Diseño de un Medio de Gestión de Servicios para Sistemas Multiagentes

Víctor R. Bravo
Postgrado de Computación. Facultad de Ingeniería
Universidad de los Andes
Mérida – Venezuela.
victorb@cptm.ula.ve

y

José L. Aguilar, Franklin Rivas, Mariela Cerrada
Centro de Microcomputación y Sistemas Distribuidos (CEMSID)
Escuela de Ingeniería de Sistemas, Facultad de Ingeniería
Universidad de los Andes
Mérida – Venezuela
{aguilar, rivas}@ula.ve

Abstract

Agent based programming is a new paradigm to build software systems. It is based on the generation of software modules with capacities like communication and autonomy on its actions that facilitates the construction of auto-organized and more efficient complex systems. In this paper a middleware based on agents for a computational platform of MultiAgents Systems is developed. Particularly, the middleware has been used like one of the components of the platform SCDIA (Distributed Control System based on agents). This middleware provides services for access and administration of hardware resources, applications, data and agents, and it has qualities associated to distributed systems such as interoperability, migration, security, naming, communication, among others.
Keywords: Middleware, Intelligent Systems, MultiAgent Systems, Software engineering, knowledge engineering.

Resumen

La programación basada en agentes constituye un nuevo paradigma en la construcción de sistemas de software. Ella se basa en la generación de módulos de software que tengan capacidades de comunicación y autonomía sobre sus acciones, lo que posibilita la construcción de sistemas complejos autorregulados y más eficientes. En este trabajo se desarrolla un Medio de Gestión de Servicios (MGS) basado en agentes que formaría parte de una plataforma computacional para ejecutar Sistemas Multiagentes. Particularmente, el MGS ha sido usado como uno de los componentes de la plataforma SCDIA (Sistema de Control Distribuido basada en agentes) propuesta en [2,5]. Este medio proporciona servicios de acceso y gestión de recursos de hardware, de aplicaciones, de datos y de agentes, y posee las cualidades asociadas a los sistemas distribuidos tales como interoperabilidad, migración, seguridad, nombramiento, comunicación, entre otras.
Palabras claves: Medio de Gestión de Servicios, Sistemas Inteligentes, Sistemas Multiagentes, Ingeniería de Software, Ingeniería de conocimiento.

1. INTRODUCCION

Los sistemas multiagentes (SMAs) son sistemas de software que agrupan un conjunto de cualidades pertenecientes a los tópicos más avanzados que en el área de computación se tratan en la actualidad. Esta conjunción de conceptos y tecnologías, hace posible dar respuestas a problemas complejos, a través de la construcción de sistemas auto-organizados y dinámicos, que pueden integrarse con aplicaciones legadas (Las aplicaciones legadas son aplicaciones operativas que cuentan con una utilidad real, por lo tanto es inconveniente su reemplazo o desecho), y sistemas heterogéneos.

El paradigma orientado a agentes tiene años de estudio e investigación en los espacios académicos, y se han propuesto una gran cantidad de modelos y prototipos de aplicaciones que dan muestra de su utilidad [3,4]. El paradigma no se ha establecido en una gran mayoría de ambientes empresariales e industriales, a excepción del sector de telecomunicaciones donde se han desarrollado una gran cantidad de aplicaciones que actualmente funcionan y resuelven una gran cantidad de problemas utilizando teorías de agentes [3].

Se percibe una creciente actividad sobre este tema en los departamentos de investigación y desarrollo de importantes empresas del ramo del software, tal es el caso de IBM (International Business Machines), Microsoft, HP, entre otras. Además, la existencia de organizaciones tales como FIPA (Foundation Intelligent Physical Agents, organización que agrupa importantes empresas del campo tecnológico que han dictado un conjunto de especificaciones para la construcción de sistemas multiagentes), evidencia que este tipo de tecnología ofrecerá mejores y nuevas soluciones a problemas que actualmente enfrentan las organizaciones que gestionan y mantienen procesos complejos.

Por otro lado, las tendencias actuales en automatización de procesos continuos están orientadas al desarrollo de sistemas de control basados en una arquitectura jerárquica de referencia. Atendiendo a estas tendencias se ha definido un Modelo de Referencia para el Desarrollo de Sistemas de Control Distribuidos Inteligentes basado en Agentes (SCDIA), representado por una jerarquía de cinco niveles, entre los cuales se distribuye la información del sistema y las tareas de control [1, 2].

