

# Guía de instalación y configuración de StateEngine v2.0

Edición 2.0  
28/09/2006  
SMARTFlow StateEngine - Proyecto Morfeo

<http://forge.morfeo-project.org/>

## Tabla de contenidos

Introducción	2
Instalación y configuración	2
Fichero configuration.properties	4
Ejecución del script de la base de datos	5
Arranque de TIDStateEngine	6
Parámetros de invocación de la componente	7
Parada de TIDStateEngine	7
Guía rápida de uso	8

## Introducción

Este documento trata de describir de una forma rápida, sencilla y comprensible el procedimiento de instalación y configuración de la distribución binaria de la componente SMARTFlow *StateEngine* v2.0 disponible en el sitio web del proyecto MORFEO.

Al mismo tiempo, en esta guía se pretende proporcionar documentación adicional de referencia relativa a configuración avanzada del motor *StateEngine*.

El actual documento se centra en la distribución binaria de *StateEngine*. El código fuente se distribuye en forma de proyecto para Eclipse de manera que sea sencilla su compilación y ejecución desde este entorno de desarrollo. Por lo tanto, desde esta guía se recomienda el empleo del IDE Java de software libre Eclipse. En cualquier caso, se asume que el personal desarrollador (los cuales serán los que empleen la versión en código fuente) posee los conocimientos suficientes para compilar el código fuente e invocar las clases ejecutables situadas en el paquete `org.morfeo.corba.TIDStateEngine` para manejar el contenedor aunque no disponga del entorno de desarrollo Eclipse.

## Instalación y configuración

Existen dos versiones binarias para poder descargar, las cuales se corresponden con las plataformas UNIX/Linux y Windows. La distribución para UNIX/Linux se basa en un conjunto de scripts para la *shell* sh, muy habitual en ambos sistemas. La distribución para Windows emplea un conjunto de *scripts* .bat.

La distribución binaria no depende de ningún sistema externo (a excepción del SGBD en el que se realizará la persistencia de los datos manejados por SMARTFlow-*StateEngine*), por lo tanto, es un entorno bastante sencillo de instalar y configurar. Actualmente, *StateEngine* funciona correctamente para los motores de bases de datos ORACLE, PostgreSQL y MySQL.

Los ficheros que contienen la distribución binaria (fichero .zip para Windows y .tar.gz para UNIX/Linux) han de ser descomprimidos en el lugar donde se desea instalar el software. Los directorios creados tras descomprimir se estructuran de una forma similar a un sistema de ficheros UNIX:

- bin: *Scripts* que permiten la ejecución y detención del sistema.
  - tools: En este subdirectorio se encuentran los *scripts* para realizar operaciones sobre el motor de estados (insertar una red, crear una tarea, etc)
- lib: Contiene las librerías necesarias para la ejecución del motor en forma .jar.
- etc: Tiene el único fichero de configuración del sistema (*configuration.properties*)
- idl: Almacena los interfaces IDL de los servicios CORBA presentes en el motor.
- sql: *Scripts* de creación de la base de datos que necesita el sistema.
- jdbc: Contiene dentro de su subdirectorio *lib*, las librerías con las clases JDBC del motor de bases de datos empleado.
- samples: Directorio donde se almacenan los ejemplos distribuidos con la aplicación (ver el documento "Tutorial de utilización básica").

- `deployCustomClasses`: Ubicación donde hay que situar las clases asociadas a la red de Petri (ver el documento “Tutorial de utilización básica”).
- `nets`: Directorio donde se sitúan las definiciones de las redes de Petri en formato PNML (XML)
- `xsd`: Directorio donde se guardan los esquemas XML que deben cumplir las redes.

En el directorio raíz de la distribución, se incluye un *script* de inicialización, que configura el entorno de ejecución (variables de entorno necesarias). Dicho script de llama `setup_TIDStateEngine.bat` (Windows) y `setup_TIDStateEngine` (UNIX/Linux). Para que el mencionado *script* funcione correctamente, se debe establecer correctamente la variable de entorno `JAVA_HOME`. El uso de dicho *script* sería el siguiente:

- Plataforma Windows (desde una *shell* `msdos`):

```
> setup_TIDStateEngine.bat
```

- Plataforma UNIX/Linux (desde *shell* `sh/bash`):

```
$ . setup_TIDStateEngine
```

A partir de este momento, tenemos configuradas las variables de entorno necesarias para la ejecución del entorno en la *shell* en que hemos lanzado el script anterior. Queda configurar el sistema en el fichero `etc/configuration.properties`.

