idVariable) throws TrException

Elimina una variable existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idVariable	Identificador de la variable a eliminar

Salida

int	Número de filas eliminadas

Obtener variable

```
TrVariable [ ] obtenerVariable( TpoPK idVariable,
                                ClausulaWhere where,
                                ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener una o varias variables, Si se pasa el parámetro *idVariable* se devuelve los datos de esa variable, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idVariable	Identificador de la variable a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrVariable
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrVariable

Salida

TrVariable[ ]	Array de objetos TrVariable

Mantenimiento de perfiles de usuario de otras tareas

Insertar perfil de usuario de otras tareas

```
TpoPK insertarBloquePermitidoPerfil(TrBloquePermitidoPerfil blq,
                                    TpoPK idPerfilUsu,
                                    TpoPK idDefProc,
                                    TpoPK idBloquePer) throws TrException
```

Permite asociar un perfil de usuario a una tarea. Si todo es correcto devuelve los identificadores de la tarea, de la definición del procedimiento y del perfil de usuario, si algo falla lanza una excepción.

Entradas

TrBloquePermitidoPerfil blq	Perfil y tarea a asociar.

Entrada / Salida

TpoPK idDefProc	Identificador de la definición del procedimiento, cero si no se ha insertado
TpoPK idBloquePer	Identificador de la tarea, cero si no se ha insertado
TpoPK idPerfilUsu	Identificador del perfil de usuario, cero si no se ha insertado

Eliminar perfil de usuario de otras tareas

```
int eliminarBloquePermitidoPerfil( TpoPK idPerfilUsu,
                                   TpoPK idDefProc,
                                   TpoPK idBloquePer) throws TrException
```

Elimina la relación entre un perfil de usuario y una tarea en fase existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idPerfilUsu	Identificador del perfil de usuario.
TpoPK idDefProc	Identificador de la definición del procedimiento.
TpoPK idBloquePer	Identificador de la tarea en fase o bloque permitido.

Salida

int	Número de filas eliminadas

Obtener perfil de usuario de otras tareas

```
TrBloquePermitidoPerfil[ ] obtenerBloquePermitidoPerfil(  TpoPK idPerfilUsu,
                                                          TpoPK idDefProc,
                                                          TpoPK idBloquePer,
                                                          ClausulaWhere where,
                                                          ClausulaOrderBy orderBy )  throws TrException
```

Permite obtener uno o varios perfiles asociados a las tareas en fase. Si no se pasan ningún parámetro se devuelven todos los perfiles, si se pasa alguno se filtra por él.

Entradas

TpoPK idPerfilUsu	Identificador del perfil de usuario.
TpoPK idDefProc	Identificador de la definición del procedimiento.
TpoPK idBloquePer	Identificador de la tarea en fase o bloque permitido.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrBloquePermitidoPerfil
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrBloquePermitidoPerfil

Salida

TrBloquePermitidoPerfil[ ]	Array de objetos TrBloquePermitidoPerfil

Mantenimiento de perfiles de usuario de documentos

Insertar perfil de usuario de un documento

```
void insertarDocumentoPermitidoPerfil( TrDocumentoPermitidoPerfil doc,
                                       TpoString permiso,
                                       TpoPK idPerfilUsu,
                                       TpoPK idTipoDoc,
                                       TpoPK idFase,
                                       TpoPK idDefProc) throws TrException
```

Permite asociar un perfil de usuario a un documento. Si todo es correcto devuelve los identificadores de la tarea o tipo de documento, de la definición del procedimiento, del perfil de usuario, de la fase y el permiso, si algo falla lanza una excepción. Los valores para el permiso puede ser “G” para Generar, “I” para Incorporar, “F” para Firmar, “E” para Editar y “T” para todo.

Entradas

TrDocumentoPermitidoPerfil doc	Perfil y documento a asociar.

Entrada / Salida

TpoString permiso	Permiso para el perfil. Sus valores pueden ser: → “**G**” – Generar → “**I**” – Incorporar → “**F**“ – Firmar → “**E**” – Editar → “**T**” – Todo

TpoPK idPerfilUsu	Identificador del perfil de usuario, cero si no se ha insertado
TpoPK idTipoDoc	Identificador de la tarea o tipo de documento, cero si no se ha insertado
TpoPK idFase	Identificador de la fase, cero si no se ha insertado
TpoPK idDefProc	Identificador de la definición del procedimiento, cero si no se ha insertado

Eliminar perfil de usuario de un documento

```
int eliminarDocumentoPermitidoPerfil( String permiso,
                                      TpoPK idPerfilUsu,
                                      TpoPK idTipoDoc,
                                      TpoPK idFase,
                                      TpoPK idDefProc ) throws TrException
```

Elimina la relación entre un perfil de usuario y una tarea en fase existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

String permiso	Identificador del perfil de usuario.
TpoPK idPerfilUsu	Identificador del perfil de usuario.
TpoPK idTipoDoc	Identificador de la tarea o tipo de documento.
TpoPK idFase	Identificador de la fase.
TpoPK idDefProc	Identificador de la definición del procedimiento.

Salida

int	Número de filas eliminadas

Obtener perfil de usuario de otras tareas

```
TrDocumentoPermitidoPerfil[ ] obtenerDocumentoPermitidoPerfil( String permiso
,
                                      TpoPK idPerfilUsu,
                                      TpoPK idTipoDoc,
```

```
                                    TpoPK idFase,
                                    TpoPK idDefProc,
                                    ClausulaWhere where,
                                    ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener uno o varios perfiles asociados a las tareas en fase o documentos. Si no se pasan ningún parámetro se devuelven todos los perfiles, si se pasa alguno se filtra por él.

Entradas

String permiso	Identificador del perfil de usuario.
TpoPK idPerfilUsu	Identificador del perfil de usuario.
TpoPK idTipoDoc	Identificador de la tarea o tipo de documento.
TpoPK idFase	Identificador de la fase.
TpoPK idDefProc	Identificador de la definición del procedimiento.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrDocumentoPermitidoPerfil
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrDocumentoPermitidoPerfil

Salida

TrDocumentoPermitidoPerfil[ ]	Array de objetos TrDocumentoPermitidoPerfil

Mantenimiento de firmas de tipos de documentos

Insertar firma de tipo de documento

```
void insertarFirmaTipoDocumento(TrFirmaTipoDocumento firmaTipoDoc,
                                TpoPK idTipoDoc,
                                TpoPK idFirmanteDef) throws TrException
```

Inserta una nueva firma para un tipo de documento. Si todo es correcto devuelve los identificadores del tipo de documento y del firmante definido, si algo falla lanza una excepción.

Entradas

TrFirmaTipoDocumento firmaTipoDoc	Firma para un tipo de documento a insertar

Entrada / Salida

TpoPK idTipoDoc	Identificador del tipo de documento, cero si no se ha insertado
TpoPK idFirmanteDef	Identificador del firmante definido, cero si no se ha insertado

Modificar firma de tipo de documento

```
int modificarFirmaTipoDocumento( TrFirmaTipoDocumento firmaTipoDoc ) throws TrException
```

Modifica una firma para un tipo de documento existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrFirmaTipoDocumento firmaTipoDoc	Firma para un tipo de documento a modificar

Salida

int	Número de filas modificadas

Eliminar firma de tipo de documento

```
int eliminarFirmaTipoDocumento( TpoPK idTipoDoc,
                                TpoPK idFirmanteDef ) throws TrException
```

Elimina una firma para un tipo de documento existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoDoc	Identificador del tipo de documento
TpoPK idFirmanteDef	Identificador del firmante definido

Salida

int	Número de filas eliminadas

Obtener firma de tipo de documento

TrFirmaTipoDocumento[ ] obtenerFirmaTipoDocumento(TpoPK idTipoDoc, TpoPK idFirmanteDef, ClausulaWhere where, ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias firmas para un tipo de documento. Si se pasa el parámetro *idTipoDoc* se devuelve los datos de las firmas para ese tipo de documento y si se pasa el *idFirmanteDef* se filtra por ese firmante, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idTipoDoc	Identificador del tipo de documento. Si se pasa *null* se obtienen todos.
TpoPK idFirmanteDef	Identificador del firmante definido. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrFirmaTipoDocumento
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrFirmaTipoDocumento

Salida

TrFirmaTipoDocumento[ ]	Array de objetos TrFirmaTipoDocumento

Mantenimiento de parámetros de otras tareas

Insertar parámetro de otras tareas

void insertarParametroBloque(TrParametroBloque paramBloque, TpoPK idParametro, TpoPK idBloque) throws TrException

Inserta un nuevo parámetro para otras tareas. Si todo es correcto devuelve el identificador del parámetro y de la tarea o bloque, si algo falla lanza una excepción.

Entradas

TrParametroBloque paramBloque	Parámetro para otras tareas a insertar

Entrada / Salida

TpoPK idParametro	Identificador del parámetro, cero si no se ha insertado

TpoPK idBloque	Identificador del bloque o tarea, cero si no se ha insertado

Modificar parámetro de otras tareas

```
int modificarParametroBloque( TrParametroBloque paramBloque ) throws TrException
```

Modifica un parámetro para otras tareas existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrParametroBloque paramBloque	Parámetro para otras tareas a modificar

Salida

int	Número de filas modificadas

Eliminar parámetro de otras tareas

```
int eliminarParametroBloque( TpoPK idParametro,
                             TpoPK idBloque) throws TrException
```

Elimina un parámetro para otras tareas existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idParametro	Identificador del parámetro
TpoPK idBloque	Identificador del bloque o tarea

Salida

int	Número de filas eliminadas

Obtener parámetro de otras tareas

```
TrParametroBloque[ ] obtenerParametroBloque( TpoPK idParametro,
                                             TpoPK idBloque,
                                             ClausulaWhere where,
                                             ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener uno o varios parámetros asociados a otras tareas. Si se pasa el parámetro *idParametro* se obtienen los datos de ese parámetro para otras tareas y si se pasa el parámetro *idBloque* se obtienen los parámetros para ese bloque o tarea, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idParametro	Identificador del parámetro
TpoPK idBloque	Identificador del bloque o tarea
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrParametroBloque
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrParametroBloque

Salida

TrParametroBloque[ ]	Array de objetos TrParametroBloque

Mantenimiento de parámetros de variables

Insertar parámetro de variable

```
void insertarParametroVariable(TrParametroVariable paramVar,
                               TpoPK idParametro,
                               TpoPK idVariable) throws TrException
```

Inserta un nuevo parámetro para las variables. Si todo es correcto devuelve el identificador del parámetro y de la variable, si algo falla lanza una excepción.

Entradas

TrParametroVariable paramVar	Parámetro para las variables a insertar

Entrada / Salida

TpoPK idParametro	Identificador del parámetro, cero si no se ha insertado
TpoPK idVariable	Identificador de la variable, cero si no se ha insertado

Modificar parámetro de variable

```
int modificarParametroVariable( TrParametroVariable paramVar ) throws
```

TrException

Modifica un parámetro para una variable existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrParametroVariable paramVar Parámetro para la variable a modificar

Salida

int Número de filas modificadas

Eliminar parámetro de variable

int eliminarParametroVariable(TpoPK idParametro,
TpoPK idVariable) throws TrException

Elimina un parámetro para una variable existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idParametro Identificador del parámetro

TpoPK idVariable Identificador de la variable

Salida

int Número de filas eliminadas

Obtener parámetro de variable

TrParametroVariable[ ] obtenerParametroVariable(TpoPK idParametro,
TpoPK idVariable,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios parámetros asociados a variable. Si se pasa el parámetro *idParametro* se obtienen los datos de ese parámetro para las variables y si se pasa el parámetro *idVariable* se obtienen los parámetros de esa variable, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idParametro	Identificador del parámetro
TpoPK idVariable	Identificador de la variable
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrParametroVariable
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrParametroVariable

Salida

TrParametroVariable[ ]	Array de objetos TrParametroVariable

Mantenimiento de perfiles de usuario de un usuario

Insertar perfil de usuario de un usuario

