Agentes de software: tecnologías, herramientas y aplicaciones*

Software agents: technologies, tools and applications

Olga Lucía Roa
Magíster, profesora tiempo completo Universidad de San Buenaventura Cali.
Integrante del Laboratorio de Investigaciones para el Desarrollo de la Ingeniería de Software LIDIS
olroa@usb.edu.co

Grupo de Investigación *Laboratorio de Investigaciones para el Desarrollo de la Ingeniería de Software-LIDIS*
Universidad de San Buenaventura Cali

Resumen

Los agentes de software, en su recorrido por distintos sistemas, tienen la posibilidad de obtener servicios ofertados localmente y dialogar con otros agentes. Estas y otras características de los Sistemas Multi Agentes (SMA) hacen que su diseño, implementación, mantenimiento, etc., sea una tarea poco trivial. Por ello es deseable contar con herramientas apropiadas que asistan las diferentes etapas de su construcción. El presente artículo pretende mostrar un estudio sobre las herramientas que existen en la actualidad para el desarrollo de los SMA, partiendo desde su definición y los servicios que brindan, hasta llegar al análisis de las tecnologías utilizadas en la actualidad. Este artículo termina formulando una hipótesis de las tendencias en construcción de software.

Palabras claves: Agentes de software, agentes móviles, FIPA (Foundation for Intelligent Physical Agents), FIPA-OS.

Abstract

During their route through different systems, software agents have the possibility of obtaining services supplied locally in the visited systems and engaging in a dialog with other agents. These and other characteristics of the Multi-Agent Systems (MAS) make their design, implementation, maintenance, etc. a non trivial task. For this reason, it is desirable to count on tools that support the stages involved in the construction of MAS. This article presents a study on the current tools for the development of MAS, starting from their definition and the services they offer until the analysis of the technologies and tools used nowadays. At the end of the article a hypothesis on the trends of software construction is made.

Key Words: Software agents, mobile agents, FIPA, FIPA-OS.

* Este documentohace parte del trabajo de investigación de la línea de *Computación móvil*, del Laboratorio de Investigaciones para el Desarrollo de la Ingeniería de Software - LIDIS.
Fecha de recepción: Julio de 2004
Aceptado para su publicación: Septiembre de 2004

Introducción

La ingeniería de software basada en agentes, ha surgido como respuesta a la necesidad de generar nuevos paradigmas que permitan modelar problemas de una forma cada vez más transparente. Ante esto, los sistemas multi-agentes (SMA) interactúan a fin de satisfacer estos objetivos.

Los agentes de software, al ser aplicados a una gran cantidad de sistemas de hardware y software, comparten una serie de características como la distribución de la información, el conocimiento parcial que tienen las entidades, la computación asincrónica y la ausencia de un sistema de control central.

Estas entidades computacionales persisten en el tiempo; es decir, que no terminan su ejecución cuando han finalizado una tarea, sino que continúan observando su entorno, decidiendo qué acción ejecutar en el próximo instante de tiempo.

El presente estudio pretende mostrar una evaluación de herramientas para el desarrollo de sistemas multi-agentes (SMA) y las tecnologías que utilizan internamente estos frameworks. La metodología utilizada es la aplicación de la deducción en la elaboración de hipótesis y la inducción en los hallazgos. Partiendo de un estado del arte se analiza la documentación en el tema de herramientas y tecnologías de agentes de software logrando deducir la hipótesis de que no existe una herramienta que cumpla completamente con los requerimientos ideales de un framework para el desarrollo de estos sistemas. En este trabajo se realiza un cuadro comparativo de las herramientas que existen para el desarrollo de estos sistemas y un resumen de sus principales características, ventajas, desventajas, servicios y aplicaciones. A continuación se expone la característica de movilidad y se dejan propuestos otros temas para futuros desarrollos.

Agentes móviles

En la arquitectura global de un sistema de agentes móviles se pueden observar las agencias, las cuales pueden estar configuradas en otras máquinas, conformando ambientes de ejecución. En cada una de estas agencias se publican los servicios que se ofrecen a las otras. Todas están dirigidas por un administrador (Ver Figura 1).

