HERRAMIENTA PARA TRATAR TAREAS EN UN GRID DINÁMICO

PROYECTO DE SISTEMAS INFORMÁTICOS
2002-2003
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

ALUMNOS DEL PROYECTO
Bahia Garbajo Horcajo
Antonio Martínez Alfaya
Jorge Merino Granizo

PROFESOR DIRECTOR DEL PROYECTO
Ignacio Martín Llorente
OTROS TUTORES DEL PROYECTO
Rubén Santiago Montero

ÍNDICE

Nº Pág

Bahia Carbajo Horcajo
Antonio Martínez Alfaya
Jorge Merino Granizo

CREACIÓN DE UNA HERRAMIENTA PARA TRATAR TAREAS EN UN GRID DINÁMICO

RESUMEN DEL PROYECTO (Castellano)

Dada la existencia de una herramienta desarrollada por el departamento de Arquitectura y Sistema Distribuidos de Seguridad, capaz de adaptar la ejecución de trabajos en un entorno Grid Dinámico, este proyecto consiste en crear una herramienta nueva para tratar esos trabajos de forma más fácil para el usuario, ya que la preparación de este tipo de tareas es muy tediosa y propensa a errores.

Nuestra herramienta permitirá, mediante una interfaz gráfica y la generación del código necesario, configurar y activar el Grid, definir las tareas, editarlas y prepararlas para su ejecución, lanzarlas, así como obtener y manejar todo tipo de información sobre las tareas y los hosts activos en el Grid.

RESUMEN DEL PROYECTO (Inglés)

Given the existence of a tool that has been developed by the department of architecture and distributed systems of security which is capable of adapting the executions of jobs I a dynamic grid environment. The main aim of this project is to create a new tool to make the jobs easier to handle for the users, due to the fact that the preparation of these types of tasks is very tedious and inclined to errors.

By means of a graphic interface and the generation of the necessary code, our tool will permit the configuration and activation of the grid, the definition, edition, preparation, and launching of the task for their execution, as well as obtaining and managing different types of information about the task and the active hosts in the grid.

DESCRIPCIÓN DEL PROYECTO

➢ PROFESOR DIRECTOR DEL PROYECTO.

Ignacio Martín Llorente.

➢ OTROS PROFESORES DEL PROYECTO.

Rubén Santiago Montero.

➢ ALUMNOS ASIGNADOS AL PROYECTO.

Bahia Carbajo Horcajo.
Antonio Martínez Alfaya.
Jorge Merino Granizo

➢ DESCRIPCIÓN GENERAL

Fruto de un proyecto de investigación y desarrollo llevado a cabo en el departamento, se ha creado una herramienta capaz de adaptar la ejecución de trabajos en un entorno Grid Dinámico usando la tecnología Globus (www.globus.org).

Actualmente la herramienta es capaz de ejecutar tareas complejas compuestas por trabajos, donde la salida de uno o más trabajos es la entrada de uno o más trabajos.

Sin embargo, la preparación de este tipo de tareas es muy tediosa y propensa a errores. Por tanto, se pretende simplificar el uso de esta herramienta desarrollando una herramienta que permita la definición de este tipo de tareas y que realice la preparación de las mismas de forma automática.

➢ OBJETIVOS

- Creación de un interfaz para crear, almacenar y editar tareas compuestas por trabajos.
- Generación del código necesario para realizar la preparación de la tarea.
- Creación de un interfaz para invocar la ejecución de la tarea usando la herramienta disponible en el centro.
- Creación de un interfaz de monitorización de la tarea y de los trabajos que la componen.

- Integración de la herramienta disponible en el departamento con el DataGrid.

➢ HERRAMIENTAS

- Graph Editing Framework (GEF): http://gef.tigris.org
- Java Development Kit (JDK): http://java.sun.com
- Apache HTTP Server: http://httpd.apache.org
- Axtensive Meta Language (XML): http://www.xml.org
- mySQL: http://www.mysql.org

REQUISITOS DEL SISTEMA

Los requisitos exigidos para el sistema son:

- Simplificación al usuario el lanzamiento de trabajos en el Grid.
- Creación de una interfaz para gestionar órdenes que ya existen.
- Monitorización del Grid.
- Entorno de uso (formularios) pequeño y manejable, pero eficaz.
- Robustez: ejecución de la herramienta en el mayor número de plataformas posible (sobre todo bajo Linux, Solaris…)
- La ubicación del diectorio GridWay y del editor que vamos a utilizar deben estar indicadas en un fichero en el home de usuario (*gridtoolkit.ini*).

INSTALACIÓN Y USO DE LAS HERRAMIENTAS NECESARIAS

Primeramente se instalaron las herramientas adecuadas para la creación del interfaz y la generación de código.

En un principio se intentó utilizar la herramienta de Java *"Borland JBuilder 4 Enterprise"*, comenzando con la realización de un prototipo del interfaz bajo el sistema operativo Windows. Sin embargo no fue posible ejecutar el prototipo bajo el sistema operativo Linux, que era lo que se pretendía.

Buscamos en internet una herramienta de libre distribución para implementar la interfaz en sistemas Solaris y familia Unix: *"Sun ONE Studio 4 CE"*

Se instaló entonces la *herramienta "Sun ONE Studio 4 CE",* de la que existen versiones para ambos sistemas operativos y por tanto permite que la aplicación creada sea portable.

Esta herramienta se instaló en las máquinas disponibles en el departamento de Arquitectura de Sistemas Distribuidos y Seguridad de la universidad (en principio bajo el sistema operativo Linux) y en nuestros pc personales (la versión para Windows), de modo que podíamos trabajar tanto en casa como en la universidad.

Más adelante, por problemas de velocidad en la máquina utilizada, se volvió a instalar el *"Sun ONE Studio 4 CE",* esta vez en una máquina cuyo sistema operativo era Solaris.

Dado que es un requisito del sistema la portabilidad del mismo (que se pueda ejecutar en el mayor número de plataformas) el prototipo se iba probando tanto en Linux como en SolarIs constantemente.

Según iba avanzando el prototipo, la interfaz se iba complicando y los problemas de velocidad ejecutándolo en *"Sun ONE Studio 4 CE"* aumentaban, así que lo siguiente que se hizo fue intentar ejecutar el sistema con el *"jdk 1.3"*. Esto provocó una vez más incompatibilidades entre las herramientas utilizadas, ya que el *"jdk 1.3*" no reconocía determinadas propiedades de los controles usados en *"Sun ONE Studio 4 CE"*.

Solucionados estos problemas, a partir de ese momento el prototipo se continuó elaborando con *"Sun ONE Studio 4 CE"* y probando con el *"jdk 1.3"*.

MANUAL DE LA APLICACIÓN

DESCRIPCIÓN GENERAL DE LA APLICACIÓN

➢ ENTORNO Y ARCHIVOS DE LA APLICACIÓN.

La herramienta consiste en una aplicación que permite al usuario la gestión completa de las tareas que se ejecutan en el Grid Dinámico.

La aplicación consta de 7 formularios:

- FORMULARIO PRINCIPAL
- EDITOR DE EXPERIMENTOS
- ENVÍO DE TRABAJOS
- MONITOR DE TRABAJOS
- MONITOR DE HOSTS
- CONFIGURACIÓN
- TRABAJOS ADJUNTOS

Cada uno de estos formularios está implementado en una clase y tiene asociado un formulario y un .java:

NOMBRE FORMULARIO	ARCHIVOS		
	CLASE	**FORMULARIO**	**.JAVA**
FORMULARIO PRINCIPAL	FramePpal.class	FramePpal.form	FramePpal.java
EDITOR DE EXPERIMENTOS	EditorExper.class	EditorExper.form	EditorExper.java
ENVÍO DE TRABAJOS	JobSubmission.class	JobSubmission.form	JobSubmission.java
MONITOR DE TRABAJOS	JobMonitoring.class	JobMonitoring.form	JobMonitoring.java
MONITOR DE HOSTS	HostMonitoring.class	HostMonitoring.form	HostMonitoring.java
CONFIGURACIÓN	Configuration.class	Configuration.form	Configuration.java
TRABAJOS ADJUNTOS	TrabajosAdjuntos.class	TrabajosAdjuntos.form	TrabajosAdjuntos.java

Además de estas clases también se utilizan las siguientes:

- **Gwps.class**: implementa la estructura de cada trabajo.

- **host.class**: implementa la estructura del host.
- **hilo.class**: implementa la interfaz del *thread* del demonio del Grid.
- **hiloTail.class**: implementa la interfaz del *thread* de la orden *tail.*

NOMBRE CLASE	ARCHIVOS	
	CLASE	**.JAVA**
Gwps	Gwps.class	Gwps.java
hilo	hilo.class	hilo.java
hiloTail	hiloTail.class	hiloTail.java
host	host.class	host.java

Otros ficheros utilizados son:

- **Ayuda.txt**: archivo de ayuda de la aplicación.
- **gridtoolkit.ini**: archivo de inicialización del Grid (contiene el directorio GridWay y el editor que vamos a usar).
- **gwd.conf**: archivo con información de la configuración del Grid.
- **miscript**: script utilizado en el Host Monitoring.

FUNCIONALIDAD DE LA APLICACIÓN

➢ FORMULARIO PRINCIPAL.

Este formulario contiene los 5 botones principales que manejarán la aplicación:

- Botón de acceso al Experiment TemplateEditor (Editor de Experimentos).
- Botón de acceso al Job Submission (Envío de Trabajos).
- Botón de acceso al Job Monitoring (Monitor de Trabajos).
- Botón de acceso al Host Monitoring (Monitor de Hosts).

- Botón de acceso a la Configuration(Configuración).

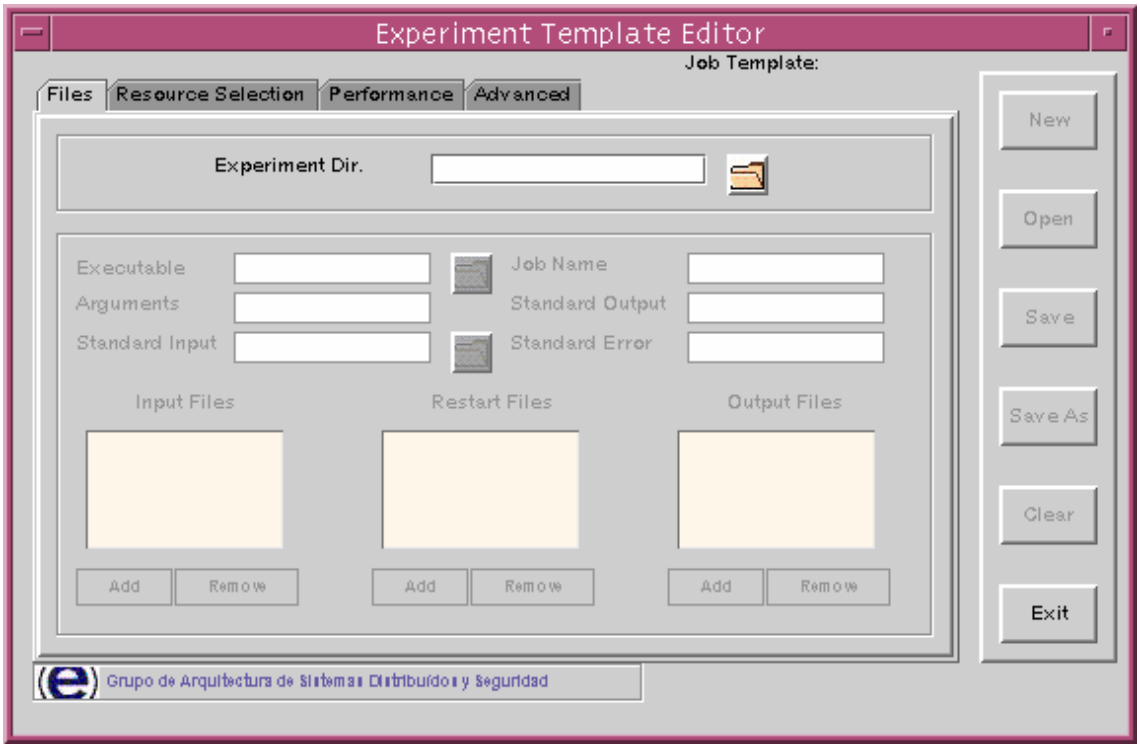
Además contiene oto botón para salir de la aplicación:

- Botón Exit.

Estos botones están accesibles desde el momento en que se ejecuta la aplicación por primera vez, ya que el usuario puede querer acceder a cualquiera de las opciones, ya que son independientes.

Las dependencias de datos entre los formularios se controlan con restricciones, una vez dentro de cada uno de ellos.

➢ EDITOR DE EXPERMENTOS.



El editor de experimentos sirve para crear y configurar un nuevo trabajo.

Este formulario consta de un panel con 4 pestañas que contienen toda la información del trabajo abierto:

- Pestaña de Files.
- Pestaña de Resource Selection.
- Pestaña de Performance.
- Pestaña de Advanced.

y de otro panel con 6 botones:

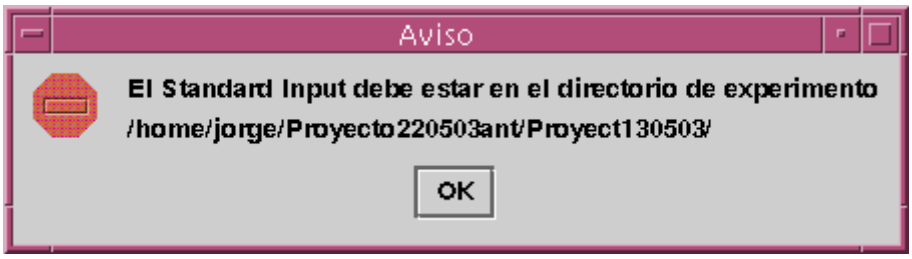
- Botón New.
- Botón Open.
- Botón Save.
- Botón Save As.
- Botón Clear.

- Botón Exit.