```
void insertarUsuarioPerfilUsuario ( TrUsuarioPerfilUsuario usuPer,
                                    TpoPK idPerfilUsu,
                                    TpoString usuario) throws TrException
```

Permite asignar un perfil de usuario a un usuario. Si todo es correcto devuelve los identificadores del perfil de usuario y del usuario, si algo falla lanza una excepción.

Entradas

TrUsuarioPerfilUsuario usuPer	Usuario y perfil de usuario a relacionar

Entrada / Salida

TpoPK idPerfilUsu	Identificador del perfil de usuario, cero si no se ha insertado
TpoString usuario	Identificador del usuario, cero si no se ha insertado

Eliminar perfil de usuario de un usuario

```
int eliminarUsuarioPerfilUsuario( TpoPK idPerfilUsu,
                                  String usuario ) throws TrException
```

Elimina una relación existente entre un perfil de usuario y un usuario. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idPerfilUsu	Identificador del perfil de usuario
String usuario	Identificador del usuario

Salida

int	Número de filas eliminadas

Obtener perfil de usuario de un usuario

```
TrUsuarioPerfilUsuario[ ] obtenerUsuarioPerfilUsuario(
                                TpoPK idPerfilUsu,
                                String usuario,
                                ClausulaWhere where,
                                ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener uno o varios perfiles de usuario que tiene asignados un usuario. Si se pasa el parámetro *idPerfilUsu* se devuelve los datos de ese perfil de usuario para todos los usuarios, si se pasa el parámetro *usuario* se devuelven los perfiles de ese usuario y si se pasa *null*, se devuelven todos.

Entradas

TpoPK idPerfilUsu	Identificador del perfil de usuario. Si se pasa *null* se obtienen todos.
String usuario	Identificador del usuario. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrUsuarioPerfilUsuario
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrUsuarioPerfilUsuario

Salida

TrUsuarioPerfilUsuario[ ]	Array de objetos TrUsuarioPerfilUsuario

Mantenimiento de ámbitos de ley

Insertar ámbito de ley

```
TpoPK insertarAmbitoLey(TrAmbitoLey ambitoLey ) throws TrException
```

Inserta un nuevo ámbito de ley. Si todo es correcto devuelve el identificador del nuevo ámbito de ley, si algo falla lanza una excepción.

Entradas

TrAmbitoLey ambitoLey	Ámbito de ley a insertar

Salida

TpoPK	Identificador del nuevo ámbito de ley, cero si no se ha insertado

Modificar ámbito de ley

int modificarAmbitoLey(TrAmbitoLey ambitoLey) throws TrException

Modifica un ámbito de ley existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrAmbitoLey ambitoLey	Ámbito de ley a modificar

Salida

int	Número de filas modificadas

Eliminar ámbito de ley

int eliminarAmbitoLey(TpoPK idAmbitoLey) throws TrException

Elimina un ámbito de ley existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idAmbitoLey	Identificador del ámbito de ley a eliminar

Salida

int	Número de filas eliminadas

Obtener ámbito de ley

TrAmbitoLey [ ] obtenerAmbitoLey(TpoPK idAmbitoLey,

ClausulaWhere where, ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios ámbitos de ley. Si se pasa el parámetro *idAmbitoLey* se devuelve los datos de ese ámbito de ley, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idAmbitoLey	Identificador del ámbito de ley a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrAmbitoLey
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrAmbitoLey

Salida

TrAmbitoLey[ ]	Array de objetos TrAmbitoLey

Mantenimiento de razones de interés

Insertar razón de interés

TpoPK insertarRazonInteres(TrRazonInteres razonInt) throws TrException

Inserta una nueva razón de interés. Si todo es correcto devuelve el identificador de la nueva razón de interés, si algo falla lanza una excepción.

Entradas

TrRazonInteres razonInt	Razón de interés a insertar

Salida

TpoPK	Identificador de la nueva razón de interés, cero si no se ha insertado

Modificar razón de interés

int modificarRazonInteres(TrRazonInteres razonInt) throws TrException

Modifica una razón de interés existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrRazonInteres razonInt	Razón de interés a modificar

Salida

int	Número de filas modificadas

Eliminar razón de interés

int eliminarRazonInteres(TpoPK idRazonInt) throws TrException

Elimina una razón de interés existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idRazonInt	Identificador de la razón de interés a eliminar

Salida

int	Número de filas eliminadas

Obtener razón de interés

TrRazonInteres [ ] obtenerRazonInteres(TpoPK idRazonInt,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias razones de interés. Si se pasa el parámetro *idRazonInt* se devuelve los datos de esa razón de interés, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idRazonInt	Identificador de la razón de interés a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrRazonInteres
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrRazonInteres

Salida

TrRazonInteres[ ]	Array de objetos TrRazonInteres

Mantenimiento de tipos de componentes

Insertar tipo de componente

TpoPK insertarTipoComponente(TrTipoComponente tipoComp) throws TrException

Inserta un nuevo tipo de componente. Si todo es correcto devuelve el identificador del nuevo tipo de componente, si algo falla lanza una excepción.

Entradas

TrTipoComponente tipoComp	Tipo de componente a insertar

Salida

TpoPK	Identificador del nuevo tipo de componente, cero si no se ha insertado

Modificar tipo de componente

int modificarTipoComponente(TrTipoComponente tipoComp) throws TrException

Modifica un tipo de componente existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrTipoComponente tipoComp	Tipo de componente a modificar

Salida

int	Número de filas modificadas

Eliminar tipo de componente

int eliminarTipoComponente(TpoPK idTipoComp) throws TrException

Elimina un tipo de componente existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoComp	Identificador del tipo de componente a eliminar

Salida

int	Número de filas eliminadas

Obtener tipo de componente

TrTipoComponente [ ] obtenerTipoComponente(TpoPK idTipoComp,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios tipos de componentes. Si se pasa el parámetro *idTipoComp* se devuelve los datos de ese tipo de componente, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idTipoComp	Identificador del tipo de componente a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoComponente
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoComponente

Salida

TrTipoComponente[ ]	Array de objetos TrTipoComponente

Mantenimiento de tipos de contacto

Insertar tipo de contacto

TpoPK insertarTipoContacto(TrTipoContacto tipoCont) throws TrException

Inserta un nuevo tipo de contacto. Si todo es correcto devuelve el identificador de la nueva tipo de contacto, si algo falla lanza una excepción.

Entradas

TrTipoContacto tipoCont	Tipo de contacto a insertar

Salida

TpoPK	Identificador del nuevo tipo de contacto, cero si no se ha insertado

Modificar tipo de contacto

int modificarTipoContacto(TrTipoContacto tipoCont) throws TrException

Modifica un tipo de contacto existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrTipoContacto tipoCont	Tipo de contacto a modificar

Salida

int	Número de filas modificadas

Eliminar tipo de contacto

int eliminarTipoContacto(TpoPK idTipoCont) throws TrException

Elimina un tipo de contacto existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoCont	Identificador del tipo de contacto a eliminar

Salida

int	Número de filas eliminadas

Obtener tipo de contacto

TrTipoContacto [ ] obtenerTipoContacto(TpoPK idTipoCont,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios tipos de contacto. Si se pasa el parámetro *idTipoCont* se devuelve los datos de ese tipo de contacto, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idTipoCont	Identificador del tipo de contacto a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoContacto

	ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoContacto

Salida

TrTipoContacto[ ]	Array de objetos TrTipoContacto

Mantenimiento de tipos de indicaciones

Insertar tipo de indicación

TpoPK insertarTipoIndicacion(TrTipoIndicacion tipoInd) throws TrException

Inserta un nuevo tipo de indicación. Si todo es correcto devuelve el identificador del nuevo tipo de indicación, si algo falla lanza una excepción.

Entradas

TrTipoIndicacion tipoInd	Tipo de indicación a insertar

Salida

TpoPK	Identificador del nuevo tipo de indicación, cero si no se ha insertado

Modificar tipo de indicación

int modificarTipoIndicacion(TrTipoIndicacion tipoInd) throws TrException

Modifica un tipo de indicación existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrTipoIndicacion tipoInd	Tipo de indicación a modificar

Salida

int	Número de filas modificadas

Eliminar tipo de indicación

int eliminarTipoIndicacion(TpoPK idTipoInd) throws TrException

Elimina un tipo de indicación existente. Si todo es correcto devuelve el número de filas

eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoInd	Identificador del tipo de indicación a eliminar

Salida

int	Número de filas eliminadas

Obtener tipo de indicación

```
TrTipoIndicacion [ ] obtenerTipoIndicacion( TpoPK idTipoInd,
                                            ClausulaWhere where,
                                            ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener uno o varios tipos de indicación. Si se pasa el parámetro *idTipoInd* se devuelve los datos de ese tipo de indicación, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idTipoInd	Identificador del tipo de indicación a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoIndicacion
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoIndicacion

Salida

TrTipoIndicacion[ ]	Array de objetos TrTipoIndicacion

Mantenimiento de tipos de normativas

Insertar tipo de normativa

```
TpoPK insertarTipoNormativa(TrTipoNormativa tipoNorma ) throws TrException
```

Inserta un nuevo tipo de normativa. Si todo es correcto devuelve el identificador del nuevo tipo de normativa, si algo falla lanza una excepción.

Entradas

TrTipoNormativa tipoNorma	Tipo de normativa a insertar

Salida

TpoPK	Identificador del nuevo tipo de normativa, cero si no se ha insertado

Modificar tipo de normativa

int modificarTipoNormativa(TrTipoNormativa tipoNorma) throws TrException

Modifica un tipo de normativa existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrTipoNormativa tipoNorma	Tipo de normativa a modificar

Salida

int	Número de filas modificadas

Eliminar tipo de normativa

int eliminarTipoNormativa(TpoPK idTipoNorma) throws TrException

Elimina un tipo de normativa existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoNorma	Identificador del tipo de normativa a eliminar

Salida

int	Número de filas eliminadas

Obtener tipo de normativa

TrTipoNormativa [ ] obtenerTipoNormativa(TpoPK idTipoNorma,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios tipos de normativas. Si se pasa el parámetro *idTipoNorma* se devuelve los datos de ese tipo de normativa, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idTipoNorma	Identificador del tipo de normativa a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoNormativa
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoNormativa

Salida

TrTipoNormativa[ ]	Array de objetos TrTipoNormativa

Mantenimiento de tipos de publicación

Insertar tipo de publicación

TpoPK insertarTipoPublicacion(TrTipoPublicacion tipoPubli) throws TrException

Inserta un nuevo tipo de publicación. Si todo es correcto devuelve el identificador del nuevo tipo de publicación, si algo falla lanza una excepción.

Entradas

TrTipoPublicacion tipoPubli	Tipo de publicación a insertar

Salida

TpoPK	Identificador del nuevo tipo de publicación, cero si no se ha insertado

Modificar tipo de publicación

int modificarTipoPublicacion(TrTipoPublicacion tipoPubli) throws TrException

Modifica un tipo de publicación existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrTipoPublicacion tipoPubli	Tipo de publicación a modificar

Salida

int	Número de filas modificadas

Eliminar tipo de publicación

int eliminarTipoPublicacion(TpoPK idTipoPubli) throws TrException

Elimina un tipo de publicación existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoPubli	Identificador del tipo de publicación a eliminar

Salida

int	Número de filas eliminadas

Obtener tipo de publicación

TrTipoPublicacion [ ] obtenerTipoPublicacion(TpoPK idTipoPubli,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios tipos de publicación. Si se pasa el parámetro *idTipoPubli* se devuelve los datos de ese tipo de publicación, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idTipoPubli	Identificador del tipo de publicación a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoPublicacion
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoPublicacion

Salida

TrTipoPublicacion[ ]	Array de objetos TrTipoPublicacion

Mantenimiento de variables para un tipo de documento

Insertar variable para un tipo de documento

void insertarVariableTipoDocumento(TrVariableTipoDocumento variable,

TpoPK idTipoDoc, TpoPK idVariable) throws TrException

Permite relacionar una variable con un tipo de documento. Si todo es correcto devuelve los identificadores del tipo de documento y de la variable, si algo falla lanza una excepción.

Entradas

TrVariableTipoDocumento variable	Variable y tipo de documento a relacionar.

Entrada / Salida

TpoPK idTipoDoc	Identificador del tipo de documento, cero si no se ha insertado
TpoPK idVariable	Identificador de la variable, cero si no se ha insertado

Eliminar variable para un tipo de documento

int eliminarVariableTipoDocumento(TpoPK idTipoDoc, TpoPK idVariable) throws TrException

Elimina una relación existente entre una variable y un tipo de documento. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoDoc	Identificador del tipo de documento
TpoPK idVariable	Identificador de la variable

Salida

int	Número de filas eliminadas

Eliminar todas las variables para un tipo de documento

int eliminarVariableTipoDocumento(TpoPK idTipoDoc) throws TrException

Permite eliminar todas las variables asociadas a tipo de documento que se indique en el parámetro *idTipoDoc*. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoDoc	Identificador del tipo de documento

Salida

int	Número de filas eliminadas

Obtener variable para un tipo de documento