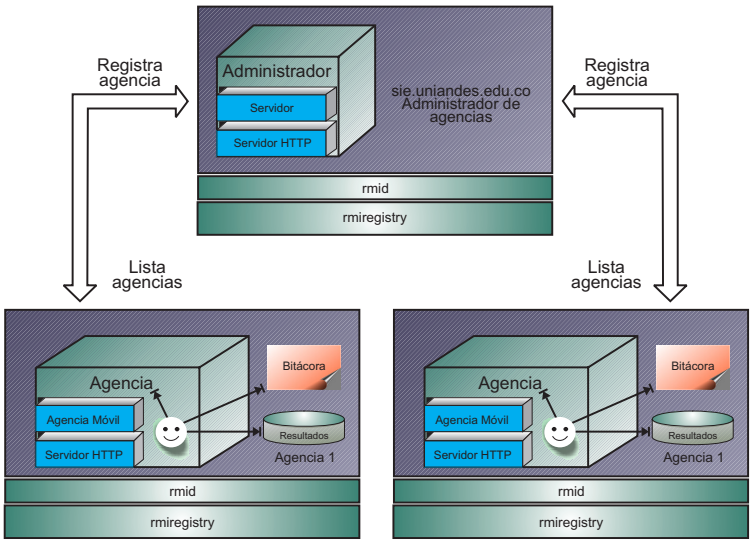
Un agente móvil posee los siguientes estados:

0: Estacionario. El agente no es móvil, no envía mensajes y envía un mensaje de no entendido (not-understood) si recibe cualquier mensaje. Si un agente es estacionario, estará todo su ciclo de vida en estado cero (0).

1: Salida. El agente es móvil. Cuando está en 1 se dice que ha inicializado por primera vez. Cuando lo hace envía inmediatamente una solicitud de mover. Entonces, el mensaje informe (inform) llega con el contenido del mensaje de movimiento (move), pasando al estado 2 y se cerrará.

2: Movido. El agente es móvil y simplemente ha llegado a su destino. Cuando es inicializado esperará el mensaje ejecutar

Figura 1
Arquitectura global de un sistema de agentes móviles



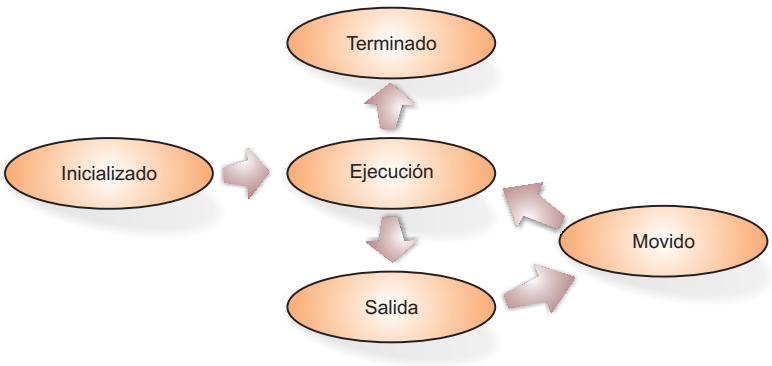
(execute), para realizar la búsqueda y luego envía una solicitud para el movimiento. Cuando el mensaje informe llega con el contenido de movimiento, este se moverá al estado 3 y se cerrará.

3: Terminando. El agente es móvil y simplemente ha regresado a la localización original. Cuando inicializó estaba esperando el mensaje execute, después despliega los resultados y se cierra (Ver Figura 2).

En esta arquitectura se trabaja un protocolo de movilidad simple. Según FIPA (Foundation for Intelligent Physical Agents), este protocolo se define así: El AMS (Agent Management System) es el responsable de realizar toda la gestión necesaria. El agente solicita la transferencia y el AMS se ocupa de llevarla a cabo.

El agente confía en un protocolo de alto nivel que ejecuta el movimiento a una plataforma destino en una sola acción.

Figura 2
Ciclo de vida de un agente móvil



En este caso, la plataforma tendrá que llevar a cabo el protocolo necesario para realizar la migración entera. Así, un agente delega la ejecución de la movilidad a la plataforma. Una de las principales ventajas percibidas en los protocolos de movilidad simples es que hay poca complejidad en el desarrollo de la aplicación de agentes ya que la movilidad es soportada por la plataforma; adicionalmente, existe un número reducido de interacciones remotas (GarcíaA98).