La adaptación de SMAs dentro de plataformas tecnológicas ya operativas, no puede considerarse como un problema trivial. La heterogeneidad trae como consecuencia un extenso y difícil trabajo de acoplamiento de interfaces; aunado a esto, el nuevo paradigma basado en agentes trae consigo nuevos conceptos que deben ajustarse a los ya existentes. Por ejemplo, cuando se habla de "conversaciones" entre agentes, es necesario trasladar este concepto al contexto de computación distribuida en el cuál se manipulan objetos, mensajes, protocolos, procesos, con la finalidad de implantar la comunicación entre entes que llamaremos agentes. La necesidad de contar con un conjunto de servicios que brinden conexión con recursos y aplicaciones puede constituirse en una herramienta indispensable para la integración del paradigma de agentes a las plataformas computacionales de las organizaciones de hoy en día. En este proyecto se diseña e implementa un Medio de Gestión de servicios (middleware) que permite la comunicación entre recursos, aplicaciones, y agentes. En nuestra propuesta se hace uso de conceptos tales como componentes, reflexión y computación distribuida; que brindan la base teórica para el trabajo planteado.

2. ASPECTOS TEÓRICOS

2.1 Sistemas Multiagentes (SMAs)

Cuando se habla de agentes en computación no existe un consenso sobre su definición. El concepto de "agente" tiene asociado un conjunto de cualidades, que, aunque cada una de ellas tiene una definición clara, no tienen ni una única interpretación ni una sola forma de implementarse. Ahora bien, un agente debe reunir algunas de las siguientes características:

- *Autonomía:* los agentes son autónomos en la medida en que actúan sin la intervención humana ni de otros sistemas externos [12]. Se puede dar una noción general de autonomía, definiéndola como la capacidad que tiene un agente de tener un comportamiento propio, y reaccionar a los estímulos externos basándose en su estado interno.
- *Comunicación:* la comunicación entre agentes tiene entre sus objetivos acercarse a lo que es la comunicación entre personas, a través de intervenciones dinámicas con oraciones y frases que contienen significados. La capacidad de cada agente de conversar utilizando un lenguaje basado en ontologías y realizar intervenciones asíncronas, constituye un paso adelante en llevar el concepto real de conversación al ámbito computacional.
- *Sociabilidad:* esta capacidad está muy relacionada con el aspecto comunicacional. Una sociedad de agentes es un grupo de agente que interactúan, se comunican, conversan, "piensan", y actúan en conjunto para lograr un objetivo común.
- *Reactividad:* Es la capacidad de emitir una acción inmediata al recibir una señal o percibir un estado en el ambiente.
- *Inteligencia:* se pueden utilizar un conjunto variado de técnicas inteligentes en la construcción de agentes, entre las cuales se pueden nombrar las reglas difusas para analizar situaciones dinámicas, las redes neuronales para predecir comportamientos y variables del ambiente, las colonias de hormigas como una técnica de coordinación entre agentes, los algoritmos genéticos como método de búsqueda, entre otros. En la medida en que se integren más técnicas inteligentes a la construcción de agentes, se logrará un acercamiento más certero a la característica inteligente de los agentes.

- Movilidad: *es la capacidad que tiene un agente de mover su estado y código de ejecución de un nodo a otro en un sistema distribuido.*

No existe una única arquitectura para diseñar y construir agentes, por ejemplo, una corriente se ha inclinado por el diseño de agentes utilizando programación lógica, lo que supone una estrecha vinculación a un motor de inferencia con colecciones de hechos; otra corriente ha vinculado los agentes con los sistemas de control, y también, existen arquitecturas específicas basadas en creencias y deseos.

2.2 Sistemas de Control Distribuidos Inteligentes basados en Agentes

Los Sistemas de Control Distribuidos Inteligentes basados en agentes (SCDIA) es específicamente una plataforma de SMA diseñada para sistemas de automatización industrial [1, 2]. Propone una colección de agentes caracterizando los elementos de control de procesos y definiendo un mecanismo genérico para el manejo de las actividades organizadas relacionadas con la automatización industrial. Así, los agentes del SCDIA son:

- Agente Medición: recoge la información necesaria para obtener el estado del proceso.
- Agente Controlador: toma acciones basadas en el estado del sistema.
- Agente Coordinador: modifica las decisiones del agente controlador y establece nuevos objetivos y servicios. Coordina la comunidad de agentes.
- Agente Actuador: ejecuta las decisiones tomadas por el agente controlador, agente coordinador, y/o agentes especializados.
- Agentes Especializados: ellos ejecutan las tareas especiales de la comunidad de los agentes.

El SCDIA es dividido en dos niveles: un nivel de la interacción con el ambiente dónde el agente medición y el agente actuador viven; y un nivel de decisión donde los otros agentes de la comunidad viven (ver figura 1). Nosotros llamamos a este conjunto de cinco agentes, la Comunidad de Agentes de Control.