El fichero `configuration.properties` permite especificar ciertos aspectos de la componente en base a los valores de propiedades almacenadas en dicho fichero. Este archivo permite configurar el comportamiento del motor de una forma bastante profunda. En este apartado comentaremos un pequeño grupo de parámetros. Para un uso más avanzado de este fichero, consulte el apartado “Fichero `configuration.properties`”.

La mayoría de parámetros del fichero que acompaña a la distribución, están configurados para que funcionen los ejemplos que se distribuyen en la versión binaria. Por lo tanto, desde esta guía rápida, se recomienda que se modifiquen únicamente los parámetros relativos a la conexión JDBC con la base de datos. Cuando el usuario sea capaz de ejecutar completamente el ejemplo que acompaña a la distribución binaria, entonces creemos que podría plantearse realizar configuraciones personalizadas.

En principio, para ejecutar el ejemplo distribuido, los únicos parámetros a modificar serían los reflejados en el siguiente cuadro. Lógicamente se debe dar los valores adecuados para la conexión JDBC que desee establecer el usuario.

```
#CONFIGURACION DE LA BASE DE DATOS OPERACIONAL
#Nombre de usuario para acceder a la base de datos
db.databaseUser= user
#Password del usuario anterior en la base de datos
db.databasePass = pass
#Nombre de la base de datos
db.name =database
#Puerto en el que escucha el SGBD
db.port=port
#Máquina en la que se instala el SGBD
db.host=machine
#Protocolo de comunicaciones a emplear en la conexión
db.protocol=tcp
#Driver a emplear en la base de datos. Normalmente opcional, para ORACLE usar thin
db.driver=driver
#Clase que implementa el pool de conexiones de la base de datos
#En este ejemplo se muestra la clase para ORACLE. Existe una versión
#equivalente a este clase para MySQL y para Postgre
db.className=org.morfeo.corba.TIDStateEngine.task.StateEngineOraclePool
#Número de conexiones que se guardan en el pool
```

```
db.poolMaxConnections = conexiones
```

Una vez actualizado adecuadamente el fichero de propiedades, ya se dispone del entorno adecuadamente configurado. Para comprobar la instalación, podemos ejecutar el *script* de inicio de la componente (véase “Arranque de TIDStateEngine”)

### Fichero configuration.properties

La configuración de la componente *TIDStateEngine* viene dada por un fichero de configuración llamado *configuration.properties*. Dicho fichero se sitúa en el directorio de configuración de la componente (*etc*). A continuación se detallaran sus distintos parámetros, así como un ejemplo de los distintos valores que pueden tomar:

- *GeneratedCodePkg = org.morfeo.corba.TIDStateEngine.petrinet.GeneratedCode*  
Indica el nombre del paquete de las clases generadas por la herramienta de compilación de redes (vea la guía de usuario).
- *GeneratedCodeDir=.*  
Indica el directorio donde se guardaran las clases generadas por la herramienta de compilación de redes. Las clases generadas por al herramienta, deben ser “rellenadas” por el usuario.
- *PetriXMLFilesDir=nets*  
Indica el directorio donde se encuentran los ficheros XML de definición de redes.
- *TaskCacheSize=0*  
Tamaño de la caché de tareas, para este caso particular vale 0.
- *db.databaseUser=user*
- *db.databasePass=pass*  
Identificación del usuario (mediante *login* y *password*) en la conexión JDBC operacional.
- *db.name = name*
- *db.port = port*
- *db.host = host*
- *db.protocol = protocol {tcp, udp}*
- *db.driver = thin*  
El parámetro *db.driver* es únicamente requerido si el SGBD es Oracle.  
Los cinco anteriores parámetros son los encargados de definir el origen de los datos de la conexión JDBC operacional.
- *db.className = class*  
Indica la clase que se encarga de obtener los datos de cada fuente de datos (*data-source*). Se dispone de una clase para cada motor de bases de datos compatible con *TIDStateEngine*.
  - *org.morfeo.corba.TIDStateEngine.task.StateEngineOraclePool (Oracle)*
  - *org.morfeo.corba.TIDStateEngine.task.StateEngineMySQLPool (MySQL)*
  - *org.morfeo.corba.TIDStateEngine.task.StateEnginePostGrePool (PostGreSQL)*
- *db.poolMaxConnection = connections*  
Número de conexiones que tiene el pool.

