PROYECTO FIN DE CARRERA

DESARROLLO DE UN PROTOTIPO SOFTWARE PARA LA EJECUCIÓN DE SERVICIOS GRID SEMÁNTICOS BASADO EN TECNOLOGÍA DE AGENTES

ANEXO 1:

INSTALACIÓN Y MANUAL DE USUARIO

Índice

1. INTRODUCCIÓN **5**

2. INSTALACIÓN Y DESPLIEGUE **6**

3. INTERFACES DE LA APLICACIÓN **9**

3.1. Interfaz del consumidor de servicios 9

3.2. Interfaz del proveedor de servicios 10

3.3. Interfaz del desarrollador de software 11

4. FUNCIONAMIENTO INTERNO **13**

Índice de figuras

1. Desplegar un Servicio Grid desde eclipse 7
2. Interfaz del consumidor 9
3. Lista de servicios encontrados 10
4. Resultado de la consulta 10
5. Vista de los Servicios Grid Semánticos ofrecidos 11
6. Edición de Servicios Grid Semánticos 12
7. Diagrama de agentes en JADE 14

1. INTRODUCCIÓN

A lo largo de este trabajo se ha desarrollado un sistema multiagente capaz de automatizar las tareas que comprenden la ejecución de un Servicio Grid (descubrimiento, composición, selección e invocación). Para hacer más sencillo su uso, se ha desarrollado, conjuntamente, una aplicación Web que sirve de interfaz de comunicación entre los usuarios y el sistema. De esta forma, únicamente es necesario disponer de un ordenador con conexión a Internet y un navegador Web instalado para disfrutar de toda la funcionalidad de la plataforma.

Esta aplicación Web consta de dos interfaces: la interfaz del consumidor de servicios, que nos guiará por el proceso de ejecución del Servicio Grid; y la interfaz del proveedor de servicios, que permitirá a los proveedores registrar sus Servicios Grid, junto con su información semántica, en el Semantic Binding.

Una tercera interfaz, la del desarrollador, está prevista para el futuro. Esta interfaz permitirá a los desarrolladores personalizar la plataforma con funcionalidades como: implementación de roles, instanciación dinámica de agentes y gestión de ontologías.

2. INSTALACIÓN Y DESPLIEGUE

Para poner en funcionamiento la plataforma es necesario tener instaladas una serie de aplicaciones. Estas son:

- **Tomcat 5.5 (`http://tomcat.apache.org`):** es el contenedor de *servlets* donde será desplegada la aplicación Web. Su instalación consiste simplemente en descargar el archivo *zip* que contiene Tomcat 5.5, y descomprimirlo en un directorio.

- **MySQL 5.0 (`http://www.mysql.com`):** es un sistema gestor de bases de datos (SGBD) que almacenará las bases de datos necesarias para el correcto funcionamiento de la aplicación. La instalación se realiza descargando el archivo *exe* que contiene MySQL, y ejecutarlo. El mismo programa de instalación se encargará de guiar al usuario durante el proceso.

- **Globus Toolkit 4 (`http://www.globus.org`):** es un conjunto de herramientas de código abierto para el desarrollo basado en tecnología Grid. Incluye servicios y librerías para la monitorización, descubrimiento y gestión de recursos, además de para la seguridad y gestión de archivos. En él se desplegarán los Servicios Grid que vayan a estar disponibles para la plataforma. Para instalar Globus Toolkit 4 hay que descargar el archivo *zip* que lo contiene y descomprimirlo en un directorio. Una vez descomprimido, hay que crear una variable de entorno de nombre GLOBUS_LOCATION y cuyo valor sea la ruta del directorio que contiene la instalación de Globus Toolkit 4 (Por ejemplo, GLOBUS_LOCATION=C:\SGS\ws-core-4.0.5). Por último, hay que crear otra variable de entorno con el nombre JAVA_HOME y cuyo valor sea la ruta en la que la máquina virtual de Java está instalada (Por ejemplo, JAVA_HOME=C:\Java\jdk1.5.0_09). Es importante mencionar que, GLOBUS_LOCATION, como JAVA_HOME no deben contener espacios en sus respectivos valores. Para poner en funcionamiento Globus Toolkit 4, basta con ejecutar el siguiente comando:

  %GLOBUS_LOCATION%\bin\globus-start-container -p 18080 -nosec

  Este comando significa que arranca la herramienta Globus Toolkit 4 en el puerto 18080 sin opciones de seguridad. Y finalmente, para desplegar los Servicios Grid en Globus Toolkit 4, puede hacerse mediante el plugin GDT de eclipse, con el que se han construido los Servicios Grid (ver Figura 1); o mediante la ejecución del siguiente comando:

  %GLOBUS_LOCATION%\bin\globus-deploy-gar archivo-gar.gar[1]

[1] Un archivo *gar* (Grid Archive), es un archivo similar al *jar* (Java Archive). Contiene los archivos necesarios para la ejecución de un Servicio Grid.

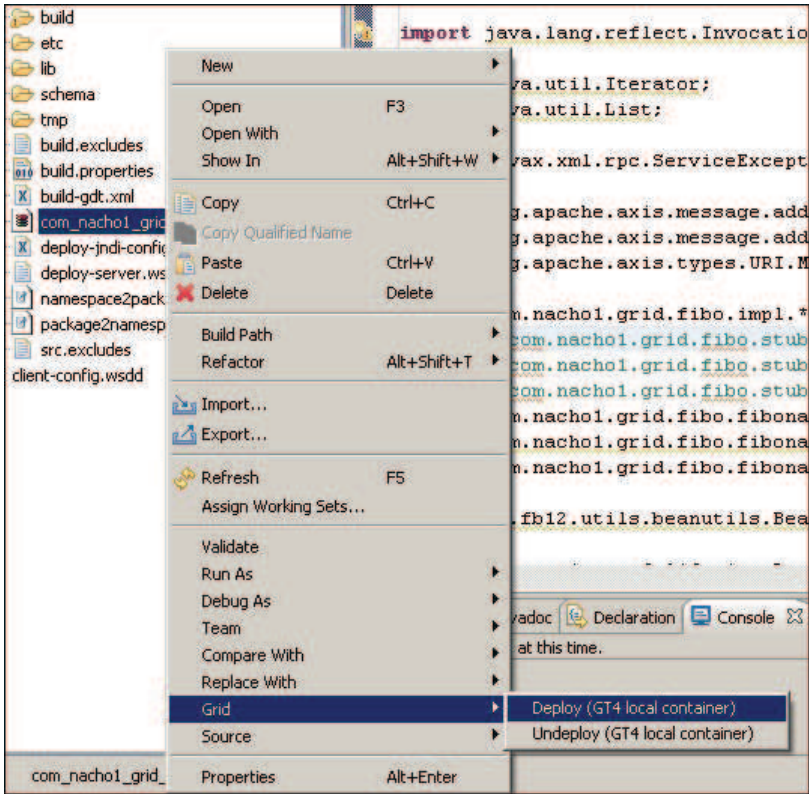

Figura 1: Desplegar un Servicio Grid desde eclipse

- **Semantic Binding Service (`http://www.cs.man.ac.uk`):** este servicio será utilizado para asociar los Servicios Grid con su información semántica. También es un Servicio Grid, por lo que deberá ser desplegado en Globus Toolkit 4. Para desplegar el servicio habrá que descargar el archivo *zip* que lo contiene y descomprimirlo en un directorio. Una vez descomprimido hay que situarse en dicho directorio y ejecutar el siguiente comando:

  %ANT_HOME%\bin\ant deploy

- **Metadata Query Service (`http://www.cs.man.ac.uk`):** gracias a este servicio es posible realizar consultas sobre los metadatos almacenados en los Semantic Bindings, para recuperar los datos de los Servicios Grid Semánticos. Como en el caso anterior, será desplegado en Globus Toolkit 4. Su instalación es igual que la de Semantic Binding Service.

Una vez estén instaladas las aplicaciones y desplegados los servicios en Globus Toolkit, se deben crear las bases de datos necesarias para el funcionamiento de la aplicación. Para ello hay que restaurar el fichero restore.sql, proporcionado junto con la documentación del proyecto. Después, se ha de desplegar el fichero sgsamas.war, también proporcionado junto con la documentación del proyecto, en Tomcat. Este fichero contendrá el framework desarrollado en este proyecto junto con todas las librerías necesarias para su ejecución.