Servicios de una plataforma de agentes

La plataforma deberá ofrecer, aparte del soporte de creación, cuando menos tres servicios básicos:

Movilidad

La infraestructura deberá incluir una librería para ofrecer movilidad total. Es decir, el agente podrá enviarse libremente a todas las direcciones que hayan sido definidas en su itinerario. Además, deberá considerar y resolver los problemas que se originan con los servidores instalados en computadores portátiles o bien servidores dentro de un cortafuegos (*firewall*).

Los agentes al viajar pueden tener problemas en alcanzar sus servidores destino, por ejemplo, cuando estos están temporalmente desconectados de la red; sin embargo, pueden usar colas privadas, llamadas almacenes de agentes (*agent box*), que están localizados en servidores remotos, de donde se pueden recuperar cuando vuelvan a estar disponibles o se hayan conectado nuevamente a la red.

Un almacén de agentes permite su recuperación desde más de un servidor si se desea. Asimismo, provee una solución segura para su aceptación dentro de un cortafuegos o dominios de seguridad.

Localización

El servidor deberá proporcionar un servicio de búsqueda y localización de agentes, no sólo de servidores locales sino también remotos, lo cual es indispensable para que posteriormente los agentes puedan interactuar con los de otros sitios.

La localización facilita una comunicación remota. Por ejemplo, el propietario de un agente remoto podría enviarle un mensaje de control (eliminarlo) o personalizar su comportamiento enviándole otro itinerario. Otro ejemplo puede ser el caso en el que se envían múltiples agentes a distintos servidores con el objeto de ejecutar tareas en paralelo, los cuales pueden requerir comunicarse con el fin de intercambiar resultados parciales o sincronizarse.

El servicio de localización será básico para la comunicación y monitoreo del sistema de agentes.

Existen tres esquemas básicos para la localización de agentes.

- Secuencias de acceso: un agente es localizado siguiendo los rastros de información que va dejando al visitar cada servidor. Es decir, cada servidor guarda una referencia de todos los agentes que lo visitan así como el destino remoto al que partió.
- Fuerza bruta: un agente es localizado al preguntar por él en múltiples servidores re-

motos. La búsqueda puede ser hecha en paralelo o en secuencia.

– Registro en un servidor de búsqueda: un agente puede actualizar su localización en un servidor de nombres establecido que permita a los agentes buscar, registrarse o borrarse. De esta manera se puede localizar un agente específico.

Comunicación

El sistema permitirá la transferencia de mensajes síncronos y asíncronos no sólo entre agentes locales sino también entre agentes remotos. La comunicación les permitirá realizar una interacción entre ellos para lograr realizar tareas en conjunto.

De esta forma, por ejemplo, se podría implementar una búsqueda en paralelo, enviando a varios agentes al encuentro de una información deseada. Cuando un agente localice la información requerida o parte de ella, podría avisar al resto a través de un mensaje que suspendan su búsqueda, o bien que continúen colaborando con la información que falta, proporcionando, además, rutas alternativas a otros agentes.

El servidor debe cuidar que no se haga un uso excesivo del servicio para que no se sature la red de mensajes.

Razones para usar agentes de software

Según la expuesto por Danny B. Lange y Mitsuru Oshima, en su artículo *Seven good reasons for mobile agents* (Danny99), de las publicaciones de ACM, se puede concluir:

Últimamente se ha producido mucha documentación sobre el potencial de los sistemas multi-agentes. Aún hay ideas planteadas que si se desarrollan efectivamente pueden generar enormes beneficios. Un ambiente de información, altamente interconectado, interdependiente y heterogéneo, plagado por una explosión de servicios disponibles, necesita métodos y soluciones que lo hagan posible de forma dinámica. Los sistemas basados en agentes permiten esta interacción de una forma natural, utilizando programación orientada a objetos y computación distribuida. El nivel de abstracción es más alto que los objetos en la dimensión de entradas disponibles para comunicarse y razonar acerca de otros en un ambiente dinámico, cada uno se esfuerza por satisfacer sus propias metas. No es ninguna sorpresa que los agentes pueden tener sus raíces en campos como la Inteligencia Artificial Distribuida (Distributed Artificial Intelligence –DAI–) y La Computación de Apoyo de Trabajo Cooperativo (Computer Supported Cooperative Work –CSCW–), los cuales requieren nociones en orden para empezar a proveer soluciones a los problemas que estos presentan. Para administración de workflow de negocios y administración de redes de telecomunicaciones para comercio electrónico y casas inteligentes, el paradigma de los agentes puede ayudar a proveer soluciones efectivas.