Además, en la parte superior del formulario se puede ver constantemente el nombre del trabajo (Job Template) con el que se está tratando en cada momento.

❖ PESTAÑA *"FILES"*

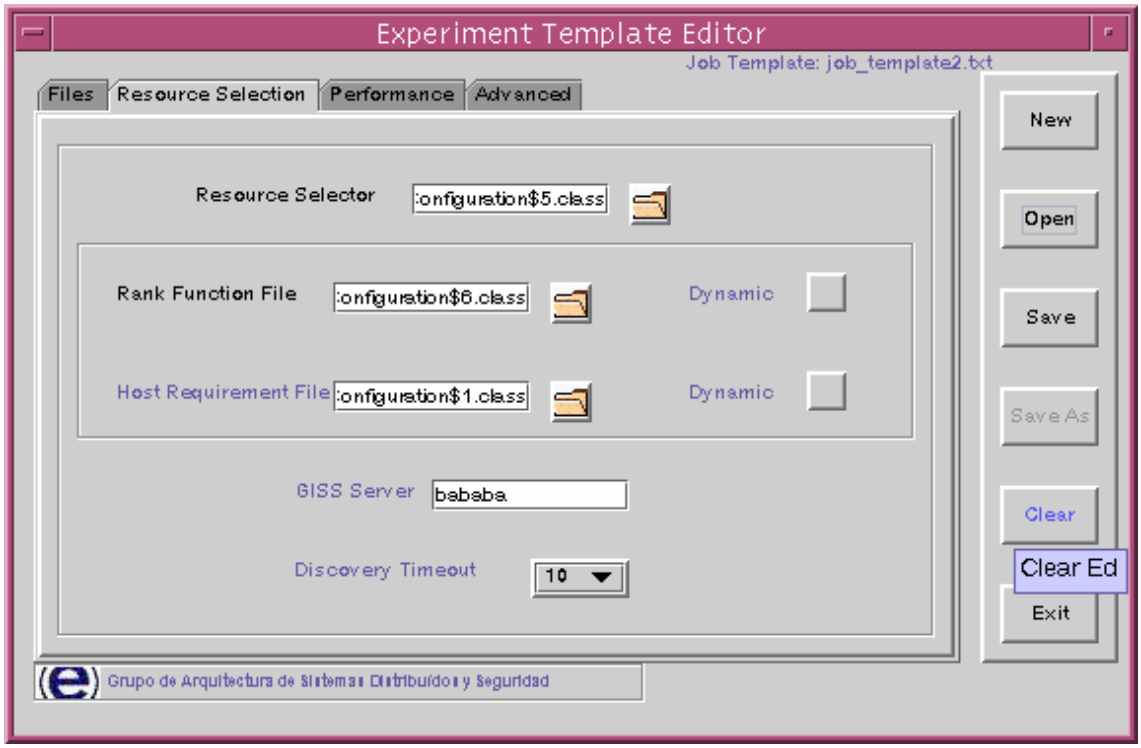
Si el campo es un archivo tiene que pertenecer al directorio de experimento. En caso contrario saltará un mensaje de error. Por ejemplo si el archivo *Standard Input* que he seleccionado con el browser no pertenece al directorio de experimento saltará el error:



Recoge los siguientes datos del trabajo:

- **Experiemnt Directory**: directorio de experimento. En él se tienen que encontrar todos los archivos relacionados con el trabajo que queramos usar, es decir, los archivos que contienen los campos de las pestañas.
- **Executable**: nombre del archivo ejecutable. Se selecciona mediante un browser.
- **Job Name**: nombre del trabajo. Se introduce manualmente.
- **Arguments**: argumentos del trabajo. Se introduce manualmente.
- **Standard Output**: salida estándar del trabajo. Se introduce manualmente.
- **Standard Input**: archivo de entrada estándar del trabajo. Se selecciona mediante un browser.
- **Standard Error**: margen de error del trabajo. Se introduce manualmente.
- **Input Files**: archivos de entrada del trabajo. Se seleccionan mediante un browser.
- **Output Files**: archivos de salida del trabajo. Se seleccionan mediante un browser.
- **Restart Files**: archivos de reinicio del trabajo. Se seleccionan mediante un browser.

❖ PESTAÑA *"RESOURCE SELECTION"*

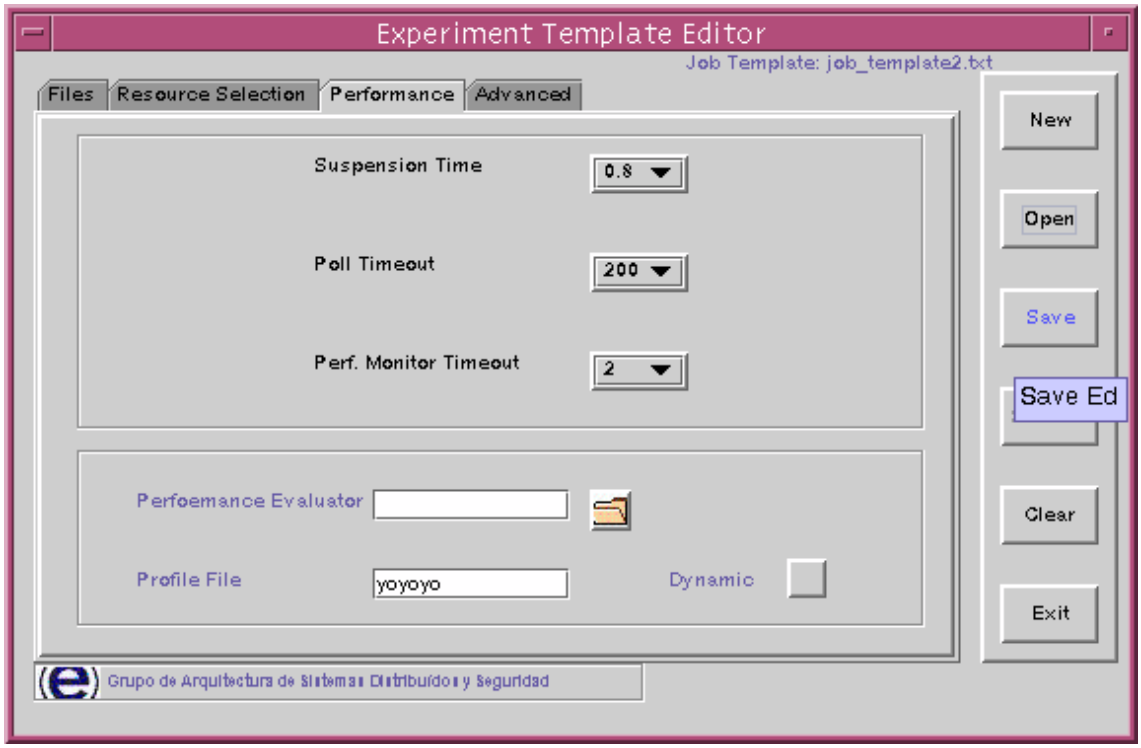


Si el campo es un archivo tiene que pertenecer al directorio de experimento.

Recoge los siguientes datos del trabajo:

- **Resource Selector**: archivo selector de la fuente. Se selecciona mediante un browser.
- **Rank Function File**: archivo de función de rank. Se selecciona mediante un browser. Puede ser dinámico (si está seleccionada la opción *Dynamic*).
- **Host Requirement File**: archivo de requisitos del host. Se selecciona mediante un browser. Puede ser dinámico (si está seleccionada la opción *Dynamic*).
- **GIIS Server**: servidor de giis. Se introduce manualmente.
- **Discovery Timeout**: tiempo restante de descubrimiento. Se introduce manualmente.

❖ **PESTAÑA *"PERFORMANCE"***

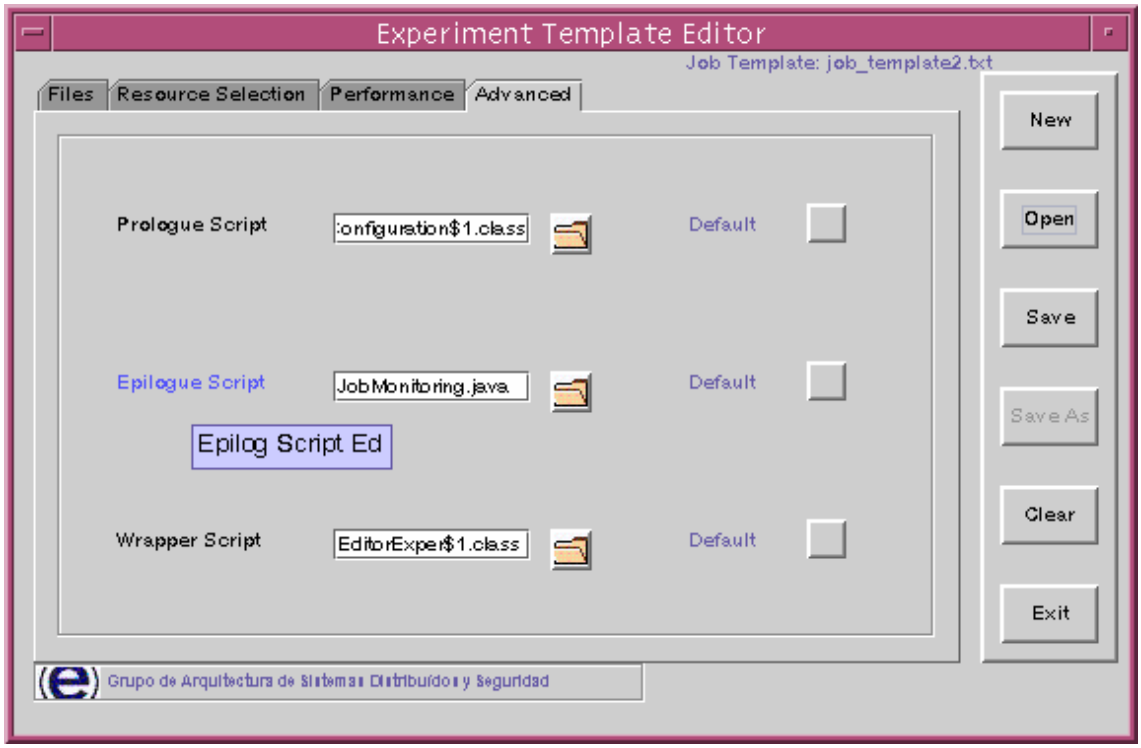


Si el campo es un archivo tiene que pertenecer al directorio de experimento.

Recoge los siguientes datos del trabajo:

- **Suspension Time**: tiempo de suspensión. Campo numérico. Se introduce manualmente.
- **PollTimeout**: tiempo de poll. Campo numérico. Se introduce manualmente.
- **Performance Monitor Timeout**: tiempo restante de monitor de performance. Campo numérico. Se introduce manualmente.
- **Performance Evaluator**: archivo de evaluador del performance. Se selecciona mediante un browser.
- **Profile File**: archivo de profile. Se introduce manualmente. Puede ser dinámico (si está seleccionada la opción *Dynamic*).

❖ **PESTAÑA *"ADVANCED"***



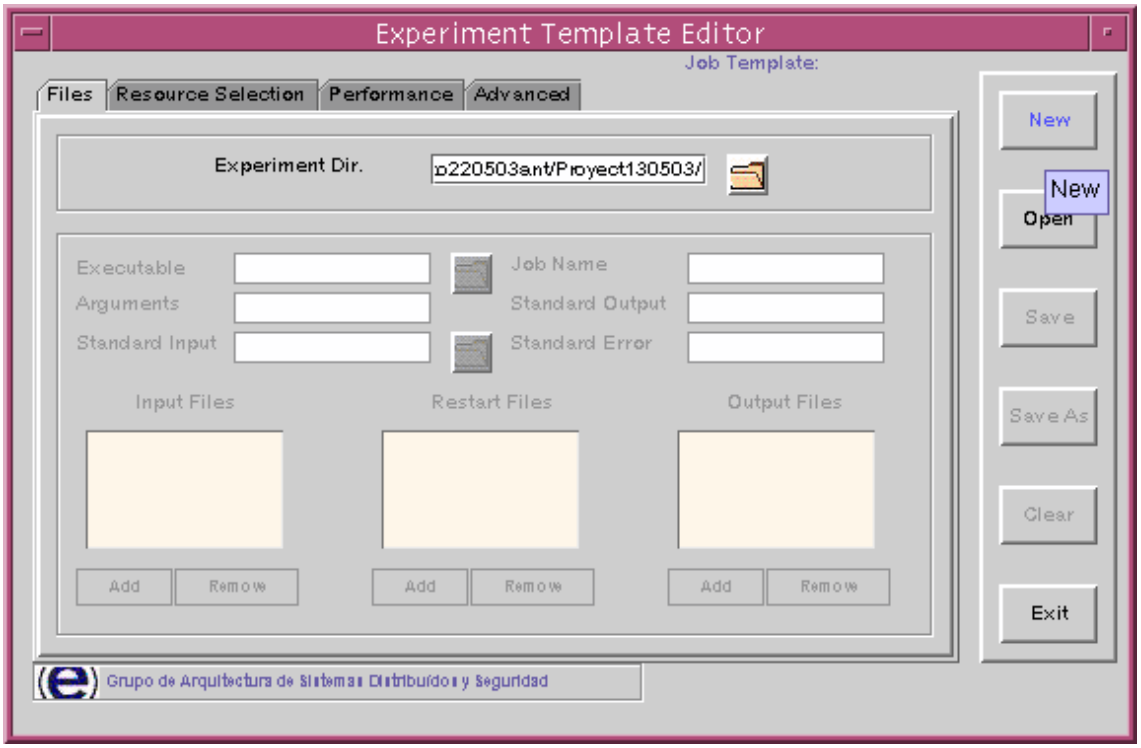
Si el campo es un archivo tiene que pertenecer al directorio de experimento.