```
TrVariableTipoDocumento[ ] obtenerVariableTipoDocumento(
                                    TpoPK idTipoDoc,
                                    TpoPK idVariable,
                                    ClausulaWhere where,
                                    ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener una o varias variables de uno o varios tipos de documentos. Si se pasa el parámetro *idTipoDoc* se obtienen todas las variables para ese tipo de documento, si se para el parámetro *idVariable* se obtiene esa variable para todos los tipos de documentos y si se pasan ambos a *null* se obtienen todas las variables para todos los tipos de documentos.

Entradas

TpoPK idTipoDoc	Identificador del tipo de documento, si se pasa *null* se obtienen todos.
TpoPK idVariable	Identificador de la variable, si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrVariableTipoDocumento
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrVariableTipoDocumento

Salida

TrVariableTipoDocumento[ ]	Array de objetos TrVariableTipoDocumento

Mantenimiento de constantes generales

Insertar constante general

```
String insertarConstanteGeneral(TrConstanteGeneral constanteGn) throws TrException
```

Inserta una nueva constante general. Si todo es correcto devuelve el identificador de la nueva constante general, si algo falla lanza una excepción.

Entradas

TrConstanteGeneral constanteGn	Constante general a insertar

Salida

String	Identificador de la nueva constante general

Modificar constante general

int modificarConstanteGeneral(TrConstanteGeneral constanteGn) throws TrException

Modifica una constante general existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrConstanteGeneral constanteGn	Constante general a modificar

Salida

int	Número de filas modificadas

Eliminar constante general

int eliminarConstanteGeneral(TpoPK idConstanteGn) throws TrException

Elimina una constante general existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idConstanteGn	Identificador de la constante general a eliminar

Salida

int	Número de filas eliminadas

Obtener constante general

```
TrConstanteGeneral [ ] obtenerConstanteGeneral( TpoPK idConstanteGn,
                                                ClausulaWhere where,
                                                ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener una o varias constantes generales. Si se pasa el parámetro *idConstanteGn* se devuelve los datos de esa constante general, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idConstanteGn	Identificador de la constante general a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrConstanteGeneral
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrConstanteGeneral

Salida

TrConstanteGeneral[ ]	Array de objetos TrConstanteGeneral

Mantenimiento de empleados

Insertar empleado

```
void insertarEmpleado( TrEmpleado empleado,
                       TpoString usuario,
                       TpoPK idOrganismo,
                       TpoString idPuestoTrab) throws TrException
```

Inserta un nuevo empleado. Si todo es correcto devuelve el identificador del nuevo empleado, si algo falla lanza una excepción.

Entradas

TrEmpleado empleado	Empleado a insertar

Entrada / Salida

TpoString usuario	Identificador del usuario
TpoPK idOrganismo	Identificador del organismo
TpoString idPuestoTrab	Identificador del puesto de trabajo

Modificar empleado

int modificarEmpleado(TrEmpleado empleado) throws TrException

Modifica un empleado existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrEmpleado empleado	Empleado a modificar

Salida

int	Número de filas modificadas

Eliminar empleado

int eliminarEmpleado(String usuario,
TpoPK idOrganismo,
String idPuestoTrab) throws TrException

Elimina un empleado existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

String usuario	Identificador del usuario
TpoPK idOrganismo	Identificador del organismo
String idPuestoTrab	Identificador del puesto de trabajo

Salida

int	Número de filas eliminadas

Obtener empleado

TrEmpleado[ ] obtenerEmpleado(String usuario,
TpoPK idOrganismo,
String idPuestoTrab,
ClausulaWhere where,

ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios empleados. Si se pasa alguno de los parámetros se filtra el resultado, si se pasan todos a *null*, se devuelven todos.

Entradas

String usuario	Identificador del usuario, si se pasa *null* se obtienen todos.
TpoPK idOrganismo	Identificador del organismo, si se pasa *null* se obtienen todos.
String idPuestoTrab	Identificador del puesto de trabajo, si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrEmpleado
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrEmpleado

Salida

TrEmpleado[ ]	Array de objetos TrEmpleado

Mantenimiento de provincias

Insertar provincia

String insertarProvincia(TrProvincia provincia) throws TrException

Inserta una nueva provincia. Si todo es correcto devuelve el identificador de la nueva provincia, si algo falla lanza una excepción.

Entradas

TrProvincia provincia	Provincia a insertar

Salida

String	Identificador de la nueva provincia

Modificar provincia

int modificarProvincia(TrProvincia provincia) throws TrException

Modifica una provincia existente. Si todo es correcto devuelve el número de filas

modificadas, si algo falla lanza una excepción.

Entradas

TrProvincia provincia	Provincia a modificar

Salida

int	Número de filas modificadas

Eliminar provincia

int eliminarProvincia(String idProvincia) throws TrException

Elimina una provincia existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

String idProvincia	Identificador de la provincia a eliminar

Salida

int	Número de filas eliminadas

Obtener provincia

TrProvincia [ ] obtenerProvincia(String idProvincia,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener una o varias provincias. Si se pasa el parámetro *idProvincia* se devuelve los datos de esa provincia, si se pasa *null*, se devuelven todas.

Entradas

String idProvincia	Identificador de la provincia a obtener. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrProvincia
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrProvincia

Salida

TrProvincia[ ] Array de objetos TrProvincia

Mantenimiento de municipios

Insertar municipio

```
void insertarMunicipio( TrMunicipio municipio,
                        TpoString idMunicipio,
                        TpoString idProvincia ) throws TrException
```

Inserta un nuevo municipio. Si todo es correcto devuelve los identificadores del municipio y provincia, si algo falla lanza una excepción.

Entradas

TrMunicipio municipio — Municipio a insertar

Salida

TpoString idMunicipio — Identificador del municipio

TpoString idProvincia — Identificador de la provincia

Modificar municipio

```
int modificarMunicipio( TrMunicipio municipio ) throws TrException
```

Modifica un municipio existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrMunicipio municipio — Municipio a modificar

Salida

int — Número de filas modificadas

Eliminar municipio

```
int eliminarMunicipio( String idMunicipio,
```

String idProvincia) throws TrException

Elimina un municipio existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

String idMunicipio	Identificador del municipio
String idProvincia	Identificador de la provincia

Salida

int	Número de filas eliminadas

Obtener municipio

TrMunicipio[ ] obtenerMunicipio(String idMunicipio,
String idProvincia,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios municipios. Si se pasa algunos de los parámetros *idMunicipio* o *idProvincia* el resultado se filtra, si se pasan ambos *null*, se devuelven todos.

Entradas

String idMunicipio	Identificador del municipio. Si se pasa *null* se obtienen todas.
String idProvincia	Identificador de la provincia. Si se pasa *null* se obtienen todas.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrMunicipio
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrMunicipio

Salida

TrMunicipio[ ]	Array de objetos TrMunicipio

Mantenimiento de países

Insertar país

String insertarPais(TrPais pais) throws TrException

Inserta un nuevo país. Si todo es correcto devuelve el identificador del nuevo pais, si algo falla lanza una excepción.

Entradas

TrPais pais	País a insertar

Salida

String	Identificador del nuevo país

Modificar país

int modificarPais(TrPais pais) throws TrException

Modifica un país existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrPais pais	País a modificar

Salida

int	Número de filas modificadas

Eliminar país

int eliminarPais(String idPais) throws TrException

Elimina un país existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

String idPais	Identificador del país a eliminar

Salida

int	Número de filas eliminadas

Obtener país

TrPais [ ] obtenerPais(String idPais,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios países. Si se pasa el parámetro *idPais* se devuelve los datos de ese país, si se pasa *null*, se devuelven todas.

Entradas

String idPais	Identificador del país a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrPais
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrPais

Salida

TrPais[ ]	Array de objetos TrPais

Mantenimiento de puestos de trabajo

Insertar puesto de trabajo

String insertarPuestoTrabajo(TrPuestoTrabajo puestoTrabajo) throws TrException

Inserta un nuevo puesto de trabajo. Si todo es correcto devuelve el identificador del nuevo puesto de trabajo, si algo falla lanza una excepción.

Entradas

TrPuestoTrabajo puestoTrabajo	Puesto de trabajo a insertar

Salida

String	Identificador del nuevo puesto de trabajo.

Modificar puesto de trabajo

int modificarPuestoTrabajo(TrPuestoTrabajo puestoTrabajo) throws TrException

Modifica un puesto de trabajo existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrPuestoTrabajo puestoTrabajo	Puesto de trabajo a modificar

Salida

int	Número de filas modificadas

Eliminar puesto de trabajo

int eliminarPuestoTrabajo(String idPuestoTrab) throws TrException

Elimina un puesto de trabajo existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idPuestoTrab	Identificador del puesto de trabajo a eliminar

Salida

int	Número de filas eliminadas

Obtener puesto de trabajo

TrPuestoTrabajo [ ] obtenerPuestoTrabajo(String idPuestoTrab,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios puestos de trabajo. Si se pasa el parámetro *idPuestoTrab* se devuelven los datos de ese puesto de trabajo, si se pasa *null*, se devuelven todos.

Entradas

String idPuestoTrab	Identificador del puesto de trabajo a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrPuestoTrabajo
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrPuestoTrabajo

Salida

TrPuestoTrabajo[ ]	Array de objetos TrPuestoTrabajo

Mantenimiento de puestos de trabajo de un organismo

Insertar puesto de trabajo de un organismo