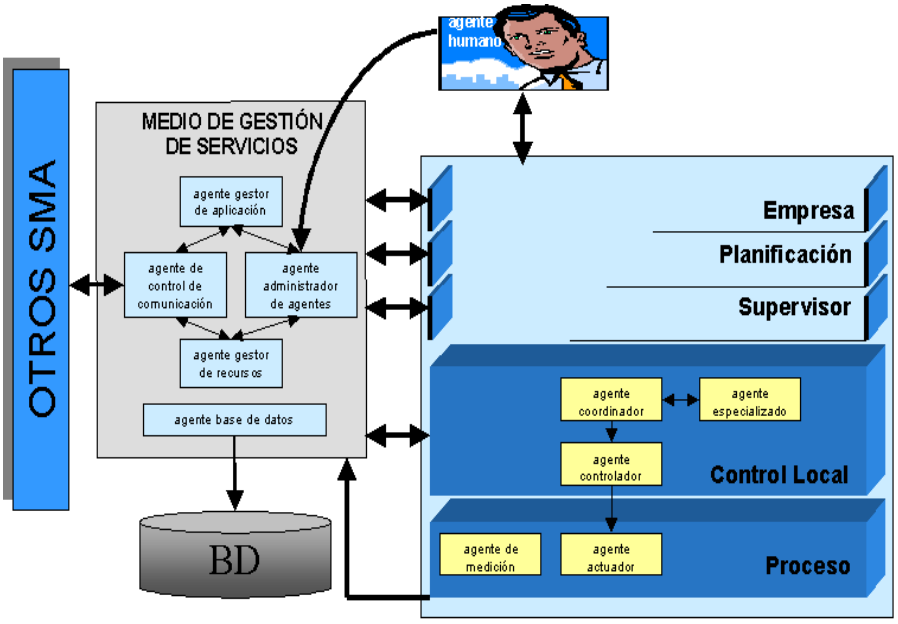

Figure 1. Modelo SCDIA

El SCDIA también propone una comunidad de agentes que manejan a los Agentes de Control. Esta comunidad se llama el Medio de Gestión de Servicios (MGS) [18]. El MGS permite la migración, la localización, la activación/desactivación de agentes, y el conocimiento del estado global del SCDIA; todo estos trabajos se llevan a cabo por el Agentes del MGS. Otro trabajo que el MGS realiza es controlar el inventario de todas las aplicaciones y recursos que se manejan y/o se proporcionan por los agentes. Por otro lado, también se necesita la administración de los datos guardados dentro del sistema; este trabajo lo lleva a cabo el Agente de la Base de datos del MGS. Finalmente, el SCDIA tiene la capacidad para comunicarse con otros SMA a través del MGS, específicamente por el Agente de Control de Comunicaciones.

2.3. Componentes

Los componentes son piezas de software reutilizables, disponibles para diferentes lenguajes y aplicaciones a través de una interfaz común. Permiten encapsular el acceso a datos, la lógica de negocio, el acceso a recursos de hardware

y de software, entre otras cosas. Se basan en una extensión de los objetos, tienen métodos y propiedades, y permiten persistencia, acceso remoto y un modelo distribuido.
En la actualidad existen varios esquemas que plantean una visión orientada a componentes Los dos esquemas más utilizados son el COM (Component Object Model) [22], propuestos para sistemas bajo sistemas operativos Windows, y el modelo basado en Beans, propuesto por Sun Microsystems para arquitecturas basadas en tecnología Java.

3. PROPUESTA DEL MEDIO DE GESTIÓN DE SERVICIOS (MIDDLEWARE)

Se desarrolló una capa intermedia de Gestión de Servicios (Middleware) constituida por componentes de software en un ambiente distribuido y heterogéneo. Cada componente puede actuar como vía de acceso al procesamiento para una determinada aplicación, como puente entre clientes remotos y fuentes de datos, o interfaz de acceso a recursos y sistemas de información. El middleware es el corazón del sistema de agentes distribuido, puesto que en él moran los agentes que manejan los servicios de comunicación y le otorgan al sistema de agentes características tales como seguridad, transparencia, nombramiento, migración e interoperabilidad. El modelo de middleware está estructurado en tres niveles (ver figura 2).

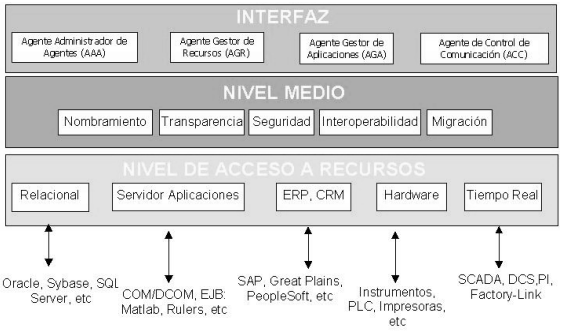

Figura 2. Arquitectura del MGS

Nivel Interfaz: es el nivel de interacción con agentes y usuarios. A esta capa pertenecen cinco agentes del SCDIA: Agente Administrador de Agentes (AAA), Agente Gestor de Recursos (AGA), Agente Gestor de Aplicaciones (AGA), Agente de Base de Datos (ABD) y Agente de Control de Comunicación (ACC). Estos interactúan con el nivel medio a través de una interfaz que permite la comunicación entre los dos niveles. El nivel interfaz es el encargado de establecer las pautas de conversación entre los componentes del sistema y los agentes, como también, definir los esquemas de coordinación con los agentes de nivel superior (por ejemplo, con la comunidad de agentes de control del SCDIA).