Recoge los siguientes datos del trabajo:

- **Prologue Script**
- **Epilogue Script**
- **Wrapper Script**

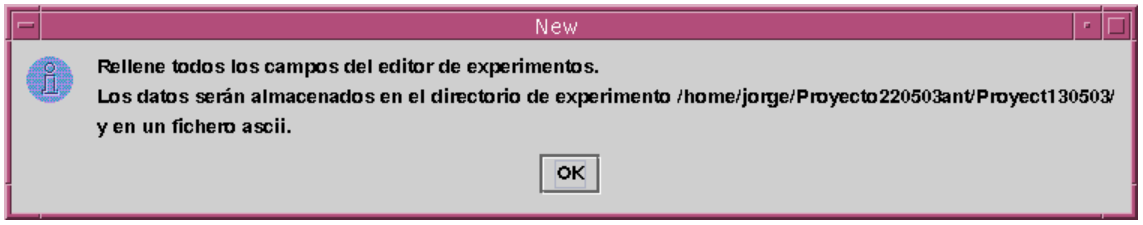
Se seleccionan mediante un browser. Si está seleccionada la opción *Default* no aparecen en el *"job_template.txt"* ya que en ese caso se cogen los valores por defecto del propio sistema.

❖ **BOTÓN *"NEW"***



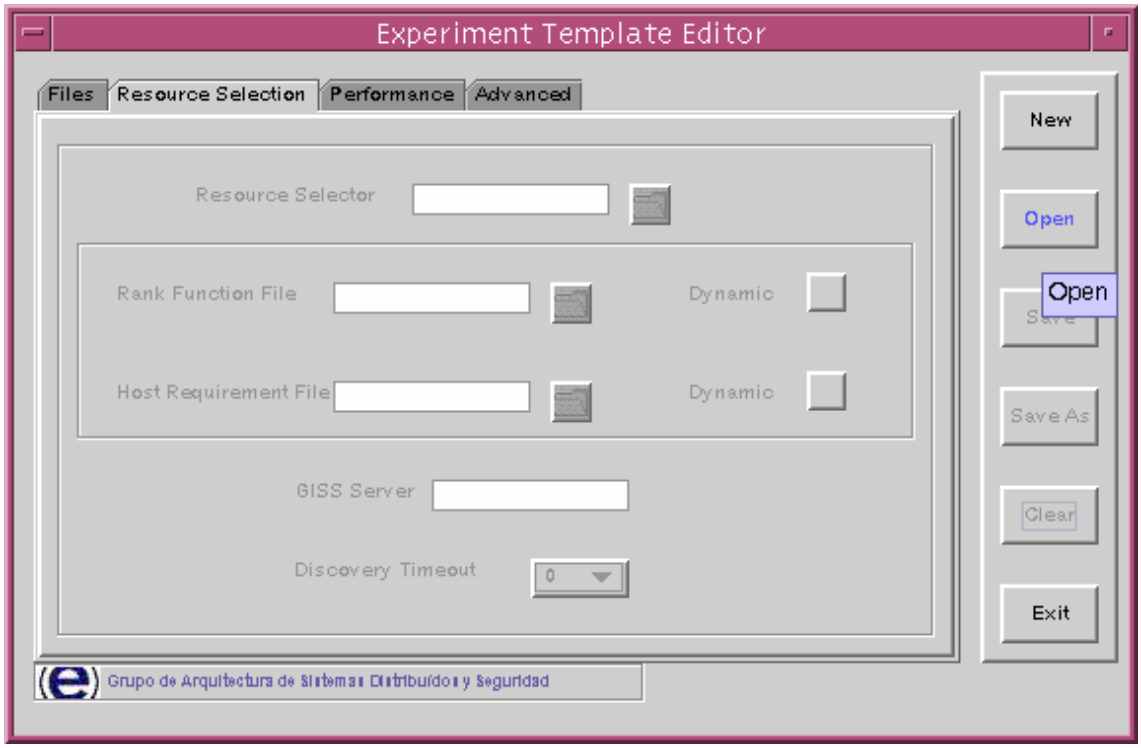
Crea un nuevo trabajo:

1. Se abre un mensaje de ayuda que nos indica que tenemos que rellenar los datos de los campos.



2. Tras rellenar los campos podemos guardar el nuevo trabajo creado (se tiene que guardar en el directorio de experimento), pulsando el botón *Save As*.
3. A partir de ese momento podemos modificar los campos y guardar las modificaciones con el botón *Save* (o con el botón Save As si se quiere guardar la información con otro nombre).

❖ **BOTÓN *"OPEN"***

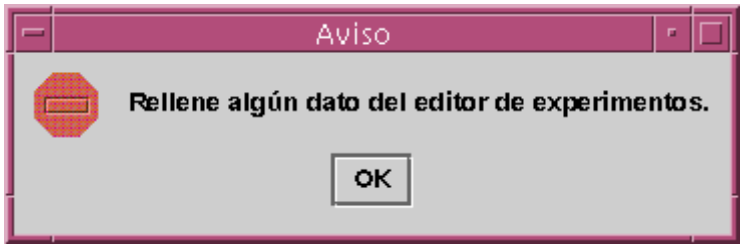


Abre un trabajo ya existente:

4. Se abre un browser en el que buscamos el trabajo (éste ha de estar en el directorio de experimento).
5. Al seleccionar el trabajo (archivo de texto) se lee el archivo seleccionado y se vuelcan todos sus datos a los campos de las pestañas del editor de experimentos.
6. A partir de ese momento podemos modificar los campos y guardar las modificaciones con el botón *Save* (o con el botón Save As si se quiere guardar la información con otro nombre).

❖ **BOTÓN *"SAVE"***

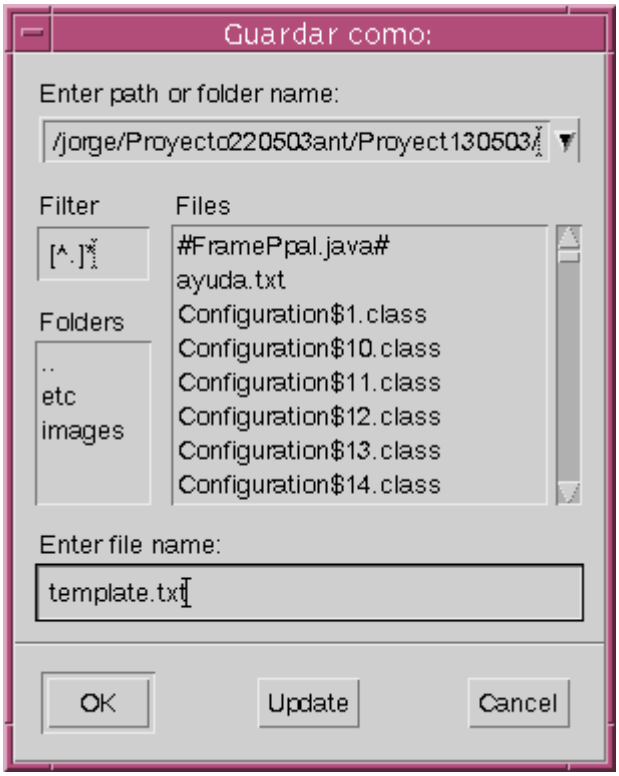
Guarda la información que haya en ese momento en los campos (no pueden estar vacíos), en el archivo del trabajo que estuviera abierto en ese momento.



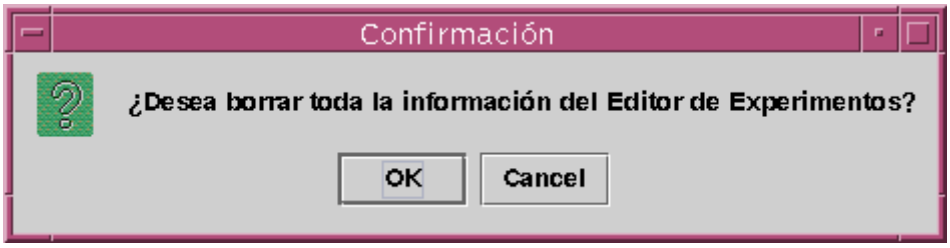
Para ello tiene que haber pulsado previamente *Open* para abrir un nuevo trabajo, o haber creado un nuevo trabajo y luego haberlo modificado.

- **BOTÓN *"SAVE AS"***

Guarda la información que se ha introducido en los campos al crear un trabajo nuevo con *New*. La información la guarda en el archivo del trabajo que le indiquemos en el browser que se activa.



- **BOTÓN *"CLEAR"***



Limpia los campos del editor de experimentos.

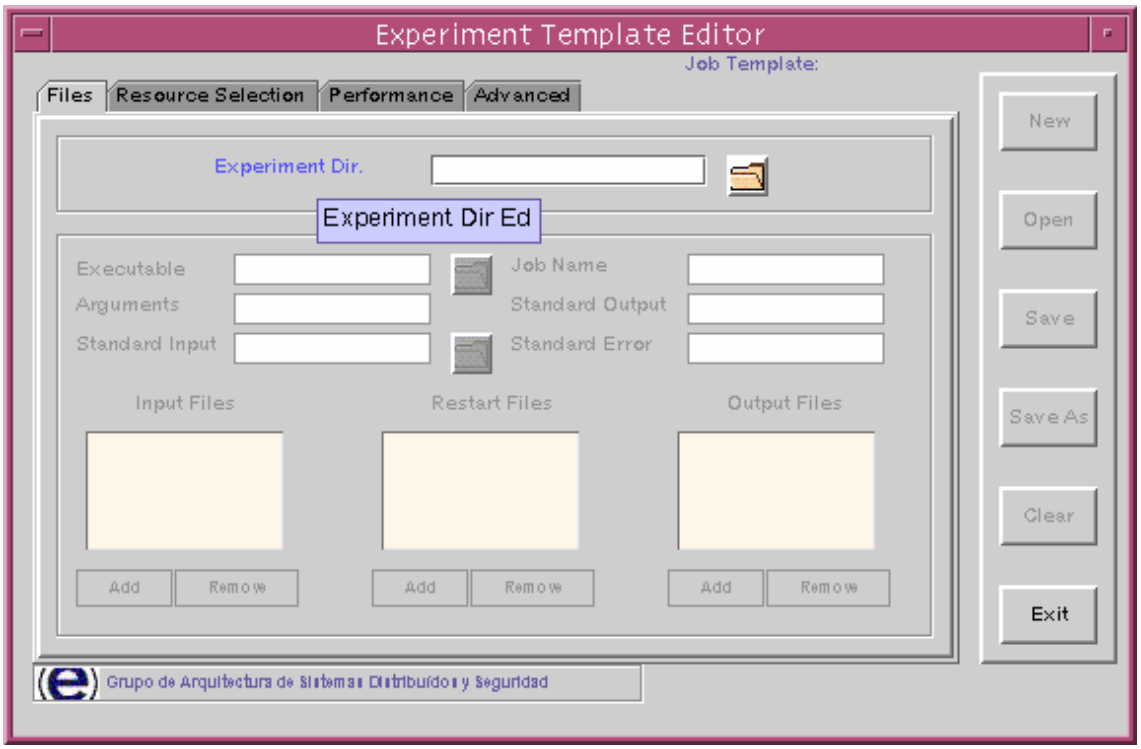
- **BOTÓN *"EXIT"***

Sale del editor de experimentos y vuelve al formulario principal.

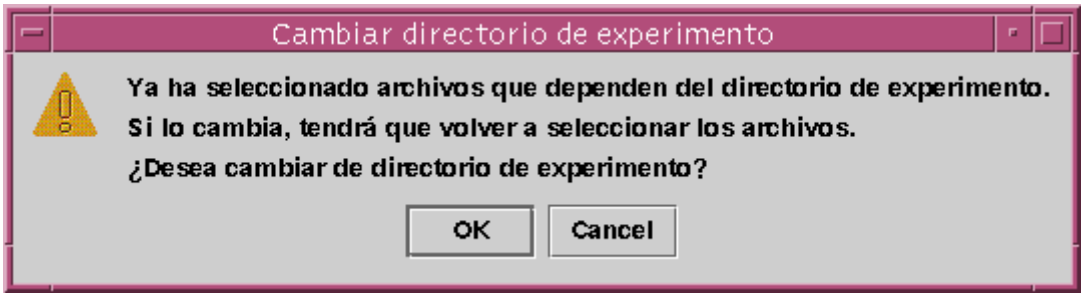
La información de los campos se mantendrá en los mismos, para la próxima vez que entremos en el editor. Esta información desaparecerá cuando cerremos la aplicación.

❖ ORDEN DE EJECUCIÓN PARA EL EDITOR DE EXPERIMENTOS

1. Inicialmente están desactivadas todas las opciones, excepto el browser para elegir un directorio de experimento.
2. Introducir el directorio de experimento, seleccionándolo con el browser. Si no está seleccionado el directorio no se nos permite realizar ninguna operación más.



3. Si en cualquier otro momento queremos modificar el directorio de experimento lo podemos hacer pulsando el browser, pero un mensaje nos avisará de que toda la información que haya en ese momento en los campos se borrará (ya que esa información debe pertenecer al directorio de experimento que elijamos).



4. Una vez seleccionado el directorio de experimento se activan las opciones Open y New.
5. Si queremos abrir un trabajo ya existente, pulsaremos *Open* y si modificamos los campos guardaremos los cambios con *Save* (o con *Save As* si queremos guardarlo con otro nombre).
6. Si queremos crear un trabajo nuevo, pulsaremos *New* y rellenaremos los campos, tras lo cuál pulsaremos *Save As* para guardar el nuevo trabajo (en el directorio de experimento).

   Si el campo a rellenar es un archivo tiene que pertenecer al directorio de experimento.
7. Para salir del Editor de Experimentos pulsaremos Exit. La información de los campos se mantendrá en los mismos, para la próxima vez que entremos en el editor. Esta información desaparecerá cuando cerremos la aplicación.