```
void insertarPtoTrabOrganismo (TrPtoTrabOrganismo ptoTrabOrg,
                               TpoString idPuestoTrab,
                               TpoPK idOrganismo) throws TrException
```

Relaciona un puesto de trabajo con un organismo Si todo es correcto devuelve los identificadores del puesto de trabajo y del organismo, si algo falla lanza una excepción.

Entradas

TrPtoTrabOrganismo ptoTrabOrg	Puesto de trabajo y organismo a relacionar.

Entrada / Salida

TpoString idPuestoTrab	Identificador del puesto de trabajo
TpoPK idOrganismo	Identificador del organismo

Eliminar puesto de trabajo de un organismo

```
int eliminarPtoTrabOrganismo( String idPuestoTrab,
                              TpoPK idOrganismo ) throws TrException
```

Elimina la relación existente entre un puesto de trabajo y un organismo. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

String idPuestoTrab	Identificador del puesto de trabajo
TpoPK idOrganismo	Identificador del organismo

Salida

int	Número de filas eliminadas

Obtener puesto de trabajo de un organismo

TrPtoTrabOrganismo[ ] obtenerPtoTrabOrganismo(String idPuestoTrab,
TpoPK idOrganismo,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios puestos de trabajos relacionados a un organismo. Si se pasa sólo el parámetro *idPuestoTrab* se obtiene ese puesto de trabajo de todos los organismos, si se pasa sólo el parámetro *idOrganismo* se obtienen todos los puestos de trabajo de ese organismo y si se pasan ambos *null*, se devuelven todos.

Entradas

String idPuestoTrab	Identificador del puesto de trabajo
TpoPK idOrganismo	Identificador del organismo
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrPtoTrabOrganismo
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrPtoTrabOrganismo

Salida

TrPtoTrabOrganismo[ ]	Array de objetos TrPtoTrabOrganismo

Mantenimiento de sistemas

Insertar sistema

TpoPK insertarSistema(TrSistema sistema) throws TrException

Inserta un nuevo sistema. Si todo es correcto devuelve el identificador del nuevo sistema, si algo falla lanza una excepción.

Entradas

TrSistema sistema	Sistema a insertar

Salida

TpoPK	Identificador del nuevo sistema, cero si no se ha insertado

Modificar sistema

```
int modificarSistema(  TrSistema  sistema ) throws TrException
```

Modifica un sistema existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrSistema sistema	Sistema a modificar

Salida

int	Número de filas modificadas

Eliminar sistema

```
int eliminarSistema( TpoPK idSistema) throws TrException
```

Elimina un sistema existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idSistema	Identificador del sistema a eliminar

Salida

int	Número de filas eliminadas

Obtener sistema