Por último hay que modificar el fichero web.xml para indicar, entre otras cosas, la localización de la base de datos, el driver que utiliza, y el nombre y contraseña de usuario que se utilizará para la conexión con dicha base de datos. El contenido del fichero web.xml podría ser el siguiente:

```
...
<web-app>
   <servlet>
      <servlet-name>FrontController</servlet-name>
      <servlet-class>webapp.controller.FrontController</servlet-class>
      <init-param>
         <param-name>repository</param-name>
         <param-value>MySQL</param-value>
      </init-param>
      <init-param>
         <param-name>dbDriver</param-name>
         <param-value>com.mysql.jdbc.Driver</param-value>
      </init-param>
      <init-param>
         <param-name>dbConnection</param-name>
         <param-value>jdbc:mysql://localhost:3306/sgsmas</param-value>
      </init-param>
      <init-param>
         <param-name>dbUser</param-name>
         <param-value>sgsmas</param-value>
      </init-param>
      <init-param>
         <param-name>dbPassword</param-name>
         <param-value>sgsmas</param-value>
      </init-param>
      <init-param>
         <param-name>PlatformConfigurationFile</param-name>
         <param-value>/WEB-INF/init.conf</param-value>
      </init-param>
      <init-param>
         <param-name>ConsumerConfigurationFile</param-name>
         <param-value>/WEB-INF/consumerInit.conf</param-value>
      </init-param>
      <init-param>
         <param-name>ProviderConfigurationFile</param-name>
         <param-value>/WEB-INF/providerInit.conf</param-value>
      </init-param>
   </servlet>
   <servlet-mapping>
      <servlet-name>FrontController</servlet-name>
      <url-pattern>*.ctrl</url-pattern>
   </servlet-mapping>
   ...
</web-app>
```

Cabe destacar que, para que los Servicios Grid estén disponibles para los usuarios deben ser desplegados en Globus Toolkit y asociados a su información semántica con Semantic Binding Service. Para las pruebas se ha utilizado el generador automático de información semántica WSRF2OWL-S. Además las ontologías que maneje la plataforma (como por ejemplo las ontologías de dominio) deben ser accesibles por su URL.

3. INTERFACES DE LA APLICACIÓN

3.1. Interfaz del consumidor de servicios

Los usuarios consumidores, una vez autenticados (obviamente han de estar registrados) con su *login* y *password*, serán dirigidos a la interfaz de consumidores de servicios. En ella podrán elegir el entorno en el que realizarán las peticiones (Bioinformática, e-Gobierno, Matemáticas, etc.). La Figura 2 muestra un ejemplo de uso de la interfaz de consumidor. En ella se ve como el usuario Nacho elige el entorno de las Matemáticas para realizar su consulta.

Figura 2: Interfaz del consumidor

Siguiendo con el ejemplo de la Figura 2, la petición que realiza el usuario es obtener la integral definida de la función $f(x) = x$ entre 0 y 3, con 10 intervalos. También es posible realizar las peticiones mediante ficheros OWL directamente, bien sea especificando su dirección local en el disco duro, o su URL. Además también es posible especificar un fichero que incluya las preferencias del usuario, las cuales serán utilizadas en el proceso de negociación. Una vez procesada la petición, se devolverá una lista con los Servicios Grid que podrían satisfacerla. El usuario puede, entonces, marcar la solución que más le satisfaga (o rechazarlas todas). La Figura 3 muestra como se listan los Servicios Grid seleccionados por la plataforma. Aparece un servicio que satisface la petición, llamado calcularIntegral.

Figura 3: Lista de servicios encontrados

Una vez elegido el servicio, el sistema lo invoca y devuelve un informe con el resultado de la ejecución. La Figura 4 muestra el resultado obtenido tras ejecutar el servicio calcularIntegral.

Figura 4: Resultado de la consulta

3.2. Interfaz del proveedor de servicios

La interfaz de proveedores de servicios facilita a los usuarios proveedores gestionar una lista de Servicios Grid que ofrecen. En todo momento pueden consultar la lista, y añadir,

eliminar y editar servicios de forma rápida y sencilla. La Figura 5 muestra el aspecto visual de la lista de servicios ofertados.