❖ EJEMPLO DE SALIDA DEL EDITOR DE EXPERIEMNTOS

Los archivos de trabajo que se crean (o se modifican) usando el editor de experimentos tienen el siguiente aspecto:

```
#-----------------------------------------------------------------------
---------
#                   Job Configuration file
# Syntax
#   VARIABLE = VALUE
#   '#' Comments
#-----------------------------------------------------------------------
----------

POLL_TIMEOUT           = 300
MAX_SUSPENSION_TIME    = 1
MAX_DISCOVERY_TIME     = 10
PERFORMANCE_CHECK_TIME = 0

# Executable Parameters

EXECUTABLE_FILE      =   sleep
EXECUTABLE_ARGUMENTS =   60
INPUT_FILES          =
OUTPUT_FILES         =

# Standard Files

STDIN_FILE  =   /dev/null
STDOUT_FILE =   stdout_file.${GW_JOB_ID}
STDERR_FILE =   stderr_file.${GW_JOB_ID}

# Scripts
# Absolute PATH here!!!!!

PROLOG  = /home/luis/tmp/GridWay/bin/ls_exp2/prolog.sh
EPILOG  = /home/luis/tmp/GridWay/bin/ls_exp2/epilog.sh
WRAPPER = /home/luis/tmp/GridWay/bin/ls_exp2/wrapper.sh
```

```
RESOURCE_SELECTOR = /home/luis/tmp/GridWay/bin/ls_exp2/rs.sh

# Migration related files

RESTART_FILES =
PROFILE_DFILE =

# Resource Selection Files

HOST_REQUIREMENTS_FILE  = requirements.ldif
RANK_FUNCTION_FILE      =

#  User defined variables

maxCpuTime = 45
maxTime    = 1000
maxMemory  = 5000
minMemory  = 50000

MDS_HOST = ursa.dacya.ucm.es
```

OBSERVACIONES

Podemos observar que toda la información que rellenamos en los campos del editor de experimentos se almacena en este archivo.

Podemos observar, por ejemplo, que en los archivos: PROLOG, EPILOG WRAPPER y RESOURCE_SELECTOR se almacena la ruta completa.

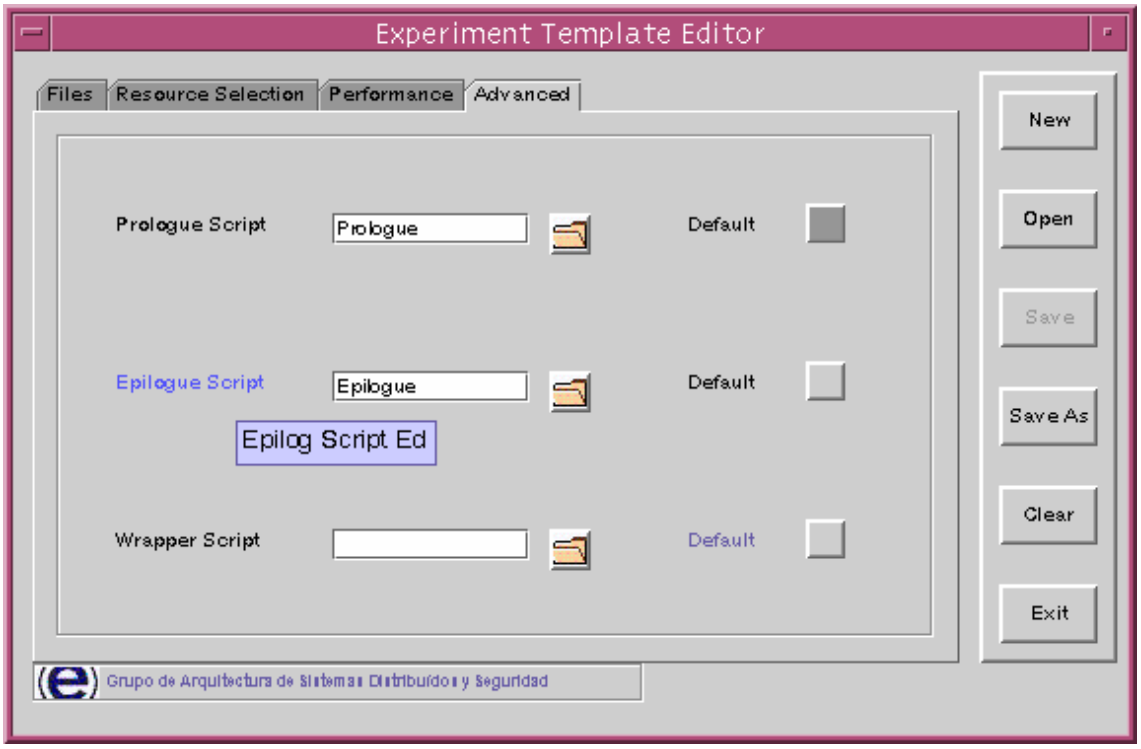
```
PROLOG  = /home/luis/tmp/GridWay/bin/ls_exp2/prolog.sh
EPILOG  = /home/luis/tmp/GridWay/bin/ls_exp2/epilog.sh
WRAPPER = /home/luis/tmp/GridWay/bin/ls_exp2/wrapper.sh
RESOURCE_SELECTOR = /home/luis/tmp/GridWay/bin/ls_exp2/rs.sh
```

Observamos también que la información definida en el epígrafe *"User defined variables"* no la hemos introducido nosotros en el editor de experimentos. Esto es debido a que esta información la introduce el usuario directamente en el archivo *"job_template.txt"*, por exigencias de la aplicación *DataGrid*.

```
#  User defined variables

maxCpuTime = 45
maxTime    = 1000
maxMemory  = 5000
minMemory  = 50000
```

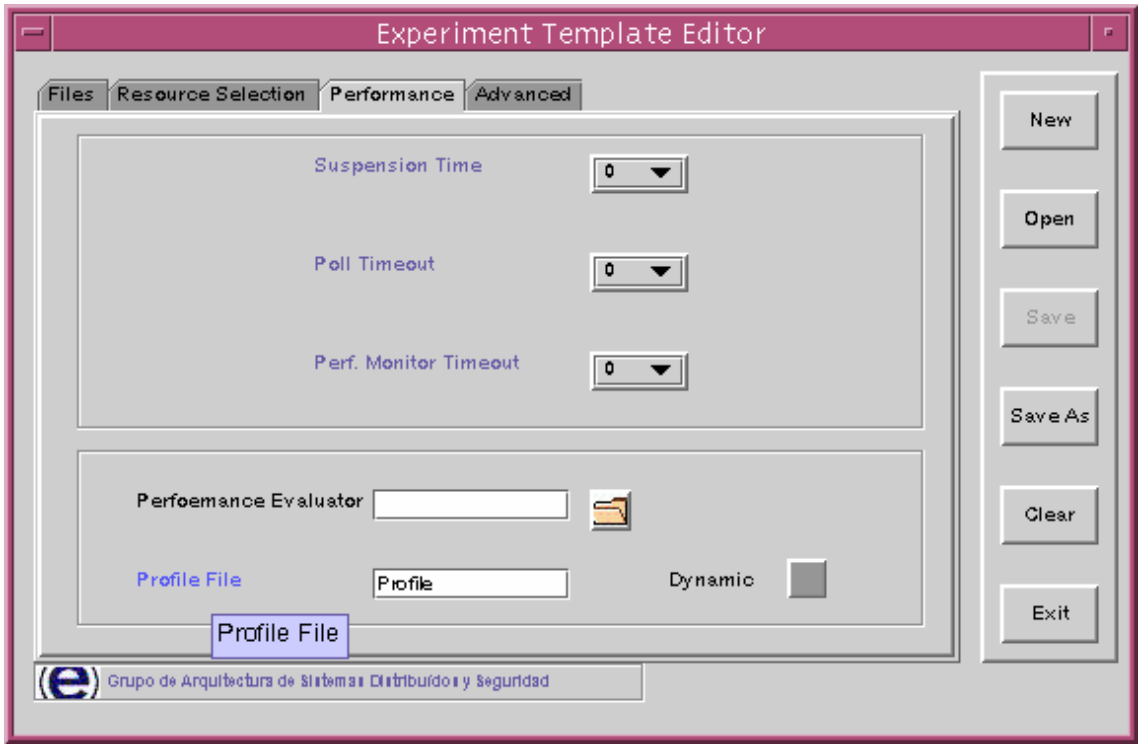
Otra observación que podemos hacer es la de la funcionalidad de las opciones *Default* y *Dynamic*.

- **Default**: si esta opción está seleccionada en el editor de experimentos:



en el *"job_template.txt"* no aparece la línea asociada a ese archivo.

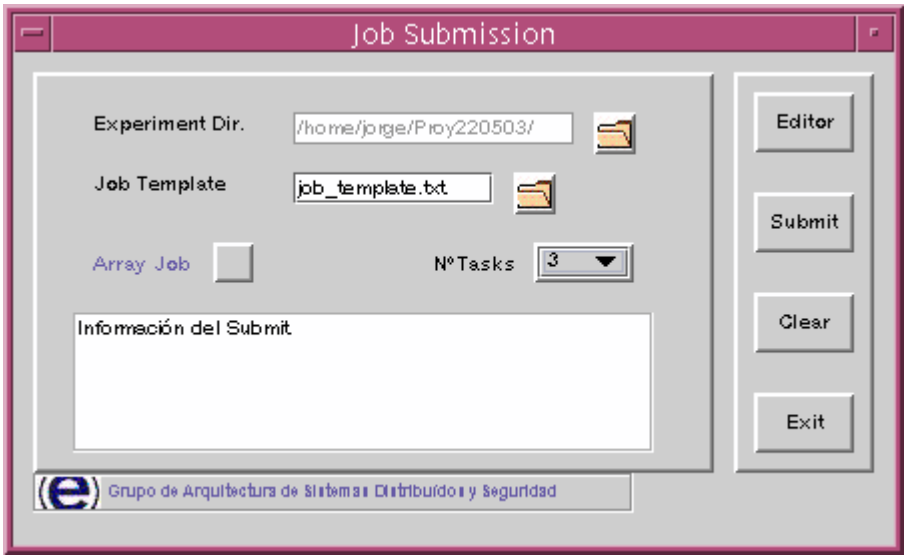
- **Dynamic**: si esta opción está seleccionada en el editor de experimentos:



en el *"job_template.txt"* aparece el nombre del archivo asociado con una D:

```
PROFILE_DFILE = Profile
```

➢ ENVÍO DE TRABAJOS.



Para poder entrar tenemos que seleccionar previamente el directorio GridWay en *Configuration*.

El formulario de envío de trabajos se utiliza para lanzar trabajos al Grid o para editarlos.

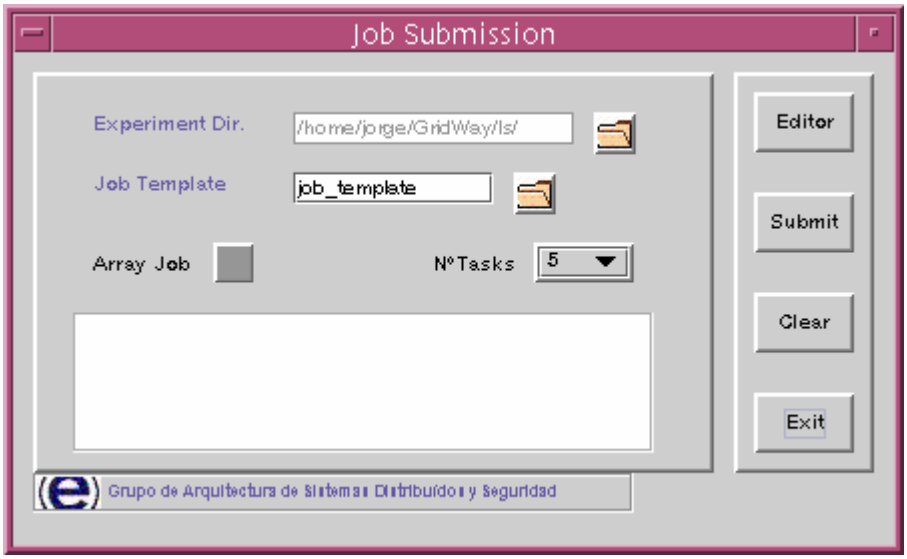
Este formulario consta de un panel con la información del trabajo, un área de texto con la información del envío y otro panel con 4 botones:

- Botón Editor.
- Botón Submit.
- Botón Clear.
- Botón Exit.

❖ INFORMACIÓN DE LOS TRABAJOS

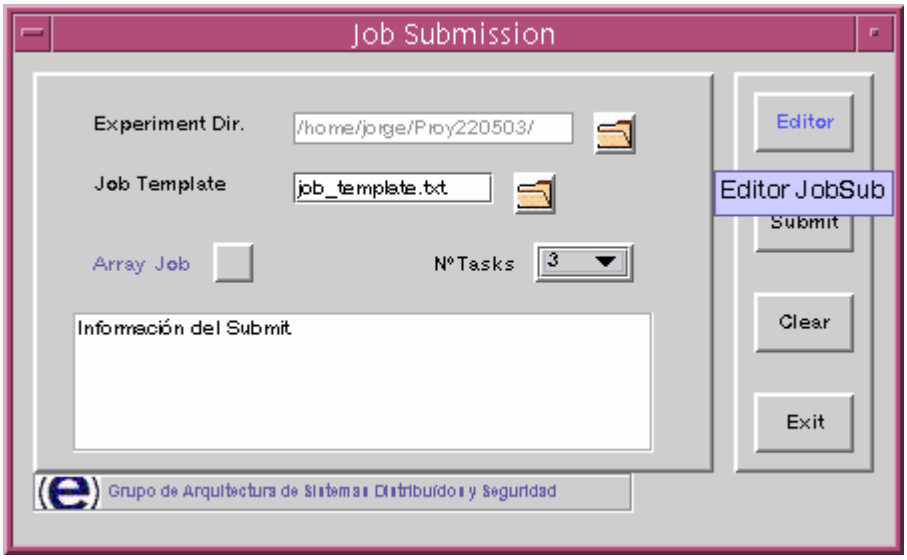
Recoge los siguientes datos del trabajo:

- **Experiment Directory**: directorio de experimento donde esté el trabajo. Se selecciona mediante un browser.
- **Job Template**: ruta del trabajo que queremos ver o lanzar. Tiene que estar en el directorio de experimento. Se selecciona mediante un browser.
- **Array Job:** si está pulsado es porque el trabajo pertenece a un array.