Nivel Medio: constituye el núcleo del sistema distribuido, provee servicios al nivel interfaz. Garantiza transparencia y seguridad en las transacciones, interoperabilidad de las aplicaciones y componentes de software, migración de agentes, objetos y/o recursos, comunicación ínter-proceso, y provee un sistema de nombramiento para la localización de agentes y/o objetos.

Nivel de Acceso a Recursos: Esta capa está integrada por todos los manejadores asociados a recursos directos, tales como, manejadores para acceso a datos relaciones, que implementan conexiones ODBC, JDBC, ADO, DAO, ADO.NET; manejadores de acceso a sistemas en tiempo real y a sistemas supervisores, acceso a sistema de planeación empresarial y gestión de recursos (ERP: Enterprise Resources Planning), acceso a sistemas de gestión de relaciones con los clientes (CRM); acceso a aplicaciones o componentes de software; y acceso directo a hardware específico.

3.1 Características Nivel Interfaz

Como se dijo antes, esta capa esta compuesta por los agentes que sirven de interfaz entre el MGS y la comunidad de agentes de control del SCDIA. Esta capa está compuesta por los siguientes agentes:

Nombre: Administrador de Agentes (AAA).
Tipo: Agente de Software.
Papel: Administrador del sistema multiagentes.
Descripción: se encarga de manejar, integrar y supervisar el estado del sistema multiagente. Este agente conoce la localización y estado de todos los agentes que existan en el sistema. El AAA dirige las migraciones de los agentes; así, cada agente que se mueve de un nodo a otro debe notificar al AAA el movimiento que ha efectuado; de manera que el agente administrador siempre tenga una vista ajustada al estado del sistema en tiempo real

Nombre: Base de Datos (ABD).
Tipo: Agente de Software.

Papel: Gestor de Datos en el SCDIA.
Descripción: este agente se encarga de establecer el enlace con los lugares donde existan datos de interés para el proceso que se esté ejecutando, sea que estos datos provengan de bases de datos (relacionales, orientadas a objetos, tiempo real, etc.), de SCADAS, DCS, medidores, o cualquier otro dispositivo o aplicación que pueda almacenar datos. Además, el agente debe permitir el traslado de los datos entre los diferentes dispositivos y/o aplicaciones de una manera transparente. Responde a las peticiones de los agentes de bases de datos, agentes de aplicaciones, agentes de recursos, y los agentes de la comunidad de agentes de control del SCDIA.

Nombre: Gestor de Aplicaciones (AGA).
Tipo: Agente de Software.
Papel: Administrador de aplicaciones del SCDIA.
Descripción: este agente se encarga de ubicar las aplicaciones que puedan ser requeridas por un proceso que se esté ejecutando, como por ejemplo de acceso a redes, programas de cálculo numérico o simbólico, aplicaciones de inteligencia artificial, de envío y recepción de mensajes, etc. Dichas aplicaciones pueden estar en cualquier servidor al que se tenga acceso y son requeridas por los agentes de bases de datos, de administración de recursos, y algunos de los agentes de la comunidad de agentes de control del SCDIA (coordinadores y especializados).

Nombre: Gestor de Recursos (AGR).
Tipo: Agente de Software
Papel: Administrador de recursos de Hardware del SCDIA
Descripción: este agente se encarga de distribuir el uso de los dispositivos (hardware) necesarios en la ejecución de un proceso, como por ejemplo procesadores, dispositivos de entrada/salida, dispositivos de almacenamiento, etc. Este agente puede ser accedido por cualquier agente del SCDIA.

Nombre: Control de Comunicación (ACC).
Tipo: Agente de Software.
Papel: Administrador de comunicaciones del SCDIA.
Descripción: es el encargado de mantener y controlar la comunicación entre sistemas multiagentes. Se encarga de traducir y manipular ontologías, y mantener un estado confiable del canal de comunicación.