Figura 5: Vista de los Servicios Grid Semánticos ofrecidos

Para registrar un nuevo Servicio Grid, el proveedor debe indicar el nombre y descripción del servicio, y la ruta del fichero que contenga la descripción semántica del servicio. También tendrá la opción de indicar la ruta del fichero que contenga las preferencias de ejecución de dicho servicio.

Mediante el botón de edición, es posible ver el contenido semántico del servicio y cambiarlo si fuera necesario. Esto puede verse en la Figura 6.

3.3. Interfaz del desarrollador de software

Esta interfaz ha sido ideada con el fin de facilitar la adaptación de la plataforma según las necesidades de los usuarios, y así dotarla de una mayor flexibilidad. Su implementación está aún por desarrollar, pero se pretende que proporcione las siguientes funcionalidades:

- **Implementación de roles:** la plataforma permitirá que los desarrolladores añadan diferentes implementaciones para cada rol identificado en el sistema. Y será el propio sistema el que validará si dicha implementación se ajusta a la interfaz que implementa dicho rol.

- **Instanciación de agentes:** los desarrolladores podrán elegir el número y tipo de agentes que compondrán el núcleo de la plataforma, según lo consideren oportuno. Además el sistema deberá habilitar el enlace dinámico entre agentes y roles, ya que actualmente no es posible.

Figura 6: Edición de Servicios Grid Semánticos

- **Gestión de ontologías:** el sistema deberá permitir al desarrollador la opción de añadir y eliminar las ontologías que necesite, indicando únicamente el tipo de la ontología en cuestión.

4. FUNCIONAMIENTO INTERNO

En este punto se explicará detalladamente cada uno de los aspectos del funcionamiento interno de la plataforma cuando se ejecuta una petición de usuario. Suponemos que el sistema se encuentra plenamente operativo, esto es, desplegado en el servidor, con las ontologías almacenadas y con los Servicios Grid accesibles.

Lo primero que debe hacer el usuario es registrarse en la plataforma, especificando algunos datos personales, una contraseña y sus preferencias para las peticiones. Estas preferencias serán utilizadas más tarde durante el proceso de negociación. Deberá indicar también el tipo de usuario que es (consumidor o proveedor).

Una vez registrado el usuario (continúa el ejemplo para el caso de un usuario consumidor de servicios) y autenticado en el sistema, un *Framework Agent* creará una instancia del agente representante del usuario, *Customer Agent*. Ahora el usuario está en disposición de realizar las consultas. Como se ha explicado anteriormente, las consultas pueden realizarse en lenguaje natural, o directamente mediante el uso de ficheros OWL de descripción de objetivos. Si se hacen en lenguaje natural se transformarán en ontologías de objetivo por el analizador de lenguaje natural.

Customer Agent envía la petición al *Discovery Agent*, y este busca, por medio del *Semantic Binding Service*, los Servicios Grid que satisfagan dicha petición. Si durante la búsqueda no se ha encontrado ningún servicio que se ajuste a los requerimientos del usuario, el agente descompondrá la petición, por medio de un sistema de planificación, y buscará los servicios que satisfagan cada subpetición, formando de esta manera servicios compuestos. Con estos servicios (o servicios compuestos) encontrados, el *Discovery Agent* formará una lista y se la pasará al *Customer Agent*.

El *Customer Agent*, al recibir la lista de servicios, se la envía al *Selector Agent* para que descarte los servicios que no cumplan con la lista de preferencias que el usuario, al que representa el *Customer Agent*, introdujo durante el registro. Una vez hecho esto, el *Selector Agent* pide al *Framework Agent* que cree tantas instancias de *Service Agent* cómo servicios queden en la lista. En este punto comienza el proceso de negociación. Cada *Service Agent* creado, marca los requisitos especificados por los proveedores y hace su propuesta según sus preferencias. Por su parte, el *Selector Agent*, elige el conjunto de servicios que satisface las preferencias del cliente, y devuelve al *Customer Agent* una lista de servicios junto con las condiciones, acordadas durante la negociación, para la ejecución de cada servicio.

Ahora, la lista de servicios será mostrada al usuario para que sea el verdadero responsable de la elección del servicio que se ajuste a sus necesidades, también tiene la posibilidad de no elegir ninguno en caso de que ninguno le satisfaga. Una vez que el usuario ha elegido el servicio que desea ejecutar, se lo comunica al *Service Agent* que representa a

dicho servicio. Este *Service Agent* será el encargado de invocar el Servicio Grid y ejecutar los métodos apropiados. Una vez obtenidos los resultados son mostrados por pantalla al usuario. En la Figura 7 puede verse todo el proceso mediante el diagrama que elabora JADE.