A la pregunta de ¿por qué se está enfrentando un problema manejando el aumento en información y servicios? Las causas se encuentran en parte en el estilo de interacción

entre humanos y computadores. Actualmente el método es la manipulación directa, la cual es usada para describir la situación donde un usuario debe guiar explícitamente el computador en el desempeño de tareas por manipulación de objetos en la pantalla así como ventanas, botones, listas , boxes, etc. Estos son un avance para los paradigmas previos de instrucciones de líneas de comando, donde una causa y un efecto no están claramente relacionados y el usuario necesita memorizar instrucciones arcaicas (aún a través de algunos Unix). Sin embargo el incremento de tareas para desarrollar y su complejidad generan una interacción más dura. Sin embargo, con la ayuda de los agentes de software un nuevo estilo de interacción ha emergido y es llamado manipulación indirecta, en la cual el agente desarrollar tareas por sí mismo tal como monitoreo de eventos y reacciona a ellos en una forma apropiada.

El beneficio del paradigma de los agentes no para en el nivel de la interacción humano-computador.

Pensar en términos de agentes puede ayudar a modelar el problema en forma más intuitiva, generando mejores soluciones. Por ejemplo, en Suecia se está empezando a desarrollar un sistema multi-agente como respuesta al problema de balances de carga en una malla eléctrica. El proyecto llamado ISES (Information, Society, Energy and Systems), es un proyecto de unión entre la universidad y la industria el cual ayuda a mejorar la efectividad de los sistemas para comunicarse, donde las utilidades pueden ser manipuladas inteligentemente para ganar el mejor precio y proporciones de desempeño.

Este ejemplo muestra cómo el paradigma de los agentes da fáciles solución a problemas complejos, permitiendo brindar con sus propias herramientas las nociones de negociación multi-agente y cooperación, descentralización y una perspectiva para múltiples facetas. Pero el más poderoso elemento de las técnicas enfocadas en agentes es que estos brindan la unidad de análisis más cercana a la organización natural del mundo real, solucionando el problema en términos de entradas, donde cada una interactúa con su propia agenda para seguir por la configuración de sus metas e implementando planes basados en deseos y creencias del ambiente. Sin embargo, hasta ahora los desarrollos de sistemas basados en agentes se han enfocado más en construir infraestructuras propietarias que crear mecanismos para el descubrimiento, la comunicación o el movimiento de código de agentes en el caso de sistemas de agentes móviles.

Aplicaciones de agentes de software

Los agentes han demostrado ser una abstracción útil para diseñar sistemas distribuidos y cooperativos en muchas actividades industriales y de servicios, incluyendo las telecomunicaciones, el control de tráfico aéreo, la administración del transporte, el cuidado médico y el entretenimiento (JEN98).

Todas estas propiedades (autonomía, distribución geográfica, cooperación, aprendizaje y comunicación) hacen que los agentes de soft-

ware sean ideales para el desarrollo de un gran número de aplicaciones. A continuación se describen algunos proyectos donde se aplican agentes:

- MASCONTROL (Mascon04): este proyecto presenta un sistema multi-agentes para la identificación y control de procesos, se implementa el esquema de un regulador auto-ajustable y se cumple con el estándar FIPA (GarcíaA98). El sistema funciona tomando del entorno los valores de unas variables de ambiente y según unas condiciones determinadas como óptimas se efectúan unos cambios sobre ciertos factores.
- EVA "Espacios Virtuales de Aprendizaje" (EVA02). Este sistema tiene como meta el desarrollo de un aula virtual multi-agentes de enseñanza/aprendizaje cooperativo, esta aplicación, entre otras, se desarrolla para sistemas de educación virtual. Está basada en el framework JatLite de la Universidad de Stanford (JAT98).
- Agentes de información/Internet, como Jasper (Joint Access to Stored Pages with Easy Retrieval). Los agentes Jasper trabajan a favor de un usuario o comunidad de usuarios y son capaces de almacenar, recuperar, compendiar y utilizar otros agentes de información útiles a ellos para encontrar información en la www. Existen otros agentes de información realizados en particular para filtrado de información. Hay, también, agentes reactivos, utilizados principalmente en la industria del entretenimiento, en la programación de videojuegos y en la fabricación de juguetes.
- GUARDIÁN, agente que fue construido para el nicho de las UCI (Unidades de Cuidados Intensivos). Guardian es actualmente capaz de monitorear el estado de alrededor de 100 variables de estado de cada paciente, censándolas continuamente.

Los sistemas multi-agentes también se usan como asistentes para la búsqueda de cierta información, es así como el usuario define ciertos criterios y el agente permanece en una búsqueda constante, dicho sistema se puede programar para que envié resultados al correo electrónico (Hen05).

En *Agents group at the Massachusetts Institute of Technology Media Lab in Cambridge* (MIT05) se presentan otras aplicaciones de sistemas multi-agentes de software.

Tecnologías utilizadas para simular movilidad

Se pueden simular dos tipos de movilidad:

La movilidad de estado del agente, mediante paso de mensajes ACL: este modelo se implementa encapsulando el agente como el contenido de un mensaje ACL, enviándolo a su destino, en donde se crea una nueva instancia del objeto, se restauran los estados de los atributos y se retorna el resultado.

La segunda es la movilidad del código Java que corresponde al agente, esto se podría hacer mediante ClassLoaders, los cuales literalmente cargan las clases de Java; de esta manera se puede cargar un código Java desde diferentes máquinas en la red. Jeff Nelson, en su libro *Programming mobile objects with Java*,

define un objeto móvil "como aquel que se mueve entre dos o más aplicaciones"; por lo anterior, se puede observar que transmitir el estado de un objeto en Java no es complicado ya que se puede trabajar la serialización de los objetos y el paso de ellos mediante RMI.

En general, se puede simular movilidad mediante alguna de las siguientes tecnologías de comunicación, entre otras:

- RMI (Remote Method Invocation). Esta llamada a métodos remotos se basa en la comunicación mediante socket's, pero es más fácil de utilizar.
- JINI (su pronunciación es idéntica a "genie", genio, en inglés). Tecnología Java basada en RMI, que provee servicios, www.jini.org. Además hace posible la interacción entre dispositivos electrónicos provistos de microprocesadores.
- CORBA (Common ORB Architecture). Tecnología que permite comunicar componentes de software de diferentes lenguajes y plataformas.

El SMA se puede diseñar para que sea la agencia la encargada de hacer el paso del agente y este se ejecutará en la agencia de destino hasta que se le solicite ser enviado a otra agencia.

Herramientas para el desarrollo de sistemas multiagentes

La teoría de agentes establece una serie de mecanismos que pretenden dar un paso más allá en el tratamiento informático distribuido, añadiendo características como la localización o la situación, y permitiendo la interacción dinámica de componentes autónomos y heterogéneos.

El paradigma de agentes está centrado en conceptos de alto nivel de abstracción que permiten modelar complejos problemas de ingeniería como la administración de sistemas distribuidos de software, la recolección de información en la internet y en redes, la asistencia en procesos industriales y cooperativos de aprendizaje, entre otros. Es así como surge AUML para representar los diseños de sistemas multi-agentes y FIPA como una fundación que define su estándar e implementación, regulando las formas de comunicación, la arquitectura global de los sistemas, las reglas de negociación, las formas de manejar ontologías y bases de conocimiento. Por ejemplo, FIPA ha desarrollado el lenguaje ACL (*Agent Comunicación Language*), una evolución del lenguaje KQML (*Knowledge Query Manipulation Language*), tomado como estándar durante mucho tiempo para el intercambio de conocimiento entre agentes.