- **Nº Tasks:** El número de tareas de ese trabajo.

❖ BOTÓN *"EDITOR"*



Edita la plantilla del trabajo indicado en el *Job Template,* usando el editor seleccionado en el formulario *Configuration*. Si no se hubiera seleccionado ningún editor en *Configuration,* no nos hubiera permitido entrar en el formulario actual.

La plantilla del trabajo ha podido ser previamente creada con el *Experiment Editor.*

- **BOTÓN *"SUBMIT"***

Envía el trabajo indicado al Grid, ejecutando la orden:

gwsubmit –t <job_template>

Si el trabajo pertenece a un array, los campos *Array Job* y *Nº* tasks deben estar rellenos, y en este caso la orden sería:

gwsubmit –t <job_template> *-n* <nº tasks>

- **BOTÓN *"CLEAR"***

Limpia los campos del formulario.

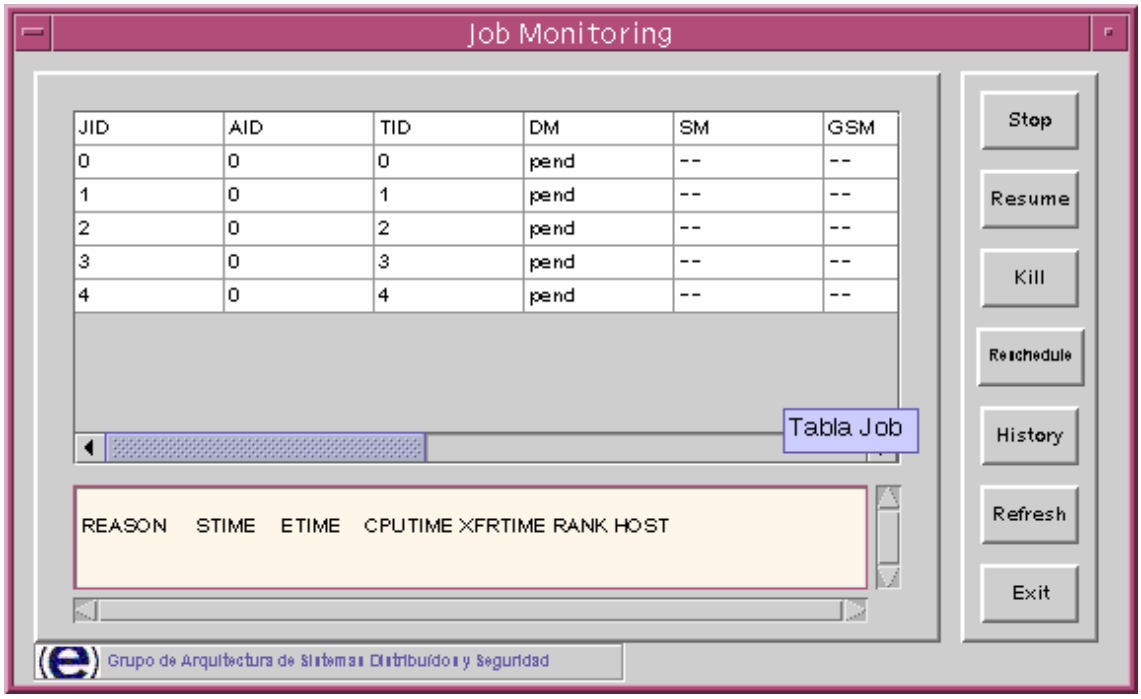
- **BOTÓN *"EXIT"***

Sale del envío de trabajos y vuelve al formulario principal.
La información de la tabla se mantendrá para la próxima vez que entremos en el formulario. Esta información desaparecerá cuando paremos el demonio (en *Envío de Trabajos*) o cerremos la aplicación.

- **ORDEN DE EJECUCIÓN PARA EL FORMULARIO DE ENVÍO DE TRABAJOS.**

1. Elegir el directorio de experimento.
2. Elegir el trabajo (tiene que pertenecer al directorio de experimento seleccionado) que queramos editar o ejecuta, seleccionando la opción *Array Job* y el nº de tareas si se trata de un array.
3. A partir de este momento podemos:
    - Editar el trabajo con el editor que previamente hayamos seleccionado en *Configuration,* pulsando *Editor.*
    - Lanzar el trabajo al Grid pulsando *Submit.*
    - Salir del formulario pulsando *Exit*.

➢ MONITOR DE TRABAJOS.



Para poder entrar tenemos que seleccionar previamente el directorio GridWay en *Configuration*.

Para ver información en la tabla tenemos que activar el demonio del Grid en *Envío de Trabajos.*

El formulario de monitor de trabajos se utiliza para visualizar información sobre los trabajos que están activos en el Grid.

Este formulario consta de una tabla con la información de los trabajos, un área de texto para mostrar la historia de un trabajo y un panel con 7 botones:

- Botón Stop.
- Botón Resume.
- Botón Kill.
- Botón Reschedule.
- Botón History.
- Botón Refresh.
- Botón Exit.

❖ TABLA (INFORMACIÓN DEL TRABAJO)

La tabla contiene información de los trabajos que están activos en el Grid (una fila por trabajo) tal como:

- **Job-ID**: identificador del trabajo.

- **Host**: identificador de su host.
- …

Para seleccionar una fila de la tabla sólo hay que pinchar en cualquiera de las columnas de esa fila.

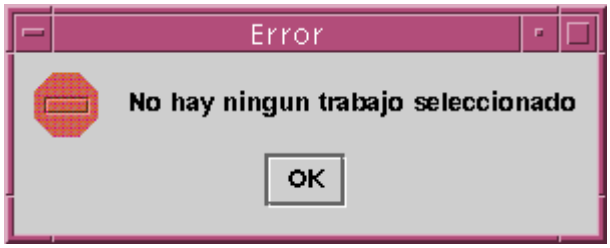
Al pinchar en una columna de la cabecera de la tabla se ordenan las filas de toda la tabla por el elemento seleccionado.

La tabla tiene el siguiente aspecto:

```
JID AID TID DM          SM          GSM         STIME ETIME CPUTIME
XFRTIME EXIT TEMPLATE         HOST
0   --  --  zombie      done        --          47:37 49:12 01:10
00:24   0    job_template     draco.dacya.ucm.es
1   --  --  zombie      done        --          46:37 48:13 01:11
00:25   0    job_template     draco.dacya.ucm.es
2   --  --  submitted   submitting active      50:27 --:-- 00:54
00:20   --   job_template     draco.dacya.ucm.es
3   --  --  submitted   submitting active      50:27 --:-- 01:03
00:11   --   job_template     solea.quim.ucm.es
4   0   0   submitted   submitting active      50:28 --:-- 00:56
00:17   --   job_template     ursa.dacya.ucm.es/jobmanager
5   0   1   submitted   submitting active      50:28 --:-- 00:52
00:21   --   job_template     draco.dacya.ucm.es
6   0   2   --          --          --          --:-- --:-- --:--   --:--
   --   job_template    --
7   0   3   --          --          --          --:-- --:-- --:--   --:--
   --   job_template    --
8   0   4   --          --          --          --:-- --:-- --:--   --:--
   --   job_template    --
9   1   0   submitted   submitting active      50:37 --:-- 00:47
00:17   --   job_template     solea.quim.ucm.es
10  1   1   submitted   submitting active      50:37 --:-- 00:48
00:16   --   job_template     ursa.dacya.ucm.es/jobmanager
11  1   2   --          --          --          --:-- --:-- --:--   --:--
   --   job_template    --
12  --  --  submitted   submitting active      50:37 --:-- 00:37
00:27   --   job_template     draco.dacya.ucm.es
```

❖ BOTÓN *"STOP"*

Hay que seleccionar previamente una fila de la tabla. Si no salta un error:



Ejecuta la orden:

gwkill –s <job-id>

que envía señales a los trabajos.

- **BOTÓN *"RESUME"***

Hay que seleccionar previamente una fila de la tabla.
Ejecuta la orden:

gwkill –r <job-id>

que envía señales a los trabajos.

- **BOTÓN *"KILL"***

Hay que seleccionar previamente una fila de la tabla.
Ejecuta la orden:

gwkill –a <job-id>

que mata el proceso seleccionado.

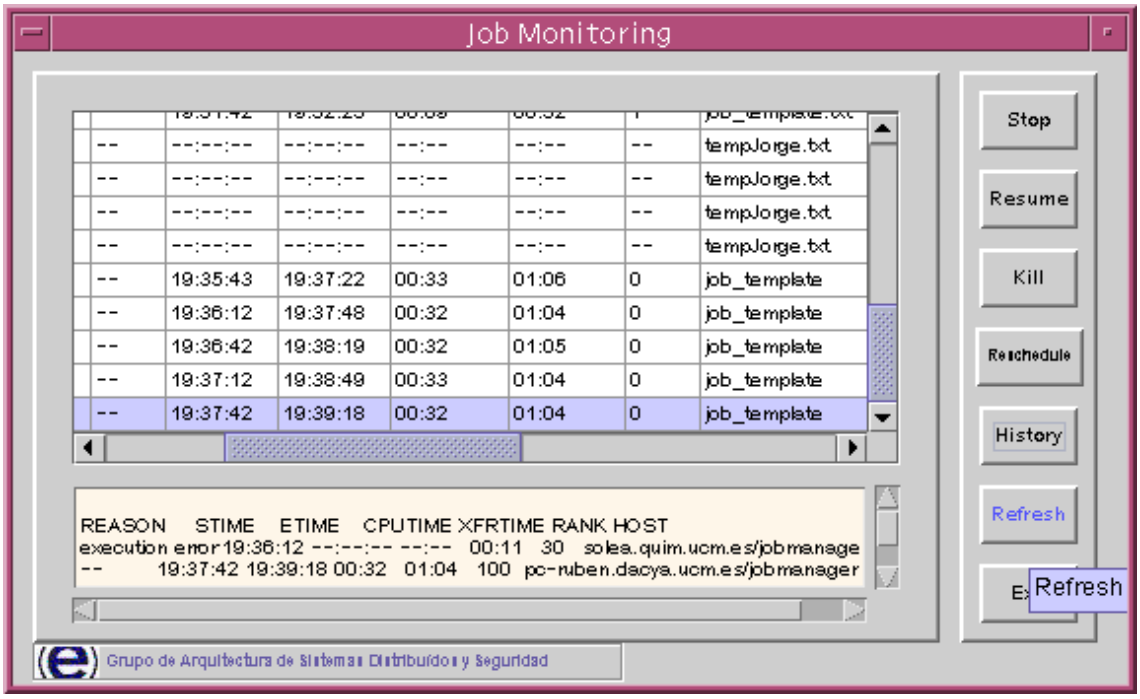
- **BOTÓN *"RESCHEDULE"***

Hay que seleccionar previamente una fila de la tabla.
Ejecuta la orden:

gwkill –m <job-id>

que envía señales a los trabajos.

- **BOTÓN *"HISTORY"***



Hay que seleccionar previamente una fila de la tabla.
Ejecuta la orden:

gwhistory <job-id>

que muestra la historia del trabajo seleccionado en el área de texto.
La historia tiene el siguiente aspecto:

```
REASON      STIME      ETIME      CPUTIME     XFRTIME     RANK HOST
--          47:37      --:--      --:--       00:03       50
draco.dacya.ucm.es
stop/resume 46:37       47:16       00:18       47:27       50
ursa.dacya.ucm.es/jobmanager
```

- **BOTÓN *"REFRESH"***

Actualiza el estado de los trabajos de la tabla.

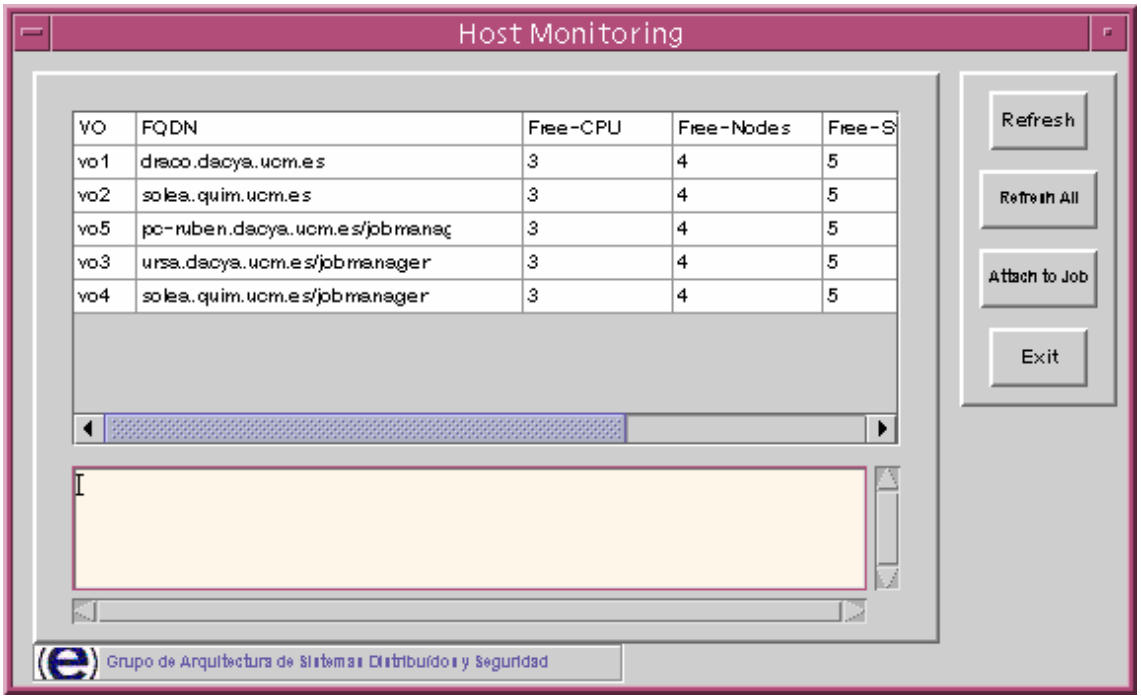
- **BOTÓN *"EXIT"***

Sale del monitor de trabajos y vuelve al formulario principal.
La información de la tabla se mantendrá para la próxima vez que entremos en el formulario. Esta información desaparecerá cuando paremos el demonio (en *Envío de Trabajos*) o cerremos la aplicación.