```
TrSistema [ ] obtenerSistema( TpoPK idSistema,
                              ClausulaWhere where,
                              ClausulaOrderBy orderBy ) throws TrException
```

Permite obtener uno o varios sistemas. Si se pasa el parámetro *idSistema* se devuelve los datos de ese sistema, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idSistema	Identificador del sistema a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrSistema

	ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrSistema

Salida

TrSistema[ ]	Array de objetos TrSistema

Mantenimiento de tipos de identificadores

Insertar tipo de identficador

String insertarTipoIdentificador(TrTipoIdentificador tipoIdent) throws TrException

Inserta un nuevo tipo de identificador. Si todo es correcto devuelve el identificador del nuevo tipo de identificador, si algo falla lanza una excepción.

Entradas

TrTipoIdentificador tipoIdent	Tipo de identificador a insertar.

Salida

String	Identificador del nueva tipo de identificador.

Modificar tipo de identficador

int modificarTipoIdentificador(TrTipoIdentificador tipoIdent) throws TrException

Modifica un tipo de identificador existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrTipoIdentificador tipoIdent	Tipo de identificador a modificar

Salida

int	Número de filas modificadas

Eliminar tipo de identficador

int eliminarTipoIdentificador(String idTipoIdent) throws TrException

Elimina un tipo de identificador existente. Si todo es correcto devuelve el número de

filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoIdent	Identificador del tipo de identificador a eliminar

Salida

int	Número de filas eliminadas

Obtener tipo de identficador

TrTipoIdentificador [ ] obtenerTipoIdentificador(String idTipoIdent,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios tipos de identificadores. Si se pasa el parámetro *idTipoIdent* se devuelve los datos de ese tipo de identificador, si se pasa *null*, se devuelven todos.

Entradas

String idTipoIdent	Identificador del tipo de identificador a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoIdentificador
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoIdentificador

Salida

TrTipoIdentificador[ ]	Array de objetos TrTipoIdentificador

Mantenimiento de tipos de organismos

Insertar tipo de organismo

TpoPK insertarTipoOrganismo(TrTipoOrganismo tipoOrg) throws TrException

Inserta un nuevo tipo de organismo. Si todo es correcto devuelve el identificador del nuevo tipo de organismo, si algo falla lanza una excepción.

Entradas

TrTipoOrganismo tipoOrg	Tipo de organismo a insertar

Salida

TpoPK	Identificador del nuevo tipo de organismo, cero si no se ha insertado

Modificar tipo de organismo

int modificarTipoOrganismo(TrTipoOrganismo tipoOrg) throws TrException

Modifica un tipo de organismo existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrTipoOrganismo tipoOrg	Tipo de organismo a modificar

Salida

int	Número de filas modificadas

Eliminar tipo de organismo

int eliminarTipoOrganismo(TpoPK idTipoOrg) throws TrException

Elimina un tipo de organismo existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoOrg	Identificador del tipo de organismo a eliminar

Salida

int	Número de filas eliminadas

Obtener tipo de organismo

TrTipoOrganismo [ ] obtenerTipoOrganismo(TpoPK idTipoOrg,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios tipos de organismos. Si se pasa el parámetro *idTipoOrg*

se devuelve los datos de ese tipo de organismo, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idTipoOrg	Identificador del tipo de organismo a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoOrganismo
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoOrganismo

Salida

TrTipoOrganismo[ ]	Array de objetos TrTipoOrganismo

Mantenimiento de tipos de organizaciones

Insertar tipo de organización

String insertarTipoOrganizacion(TrTipoOrganizacion tipoOrgz) throws TrException

Inserta un nuevo tipo de organización. Si todo es correcto devuelve el identificador del nuevo tipo de organización, si algo falla lanza una excepción.

Entradas

TrTipoOrganizacion tipoOrgz	Tipo de organización a insertar

Salida

String	Identificador del nuevo tipo de organización.

Modificar tipo de organización

int modificarTipoOrganizacion(TrTipoOrganizacion tipoOrgz) throws TrException

Modifica un tipo de organización existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrTipoOrganizacion tipoOrgz	Tipo de organización a modificar

Salida

int	Número de filas modificadas

Eliminar tipo de organización

int eliminarTipoOrganizacion(String idTipoOrgz) throws TrException

Elimina un tipo de organización existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoOrgz	Identificador del tipo de organización a eliminar

Salida

int	Número de filas eliminadas

Obtener tipo de organización

TrTipoOrganizacion [ ] obtenerTipoOrganizacion(String idTipoOrgz,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios tipos de organizaciones. Si se pasa el parámetro *idTipoOrgz* se devuelve los datos de ese tipo de organización, si se pasa *null*, se devuelven todos.

Entradas

TpoPK idTipoOrgz	Identificador del tipo de organización a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoOrganizacion
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoOrganizacion

Salida

TrTipoOrganizacion[ ]	Array de objetos TrTipoOrganizacion

Mantenimiento de tipos de vías

Insertar tipo de vía

String insertarTipoVia(TrTipoVia tipoVia) throws TrException

Inserta un nuevo tipo de vía. Si todo es correcto devuelve el identificador del nuevo tipo de vía, si algo falla lanza una excepción.

Entradas

TrTipoVia tipoVia	Tipo de vía a insertar

Salida

String	Identificador del nuevo tipo de vía.

Modificar tipo de vía

int modificarTipoVia(TrTipoVia tipoVia) throws TrException

Modifica un tipo de vía existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrTipoVia tipoVia	Tipo de vía a modificar

Salida

int	Número de filas modificadas

Eliminar tipo de vía

int eliminarTipoVia(String idTipoVia) throws TrException

Elimina un tipo de vía existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

TpoPK idTipoVia	Identificador del tipo de vía a eliminar

Salida

int	Número de filas eliminadas

Obtener tipo de vía

TrTipoVia [ ] obtenerTipoVia(String idTipoVia, ClausulaWhere where, ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios tipos de vías. Si se pasa el parámetro *idTipoVia* se devuelve los datos de ese tipo de vía, si se pasa *null*, se devuelven todas.

Entradas

TpoPK idTipoVia	Identificador del tipo de vía a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrTipoVia
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrTipoVia

Salida

TrTipoVia[ ]	Array de objetos TrTipoVia

Mantenimiento de usuarios

Insertar usuario

String insertarUsuario(TrUsuario usuario) throws TrException

Inserta un nuevo usuario. Si todo es correcto devuelve el identificador del nuevo usuario, si algo falla lanza una excepción.

Entradas

TrUsuario usuario	Usuario a insertar

Salida

String	Identificador del nuevo usuario.

Modificar usuario

int modificarUsuario(TrUsuario usuario) throws TrException

Modifica un usuario existente. Si todo es correcto devuelve el número de filas modificadas, si algo falla lanza una excepción.

Entradas

TrUsuario usuario	Usuario a modificar

Salida

int	Número de filas modificadas

Eliminar usuario

int eliminarUsuario(String idUsuario) throws TrException

Elimina un usuario existente. Si todo es correcto devuelve el número de filas eliminadas, si algo falla lanza una excepción.

Entradas

String idUsuario	Identificador del usuario a eliminar

Salida

int	Número de filas eliminadas

Obtener usuario

TrUsuario [ ] obtenerUsuario(String idUsuario,
ClausulaWhere where,
ClausulaOrderBy orderBy) throws TrException

Permite obtener uno o varios usuarios. Si se pasa el parámetro *idUsuario* se devuelve los datos de ese usuario, si se pasa *null*, se devuelven todos.

Entradas

String idUsuario	Identificador del usuario a obtener. Si se pasa *null* se obtienen todos.
ClausulaWhere where	Filtro a aplicar. Consultar campos posibles para TrUsuario
ClausulaOrderBy orderBy	Ordenación a aplicar. Consultar campos posibles para TrUsuario

Salida

TrUsuario[ ]	Array de objetos TrUsuario

Otras APIs

Establecer el usuario

void establecerUsuarioSistema(String usuario) throws TrException

Permite establecer el usuario del TrAPIADM.

Entradas

String usuario	*Usuario*

Los usuarios deben tener el rol o perfil de usuario TR_R_USUARIO o el TR_R_ADMINISTRADOR.

Obtener usuario establecido

String obtenerUsuarioSistema(TpoString rol)

Permite obtener el usuario establecido en el TrAPIADM y su rol o perfil.

Entrada/Salida

TpoString rol	*Rol del usuario establecido en el api* *Parámetro de entrada/salida*

Salida

String usuario	*Usuario establecido en el api*

Cierre de la sesión

void cerrarSesion( )

Cierra la sesión actual de trabajo, haciendo por defecto un commit si el autoCommit está a true, si no se realiza un rollback.

void cerrarSesion(boolean commit)

Cierra la sesión actual de trabajo. Realiza un commit o un rollback según se indique en el parámetro *commit*.

Entrada

boolean commit	→ "**true**" – Se realiza un commit para guardar los cambios. → "**false**" – Se realiza un rollback para no guardar los cambios, desde el último commit.

AutoCommit

void setAutoCommit(boolean value)

Establece el valor del autoCommit.

Entrada

boolean value	→ "**true**" – Se realiza un commit cada vez que se ejecute un api. → "**false**" – No se realiza commit cuando se ejecute un api, el usuario debe hacer uso del método commit() para guardar los cambios.

boolean getAutoCommit()

Devuelve el estado actual del autoCommit.

Salida

boolean	→ "**true**" – Se realiza un commit cada vez que se ejecute un api. → "**false**" – No se realiza commit cuando se ejecute un api, el usuario debe hacer uso del método commit() para guardar los cambios.

Commit y Rollback

boolean commit()

Realiza un commit sobre la base de datos guardando los últimos cambios.

Salida

boolean	→ "**true**" – El commit se ha realizado correctamente → "**false**" – El commit no se ha realizado correctamente

boolean rollback()

Realiza un rollback sobre la base de datos deshaciendo los últimos cambios.

Salida

boolean	→ "**true**" – El rollback se ha realizado correctamente → "**false**" – El rollback no se ha realizado correctamente

Comprobación de conexión

boolean hayConexion( )

Indica si existe o no una conexión válida a la base de datos.

Salida

boolean	→ "**true**" – Existe una conexión válida a la base de datos. → "**false**" – No existe conexión válida a la base de datos.

Establecer conexión fija

synchronized void establecerConexionFija(boolean fija)

Permite establecer la conexión como fija, es decir una vez que se crea no se cierra hasta que no se ejecute el método cerrarSesion. Este método sólo tiene efecto cuando la TrAPIADM se ha creado con un perfil de conexión que sea un datasource.

Si se está usando un perfil de conexión que es un datasource y el conexionFija es false se devuelve la conexión al finalizar la ejecución de un método, en caso contrario no se devuelve.

Entrada

boolean fija	Indica si la conexión es fija para la TrAPIADM o se devuelve al pool de conexiones al finalizar la ejecución de un método. → "**true**" – La conexión es fija y no se devuelve al pool. Por defecto toma este valor. → "**false**" – Se devuelve la conexión al pool al terminar la ejecución de cada método, haciendose commit al final de cada método.

Obtener estado conexión fija

synchronized boolean obtenerEstadoConexionFija( )

Devuelve el valor actual del atributo conexionFija. El estado de la conexión fija sólo tiene efecto cuando la TrAPIADM se ha creado usando un perfil de conexión que es un datasource. Si el valor devuelto es true indica que la conexión se fija y no se devuelve al pool hasta cerrar la sesión, en caso contrario si el valor es false indica que no es fija y que se devuelve la conexión al pool al finalizar la ejecución de cada método.

Salida

boolean	Valor del atributo conexionFija → "**true**" – La conexión es fija y no se devuelve al pool. Por defecto toma este valor. → "**false**" – Se devuelve la conexión al pool al terminar la ejecución de cada método, haciendose commit al final de cada método.

Descripción del TrAPIUTL

Descripción de clases involucradas

Interfaz J-TrAPIUTL

La interfaz J-TrAPIUTL define todos los métodos para facilitar el intercambio de definiciones de procedimiento entre diferentes sistemas con TREW@. En la librería este acceso se realiza a través de la implementación de la interfaz **TrAPIUTL** que se describe en este apartado.

TrAPIUTL

trewa.bd.trapi.trapiutl.TrAPIUTL

Interfaz de acceso al motor de tramitación TREW@.

Para poder trabajar con la TrAPIUTL es preciso crear una instancia de la misma a través de la clase Factoría: *TrAPIUTLFactory*, descrita en el apartado siguiente.

Métodos para la creación de instancias del TrAPIUI

Descarga de procedimientos

→ `InputStream descargaDDP(TpoPK idDDP, TpoString error) )`
deprecated

Descarga el XML asociado a una definición de procedimientos

Entradas	`TpoPK idDDP`	*Identificador de la definición de procedimiento.*
	`TpoString error`	*Mensaje de error*
		Entrada/Salida
Salidas	*InputStream*	*Fichero XML con la información referente a la definición de procedimiento.*

→ `void descargaDDP(TpoPK idDDP, OutputStream out, TrAPIUTLIO archivos) throws TrException`

Descarga el XML asociado a una definición de procedimiento y la referencia a los archivos asociados

Entradas	`TpoPK idDDP`	*Identificador del expediente.*
	`OutputStream out`	*Flujo de salida del fichero XML del DDP*
	`TrAPIUTLIO archivos`	*Lista con los nombres de los archivos adjuntos que se tendrán que descargar.* *Ver API de descarga de archivos.*

→ `void descargaDDP(TpoPK idDDP, OutputStream out, TrAPIUTLIO archivos, boolean reutilizables) throws TrException`

Descarga el XML asociado a una definición de procedimiento y la referencia a los archivos asociados. Permite indicar si se obtienen los procedimientos reutilizables dentro del principal.

Entradas	`TpoPK idDDP`	*Identificador del expediente.*
	`OutputStream out`	*Flujo de salida del fichero XML del DDP*
	`TrAPIUTLIO archivos`	*Lista con los nombres de los archivos adjuntos que se tendrán que descargar.* *Ver API de descarga de archivos.*
	`boolean reutilizables`	*Indica si se obtienen los procedimientos reutilizables.*

Descarga de expedientes

→ `InputStream descargaExpediente(TpoPK idExpediente, TpoString error)` *`deprecated`*

Descarga el XML asociado a la evolución de un expediente

Entradas	`TpoPK idExpediente`	*Identificador del expediente.*
	`TpoString error`	*Mensaje de error* *Entrada/Salida*
Salidas	*InputStream*	*Fichero XML con la información referente a la evolución de un expediente.*

→ `void descargaExpediente(TpoPK idExpediente, OutputStream out, TrAPIUTLIO archivos) throws TrException`

Descarga el XML asociado a la evolución de un expediente

Entradas	`TpoPK idExpediente`	*Identificador del expediente.*
	`OutputStream out`	*Flujo de salida del fichero XML del Expediente*

`TrAPIUTLIO archivos`	*Lista con los nombres de los archivos adjuntos que se tendrán que descargar.* *Ver API de descarga de archivos.*

Subida de definición de procedimientos

→ `String subidaDDP(boolean actualizar, InputStream ficheroDDP, int size)` *`deprecated`*

Sube un XML que representa una definición de procedimiento.

Entradas	`Boolean`	*Indica si se actualiza o no la información existente en Trew@ con la que viene en el XML.* *→ "**S**" – Se actualiza la información que exista con la que viene en el XML.* *→ "**N**" – No se actualiza la información que exista con la que viene en el XML.*
	`InputStream ficheroDDP`	*Archivo XML con la definición de procedimiento.*
	Int size	*Tamaño del archivo XML.*
Salidas	*String*	*Mensaje de error.*

→ `TpoPK subidaDDP(boolean actualizar,InputStream ficheroDDP,TrAPIUTLIO flujos) throws TrException`

Sube un XML que representa una definición de procedimiento.

Entradas	`boolean`	*Indica si se actualiza o no la información existente en Trew@ con la que viene en el XML.* *→ "**S**" – Se actualiza la información que exista con la que viene en el XML.* *→ "**N**" – No se actualiza la información que exista con la que viene en el XML en este caso el procedimiento intentará ser borrado previo a su inserción.*
	`InputStream ficheroDDP`	*Archivo XML con la definición de procedimiento.*
	`TrAPIUTLIO flujos`	*Flujos flujos de entrada correspondientes a los archivos.* *Ver API de subida de archivos.*
Salidas	*TpoPK*	*Identificador del nuevo procedimiento o del procedimiento actualizado en la base de datos.*

Descarga de entidades del sistema

→ `void descargaDES(TpoPK idSistema, OutputStream out, TrAPIUTLIO archivos) throws TrException`

Descarga el XML asociado a la evolución de un expediente

Entradas	`TpoPK idSistema`	*Identificador del sistema*
	`OutputStream out`	*Flujo de salida del fichero XML del DES*
	`TrAPIUTLIO archivos`	*Lista con los nombres de los archivos adjuntos que se tendrán que descargar.* *Ver API de descarga de archivos.*

Obtención de archivos

→ `void obtenerArchivo(TrAPIUTLArchivo archivo,OutputStream oArchivo) throws TrException;`

Se obtiene el archivo de BBDD

Entradas	`TrAPIUTLArchivo archivo`	*Información del archivo*
	`OutputStream oArchivo`	*Flujo de salida hacia el archivo*

Anexar archivos

→ `void anexarArchivo(TrAPIUTLArchivo archivo,InputStream iArchivo) throws TrException`

Inserta una archivo en la base de datos

Entradas	`TrAPIUTLArchivo archivo`	*Identificador del sistema*
	`InputStream iArchivo`	*Flujo de entrada desde el que se lee el archivo*

Bloqueo y desbloqueo de procedimientos

→ `void bloquearDefProcedimiento( TpoPK idDefProc ) throws TrException`

Bloquea el procedimiento indicado siempre que no esté bloqueado por otro usario

Entradas	`TpoPK idDefProc`	*Identificador de la definición de procedimiento*

→ `void desbloquearDefProcedimiento( TpoPK idDefProc ) throws TrException`

Desbloquea el procedimiento indicado si está bloqueado por el mismo usuario

Entradas	`TpoPK idDefProc`	*Identificador de la definición de procedimiento*

→ `String procedimientoBloqueadoPor (TpoPK idDefProc, TpoString nombre) throws TrException`

Indica qué usuario tiene bloqueado un procedimiento

Entradas	`TpoPK idDefProc`	*Identificador de la definición de procedimiento*
	`TpoString nombre`	*Nombre y apellidos del usuario que tiene bloqueado el procedimiento (Salida)*
Salida	`String`	*Código de usuario que tiene bloqueado el procedimiento*

Borrar procedimiento

→ `void borrarProcedimiento(TpoPK idDefProc, String tipoBorrado) throws TrException`

Borra el procedimiento indicado. Mediante el parámetro tipoBorrado indicamos hasta qué nivel queremos borrar, invalidando las entidades que no sea posible borrar.

Entradas

`TpoPK idDefProc`	*Identificador del procedimiento*
`String tipoBorrado`	→ *"**B**" – Básico. Se realiza un borrado (o invalidación) básico del procedimiento, es decir sin borrar fases, tipos de documentos, tareas, etc.*
	→ *"**A**" – Avanzado. Se realiza un borrado (o invalidación) avanzado, es decir borra fases, metafases, transiciones, tipos documentos, tareas, etc.*
	→ *"**I**" – Intermedio. Se realiza un borrado (o invalidación) intermedio, similar al avanzado, sin eliminar versiones del procedimiento ni relaciones.*

Otros métodos

→ `boolean hayConexion( )`

Indica si existe o no una conexión válida a la base de datos

Salidas *boolean*

→ "**true**" – Existe una conexión válida a la base de datos.
→ "**false**" – No existe conexión válida a la base de datos.

→ `void cerrarSesion( )`

Cierra la sesión actual de trabajo haciendo un commit por defecto si está activado el autoCommit

→ `void cerrarSesion(boolean commit )`

Cierra la sesión actual de trabajo indicándose si se realiza commit o no

Entradas

`boolean`	*Indica si se hace o no el autoCommit*
	→ *"**S**" – Hacer commit*
	→ *"**N**" – No hacer commit*

→ `void setAutoCommit(boolean value)`

Permite establece el valor del autoCommit

Entradas	`boolean`	*Indica si se activa o no el autoCommit* → *"**S**" – Autocommit activado* → *"**N**" – AutoCommit no activado. Se precisará de control de commit explícito por parte de la aplicación cliente..*

→ `void getAutoCommit()`

Devuelve el valor del autoCommit

Salida	`boolean`	*Valor de autoCommit establecido actualmente*

→ `void commit()`

Realiza un commit en la base de datos, salvando los cambios

→ `boolean rollback()`

Realiza un rollback en la base de datos lo que deshace los últimos cambios

Salida	`boolean`	*Devuelve true si el rollback se hizo con éxito y false en caso contrario*

→ `synchronized void establecerConexionFija(boolean fija)`

Permite establecer la conexión como fija, es decir una vez que se crea no se cierra hasta que no se ejecute el método cerrarSesion. Este método sólo tiene efecto cuando la TrAPIUTL se ha creado con un perfil de conexión que sea un datasource.

Si se está usando un perfil de conexión que es un datasource y el conexionFija es false se devuelve la conexión al finalizar la ejecución de un método, en caso contrario no se devuelve.

Entrada	`boolean fija`	Indica si la conexión es fija para la TrAPIUTL o se devuelve al pool de conexiones al finalizar la ejecución de un método. → "**true**" – La conexión es fija y no se devuelve al pool. Por defecto toma este valor. → "**false**" – Se devuelve la conexión al pool al terminar la ejecución de cada método, haciendose commit al final de cada método.

→ `synchronized boolean obtenerEstadoConexionFija()`

Devuelve el valor actual del atributo conexionFija. El estado de la conexión fija sólo tiene efecto cuando la TrAPIUTL se ha creado usando un perfil de conexión que es un datasource. Si el valor devuelto es true indica que la conexión se fija y no se devuelve al pool hasta cerrar la sesión, en caso contrario si el valor es false indica que no es fija y que se devuelve la conexión al pool al finalizar la ejecución de cada método.

Salida	`boolean`	Indica si la conexión es fija para la TrAPIUTL o se devuelve al pool de conexiones al finalizar la ejecución de un método. → "**true**" – La conexión es fija y no se devuelve al pool. Por defecto toma este valor. → "**false**" – Se devuelve la conexión al pool al terminar la ejecución de cada método, haciendose commit al final de cada método.

→ `void establecerUsuarioSistema(String usuario) throws TrException`

Establece un usuario no ocracle

Entradas	`String usuario`	*Nombre del usuario*

→ `TrSistema[] obtenerSistemas() throws TrException`

Devuelve los sistemas disponibles en Trew@

Salida	`TrSistema[]`	*Array de objetos TrSistema*

Clase Factoría

La clase Factoría se encarga de crear una instancia correcta de la J-TrAPIUTL de forma que los métodos de esta última aparezcan disponibles para las aplicaciones.

TrAPIUTLFactory

trewa.bd.trapi.trapiutl.TrAPIUTLFactory

Clase que aporta funcionalidad para la correcta creación de una instancia de la TrAPIUTL.

Métodos para la creación de instancias del TrAPIUTL

→ `TrAPIUTL crearAPIUTL()`

Devuelve una instancia del TrAPIUTL. La información para el establecimiento de la conexión la obtiene a través del perfil "default".

Salidas	`TrAPIUTL`	Instancia del TrAPIUTL o "null" si no se pudo crear.

→ `TrAPIUTL crearAPIUTL( String usuario, String clave)`

Devuelve una instancia del TrAPIUTL. La información para el establecimiento de la conexión la obtiene a través del perfil "default" pero en este caso hace uso del usuario y password que recibe como parámetros.

Entradas	*String* usuario	Usuario
	String clave	Clave de usuario
Salidas	*TrAPIUTL*	Instancia del TrAPIUTL o "null" si no se pudo crear.

→ `TrAPIUTL crearAPIUTL( String perfil )`

Devuelve una instancia del TrAPIUTL. La información para el establecimiento de la conexión la obtiene a través del perfil especificado en la llamada como parámetro.

Entradas	*String* perfil	*Nombre del perfil para el establecimiento de la conexión.*
Salidas	*TrAPIUTL*	Instancia del TrAPIUTL o "null" si no se pudo crear.

→ `TrAPIUTL crearAPIUTL( String perfil )`

Devuelve una instancia del TrAPIUTL. La información para el establecimiento de la conexión la obtiene a través del perfil especificado en la llamada como parámetro.

Entradas	*String perfil*	*Nombre del perfil para el establecimiento de la conexión.*
	String usuario	*Usuario*
	String clave	*Clave de usuario*
Salidas	*TrAPIUTL*	*Instancia del TrAPIUTL o "null" si no se pudo crear.*

Ejemplo de uso

En el siguiente ejemplo se muestra cómo obtener una instancia del TrAPIUTL a través de la clase TrAPIUTLFactory:

```
TrAPIUTL apiUTL = null;
Imputstream is = null;
TpoPK idExpediente = null;
TpoString strErr = null;

apiUTL = TrAPIUTLFactory.crearAPIUI( "perfilTrewa" );
if (apiUTL!= null ){
       // Ya se puede acceder a los métodos del TrAPIUTL
       idExpediente = new TpoPK( BigDecimal.valueOf(2) );
       strErr = new TpoString("");
       try{
              is = apiUTL.descargaExpediente( idExpediente, strErr );
              ...
       }
       catch( TrException ex )
       {
          ex.printStackTrace();
       }
       catch( Exception ex )
       {
          ex.printStackTrace();
       }
}
```

Clases de entidad

Estas clases se usan para el intercambio de información entre las aplicaciones cliente y la interfaz TrAPIUTL.

TrAPIUTLArchivo

trewa.bd.trapi.trapiutl.TrAPIUTLArchivo

Clase que representa a un archivo para insertar u obtener de un sistema Trew@.

Atributos accesibles mediante métodos get/set

Atributo	Tipo	Descripción
→ **id**	TpoPK	Identificador del archivo.
→ **nombre**	String	Nombre del archivo.
→ **tipo**	String	Tipo de archivo
→ **mime**	String	Tipo de mime del archivo

TrAPIUTLIO

trewa.bd.trapi.trapiutl.TrAPIUTLIO

Clase que representa a una conjunto de archivos a descargar del sistema Trew@.

Métodos públicos

→ `TrAPIUTLArchivo[] getArchivos()`
Método que devuelve el conjunto de archivos.

Servlets de subida y descarga de XML (DDP, Expedientes y Entidades del sistema)

Para la subida y descarga de archivos XML, junto con la TrAPIUTL, se aportan tres servlets que interactúan con la misma.

Servlet DescargaXML

El servlet DescargaXML obtiene de TREW@ un fichero XML que puede ser una Definición de Procedimiento, un Expediente o una definición de componentes de un sistema según se le indique en la petición.

Los parámetros del servlet se indican en la siguiente tabla:

Parámetro	Ámbito	Ob.	Descripción
tipo	Request	Sí	Indica el tipo de XML que se desea obtener de entre los siguientes: → "ddp" -- Definición de procedimiento. → "expediente" -- Expediente. → "des" -- Definición de componentes del sistema (en este caso devuelve un zip si existen archivos asociados)
id	Request	Sí	Identificador clave de la Definición de procedimiento, del expediente o del sistema, según el tipo de archivo que se desee obtener.
perfil	Request	Sí	Perfil con el que se intentará realizar la conexión. Por defecto tomará el perfil "default"
usuario	Session	No	Usuario con el que se realiza la conexión.
clave	Session	No	Password de usuario.
apiUTL	Session	No	Interfaz TrAPIUTL. Si no se le pasa intentará crear una interfaz con los parámetros por defecto.

El servlet DescargaXML necesita una conexión con la base de datos para obtener la información necesaria. Para ello se hace uso de la interfaz TrAPIUTL. Existen varias formas de establecer dicha conexión:

1. Crear un objeto de la clase TrAPIUTL con la conexión a la base de datos establecida y poner dicho objeto en la sesión.

2. Pasar como parámetros un perfil que tenga usuario y clave válidos

3. Pasar todos los parámetros, perfil, usuario y clave.

4. En caso de que no se logre realizar la conexión, existe la posibilidad de pasar como parámetro (login) una página jsp en la que el usuario deberá identificarse.

Ejemplo de llamada al servlet DescargaXML

→ Llamada desde una URL
Descarga de procedimiento
`http://servidorWeb/aplicacionJSP/trewa/DescargaXML?tipo=ddp&id=406`
Descarga de entidades del sistema
`http://servidorWeb/aplicacionJSP/trewa/DescargaXML?tipo=des&id=5`
 	
→ Llamada desde un javascript
`location.href='trewa/DescargaXML?tipo=des&id=2`

Servlet SubidaXML

El servlet SubidaXML inserta en TREW@ la información obtenida de una archivo XML que consistirá en la Definición de un Procedimiento.

Los parámetros del servlet se indican en la siguiente tabla:

Parámetro	Ámbito	Ob.	Descripción
perfil	Request	Sí	Perfil con el que se intentará realizar la conexión.
act	Request	No	Indica si se actualiza la información existente para la Definición de procedimiento. Puede ser “true” o “false”. El valor por defecto es “false”.
usuario	Session	No	Usuario con el que se realiza la conexión.
clave	Session	No	Password de usuario.
apiUTL	Session	No	Interfaz TrAPIUTL

Servlet SubidaXML en formato ZIP

El servlet SubidaXML inserta en TREW@ la información obtenida de un archivo ZIP que contiene la Definición de un Procedimiento junto con archivos adicionales asociados a la definición.

Los parámetros del servlet se indican en la siguiente tabla:

Parámetro	Ámbito	Ob.	Descripción
apiUTL	Session	Sí	Interfaz TrAPIUTL
act	Request	No	Indica si se actualiza la información existente para la Definición de procedimiento. Puede ser “true” o “false”. El valor por defecto es “false”.

Despliegue y configuración de las J-TrAPIs

Despliegue de las J-TrAPIs

Para que las J-TrAPIs puedan estar accesibles desde diferentes aplicaciones, deben estar desplegadas en el mismo servidor de aplicaciones donde dichas aplicaciones residan. El despliegue de las mismas variará en función del servidor de aplicaciones donde se vaya a realizar y en todo caso también se tendrá que realizar una configuración para habilitar el acceso a los servlets.

Despliegue en Tomcat

Incluir la librería **trewa.jar** en el directorio **%Tomcat%\common\lib**, o desplegar de forma completa las clases en el directorio **%Tomcat%\common\classes**.

Despliegue en Jboss

Incluir la librería **trewa.jar** en el directorio **%JBOSS%\server\default\lib** para que todas las aplicaciones desplegadas a nivel default puedan acceder a ella.

Configuración de servlets

Para que una aplicación pueda acceder a los servlets de subida y descarga de documentos (TrAPIUI), para los de subida y descarga de XML (TrAPIUTL) y para 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” del servidor 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>
```