En la actualidad existen diversas herramientas para la construcción de agentes, de las cuales se pueden mencionar las siguientes:

- Aglets Software Development Kit (ASDK) de IBM, la cual presenta una interfaz gráfica identificada como una agencia, en la cual los agentes se ejecutan y existen; brindándole servicios de movilidad cifrando el código y los datos de un aglet (agente) utilizando el método de serialización de Java (JOS) y trasladando agentes utilizan-

do el Protocolo para el Transporte de Aglets (ATP). Adicional a esto, ofrece el servicio de mensajería y manejo de eventos como: creación de agentes, clonación, expedición, retractación, eliminación, activación, desactivación y paso de mensaje entre agentes (Aglets98).

– Aglets provee un control de seguridad mediante definición de autoridades, de sus privilegios y preferencias. La configuración de estas propiedades de seguridad es un poco complicada, ya que se hace por un archivo externo, mediante líneas de comando. Aunque Aglets provee una plataforma para la construcción de agentes de software, no es muy estable, según dicen sus autores, limitando los desarrollos adicionales que se trabajen sobre tal aplicación. Adicional a esto, no contempla el manejo de conocimiento dentro de los agentes (Aglets98).

– **Voyager ORB** es una plataforma para el desarrollo de agentes que se centra en el manejo de su movilidad, obviando las otras características propias de los mismos. En cuanto a seguridad, provee soporte para comunicaciones de red seguras sobre el protocolo estándar SSL, permitiendo la comunicación remota sobre un canal encriptado y autenticado. Además soporta *tunneling*[1] a través de *firewalls*.

  Posee soporte de transacciones a través de interfaces, que son fáciles de utilizar y aseguran la terminación completa de las mismas (usando "*two-phase commit*"). *Voyager* permite a múltiples recursos participar en transacciones a través de múltiples máquinas virtuales (Voyager02).

– JavaLog integra el lenguaje orientado a objetos Java y el lenguaje lógico Prolog. Esta combinación permite que los agentes sean construidos como objetos, manipulando un estado mental definido a través de cláusulas lógicas que son encapsuladas en módulos lógicos. Estos módulos permiten combinar dinámicamente actitudes mentales para adaptar el comportamiento de agentes considerando diferentes contextos o circunstancias. Este lenguaje de programación carece de una plataforma propia para su manipulación, por lo tanto se necesita manejar conceptos de programación en Java y Prolog. No provee mecanismos de comunicación entre agentes (Javalog99).

– FIPA-OS, como plataforma para el desarrollo de agentes, cumple con el estándar FIPA para el diseño e implementación de sistemas multi-agentes, así como permite desarrollos en Java. Está centrada en proveer mecanismos de comunicación entre agentes. Carece de la funcionalidad del manejo del conocimiento para lo cual se apoya en JESS (Java Expert System Shell) y del manejo de la movilidad de agentes (Mikko00). Este framework se puede integrar con CATNAgentToolkit para el diseño de comportamientos sociales, aunque es dispendioso establecer las relaciones necesarias.

1. Tunneling. El proceso de encapsulamiento, que corresponde a la parte virtual en una Red Privada Virtual, la cual consiste en ocultar la información del emisor (IP origen) y la del receptor (IP destino). Técnica que permiten establecer una comunicación segura

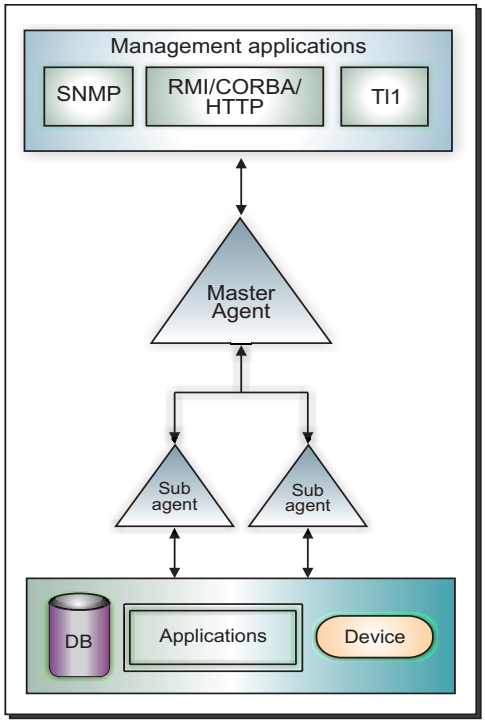
Entre las herramientas para la construcción de agentes evaluadas en este estudio, se puede decir que esta es una de las más completas y más favorables para trabajar, ya que cuenta con una documentación completa tanto para el desarrollador de sistemas multi-agentes, como para los desarrolladores que deseen realizar aportes al código existente por ser una aplicación de código abierto.