❖ **ORDEN DE EJECUCIÓN PARA EL FORMULARIO DE MONITOR DE TRABAJOS.**

1. Podemos pulsar los botones *Stop*, *Resume*, *Kill*, *Reschedule y History* seleccionando previamente una fila de la tabla.
2. Podemos pulsar los botones *Refresh* y *Exit.*

➢ MONITOR DE HOSTS.



Para poder entrar tenemos que seleccionar previamente el directorio GridWay en *Configuration*.

El formulario de monitor de hosts se utiliza para visualizar información sobre los hosts que están activos en el Grid.

Este formulario consta de una tabla con la información de los hosts, un área de texto para mostrar la evolución de un trabajo y un panel con 4 botones:

- Botón Refresh.
- Botón Refresh All.
- Botón Attach to job.
- Botón Exit.

❖ TABLA (INFORMACIÓN DEL HOST)

La tabla contiene información de los hosts que están activos en el Grid (una fila por host) tal como:

- **FQDN**: identificador del host.
- **Jobs(Total)**: número de trabajos asociados a ese host.
- …

Para seleccionar una fila de la tabla sólo hay que pinchar en cualquiera de las columnas de esa fila.

Al pinchar en una columna de la cabecera de la tabla se ordenan las filas de toda la tabla por el elemento seleccionado.

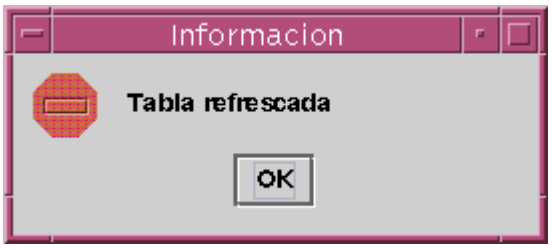
❖ BOTÓN *"REFRESH"*

Hay que seleccionar previamente una fila de la tabla.
Refresca un host (pendiente de realizar, al no disponer los alumnos de la orden necesaria).

Se simuló la orden, mediante un script propio "miscript" situado en el directorio */GridWay/bin/*, junto con las demás órdenes GridWay, que saca una serie de host activos en el Grid.

❖ BOTÓN *"REFRESH ALL"*

Actualiza el estado de todos los host de la tabla (y los muestra todos).



❖ BOTÓN *"ATTACH TO JOB"*

Hay que seleccionar previamente una fila de la tabla.

Recoge los identificadores de los trabajos asociados a ese host y muestra el formulario de *Trabajos Asociados.*

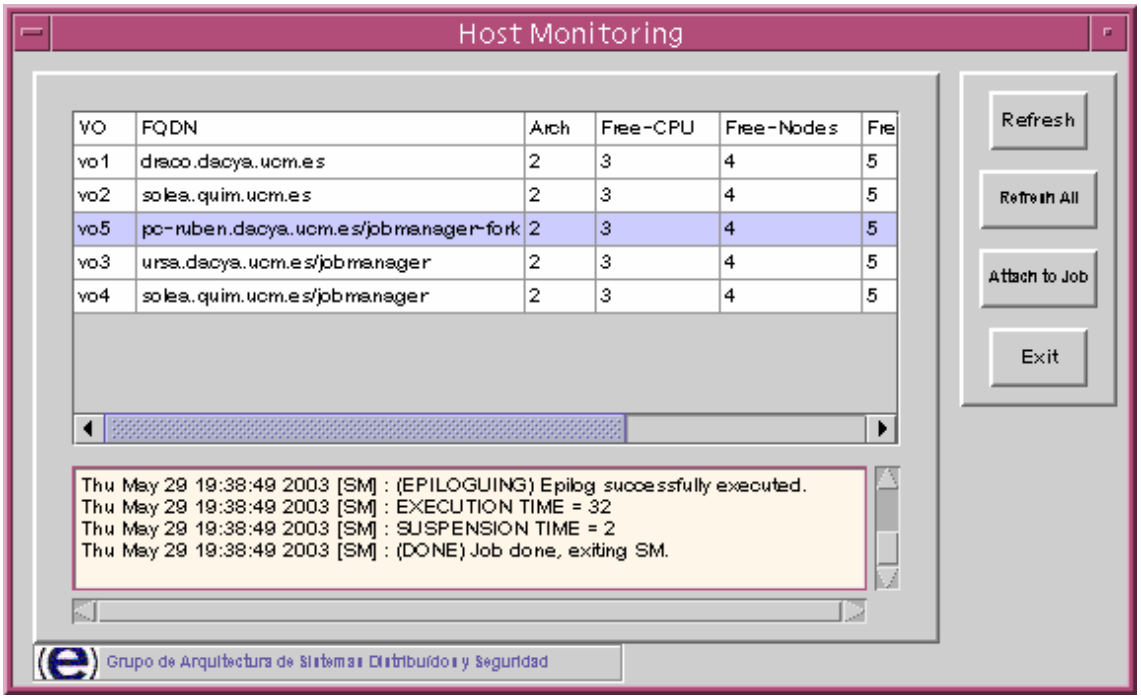
Aquí seleccionamos del combo un trabajo y al pulsar *Aceptar* se lanzará la orden *tail* del archivo *.log* guardado en el directorio del identificador del trabajo, que es el siguiente:

<idTrabajo>/<archivo.log>

y se cerrará el formulario de *Trabajos Asociados*.

Con esta orden se mostrará en el área de texto del formulario de M*onitor de Hosts* la información de ese *tail* es decir, la evolución del trabajo, que se irá refrescando constantemente (ya que está en un *thread* de forma concurrente).

Esta información dejará de refrescarse cuando ya se haya mostrado toda, aunque el demonio siga activo.



- **BOTÓN *"EXIT"***

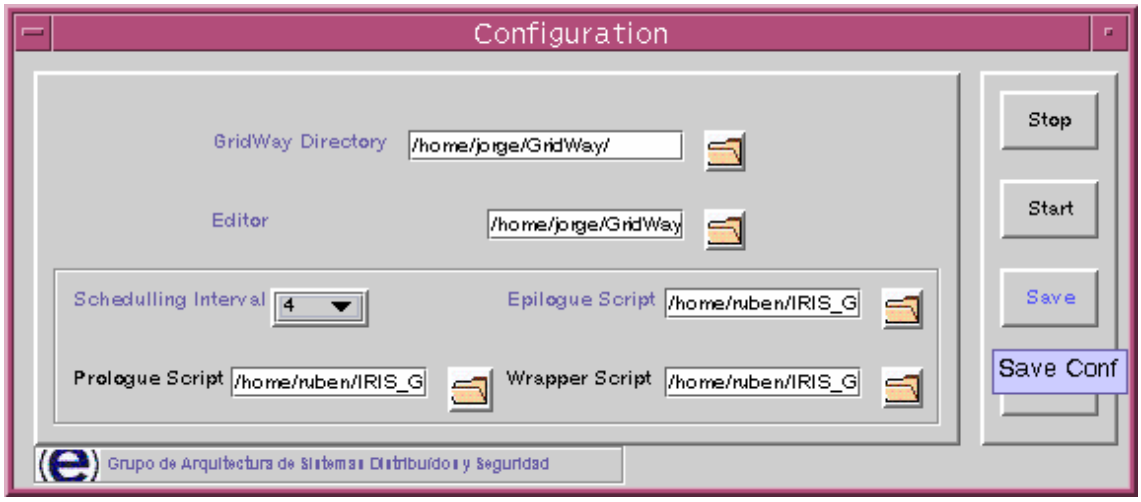
Sale del monitor de hosts y vuelve al formulario principal.

La información de los campos se mantendrá en los mismos, para la próxima vez que entremos en el formulario. Esta información desaparecerá cuando cerremos la aplicación.

- **ORDEN DE EJECUCIÓN PARA EL FORMULARIO DE MONITOR DE HOSTS.**

1. Podemos pulsar los botones *Refresh* y *Attach to job*, seleccionando previamente una fila de la tabla.
2. Podemos pulsar los botones *Refresh All* y *Exit.*

➢ CONFIGURACIÓN.



El formulario de la configuración, como su propio nombre indica, sirve para configurar la herramienta GridWay.

Este formulario consta de un panel con la información de la configuración y de otro panel con 4 botones:

- Botón Start.
- Botón Stop.
- Botón Save.
- Botón Exit.

❖ INFORMACIÓN DE LA CONFIGURACIÓN

Recoge los siguientes datos del trabajo:

- **GridWay Directory**: directorio del GridWay.
  Se selecciona mediante un browser.
  Contiene el archivo *gwd.conf*.
  Es necesario que esté definido para poder trabajar con los siguientes formularios:
  - Envío de trabajos
  - Monitor de trabajos
  - Monitor de Hosts

- **Editor**: ruta del ejecutable del editor de texto con el que vamos a querer trabajar (vi, Netscape,…). Este editor se usará cuando queramos editar una plantilla de un trabajo en

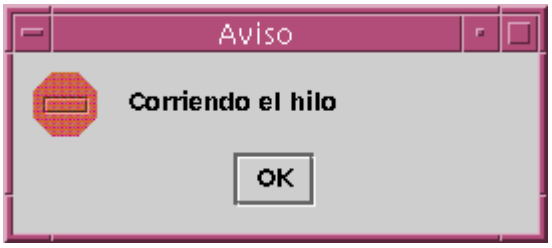
el fomulario *Job Submission.* Se selecciona mediante un browser.

- **Schedulling Interval**: intervalo de xxx. Campo numérico. Se introduce manualmente.
- **Prologue Script**: archivo de xxxx. Se selecciona mediante un browser.
- **Epilogue Script**: archivo de xxxx. Se selecciona mediante un browser.
- **Wrapper Script**: archivo de xxxx. Se selecciona mediante un browser.

❖ **BOTÓN *"START"***

Activa el demonio del GridWay. Este demonio está en un *thread*, cuya interfaz implementa la clase *hilo.* Para activar el demonio se usa el procedimiento *Start* de la clase *hilo,* que a su vez llama a *run,* ejecutando la orden:

gwd.



La orden muestra el siguiente aspecto por el terminal:

```
+------------------------------------------------------------------------
--+
| GridWay 1.0 (C) 2002-2003 ASDS/UCM & CAB
http://www.dacya.ucm.es/asds |
+------------------------------------------------------------------------
--+

DM: Setting up job-pool...done.
Activating globus modules...done.
Starting local file server...done, URL is
https://ursa.dacya.ucm.es:60095
DM: Dispatch manager waiting for events...
RM: Request Manager waiting for events...
DM: *************** Scheduling 0 jobs ****************
    ************************************************
DM: Allocating new job...
DM: Got new job with id 0
DM: Initializing structures for job 0...
DM: Job 0 successfully sumitted
DM: Allocating new job...
DM: Got new job with id 1
```

```
DM: Initializing structures for job 1...
DM: Job 1 successfully sumitted
DM: *************** Scheduling 2 jobs ***************
        Scheduling pending job 0
        Resource discovery:
                Host 0: ursa.dacya.ucm.es/jobmanager 50 1
        Resource selection:
        done. Submitting job to ursa.dacya.ucm.es/jobmanager
        Starting Submission Manager for job 0
        job 0 successfully scheduled

        Scheduling pending job 1
        Resource discovery:
                Host 0: draco.dacya.ucm.es 50 1
        Resource selection:
        done. Submitting job to draco.dacya.ucm.es
        Starting Submission Manager for job 1
        job 1 successfully scheduled

    ************************************************
DM: Stopping job 0...
DM: Cannot stop job 0, wrapper is not running
DM: *************** Scheduling 2 jobs ***************
    ************************************************
DM: Stopping job 0...
DM: *************** Scheduling 2 jobs ***************
    ************************************************
DM: *************** Scheduling 2 jobs ***************
    ************************************************
DM: job 0 done, exit status...STOPPED.
DM: *************** Scheduling 2 jobs ***************
        Re-Scheduling job 1
        Resource discovery:
                Host 0: solea.quim.ucm.es 50 1
done
        Resource selection:
done
        Evaluating migration:
         rejected!
    ************************************************
DM: *************** Scheduling 2 jobs ***************
        Re-Scheduling job 1
        Resource discovery:
                Host 0: ursa.dacya.ucm.es/jobmanager 50 1
done
        Resource selection:
done
        Evaluating migration:
         rejected!
    ************************************************
DM: Resuming job 0...
DM: *************** Scheduling 2 jobs ***************
        Scheduling pending job 0
        Resource discovery:
                Host 0: draco.dacya.ucm.es 50 1
        Resource selection:
        done. Submitting job to draco.dacya.ucm.es
```