3.2 Características del Nivel Medio

En cuanto al nivel medio, este le proporciona ciertas funcionalidades al MGS, tales como:

- *Transparencia:* se refiere al manejo de servicios siguiendo la filosofía de ocultar detalles de "cómo" se provee el servicio concentrándose en "que" provee el servicio, es decir, al agente se le ocultan detalles sobre la ejecución del servicio que no se consideran relevantes. Los servicios se proveen manejando en una capa oculta los detalles de implementación. La transparencia facilita la implantación de un sistema multiagente en una plataforma heterogénea. Para lograr transparencia, se debe utilizar una interfaz uniforme entre componentes, usuarios y agentes. En ciertos casos en necesario utilizar técnicas de envoltorio (wrapper), computación reflexiva, integración de código (.NET), con la finalidad de normalizar la interacción entre los componentes del sistema multiagente.

- *Seguridad:* debido a que el esquema distribuido opera bajo un sistema en red y de acceso externo, es necesario implementar mecanismos para mantener un nivel de seguridad que imposibiliten el acceso a aplicaciones y a datos por una fuente no acreditada. Cada agente debe ser autentificado cuando ingrese a un nodo de la plataforma, es decir, el nodo que acepta el agente debe saber el perfil del agente, su propietario, permisos con los que cuenta, y tareas que desempeña. En el mercado existen diferentes y variadas aplicaciones que implementan estos mecanismos, entre estos mecanismos podemos citar los algoritmos que utilizan criptografía, los algoritmos de validación de contraseñas, y los algoritmos de clave pública y privada, entre otros.

- *Interoperabilidad:* un problema recurrente en sistemas computacionales en red es el de integrar diferentes plataformas de software, diferentes aplicaciones corriendo en sistemas operativos diferentes que deben cumplir una función en conjunto, etc. La aparición de nuevos lenguajes de programación que implementan máquinas virtuales o código 100% portable, sistemas basados en objetos como CORBA, y métodos menos restrictivos como SOAP (**S**imple **O**bject **A**ccess **P**rotocol), han dado una respuesta a este problema.

- *Migración:* en sistemas basados en tecnología Java o CORBA, u otra tecnología similar, es posible implantar agentes que migran o que se mueven entre los nodos de una red. El agente administrador de agentes es el encargado de controlar el proceso de migración. Existen varias técnicas y modos de migración para agentes y objetos. Para lograr migración de agentes es necesario contar con una plataforma comunicacional con ciertas características, por lo general esta plataforma debe permitir el uso del protocolo TCP/IP. Los agentes móviles, generalmente se utilizan para recolectar información de varios servidores; se mueven o migran utilizando un protocolo común, y establecen sus operaciones localmente. Las plataformas distribuidas proveen herramientas para migración de objetos, estas herramientas pueden ser utilizadas en el proceso de migración de agentes. En el caso de los sistemas multi-agentes se proveen canales que permiten que los agentes viajen a través de los

diferentes nodos de la red. Es este tipo de ambiente un elemento importante a proveer es la seguridad, ya que una incursión de un agente o software externo puede reportar graves daños.

- *Mensajería y comunicación interprocesos*: es posible comunicar aplicaciones que corren en localidades diferentes a través de sistemas de mensajerías. Las aplicaciones pueden comunicarse entre ellas a bajo nivel, por ejemplo, a través de sockets (bajo protocolo TCP/IP), y a niveles más altos con lenguajes de marcas como XML (e**X**tended **M**arkup **L**anguage). También existen otros mecanismos de comunicación que son provistos por lenguajes o plataformas como Java o CORBA. Sistemas de pases de mensajes o de pizarrón (memoria compartida) también son aplicables. Todos estos sistemas de mensajería sirven de apoyo a lenguajes de alto nivel provistos para comunicación entre agentes, tales como KQML o ACL [12].

- *Sistema de Nombramiento:* para poder localizar eficientemente agentes, objetos y/o recursos es necesario contar con un sistema de nombramiento confiable que indique la función y asignación del agente u objeto en cuestión. También debe permitir acceder y manipular estos agentes u objetos de forma no ambigua. El sistema de nombramiento, también llamado "páginas blancas", debe ser administrado por un agente administrador. Este agente tiene la potestad de crear, iniciar, suspender y autorizar migraciones de todos los agentes. Los sistemas de nombramiento se implantan a través del uso de una base de datos que asocia recursos con nombres descriptivos, a los cuales se les puede asociar una jerarquía. Uno de los ejemplos de sistemas de nombramiento es el DNS (Domain Naming Server); sistema encargado de dirigir el nombramiento en la Internet y que implementa "espacios de nombres*" (namespace) o "dominios de nombres*", para soportar dominios de localidad por función o por servicio.