– JADE (*Java Agent Development Framework*), desarrollada en Italia, cumple con las especificaciones FIPA. Las aplicaciones de este framework se pueden ejecutar en un amplio rango de ambientes, entre ellos en dispositivos móviles (PDA's y celulares). Es de notar que esta aplicación, como la gran mayoría de las desarrolladas en Java, poseen librerías adicionales o se integran fácilmente con otras como JESS (Java Expert System Shell); esta librería permite representar el conocimiento heurístico de un experto humano.

Otras herramientas, las cuales tienen un valor comercial, son:

– Agent Toolkit Java Edition. Esta herramienta para desarrollar sistemas multi-agentes con acceso multi-protocolo, permite administrar información común a través de SNMP (Simple Network Management Protocol), RMI, HTTP, CORBA y TL1 (Protocolo definido dentro de la herramienta y basado en XML). Este software es uno de los más llamativos por permitir implementar el acceso por varios protocolos de una manera muy intuitiva y transparente (ver Figura 4).

Figura 4
Arquitectura de un sistema de agentes



Ventajas de los agentes móviles respecto a técnicas de sistemas distribuidos

– Los usuarios que utilizan computadores portátiles o con baja capacidad de proceso, los que no tienen banda ancha en la red o los que disponen de poco espacio de almacenamiento, pueden utilizar agentes móviles para solucionar sus problemas, desconectando su computador de la red o pidiendo que otra máquina realice los cálculos y recogiendo posteriormente los resultados.

– El usuario puede personalizar las respuestas de un servidor utilizando agentes móviles que interactúen con las interfaces para programación de aplicaciones API, exportadas por dicho servidor.

– Se flexibiliza la búsqueda semántica de información, ya que el usuario puede expresar sus necesidades en un lenguaje natural, enviar los datos a un agente consultor para que los traduzca al lenguaje utilizado por el sistema multi-agente y volver a enviar la petición con el formato adecuado a los servidores.

– Mejoran las posibilidades del comercio electrónico, ya que el usuario puede dar instrucciones a un agente para que realice la compra cuando se cumplan las condiciones necesarias. El proceso es más rápido y económico porque evita tener que hacer comunicaciones periódicas para comprobar el estado del producto.

Desventajas de los agentes móviles respecto a técnicas de sistemas distribuidos

– Es más rentable dotar de servidores representantes (proxy) a los usuarios de computadores portátiles.

– Las transacciones y peticiones semánticas pueden realizarse con otro tipo de técnicas obteniendo un resultado similar, ya que los servidores de información pueden restringir el acceso a los agentes ejecutados localmente.

– Hasta ahora no existe una metodología que permita convertir el código de un agente escrito en un lenguaje de programación determinado a otro lenguaje distinto.

– Los virus informáticos pueden contaminar los sistemas de agentes móviles y utilizarlos para reproducirse rápidamente.

– Existen deficiencias de seguridad en los modelos de agentes móviles.

Conclusiones

– Los sistemas de agentes móviles constituyen una serie de herramientas para la informática distribuida y heterogénea.

– Se supone que el esfuerzo realizado por FIPA para establecer un estándar, seguirá dando sus frutos y llegaremos a un estado en el que un agente podrá trasladarse sin dificultad a distintos entornos de operación, situados en máquinas con sistemas operativos diferentes. Por lo tanto, resulta importante recalcar que los programas y entornos comerciales deben ser compatibles al máximo con las normas de los estándares de FIPA.

– A futuro se espera contar con herramientas de desarrollo completas que faciliten la implementación de sistemas multi-agentes, permitiendo al usuario final enfocarse en los conceptos propios del negocio para el que se plantea implementar el sistema multi-agentes; olvidándose de los detalles e inconvenientes normales de la implementación.

– Se puede pensar que así como se pasó del paradigma de programación estructurada al paradigma de programación orientada a objetos, se pasará de este último al

paradigma orientado a agentes, debido a la transparencia que brindan para modelar el mundo real en un sistema computacional.