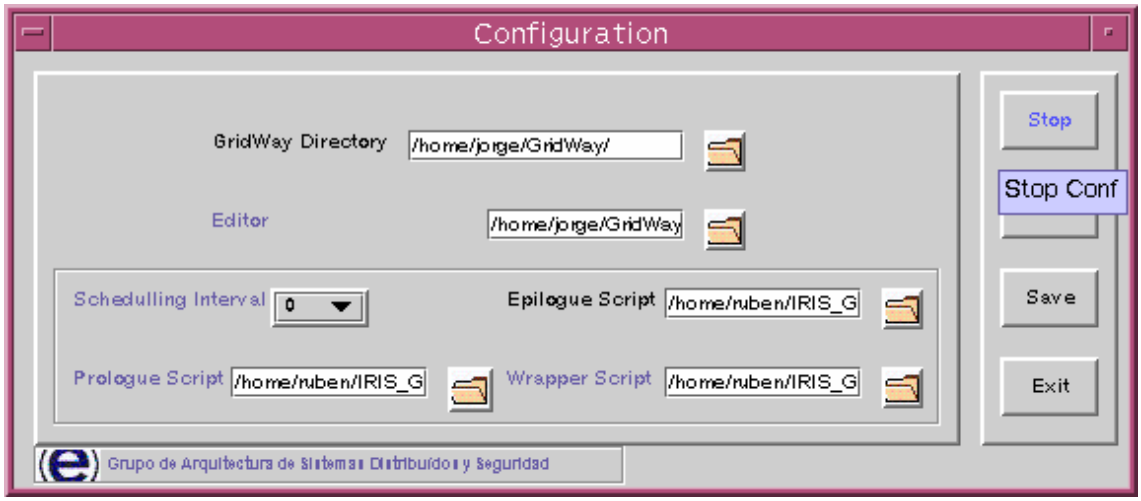
```
        Starting Submission Manager for job 0
        job 0 successfully scheduled

        Re-Scheduling job 1
        Resource discovery:
                Host 0: solea.quim.ucm.es 50 1
done
        Resource selection:
done
        Evaluating migration:
         rejected!
    ************************************************
DM: *************** Scheduling 2 jobs ***************
        Re-Scheduling job 1
        Resource discovery:
                Host 0: ursa.dacya.ucm.es/jobmanager 50 1
done
        Resource selection:
done
        Evaluating migration:
         rejected!
    ************************************************
DM: *************** Scheduling 2 jobs ***************
        Re-Scheduling job 0
        Resource discovery:
                Host 0: draco.dacya.ucm.es 50 1
done
        Resource selection:
Error!. No host selected
        Re-Scheduling job 1
        Resource discovery:
                Host 0: solea.quim.ucm.es 50 1
done
        Resource selection:
done
        Evaluating migration:
         rejected!
    ************************************************
DM: *************** Scheduling 2 jobs ***************
        Re-Scheduling job 0
        Resource discovery:
                Host 0: ursa.dacya.ucm.es/jobmanager 50 1
done
        Resource selection:
done
        Evaluating migration:
         rejected!
        Re-Scheduling job 1
        Resource discovery:
                Host 0: draco.dacya.ucm.es 50 1
done
        Resource selection:
Error!. No host selected
    ************************************************
DM: job 1 done, exit status...DONE.
DM: New state for job 1 is ZOMBIE
DM: *************** Scheduling 2 jobs ***************
```

```
        Re-Scheduling job 0
        Resource discovery:
                Host 0: solea.quim.ucm.es 50 1
done
        Resource selection:
done
        Evaluating migration:
         rejected!
    ************************************************
DM: *************** Scheduling 2 jobs ***************
        Re-Scheduling job 0
        Resource discovery:
                Host 0: ursa.dacya.ucm.es/jobmanager 50 1
done
        Resource selection:
done
        Evaluating migration:
         rejected!
    ************************************************
DM: *************** Scheduling 2 jobs ***************
        Re-Scheduling job 0
        Resource discovery:
                Host 0: draco.dacya.ucm.es 50 1
done
        Resource selection:
Error!. No host selected
    ************************************************
DM: *************** Scheduling 2 jobs ***************
        Re-Scheduling job 0
        Resource discovery:
                Host 0: solea.quim.ucm.es 50 1
done
        Resource selection:
done
        Evaluating migration:
         rejected!
    ************************************************
DM: *************** Scheduling 2 jobs ***************
        Re-Scheduling job 0
        Resource discovery:
                Host 0: ursa.dacya.ucm.es/jobmanager 50 1
done
        Resource selection:
done
        Evaluating migration:
         rejected!
    ************************************************
DM: *************** Scheduling 2 jobs ***************
    ************************************************
DM: job 0 done, exit status...DONE.
DM: New state for job 0 is ZOMBIE
DM: *************** Scheduling 2 jobs ***************
    ************************************************
DM: *************** Scheduling 2 jobs ***************
    ************************************************
DM: *************** Scheduling 2 jobs ***************
    ************************************************
DM: *************** Scheduling 2 jobs ***************
```

```
    ************************************************
DM: *************** Scheduling 2 jobs ****************
    ************************************************
DM: *************** Scheduling 2 jobs ****************
    ************************************************
DM: *************** Scheduling 2 jobs ****************
    ************************************************
DM: Allocating new job...
DM: Got new job with id 2
DM: Initializing structures for job 2...
DM: Job 2 successfully sumitted
DM: Allocating new job...
DM: Got new job with id 3
DM: Initializing structures for job 3...
DM: Job 3 successfully sumitted
DM: Allocating new array...
DM: Got new array with id 0
DM: Allocated job 4, as task 0 of array 0
DM: Initializing job 4, of array 0
DM: Allocated job 5, as task 1 of array 0
DM: Initializing job 5, of array 0
DM: Allocated job 6, as task 2 of array 0
DM: Initializing job 6, of array 0
DM: Allocated job 7, as task 3 of array 0
DM: Initializing job 7, of array 0
DM: Allocated job 8, as task 4 of array 0
DM: Initializing job 8, of array 0
DM: *************** Scheduling 9 jobs ****************
DM: Schedulling 2 of 5 tasks of array 0
        Scheduling pending job 2
        Resource discovery:
                Host 0: draco.dacya.ucm.es 50 1
        Resource selection:
        done. Submitting job to draco.dacya.ucm.es
        Starting Submission Manager for job 2
        job 2 successfully scheduled

        Scheduling pending job 3
        Resource discovery:
                Host 0: solea.quim.ucm.es 50 1
        Resource selection:
        done. Submitting job to solea.quim.ucm.es
        Starting Submission Manager for job 3
        job 3 successfully scheduled

        Scheduling pending job 4
        Resource discovery:
                Host 0: ursa.dacya.ucm.es/jobmanager 50 1
        Resource selection:
        done. Submitting job to ursa.dacya.ucm.es/jobmanager
        Starting Submission Manager for job 4
        job 4 successfully scheduled

        Scheduling pending job 5
        Resource discovery:
                Host 0: draco.dacya.ucm.es 50 1
        Resource selection:
```

```
        done. Submitting job to draco.dacya.ucm.es
        Starting Submission Manager for job 5
        job 5 successfully scheduled

    ************************************************
DM: Allocating new array...
DM: Got new array with id 1
DM: Allocated job 9, as task 0 of array 1
DM: Initializing job 9, of array 1
DM: Allocated job 10, as task 1 of array 1
DM: Initializing job 10, of array 1
DM: Allocated job 11, as task 2 of array 1
DM: Initializing job 11, of array 1
DM: Allocating new job...
DM: Got new job with id 12
DM: Initializing structures for job 12...
DM: Job 12 successfully sumitted
DM: *************** Scheduling 13 jobs ***************
DM: Schedulling 2 of 3 tasks of array 1
        Scheduling pending job 9
        Resource discovery:
                Host 0: solea.quim.ucm.es 50 1
        Resource selection:
        done. Submitting job to solea.quim.ucm.es
        Starting Submission Manager for job 9
        job 9 successfully scheduled

        Scheduling pending job 10
        Resource discovery:
                Host 0: ursa.dacya.ucm.es/jobmanager 50 1
        Resource selection:
        done. Submitting job to ursa.dacya.ucm.es/jobmanager
        Starting Submission Manager for job 10
        job 10 successfully scheduled

        Scheduling pending job 12
        Resource discovery:
                Host 0: draco.dacya.ucm.es 50 1
        Resource selection:
        done. Submitting job to draco.dacya.ucm.es
        Starting Submission Manager for job 12
        job 12 successfully scheduled

    ************************************************
DM: *************** Scheduling 13 jobs ***************
    ************************************************
```

❖ **BOTÓN *"STOP"***



Detiene el demonio del GridWay lanzando la orden:

pkill -9 gwd

❖ **BOTÓN *"SAVE"***

Guarda la información que haya en ese momento en los campos del formulario *Configuration* en los ficheros *gridtoolkit.ini* y *gwd.conf*, modificándolos.

❖ **BOTÓN *"EXIT"***

Sale de la configuración y vuelve al formulario principal.
La información de los campos se mantendrá en los mismos, para la próxima vez que entremos en el formulario. Esta información desaparecerá cuando cerremos la aplicación.

❖ **ORDEN DE EJECUCIÓN PARA EL FORMULARIO DE CONFIGURACIÓN.**

1. Si es la primera vez que se ejecuta la aplicación tenemos que:

   - Seleccionar el directorio GridWay.
   - Seleccionar el editor con el que queramos trabajar (si no hacemos esto último, no se nos permitirá entrar en *Job Submission*).
   - Rellenar el resto de los datos de *Configuration.*
   - Salvarlo (botón *Save*).
   - Se modificará el archivo *gwd.conf* y se rellenará el archivo *gridtoolkit.ini.*

2. Si no es la primera vez que se ejecuta la aplicación tendremos en los campos de *Cofiguration* la información leída de los archivos *gridtoolkit.ini y gwd.conf.*
3. A partir de este momento podemos:
    - Modificar estos datos cuando queramos y salvarlos (se modificarán los archivos *gwd.conf y gridtoolkit.ini*)
    - Lanzar el demonio del GridWay con el botón *Start.*
    - Parar el demonio del GridWay con el botón *Stop.*

❖ EJEMPLO DE FICHEROS UTILIZADOS POR EL FORMULARIO DE CONFIGURACIÓN.

Este formulario lee los archivos:

- ***gridtoolkit.ini***

```
#Archivo de configuracion del GridToolkit

GRIDWAY_DIRECTORY=/home/jorge/GridWay/
EDITOR=/usr/ld/bin/netscape/
```

Este fichero se puede modificar también manualmente, pero manteniendo el formato y sin modificar los identificadores:

<identificador> = <valor>

- ***gwd.conf***

```
# Scheduler Configuration Parameters

# Number of seconds between two schedules.
# Max number of jobs (not tasks) scheduled in each scheduling step

SCHEDULING_INTERVAL       = 30
MAX_NUMBER_OF_SCHEDULINGS = 5

# Interval to check job performance
PERFORMANCE_INTERVAL      = 5

# Maximum number of jobs, arrays and resources

MAX_NUMBER_OF_JOBS        = 200
MAX_NUMBER_OF_ARRAYS      = 10

MAX_NUMBER_OF_RESOURCES   = 100

# Uncommente this line to set a different log file
# Default is $GRIDWAY_HOME/var/gwd.log
```

```
# GWD_LOG=/home/luis/gwd.log

#Default script location
PDE=
RS=
PROLOGUE=/home/ruben/IRIS_GRID/GridWay/scripts/prolog.sh
EPILOGUE=/home/ruben/IRIS_GRID/GridWay/scripts/epilog.sh
WRAPPER=/home/ruben/IRIS_GRID/GridWay/scripts/wrapper.sh
```

Fichero de configuración del GridWay. Contiene información sobre la ejecución de trabajos en el Grid.

➢ TRABAJOS ADJUNTOS.

El formulario de trabajos adjuntos lo utiliza el formulario *Monitor de Hosts* para visualizar el *tail* de uno de los trabajos de un host.

Este formulario consta de un combo y 2 botones:

- Botón Aceptar.
- Botón Cancelar.

❖ COMBO DE TRABAJOS

Contiene los identificadores de los trabajos del host seleccionado en *Monitor de Hosts*

❖ BOTÓN *"ACEPTAR"*

Lanzará la orden *tail* del archivo *.log* guardado en el directorio del identificador del trabajo, que es el siguiente:

<idTrabajo>/<archivo.log>

y se cerrará el formulario de *Trabajos Asociados*.

Con esta orden se mostrará en el área de texto del formulario de M*onitor de Hosts* la información de ese *tail* es decir, la evolución del trabajo, que se irá refrescando constantemente (ya que está en un *thread* de forma concurrente).

Esta información dejará de refrescarse cuando ya se haya mostrado toda, aunque el demonio siga activo.

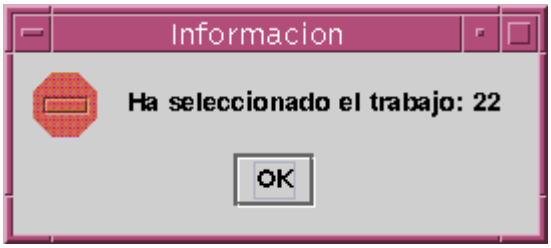
❖ **BOTÓN *"CANCELAR"***

Se cerrará el formulario de *Trabajos Asociados*.

❖ **ORDEN DE EJECUCIÓN PARA EL FORMULARIO DE TRABAJOS ADJUNTOS.**

1. Seleccionar un identificador de trabajo del combo.

2. Pulsar *Aceptar* (para mostrar su *tail*) o *Cancelar* (para no hacer nada).



PUESTA EN FUNCIONAMIENTO DE LA APLICACIÓN

➢ REQUISITOS PARA LA PUESTA EN FUNCIONAMIENTO.

Para ejecutar la aplicación en cualquier momento se necesitan los siguientes requisitos:

1. En la máquina en la que vayamos a ejecutar la aplicación tiene que estar instalado el *"jdk 1.3"*.
2. Además, por supuesto, debe estar instalada su herramienta Grid.
3. En el directorio donde vayamos a ejecutar nuestra aplicación (en el home de usuario) tienen que estar los archivos:

gridtoolkit.ini: archivo de inicialización necesario para la configuración. La primera vez que se ejecuta la aplicación no contiene nada. Luego contendrá la ruta del directorio Gridway y la del editor que vamos a usar.

Ayuda.txt: archivo que contiene la ayuda de la aplicación.

Archivos ".java" de todas las clases que vamos a usar.

Archivos ".class" de todas las clases que vamos a usar. Si no están creados se crean compilando la aplicación con la orden:

*javac *.java*

Archivos auxiliares de lectura y volcado de datos para la ejecución de los procesos: al ejecutar un proceso se suele volcar la información a un archivo y luego se trata ese archivo, aunque a veces se coge la información directamente de la salida estándar y se trata línea a línea.
Se trata de los archivos *salhost.dat* y *salgwps.dat*.