3.3 Características del Nivel de Acceso a Recursos

Un esquema para acceso a recursos integra los componentes y recursos al sistema multiagente. El esquema se realiza implementado un traductor o proxy que comunica el recurso con los otros componentes del sistema. El traductor hace uso del concepto de reflexión. La reflexión se refiere a la capacidad de un programa de manipular en tiempo de ejecución objetos, métodos, parámetros y variables propias, es decir, la propiedad de manipular los elementos que lo componen. También es posible, a través de la reflexión manipular los elementos de otras aplicaciones, de tal manera, que en tiempo de ejecución se conocen nombres, tipos y se pueden ejecutar métodos y funciones sin conocer previamente su nombre, tipo de valor devuelto, cantidad o los tipos de parámetros.
El traductor evalúa y compara los objetos del tipo "conceptos" y "acciones" que se tratan en las conversaciones de los agentes con los componentes donde se encapsulan los recursos, y realiza una sincronización entre ellos. Esta sincronización se realiza en tiempo de ejecución, cuando se produce un "acto de habla", un agente recibe un mensaje contentivo de los elementos ontológicos (concepto, acción, etc.), los evalúa y sincroniza con el componente indicado, ejecutando la acción descrita por el acto de habla.
Para encapsular los recursos es posible utilizar tecnologías como DCOM/COM (Distributed/Component Object Model), Java Beans, JNI (Java Native Interface), CORBA y XML (eXtended Meta Language), que permiten acceder al recurso exportando su interfaz a un lenguaje de definición común.
Así, en este nivel se define un esquema para acceder y lograr la comunicación con todos los recursos ubicados en las diferentes plataformas computacionales, de una manera estándar y transparente.

4. IMPLEMENTACIÓN

Para la fase de implementación se siguieron los requerimientos planteados en el diseño. Estos requerimientos están diseñados en función de las características fundamentales de los agentes, garantizando funcionalidades inherentes a los sistemas distribuidos. En este sentido se utiliza JADE[4] (Java Agent Development Enviroment), el cual es un entorno de desarrollo para Sistemas Multiagentes escrito en Java. Permite comunicación bajo el modelo RMI y IIOP-CORBA. Fue desarrollado por el TILAB (Telecom Italia Laboratory) y la Universidad de Parma, Italia. Se distribuye bajo licencia LGPL (Lesser General Public License)[1]. JADE ofrece un conjunto de bibliotecas de clases, que ofrecen servicios y funcionalidades para la construcción y administración de sistemas multiagentes. Entre las capacidades de la plataforma está la característica de interoperabilidad, heredada de Java, que permite acceder a plataformas heterogéneas. También ofrece clases para la gestión de ontologías y de comunicación, sistema de nombramiento, y de administración de agentes, y otras características importantes inherentes a los sistemas multiagentes.
JADE cumple con las especificaciones FIPA, en consecuencia se heredan las características de esta plataforma. Se implementa un modelo de comunicación basado en el lenguaje ACL; que utiliza ontologías y pase de mensajes para establecer conversaciones entre agentes. Además, es posible utilizar CORBA y protocolos para la Internet como el SMTP y HTTP.
Para la implementación de los agentes del MGS se utiliza un modelo basado en comportamientos. Los comportamientos son lógicas de acciones a seguir que pueden catalogarse en secuenciales, cíclicas, compuestas, paralelas, basadas en máquinas de estado, entre otras. Para cada agente se establece un tipo comportamiento,

[1] Esta tipo de licencia para código abierto puede accederse en el sitio web http://www.gnu.org

ejecutado en función de las variables de ambiente y la comunicación que el agente establezca con los demás agentes del sistema.
Los agentes AAA, AGA y AGR constituyen el núcleo del medio de servicios, ya que estos agentes controlan, supervisan, y modifican el estado global del sistema multiagente. Existen dos importantes servicios que brindan estos tres tipos de agentes, el primero es el control general de los agentes, que incluye creación, iniciación, migración, suspensión, destrucción, reactivación, clonación, nombramiento y búsqueda de agentes en todo el sistema. Estas tareas son coordinadas por el AAA, por lo cual todo agente debe tener comunicación con el administrador.
El segundo servicio es el de acceso a recursos de software y hardware, este servicio es administrado por los agentes AGA y AGR, y provee una interfaz basada en comunicación ACL (Agent Comunication Language). Para eso, se tiene una base de datos de componentes que envuelven el acceso a aplicaciones y recursos de hardware. Cuando es necesario integrar una aplicación se construye un componente J2EE (Java 2 Enterprise Edition) o COM y se agrega a la base de datos. El AGA posee una ontología para esta base de componentes que es compartida con los demás agentes. Para extender el acceso a recursos de hardware y de software se utiliza una interfaz JNI (Java Native Interface), que permite el acceso a componentes COM [11].
Para el acceso a recursos de hardware se cuenta con el AGR. Para cada nodo del sistema se tiene un AGR local que tiene el inventario de los recursos locales, además de que es quien administra y aplica algoritmos de balanceo de carga. Este agente posee una ontología que es utilizada para entenderse con los demás agentes del sistema.
El agente de base datos provee el servicio de acceso a datos. Los datos pueden ser accedidos desde distintas fuentes: servidores de base de datos, archivos de base de datos, archivos de texto o de hojas de cálculo, registros del sistema, etc. Para proveer heterogeneidad se utilizan los estándares JDBC para sistemas Java, y ADO para sistemas Windows.
Por último, el agente de control de comunicación es el encargado de establecer comunicación con otros sistemas multiagentes, posee una serie de protocolos que utiliza para traducir y establecer conversaciones.