Bibliografía

- (GarcíaA98) GARCÍA ALONSO, Daniel. *Introducción al estándar FIPA*. Departamento de Sistemas Informáticos y Programación, UCM- Informe Técnico UCM-DSIP 98-00 - Versión 1.0
- (Aglets98) The Aglets Software Development Kit (IBM Corp.). En línea, 1998. Consulta: septiembre 2005. Disponible en: http://sourceforge.net/projects/aglets/
- (Voyager02) Voyager Application Server. En línea, 2002. Consulta: septiembre 2005. Disponible en: http://www.recursionsw.com/products/voyager/voyager.asp
- (Javalog99) JavaLog: una integración de objetos y lógica para la programación de agentes. Ramiro Iturregui, Alejandro Zunino, Analía Amandi en los Proceedings del V Congreso Argentino de Ciencias de la Computación (CACIC'99). En línea, octubre 1999, Tandil, Buenos Aires. Argentina. Consulta: septiembre 2005. Disponible en: http://www.exa. unicen. edu.ar/~azunino/index.php?page=zunino
- (Mikko00) Mikko Laukkanen. Evaluation of FIPA-OS 1.03, Helsinki. En línea, 16th February 2000. Consulta: septiembre 2005. Disponible en: http://fipa-os. sourceforge.net/docs/papers/soneraevaluation. pdf
- (FIPA-OS02) FIPA-OS V2.0.0. Reference FIPA-OS V2.0.0 Distribution Notes. En línea. 2002. Consulta: septiembre 2005. Disponible en: http://fipa-os. sourceforge.net/installation.htm
- (JEN98) N.R. Jennings, et. al.. ADEPT: an agent-based approach for to business process management, ACM Sigmod Record, 27: 32-39. En línea, 1998. Consulta: septiembre 2005. Disponible en: http://portal.acm. org/citation.cfm?id=306112
- (Hen05) Henry Lieberman, MIT Media Lab, Christopher Fry, Bow Street Software, Louis Weitzman, IBM, Why Surf Alone?: Exploring the web with reconnaissance agents. En línea. 2005. Consulta: septiembre 2005. Disponible en: lieber.www.media.mit.edu/people/ lieber/Lieberary/Letizia/Why-Surf/Why-Surf.html
- (Mascon04) E.J. González. et. al. Mascontrol: Sistema multiagente para la identificación y control de sistemas. En línea, XXV Jornadas de Automática, Ciudad Real, del 8 al 10 de Septiembre de 2004. Tenerife. Consulta: septiembre 2005. Disponible en: www.isa-cr.uclm.es/xxvjornadas/ ConfMan_1.7/SUBMISSIONS/6-velezgoose.pdf
- (MIT05) MIT, Software Agents group at the Massachusetts Institute of Technology Media Lab in Cambridge. Consulta: septiembre 2005. Disponible en:<http://agents.media.mit.edu/publications.html>
- (JAT98) JATLite Beta Complete Documentation (1998), Stanford University. En línea. Consulta: septiembre 2005. Disponible en: http://Java.stanford. edu
- (EVA02) SHEREMETOV, Leonid; NÚÑEZ, Gustavo; GUZMÁN, Adolfo. Tecnologías de inteligencia artificial y de agentes computacionales en la educación: el proyecto EVA, Centro de Investigación en Computación, IPN. En línea. Consulta: septiembre 2005). Disponible en: http://www.dirinfo.unsl.edu.ar/~profeso/PagProy/articulos/chiariani2.htm
- NELSON, Jeff. *Programming mobile objects with Java.* Editorial Wiley Computer Publishing. Año. 1999.
- CORBA (norma genérica para la comunicación y la interacción de sistemas basados en objetos). www.informatica.us.es/~ramon/tesis/CORBA/Seminario-MASIF/
- (Danny99) DANNY B. Lange y Mitsuru Oshima. *Seven good reasons for mobile agents.* Communications of the ACM, 42(3):88-89,. Artículo de la ACM. En línea. March 1999. (Consulta: septiembre 2005), Disponible en: http://www.acm.org/pubs/citations/journals/cacm/1999-42-3/p88-lange/