4. Debe estar creado el directorio GridWay para la configuración del Grid.
5. En el directorio */GridWay/bin/* debe estar el archivo miscript y las órdenes con las que vamos a trabajar:

gwd: activa el demonio.
gwhistory: saca la historia de un trabajo.
gwkill: mata trabajos, entre otras funciones.
gwps: muestra los trabajos que están activos en el Grid.
gwsubmit: lanza un trabajo (template) al Grid.
tail: recoge las últimas líneas del archivo *.log* de un trabajo. Se actualiza constantemente.

➢ PASOS PARA LA PUESTA EN FUNCIONAMIENTO.

Para poner en funcionamiento la aplicación en cualquier momento hay que seguir los pasos:

1. Habilitarse como usuario del Grid.

   A. Ir al directorio donde estén los archivos del Grid:

   cd /usr/local/globus/bin/

B. Ejecutar la orden:

grid-proxy-init

C. Nos pide la contraseña. Introducirla:

jorge

De esta forma hemos creado el Proxy.

2. Comprobar si la habilitación es correcta ejecutando la orden:

globussun –a –r <nombre de la máquina donde estés>

Si la habilitación ha sido correcta se mostrará un mensaje como el siguiente:

"Autentification test succesfull"

3. Exportar la variable GRIDWAY_HOME, ejecutando la siguiente orden desde el directorio donde estén los archivos *.java*:

export GRIDWAY_HOME = <ruta del directorio GridWay>

4. Llamada a la ejecución del programa, desde el directorio donde estén los archivos *.java*, mediante la orden:

java FramePpal <ruta del home de usuario>

Donde *FramePpal* es el formulario principal de la aplicación. La ruta del home de usuario se le pasará al *main* (programa principal del *FramePpal*) de la aplicación como parámetro. Así evitamos el uso de rutas absolutas dentro del programa.

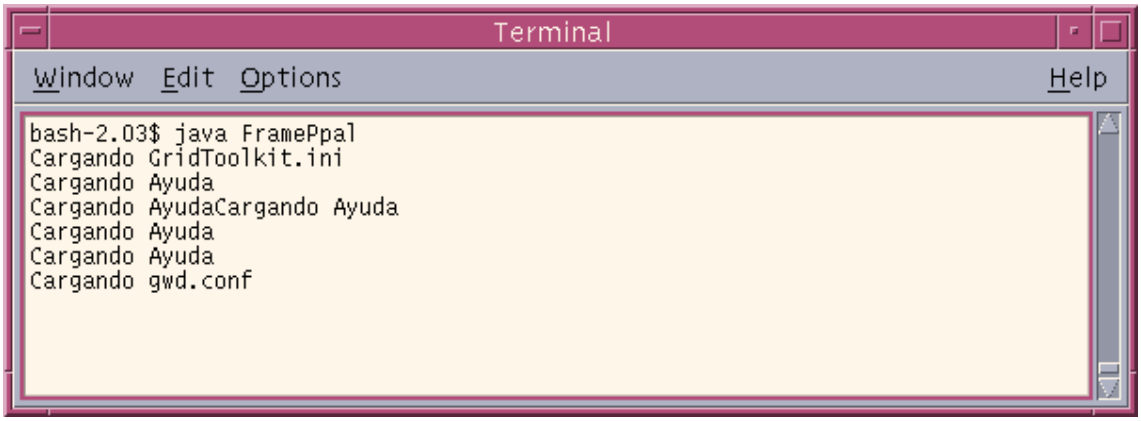
➢ PASOS DE LA EJECUCIÓN.

Tras la llamada para arrancar la aplicación, lo primero que ocurre es:

2. Se carga el archivo *gridtoolkit.ini.*
3. Se carga la ayuda de la aplicación, leyendo el archivo *Ayuda.txt*.
4. Si es la primera vez que se ejecuta la aplicación el archivo *gwd.conf* no contiene información útil.

Si no es la primera vez que se ejecuta la aplicación, se cargará el archivo *gwd.conf* (con información útil) en este momento. Este archivo está en el directorio:

<ruta del directorio GridWay>*/etc/gwd.conf*



La información de este archivo la volcará a la pantalla de *Configuration* (se verá cuando se abra la aplicación).

5. Se abre la aplicación.
   - Si es la primera vez que se ejecuta la aplicación:
     - El archivo *gridtoolkit.ini* no contiene nada, por lo tanto no están definidos el directorio GridWay ni el editor que queremos utilizar.
     - Salta un mensaje de aviso informándonos de que tenemos que seleccionar el directorio GridWay en la opción de Configuración.

     - Tenemos que entrar en *Configuration* y seleccionar tanto el directorio GridWay como el editor (si no hacemos esto último, no se nos permitirá entrar en *Job Submission*).
     - Además el archivo *gwd.conf* no contiene información válida. Tenemos que rellenar los datos de *Configuration* y, al salvarlos, modificará el archivo *gwd.conf* con ellos (información válida).

   - Si no es la primera vez que ejecutamos la aplicación:
     - El archivo *gridtoolkit.ini* contendrá el directorio GridWay y el editor seleccionados.
     - Lo lee y vuelca esa información a la pantalla de *Configuration.*

- Vemos la información (válida) del archivo *gwd.conf* (Schedulling Interval y Prologue, Epilogue y Wrapper Scripts) cargada también en la pantalla *Configuration.*

6. Ya estamos en disposición de seleccionar cualquier opción del formulario principal y trabajar con ella.

AYUDA DE LA APLICACIÓN

La aplicación contiene un sistema de ayuda que pueden configurar los propios usuarios.

Consiste en un fichero *Ayuda.txt*, que se carga al ejecutar la aplicación y que contiene todos los nombres de las etiquetas y botones del entorno, para que el usuario asocie la ayuda que quiera a cada uno de ellos.

Es importante mantener el formato de este archivo, que es:

TÍTULO
Etiqueta: <valor asociado de ayuda>
FIN TÍTULO

Podemos introducir la frase de ayuda que queramos asociar a esa etiqueta tras los dos puntos y hasta introducir un *enter*, manteniendo intactos los títulos, las etiquetas y los dos puntos.

El archivo tiene el siguiente aspecto:

```
EDITOR  DE EXPERIMENTOS
      PESTAÑA FILES
            Experiment Dir Ed:Experiment Dir Ed
            Executable:Executable
            Arguments:Arguments
            Standard Input:Standard Input
            Job Name:Job Name
            Standard Output:Standard Output
            Standard Error:Standard Error
            Input Files:Input Files
            Restart Files:Restart Files
            Output Files:Output Files
      PESTAÑA RESOURCE SELECTION
            Resource Selector:Resource Selector
            Rank Function File:Rank Function File
            Host Requirement File:Host Requirement File
            Dynamic Rank:Dynamic Rank
            Dynamic Host:Dynamic Host
            GIIS Server:GIIS Server
            Discovery Timeout:Discovery Timeout
      PESTAÑA PERFORMANCE
            Suspension Time:Suspension Time
            Poll Timeout:Poll Timeout
            Perf.Monitor Timeout:Perf.Monitor Timeout
            Performance Evaluator:Performance Evaluator
            Profile File:Profile File
            Dynamic Profile:Dynamic Profile
      PESTAÑA ADVANCED
            Prologue Script Ed:Prologue Script Ed
            Epilog Script Ed:Epilog Script Ed
            Wrapper Script Ed:Wrapper Script Ed
```

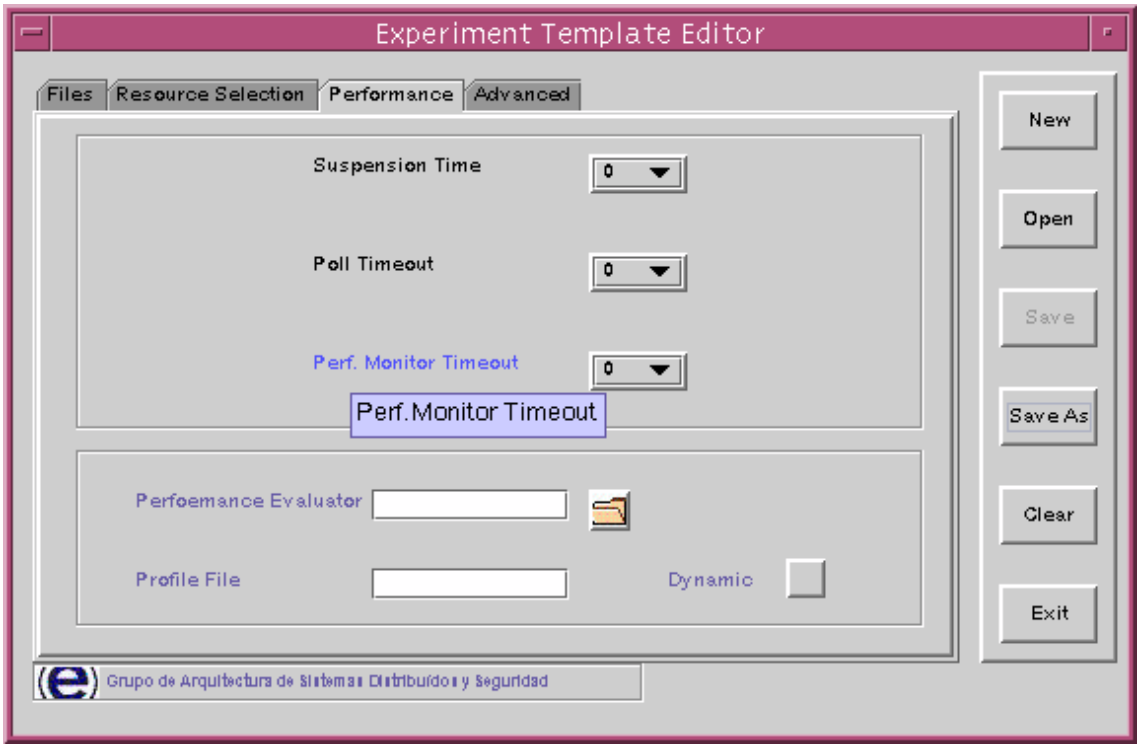
```
            Default Prologue:Default Prologue
            Default Epilogue:Default Epilogue
            Default Wrapper:Default Wrapper
      BOTONES
            New:New
            Open:Open
            Save Ed:Save Ed
            Save As:Save As
            Clear Ed:Clear Ed
            Exit Ed:Exit Ed
                Job Template Ed:Job Template Ed

FIN EDITOR  DE EXPERIMENTOS

CONFIGURATION
      GridWay Directory:GridWay Directory
      Editor Conf:Editor Conf
      Schedulling Interval:Schedulling Interval
      Prologue Script Conf:Prologue Script Conf
      Epilogue Script Conf:Epilogue Script Conf
      Wrapper Script Conf:Wrapper Script Conf
      BOTONES
            Stop Conf:Stop Conf
            Start:Start
            Save Conf:Save Conf
            Exit Conf:Exit Conf
FIN CONFIGURATION

HOST MONITORING
      Tabla Host:Tabla Host
      Text Area Tail:Text Area Tail
      BOTONES
            Refresh:Refresh
            Refresh All:Refresh All
            Attach to Job:Attach to Job
            Exit HostMon:Exit HostMon
FIN HOST MONITORING
JOB MONITORING
      Tabla Job:Tabla Job
      Text Area Historia:Text Area Historia
      Text Area Leyenda:Text Area Leyenda
      BOTONES
            Stop JobMon:Stop JobMon
            Resume:Resume
            Kill:Kill
            Reschedule:Reschedule
            History:History
            Refresh:Refresh
            Exit JobMon:Exit JobMon
FIN JOB MONITORING
JOB SUBMISION
      Experiment Dir JobSub:Experiment Dir JobSub
      JobTemplate JobSub:JobTemplate JobSub
      Array Job:Array Job
      NºTasks:NºTasks
      Text Area InfoSubmit:Text Area InfoSubmit
      BOTONES
            Editor JobSub:Editor JobSub
```

```
            Submit:Submit
            Clear JobSub:Clear JobSub
            Exit JobSub:Exit JobSub
FIN JOB SUBMISION
```

Una vez cargada, la ayuda asociada aparecerá siempre que nos posemos encima de una etiqueta o de un botón, en el ToolTipText.

En esta imagen el ratón está posado sobre la etiqueta *Performance monitor Timeout,* y podemos ver la ayuda asociada (aún simbólica) a esa etiqueta

Perf.Monitor Timeout



BIBLIOGRAFÍA

- Página web " *www.sun.com*" para descargar la aplicación *"Sun ONE Studio 4 CE".*
- Api's de Java *"Java™ Platform 1.2 API Specification"*.
- Ayuda, tutoriales y ejemplos de *"Borland JBuilder 4 Enterprise"*.
- Manual de la Escuela Superior de Ingenieros Industriales *"Aprenda Java como si estuviera en primero"*.
- Libro de Java "*Thinking in Java"* de Bruce Eckel.

LISTADO DE PALABRAS CLAVE

- Grid Dinámico (DataGrid): herramienta capaz de ejecutar tareas complejas compuestas por trabajos, donde la salida de uno o más trabajos es la entrada de uno o más trabajos.
- Tarea: cada uno de los trabajos que podemos asignar a un host mediante el GridWay.
- Host: equipo activo en el GridWay.
- Demonio: programa que lanza la ejecución del GridWay y nos permite realizar la gestión de tareas del mismo.
- Thread: clase predefinida de java que permite lanzar un proceso de forma concurrente al resto de la aplicación, para asi no bloquearla.

AUTORIZACIÓN

Los alumnos abajo firmantes, asignados para la realización de este proyecto, autorizan a la Universidad Complutense de Madrid a difundir y utilizar con fines académicos, no comerciales, tanto la memoria, como el código, la documentación y/o el prototipo desarrollado de dicho proyecto.

Bahia Carbajo Horcajo Antonio Martínez Alfaya Jorge Merino Granizo