- *HistoryDb.databaseUser = user*
- *HistoryDb.databasePass=pass*

Parámetro de identificación del usuario de la base de datos histórica. Se pueden comentar las entradas en caso de no desear base de datos histórica.

- *HistoryDb.name = name*
- *HistoryDb.port = port*
- *HistoryDb.host = host*
- *HistoryDb.protocol = protocol {tcp, udp}*
- *HistoryDb.driver = thin*

El parámetro *db.driver* es únicamente requerido si el SGBD es Oracle.

Los cinco anteriores parámetros son los encargados de definir el origen de los datos de la conexión JDBC histórica.

- *HistoryDb.className = class*

Indica la clase que se encarga de obtener los datos de cada fuente de datos (*data-source*) histórica. Se dispone de una clase para cada motor de bases de datos compatible con *TIDStateEngine*.

- *org.morfeo.corba.TIDStateEngine.task.StateEngineOraclePool (Oracle)*
- *org.morfeo.corba.TIDStateEngine.task.StateEngineMySQLPool (MySQL)*
- *org.morfeo.corba.TIDStateEngine.task.StateEnginePostGrePool (PostGreSQL)*

- *HistoryDb.poolMaxConnection = connections*

Número de conexiones que tiene el *pool* de conexiones histórico.

- *NetAdministratorPOAManager\_MaxThreads = 10*
- *NetAdministratorPOAManager\_MinThreads = 10*
- *TasksPOAManager\_MaxThreads = 10*
- *TasksPOAManager\_MinThreads = 10*

Configuración de los *threads* de petición de los distintos objetos del servicio.

- *OperationalPoolMaxConnections = 20*
- *HistoryPoolMaxConnections = 20*

Configuración de los tamaños de los *pools* de datos.

- *MaxQueuedRequestForTask = 4*

Tamaño máximo de la cola de peticiones para una tarea determinada.

- *NotifierChannelIOR = channel.ior*

Fichero conteniendo una referencia a un canal válido, por el que la componente enviará las notificaciones internas. Caso de no existir el motor no realizará dichas notificaciones.

### Ejecución del script de la base de datos

La componente *SMARTFlow-StateEngine* realiza la persistencia de sus datos internos en base a un sistema gestor de bases de datos, por lo que, antes de arrancar el entorno hay que crear la base de datos que emplea el motor. Para ello, se proporciona en el directorio *sql* de la distribución los *scripts* de creación y borrado de la base de datos.

En el directorio *sql* de la distribución, se presentan varias carpetas que contienen los propios *scripts* de gestión de la base de datos para distintos SGBD. En la actualidad, *StateEngine* permite trabajar con los motores de bases de datos Oracle, MySQL y PostgreSQL. Dependiendo del motor a emplear, se seleccionarán los *scripts* adecuados para el motor en cuestión.

Para crear la base de datos, se elijen los *scripts* adecuados para el motor a emplear, se accede al mismo y se ejecutan los siguientes *scripts*:

- Ambas plataformas:

```
> sql/creacion_bd.sql (persistencia de datos)
$ sql/creacion_bd_history (histórico de la componente)
```

## Arranque de StateEngine

Se puede comprobar que todo está correctamente instalado, una vez ejecutado el *script* de inicialización y haber configurado adecuadamente el fichero *configuration.properties*. Para ello, se debe ejecutar desde la *shell* que ejecutó el *script* de inicialización (para conservar los valores de las variables de entorno) el siguiente comando:

- Plataforma Windows (desde la *shell* msdos anterior):

```
> bin/TIDStateEngineStart.bat
```

- Plataforma UNIX/Linux (desde *shell* sh/bash anterior):

```
$ bin/TIDStateEngineStart.sh
```

El comando anterior arranca el sistema que se queda a la espera de las peticiones de los clientes. Si el *script* de arranque proporciona una salida parecida a la mostrada en el siguiente listado, tenemos una versión correctamente instalada y configurada del motor.

```
Starting Service...
Set User: user
Set Password
Set Data Base name:database
Set connection port:1521
Set host:machine
Set Driver:driver
Set Max Connections:21
IOR:00000000000000254944c3a5374617465456e67696e652f4e657441646d696e697374726174
6f723a312e30000000000000000100000000000007000010200000000b50632d6d6163616e6173
00000bbb00000000003d000000000000001000000144e657441646d696e6973747261746f72504f
41000000000000000000000000116e65745f61646d696e6973747261746f7200000000000010000
00000000008000000000000029a
Service started.
```

El sistema está desarrollado como una componente CORBA, que permite la invocación de sus servicios a clientes CORBA. Para que un cliente se pueda comunicar con el sistema, necesita conocer la ubicación del mismo. En los entornos CORBA, la forma para localizar de manera unívoca un objeto es mediante una IOR.