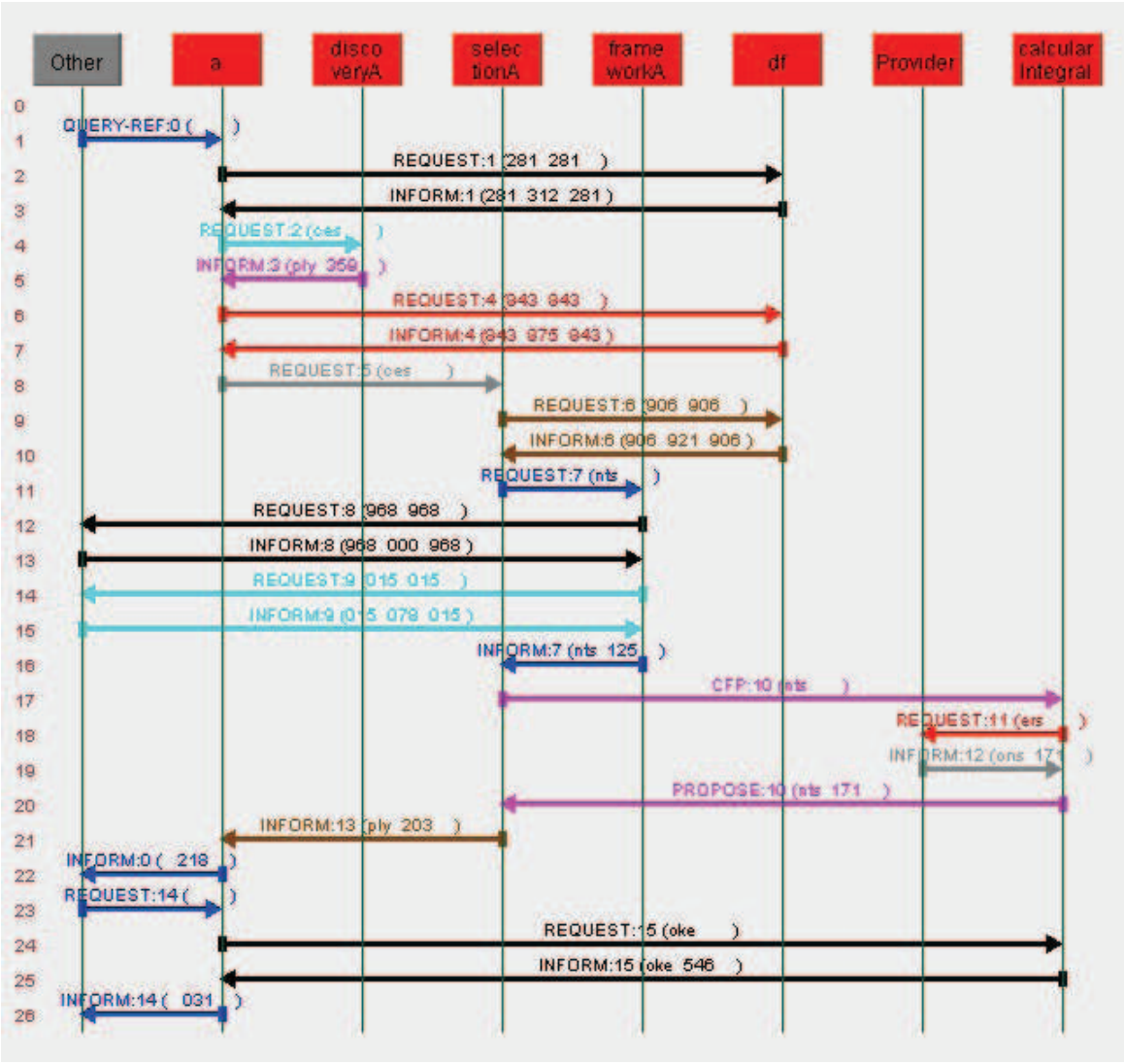

Figura 7: Diagrama de agentes en JADE