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>
```

La ubicación por defecto del archivo "**web.xml**" variará en función del servidor de aplicaciones:

- En **Tomcat** el archivo "web.xml" se encuentra en la carpeta **%Tomcat%\conf** .
- En **JBoss** el archivo "web.xml" se encuentra en la carpeta **%JBOSS%\server\default\deploy\jbossweb-tomcat50.sar .**

Nota: Estas ubicaciones son a modo orientativo, en todo caso se recomienda recurrir a la documentación técnica del servidor de aplicaciones en uso para garantizar una correcta instalación de las J-TrAPIs.

Configuración de perfiles de conexión

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

En la actualidad las J-TrAPIs permiten dos posibles usos de perfil de conexión:

- Configuración básica con archivo de properties
- Configuración con DataSources a nivel del servidor de aplicaciones

A nivel de aplicación el uso de un perfil u otro va a ser totalmente transparente y no se va a 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.

Configuración básica

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***.

Posibles ubicaciones para el archivo de perfiles

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

En todo caso será conveniente analizar varias posibles situaciones para asegurarse de que el perfil sea detectado por las TrAPIs. En la tabla adjunta se muestra una comparativa de la ubicación del archivo trewa.jar y del archivo de configuración

indicándose si la localización se llevará a cabo o no:

Ubicación del archivo trewa.jar en el servidor de aplicaciones	***Ruta default.properties en el servidor de aplicaciones***	***Observaciones***
TOMCAT		
common/lib	common/classes/trewa/conf/perfiles	El perfil es localizado por las TrAPIs
common/lib	webapps/<APLICACIÓN>/classes /trewa/conf/perfiles	El perfil no es localizado por las TrAPIs
webapps/<APLICACIÓN>/lib	common/classes/trewa/conf/perfiles	El perfil es localizado por las TrAPIs
webapps/<APLICACIÓN>/lib	webapps/<APLICACIÓN>/classes /trewa/conf/perfiles	El perfil es localizado por las TrAPIs
JBOSS		
server/default/lib	deploy/<APLICACIÓN.war>	El perfil no es localizado por las TrAPIs
deploy/<APLICACIÓN.war>	deploy/<APLICACIÓN.war>	El perfil es localizado por las TrAPIs

Formato del fichero de properties

La configuración del archivo de properties se realiza editando el archivo y publicando los siguientes campos:

Campo	Ob.	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	No	Nombre de usuario
claveUsuario	No	Clave de acceso

Un ejemplo de archivo de properties sería:

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

Si no se especifica usuario, este será exigido a través de las J-TrAPIs para establecer la conexión. En este caso se dejaría el campo en el archivo pero sin valor asignado.

A partir de la versión 1.2.0 de Trew@ 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 conexion
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=

Configuración con DataSources

En este caso se intenta aprovechar la configuración de DataSources que aportan la mayoría de los servidores de aplicaciones, ganando así la posibilidad de habilitar un pool de conexiones que será en todo caso el administrador del servidor el que se encargue de gestionar debidamente.

Esta gestión a través del DataSource va a redundar en beneficio del rendimiento de la base de datos. Estableciéndose una serie de conexiones (configuradas por el administrador) al inicio y reutilizándose las mismas, se reducirá el número de conexiones y desconexiones a la base de datos, reduciéndose así el uso de recursos de la base de datos. Es preferible en muchos casos tener un número de sesiones predefinidas abiertas que no ir abriendo y cerrando sesiones a la base de datos.

En los apartados siguientes se muestra una orientación básica de cómo configurar los DataSources en diferentes servidores de aplicaciones, remitiendo al lector a los manuales de configuración del servidor de aplicaciones en uso para cuestiones de configuración.

Configuración de DataSource en Tomcat (5.0.18)

Para crear un DataSource en Tomcat se debe modificar el archivo **server.xml** que está ubicado en **%TOMCAT%\conf**, incluyendo la información referente al DataSource y a la aplicación que lo usará.

Entre las etiquetas <GlobalNamingResources> y </GlobalNamingResources> se debe aportar la información del DataSource:

Etiqueta Resource		
Parámetro	**Ob.**	**Descripción**
name	Sí	Nombre del DataSource
type	Sí	Clase del DataSource
Etiqueta ResourceParams		
Parámetro	**Ob.**	**Descripción**
name	Sí	Nombre del DataSource
url	Sí	Cadena de conexión JDBC
password	Sí	Contraseña
maxActive	No	Máximo número de conexiones activas
maxWait	No	Máximo número de conexiones en espera
driverClassName	Sí	Clase que implementa el driver de conexión

maxIdle	No	Máximo de conexiones que se dejan abiertas en espera de ser utilizadas
username	Sí	Usuario

La estructura para las etiquetas *Resource* y *ResourceParams*, en el server.xml quedaría:

```
<Resource name="trOracle" type="javax.sql.DataSource"/>
<ResourceParams name="trOracle">
 <parameter>
        <name>url</name>
        <value>jdbc:oracle:thin:@desa.ja.es:1521:bdOracle</value>
 </parameter>
 <parameter>
        <name>password</name>
    <value>Pwd</value>
 </parameter>
 <parameter>
        <name>maxActive</name>
    <value>10</value>
 </parameter>
 <parameter>
        <name>maxWait</name>
        <value>5000</value>
 </parameter>
 <parameter>
         <name>driverClassName</name>
     <value>oracle.jdbc.driver.OracleDriver</value>
 </parameter>
 <parameter>
        <name>maxIdle</name>
    <value>2</value>
 </parameter>
 <parameter>
        <name>username</name>
    <value>Usuario</value>
 </parameter>
</ResourceParams>
```

Para completar la configuración, hay que indicar la aplicación (o aplicaciones) que va a usar el DataSource definido anteriormente. Para ello se debe añadir entre las etiquetas <Host> y </Host> la siguiente información (un contexto por aplicación):

```
<Context path="/AppWeb" docBase="AppWeb" debug="0">
        <ResourceLink name="trOracle" global="trOracle" type="javax.sql.DataSource"/>
</Context>
```

En el caso anterior se indica que la aplicación **AppWeb,** que se encuentra en el directorio **%TOMCAT%/webapps/AppWeb,** va a hacer uso del DataSource definido como **trOracle.**

El archivo **server.xml,** una vez modificado, quedaría del siguiente modo:

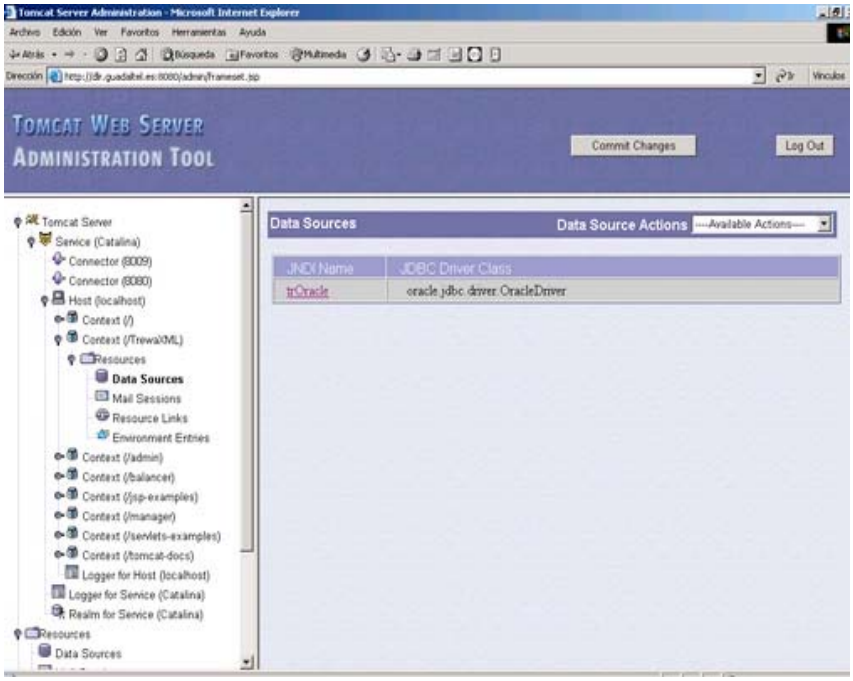
```
<?xml version='1.0' encoding='utf-8'?>
<Server>
  <Listener className="org.apache.catalina.mbeans.ServerLifecycleListener"/>
  <Listener className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener"/>
 <GlobalNamingResources>
    <Environment name="simpleValue" type="java.lang.Integer" value="30"/>
    <Resource auth="Container" description="User database that can be updated and saved"
name="UserDatabase" type="org.apache.catalina.UserDatabase"/>
    <ResourceParams name="UserDatabase">
      <parameter>
        <name>factory</name>
        <value>org.apache.catalina.users.MemoryUserDatabaseFactory</value>
      </parameter>
      <parameter>
        <name>pathname</name>
        <value>conf/tomcat-users.xml</value>
      </parameter>
    </ResourceParams>
   <Resource name="trOracle" type="javax.sql.DataSource"/>
    <ResourceParams name="trOracle">
      <parameter>
         <name>url</name>
         <value>jdbc:oracle:thin:@desa.ja.es:1521:bdOracle</value>
     </parameter>
     <parameter>
         <name>password</name>
         <value>Pwd</value>
     </parameter>
     <parameter>
         <name>maxActive</name>
```

```
        <value>10</value>
    </parameter>
    <parameter>
        <name>maxWait</name>
        <value>5000</value>
    </parameter>
    <parameter>
         <name>driverClassName</name>
         <value>oracle.jdbc.driver.OracleDriver</value>
    </parameter>
    <parameter>
         <name>maxIdle</name>
         <value>2</value>
    </parameter>
    <parameter>
         <name>username</name>
         <value>Usuario</value>
    </parameter>
  </ResourceParams>
</GlobalNamingResources>
<Service name="Catalina">
   <Connector acceptCount="100" connectionTimeout="20000" disableUploadTimeout="true"
port="8080" redirectPort="8443">
   </Connector>
   <Connector port="8009" protocol="AJP/1.3"
protocolHandlerClassName="org.apache.jk.server.JkCoyoteHandler" redirectPort="8443">
   </Connector>
   <Engine defaultHost="localhost" name="Catalina">
     <Host appBase="webapps" name="localhost">
       <Logger className="org.apache.catalina.logger.FileLogger" prefix="localhost_log."
suffix=".txt" timestamp="true"/>
   <Context path="/AppWeb" docBase="AppWeb" debug="0">
         <ResourceLink name="trOracle" global="trOracle" type="javax.sql.DataSource"/>
    </Context>
 </Host>
     <Logger className="org.apache.catalina.logger.FileLogger" prefix="catalina_log."
suffix=".txt" timestamp="true"/>
     <Realm className="org.apache.catalina.realm.UserDatabaseRealm"/>
   </Engine>
  </Service>
</Server>
```

Datasource para una aplicación cliente desde Tomcat

Se puede defininir un DataSource asociado a la aplicación desde las pantallas de administración de Tomcat. La configuración definida quedará registrada en el archivo:

<rutaTomcat>\conf\Catalina\localhost\<nombreApp>.xml



```
<?xml version='1.0' encoding='utf-8'?>
<Context displayName="Seervicios Web de Trewa" docBase="C:/Servidores/Tomcat50N/webapps/trewaws"
path="/trewaws">
  <Resource name="jdbc/trewaws" type="javax.sql.DataSource"/>
  <ResourceParams name="jdbc/trewaws">
    <parameter>
      <name>maxWait</name>
      <value>5000</value>
    </parameter>
    <parameter>
      <name>maxActive</name>
      <value>25</value>
    </parameter>
    <parameter>
      <name>password</name>
      <value>pwd</value>
    </parameter>
    <parameter>
      <name>url</name>
      <value>jdbc:oracle:thin:@máquina:1521:bd</value>
    </parameter>
    <parameter>
      <name>driverClassName</name>
      <value>oracle.jdbc.driver.OracleDriver</value>
    </parameter>
    <parameter>
      <name>maxIdle</name>
      <value>1</value>
    </parameter>
    <parameter>
      <name>username</name>
      <value>user</value>
    </parameter>
  </ResourceParams>
</Context>
```

Configuración de Datasource en JBoss (4.0.0/4.0.1)

Desde JBoss es necesario modificar o crear el archivo **oracle-ds.xml**. Este archivo debe estar ubicado en **%JBOSS%\server\default\deploy** y para el DataSource que se vaya a crear se tendrá que aportar la siguiente información:

Etiqueta	**Ob.**	**Descripción**
<jndi-name>	Sí	Nombre del DataSource
<connection-url>	Sí	Cadena de conexión JDBC
<driver-class>	Sí	Clase que implementa el driver de conexión
<user-name>	No	Nombre de usuario
<password>	No	Contraseña
<min-pool-size>	No	Mínimo número de conexiones en el pool
<max-pool-size>	No	Máximo número de conexiones en el pool
<type-mapping>	Sí	Tipo de base de datos

La estructura del oracle-ds.xml vendría a ser:

```
<datasources>
  <local-tx-datasource>
    <jndi-name>[Nombre del DataSource]</jndi-name>
    <connection-url>[Cadena de conexión jdbc]</connection-url>
    <driver-class>[Clase java que implementa el driver de conexión]</driver-class>
    <user-name>[Nombre de usuario]</user-name>
    <password>[Contraseña]</password>
    <exception-sorter-class-name>
            org.jboss.resource.adapter.jdbc.vendor.OracleExceptionSorter
    </exception-sorter-class-name>
    <min-pool-size>[mínimo de conexiones en el pool]</min-pool-size>
    <max-pool-size>[máximo de conexiones en el pool]</max-pool-size>
    <metadata>
      <type-mapping>Oracle9i</type-mapping>
    </metadata>
  </local-tx-datasource>
  <local-tx-datasource>
          .....
  </local-tx-datasource>
</datasources>
```

El caso descrito en la configuración con archivos de properties quedaría en este caso del siguiente modo:

La estructura del oracle-ds.xml sería:

```
<datasources>
  <local-tx-datasource>
    <jndi-name>trOracle</jndi-name>
    <connection-url>jdbc:oracle:thin:@desa.ja.es:1521:bdOracle</connection-url>
    <driver-class>oracle.jdbc.driver.OracleDriver</driver-class>
    <user-name>UsrComun user-name>
    <password>Pwd</password>
    <exception-sorter-class-name>
            org.jboss.resource.adapter.jdbc.vendor.OracleExceptionSorter
    </exception-sorter-class-name>
    <min-pool-size>5</min-pool-size>
    <max-pool-size>20</max-pool-size>
    <metadata>
        <type-mapping>Oracle9i</type-mapping>
    </metadata>
  </local-tx-datasource>
  <local-tx-datasource>
          .....
  </local-tx-datasource>
</datasources>
```

En este caso las aplicaciones accederán usando el perfil **trOracle** ya que así se ha definido por parte del administrador y no tendrían que validarse pues en la propia especificación del DataSource lleva incluido usuario y password.

Librerías necesarias

JDK

→ Para el correcto funcionamiento de las TrAPIs se precisa de la **versión 1.4.2 del JDK/JRE** o posterior.

JDBC Oracle

→ Las TrAPIs hacen uso de la librería **ojdbc14.jar** que contiene las clases de **JDBCDrivers Oracle10g.** Para que las TrAPIs hagan uso de estas clases, dicha librería puede estar en uno de estos directorios:

Despliegue en Tomcat

Incluir la librería **ojdbc14.jar** en el directorio **%Tomcat%\common\lib**.

Despliegue en Jboss

Incluir la librería **ojdbc14.jar** en el directorio **%JBOSS%\server\default\lib**.

El uso de la librería **ojdbc14.jar** permite, entre otras cosas, el uso de **Savepoint** y **Rollback**.

IMPORTANTE: Si durante la ejecución de la aplicación JSP aparece el siguiente error:

```
ERROR: java.lang.AbstractMethodError: java.sql.Savepoint
       java.sql.Connection.setSavepoint()
```

la causa puede ser que no haya reconocido la librería **ojdbc14.jar**. Esto puede ser por varios motivos: por que no esté colocada en el directorio correcto, o porque en el directorio donde esté colocada exista la librería **clases12.jar** que también incluye los drivers de Oracle pero no soporta Savepoint ni Rollback. Una posible solución a este último caso es renombrar la librería **ojdbc14.jar** con un nombre menor que **clases12.jar** para que encuentre antes la librería **ojdbc14.jar** que la **clases12.jar.** Se podría renombrar la librería **ojdbc14.jar** a **aojdbc14.jar** por ejemplo.

Despliegue en OC4J

Para asegurar que en oc4j se carga inicicialmente el jdbc y otras clases necesarias se debe incluir junto con la aplicación el archivo orion-web.xml con el siguiente contenido:

```
<?xml version = '1.0' encoding = 'windows-1252'?>
<!DOCTYPE orion-web-app PUBLIC "-//Evermind//DTD Orion Web Application
2.3//EN" "http://xmlns.oracle.com/ias/dtds/orion-web.dtd">
<orion-web-app servlet-webdir="/servlet/">
  <web-app-class-loader search-local-classes-first="true"/>
</orion-web-app>
```

Este archivo se ubicará en la carpeta WEB-INF de la aplicación.

Drivers Oracle Database 10g (10.1.0.4) - Versión 10.1.0.2.0 driver jdbc

Archivos

ojdbc14.jar	Clases para usar con JDK 1.4
orai18n.jar	Clases NLS (National Language Support) para usar con JDK 1.2, 1.3 y 1.4. Este jar reemplaza a los antiguos nls_charset jar/zip. NLS permite obtener e insertar información en la base de datos en cualquier juego de caracteres que soporte Oracle.
ocrs12.jar	Clases que implementan las interfaces javax.sql.rowset CachedRowSet y WebRowSet para ser usadas con JDK 1.2, 1.3 y 1.4

Drivers Oracle Database 10g Release 2 - Versión 10.2.0.1.0 driver jdbc

Archivos

ojdbc14.jar	Clases para usar con JDK 1.4 y ***JDK 1.5***
orai18n.jar	Clases NLS (National Language Support) para usar con JDK 1.2, 1.3, 1.4 y ***1.5***. Este jar reemplaza a los antiguos nls_charset jar/zip. NLS permite obtener e insertar información en la base de datos en cualquier juego de caracteres que soporte Oracle.

Otras librerías necesarias

Para el correcto funcionamiento de Trew@ se precisa una serie de librerías adicionales que se aportan junto con la distribución y que se describen a continuación:

Librerías adicionales

Jdbc PostgreSQL 8	Librería para acceso a base de datos con postgreSQL. Necesaria si se monta la base de datos sobre PostgreSQL
Jdom 1.0	Librería para parseo de XML. Necesaria para habilitar la importación y exportación de definición de procedimientos, expedientes y entidades del sistema. *(Ver APIUTL)*
Log4java 1.2.9	Librería para trazabilidad de errores. Necesaria si se quiere habilitar la trazabilidad de la aplicación para detectar posibles errores. *(Ver Trazabilidad con log4java)*
Clientes webServices	Clientes necesarios para habilitar el acceso a @visador, notific@dor, port@firmas, w@rda y trew@
Axis 1.1	Librerías necesarias para desplegar los servicios web de Trew@ *(Ver Montaje de servicios Trew@)*

Trazabilidad con log4java

Para habilitar la trazabilidad de log4java en trewa, bastará con modificar dos archivos:

1.- <*web_app*>*/WEB-INF/classes/trewa/conf/trewaconf.properties*

En este archivo se tendrá que especificar la ruta del archivo de configuración log4java:

```
#Ruta del fichero de configuración para Log4Java
log_4j_configuration_file=/trewa/log4j.properties
```

Campo	Ob.	Descripción
log_4j_configuration_file	No	Ruta donde está ubicado el fichero de configuración para log4java

2.- El archivo *log4j.properties*

A través de este archivo se podrá configurar el nivel de trazabilidad que se desea habilitar sobre trewa.

```
#Logger para aplicación TREWA
log4j.logger.trewa=DEBUG, FILE_LOG_TREWA_APPENDER
log4j.additivity.trewa=false

#Appender usado por aplicación TREWA
log4j.appender.FILE_LOG_TREWA_APPENDER=org.apache.log4j.RollingFileAppender
log4j.appender.FILE_LOG_TREWA_APPENDER.File=/trewa.log

log4j.appender.FILE_LOG_TREWA_APPENDER.layout=org.apache.log4j.PatternLayout
log4j.appender.FILE_LOG_TREWA_APPENDER.layout.ConversionPattern=%d{[yyyy-MM-dd
HH:mm:ss,SSS]} %5p: %m%n
log4j.appender.FILE_LOG_TREWA_APPENDER.MaxFileSize=10MB
log4j.appender.FILE_LOG_TREWA_APPENDER.MaxBackupIndex=10
```

Para cambiar la configuración de los niveles de trazabilidad sobre Trew@, se modificará la entrada `log4j.logger.trewa` 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 similares 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 2 niveles extras 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.

```
log4j.logger.trewa=OFF, FILE_LOG_TREWA_APPENDER
```

Configuración para WebOffice

Para el correcto funcionamiento de WebOffice 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
```

Campo	Ob.	Descripción
perfil	No	Perfil de acceso para WebOffice

Montaje de los Servicios Web de Trew@

Para el correcto funcionamiento de los servicios web de Trew@ se precisa realizar el despliegue de una aplicación con las siguientes características:

1.- Definición de los perfiles de acceso. Para esto se creará el archivo **trewaws.properties** en el paquete **trewa.conf,** donde se especificarán los perfiles a usar por los diferentes servicios. El archivo *trewaws.properties* tiene el siguiente formato:

```
perfil=java:comp/env/jdbc/trewaws
perfil_trewa=java:comp/env/jdbc/trewaws1
perfil_modelo=java:comp/env/jdbc/trewaws2
perfil_warda=java:comp/env/jdbc/trewaws3
```

Campo	Ob.	Descripción
perfil	No	Perfil por defecto para todos los servicios
perfil_trewa	No	Perfil por defecto para el servicio de trewa. Si no se especifica esta entrada se tomará la que se indique en la entrada *perfil*.
perfil_modelo	No	Perfil por defecto para el servicio de actualización de codificadoras. Si no se especifica esta entrada se tomará la que se indique en la entrada *perfil*.
perfil_warda	No	Perfil por defecto para el servicio de warda. Si no se especifica esta entrada se tomará la que se indique en la entrada *perfil*.

2.- Inclusión en la carpeta **public_html\WEB-INF** de la aplicación los archivos **web.xml** y **server-config.wsdd**. Ambos archivos se aportan junto con la plantilla de ejemplo para montar el servicio web.

3.- Inclusión de las librerías necesarias.

Condiciones, acciones y variables Java

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 y/o variable en un procedimiento con <<Tipo de 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 y **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

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

4.- La clase debe aparecer accesible a las clases de Trew@. Esto se puede conseguir de alguna de las formas siguientes (se muestra ejemplo para Tomcat):

Si trewa.jar está a nivel de aplicación

(%TOMCAT%\webapps\AplicacionJSP\WEB-INF\lib\trewa.jar)

En este caso las clases de condiciones y/o variables podrán ubicarse en:

```
- %TOMCAT%\webapps\AplicacionJSP\WEB-INF\lib ( empaquetadas en un jar )
- %TOMCAT%\webapps\AplicacionJSP\WEB-INF\classes ( los .class )
- %TOMCAT%\common\lib ( empaquetadas en un jar )
- %TOMCAT%\common\classes ( los .class )
```

Si trewa.jar está a nivel de servidor

(%TOMCAT%\common\lib\trewa.jar)

En este caso las clases de condiciones y/o variables podrán ubicarse en:

```
- %TOMCAT%\common\lib ( empaquetadas en un jar )
- %TOMCAT%\common\classes ( los .class )
```

5.- 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

Derivando 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.

```
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;
    }
}
```