Cuando el sistema se arranca de manera correcta, saca por salida estándar (como se ve en la traza anterior) y escribe en el fichero *ior.dat* (situado en el mismo directorio donde se arranca la componente) el valor de la IOR del sistema. Los clientes pueden emplear esa IOR para comunicarse con la componente *StateEngine*.

En caso de no conseguir arrancar de manera conveniente el sistema, revise todos los pasos mencionados en esta guía. Si aún así el problema persiste, revise los parámetros de configuración del fichero *configuration.properties* relativos a la conexión JDBC operacional.

## Parámetros de invocación de la componente

El *script* de arranque del sistema, se limita a lanzar la ejecución de la clase de arranque del *StateEngine* (*org.morfeo.corba.TIDStateEngine.Server*) en una JVM. Dicha clase puede recibir un gran número de parámetros de configuración que permiten personalizar el comportamiento del motor. Dichos argumentos de la componente serían:

- trace.level nivel
  - Establece el nivel de trazas para depuración.
- trace.file fichero
  - Fichero en que se escriben las trazas.
- trace.size tamaño
  - Especifica el tamaño de los ficheros.
- ior\_file fichero
  - Fichero donde se escribe la referencia IOR.
- properties\_file file
  - Ruta del fichero de configuración de la componente.
- trace.n\_fich
  - Número de ficheros de traza a emplear (trazas cíclicas).

Al mismo tiempo, dicha clase de arranque puede recibir parámetros de configuración del entorno CORBA en que se basa. En concreto, parámetros de inicialización del ORB. Algunos ejemplos serían:

- es.tid.TIDobj.iiop.orb\_port 3004  
Puerto del ORB.
- es.tid.TIDobj.trace.file fichero  
Fichero de trazas del ORB.
- es.tid.TIDobj.trace.level nivel  
Nivel de trazas del ORB.
- es.tid.TIDobj.max\_blocked\_time tiempo  
Tiempo que puede estar sin contestar una conexión IIOP.

Al ORB, en principio, se le puede pasar cualquier parámetro de configuración (veáse la especificación de *TIDorbJ*) a través de los recibidos por el método *main* de la clase *Server*. Sin embargo, los más relevantes serían los anteriores.

## Parada de StateEngine

Para detener *SMARTFlow-StateEngine*, se debe invocar un *script* con las variables de entorno debidamente configuradas, es decir, habiendo ejecutado el *script* de inicialización del entorno previamente. El comando sería el siguiente:

- Plataforma Windows (desde la *shell* msdos anterior):
 

```
> bin/TIDStateEngineShutdown.bat
```
- Plataforma UNIX/Linux (desde *shell* sh/bash anterior):
 

```
$ bin/TIDStateEngineShutdown.sh
```

Este script lanza la ejecución de la clase *org.morfeo.corba.TIDStateEngine.Murderer* y le pasa los argumentos que se le indiquen al *script*. La clase *Murderer* se comunica con la componente *StateEngine* y finaliza la ejecución de la misma.

## Guía rápida de uso

Para aclarar más la utilización de la componente, se ha desarrollado una guía rápida de forma de cualquiera pueda arrancar el sistema con un simple vistazo a la misma.

Dicho listado sería el siguiente para la plataforma Windows:

```
> setup_TIDStateEngine.bat
(... Configurar fichero configuration.properties ...)
(... Ejecutar scripts de SQL para manejar BBDD ...)
> bin/TIDStateEngineStart.bat (arranque de la componente)
> bin/TIDStateEngineShutdown.bat (finalización de la componente)
```

Para un entorno UNIX/Linux:

```
$ . setup_TIDStateEngine
(... Configurar fichero configuration.properties ...)
(... Ejecutar scripts de SQL para manejar BBDD ...)
$ bin/TIDStateEngineStart.sh (arranque de la componente)
$ bin/TIDStateEngineShutdown.sh (finalización de la componente)
```