5. CASO DE ESTUDIO

Para mostrar las características del MGS se utiliza una aplicación de optimización, que generalmente está incluida dentro del rango de aplicaciones de una plataforma de automatización.

5.1 Descripción del problema

En ambientes industriales, en ocasiones es necesario identificar procesos físicos que se realizan a diario en alguna de las plantas del complejo. Debido a que es difícil elaborar un modelo matemático exacto o aproximado siguiendo las leyes físicas del proceso, generalmente se plantea la construcción de un modelo basado en redes neuronales [3,5], que utilice datos históricos del proceso estudiado. Se tomó como ejemplo un reactor biológico, para el cual se tiene un conjunto de datos históricos de entrada y salida. Dado que uno de los objetivos del MGS es administrar eficientemente los recursos y aplicaciones de una plataforma de automatización, se toma ésta aplicación y se implementa utilizando el MGS. Para el ejemplo tratado se escogió una red neuronal de retropropagación[3,5]. La interfaz de usuario de está aplicación es un agente, por lo tanto posee todas las cualidades de los agentes del MGS. Al usuario se le muestra un cuadro de diálogo para que ingrese la ruta del archivo que contiene la configuración de la red neuronal y los archivos de datos (ver figura 3).

Figura 3. Cuadro de diálogo para entrenamiento de la red neuronal

En el archivo de configuración se especifican los nombres y tipos de variables de entrada y salida, además de los parámetros de entrenamiento, y en el archivo de datos se especifican en filas los patrones para entrenar la red:

```
// proceso.dfn
continuous entrada1u
continuous saliday
coef  0.1
momen 0.7

// proceso.dat    -------> Data
1.2 3
2.1 5
6.7 6
```

En el archivo de configuración de la red también se escriben los parámetros del entrenamiento, tales como el coeficiente de aprendizaje (*coef*) y el momento (*momen*). El coeficiente de aprendizaje es un valor numérico que ajusta el grado en que los pesos son modificados, por lo tanto, se dice que en este mismo grado la red aprende, por su parte, el parámetro "momento" permite a la red hallar en teoría, mejores errores.

5.2 Funcionamiento del MGS

Al presionar el botón "Aceptar", se inicia el proceso de gestión distribuida para el procesamiento de la información recolectada. En este proceso participan los agentes AGR, AAA; AGA, los agentes de la comunidad de agentes de control del SCDIA, y los recursos y aplicaciones que sean necesarios y estén disponibles para lograr los resultados esperados.

La aplicación se inicia realizando un proceso de negociación entre agentes. Este proceso tiene como finalidad seleccionar cuáles nodos de la red reúnen las condiciones para ejecutar las tareas de procesamiento, es decir, que nodos tienen los suficientes recursos de memoria principal y secundaria, procesadores, recursos gráficos, y conexión de la red, para ejecutar la aplicación asignada. En este proceso participa el AAA, y los agentes AGRs que se encuentran dispersos en los otros nodos de la red.

Por su parte, el AGA realiza el trabajo de elección del conjunto de agentes especializados que pueden procesar los datos y devolver los resultados esperados. Uno de los esquemas más utilizados para este tipo de procesamiento son las redes neuronales, pero existen otros esquemas tales como los árboles de decisión y métodos de aproximación que también pueden ser aplicados. Además de poder aplicar diferentes tipos de procesamiento, también es posible utilizar diferentes tipos de implementación, para el ejemplo, se utilizan dos tipos de implementaciones para redes neuronales: una desarrollada en Java y otra basada en un componente COM [11] que se comunica con MATLAB® (una aplicación comercial que provee herramientas para el cálculo matemático)[2]. Cuando se consulta las aplicaciones disponibles, el AGA sugiere estos dos agentes que están vinculados a dos aplicaciones: NNAgent y NNMatlabAgent. Luego del procesamiento, los resultados son devueltos al agente interfaz (UIAgent), y finalmente, mostrados al usuario (ver figura 4).

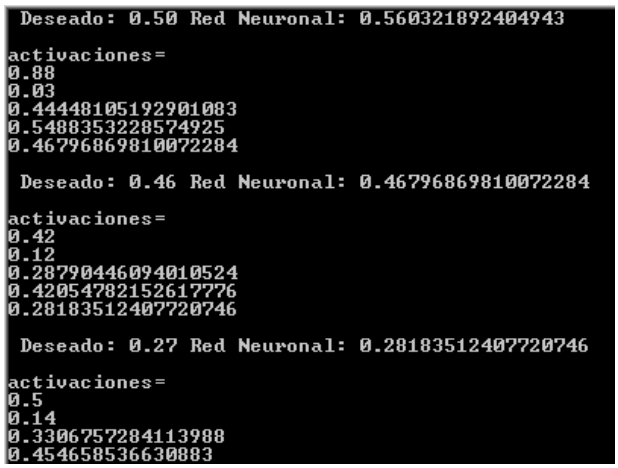

Figura 4. Resultados del entrenamiento con la red neuronal

6. CONCLUSIÓN

Aunque el modelo basado en agentes aporta líneas claras para la construcción de sistemas de software, se hace necesario tomar en cuenta el aspecto integracional. Así, la plataforma de agentes debe contar con un conjunto de servicios, propuesta del MGS, que permitan de manera eficiente la conexión con recursos y aplicaciones ya

[2] Para mayores detalles de esta aplicación puede consultar la dirección web: http://www.mathworks.com

existentes, una capa middleware que resuelva la integración, lo que es una herramienta indispensable para la implantación del paradigma de agentes en las plataformas computacionales actualmente operativas. Particularmente, para el SCDIA es vital el MGS ya que brinda integración entre cualquier plataforma computacional basada en componentes, servicios, u otro tipo de tecnología actualmente utilizada en la industria o empresa, y los agentes que componen al SCDIA.
El medio de gestión de servicios brinda características y funcionalidades en sus diferentes niveles. En el nivel interfaz se encuentran los servicios asociados con los agentes, que involucran las capacidades de comunicación, asociación y autonomía basada en modelos de inteligencia. En el nivel medio se cuenta con los servicios asociados con los sistemas distribuidos, tales como transparencia, seguridad, nombramiento, interoperabilidad, migración y mensajería. En el último nivel, o nivel de acceso a recursos, se cuenta con un modelo que permite integrar recursos heterogéneos, bajo una misma interfaz utilizando el concepto de reflexión, lo que permite un enlace y sincronización en tiempo de ejecución entre agentes y componentes. La ventaja de nuestra propuesta es que puede ser usado por cualquier otro SMA diferente al SCDIA, y puede ser incorporado sobre cualquier Sistema Distribuido.

Reconocimiento

Este trabajo fue financiado por el Proyecto 97003817: *"Esquemas generales basados en Sistemas Inteligentes para Control de Procesos"*, del programa Agenda Petróleo del CONICIT, y por el Proyecto I-620-98-02-AA: *"Gestión de Recursos en Redes de Estaciones de Trabajo mediante Sistemas MultiAgentes"* del CDCHT de la Universidad de los Andes.

Referencias

[1] Aguilar J., Cerrada, M. et al. Aplicación de Sistemas Multiagentes en Problemas del Mundo Real. *XXVII Conferencia Latinoamericana de Informática (CLEI).* Merida, Venezuela, 2001.

[2] Aguilar J., Cerrada, M. et al. Application of the Agent Reference Model for Intelligent Distributed Control Systems. in: *Advances in Systems Science: Measurement, Circuits and Control.* Edited by: N. Mastorakis and L.A. Pecorelli. World Scientific and Engineering Society Press, pp. 204-210, 2001.

[3] Bigus J., Jennifer B. Constructing Intelligent Agents using Java. *Wiley Computer Publishing*. 2001.

[4] Bellifemine F., Poggi A., Rimassi G. JADE: A FIPA-Compliant agent framework, *Proceeding Practical Applications of Intelligent Agents and Multi-Agents,* 1999. (http://sharon.cselt.it/projects/jade)

[5] Hagan M, Demut H., Beale M. Neural Network Design. *Publishing Company*. 1996.

[6] Iglesias C. Definición de una metodología para el desarrollo de sistemas multiagentes. *Tesis Doctoral.* Universidad Politécnica de Madrid. Enero 1998.

[7] Nillson, N. Inteligencia Artificial: Una nueva síntesis. *McGraw Hill.* 1998. España.

[8] Rivas, F., Aguilar, J. et al. Especificación Detallada de los Agentes del SCDIA, *Tercer Informe Técnico, Proyecto Agenda Petróleo No. 97003817,* Universidad de los Andes. 2003.

[9] Rusell S., Norvig P. Inteligencia Artificial: Un enfoque moderno. *Prentice Hall.* 1995. México.

[10] Tanenbaum A. Sistemas Operativos Distribuidos. *Prentice Hall.* 1ra Edición. 1995. México.

[11] Templeman J. Beginning MFC COM Programming. *Wrox Press.* 1997.

[12] Weiss G. Multiagent Systems. *MIT Press.* 1999.