Agentes Móviles

Lic. Aldo R. Valdez Alvarado
aldo_valdez@hotmail.com

RESUMEN

En el presente artículo se intenta dar una visión general de los Agentes Móviles desde la perspectiva de los Sistemas Distribuidos y la Inteligencia Artificial, como una forma de mostrar como la aplicabilidad de esta ultima sobre los Sistemas Distribuidos.

Palabras Clave

Agentes Móviles, Agentes de Software, Agentes Inteligentes, Agentes Móviles, Consistencia, Cliente/Servidor, Marshalling, Redes, Sistemas Distribuidos, Transparencia, Unmarshalling.

1. INTRODUCCIÓN

El uso masivo de las redes de computadoras y los problemas aun existentes en los sistemas distribuidos, permiten la aparición de los *Agentes Móviles* a través de la *Programación Remota* que representa un nuevo paradigma de programación para el desarrollo de aplicaciones en red, alternativo a los modelos típicos *Cliente/Servidor*. En determinadas aplicaciones estos agentes móviles representan una solución ideal debido a que, permiten la reducción de la carga de tráfico por la red, mejoran la latencia de la red, eliminan potenciales limitaciones físicas en la máquina del cliente. Además los agentes móviles tienen como principal característica, la delegación de tareas que requieren la visita de diversas plataformas o entornos de ejecución por la red, por eso estos tienen la capacidad de operar asíncrona y autónomamente del proceso que los creó. Por todo lo anterior, un agente móvil puede ejecutar sus tareas aun cuando la conexión con la red no esté operativa; es decir, si el agente necesita trasladarse o volver y la red no está activa, el agente puede esperar o desactivarse hasta que la conexión se restablezca. Por consiguiente, la conexión se puede cortar y el agente móvil puede seguir trabajando, cosa que no ocurre en los modelos cliente-servidor. Todo ello, hace que los agentes móviles sean particularmente idóneos en conexiones frágiles y costosas o con un ancho de banda reducido.

2. SISTEMAS DISTRIBUIDOS

Desde la aparición de los primeros ordenadores comercialmente disponibles halla por los años 50, los grandes ordenadores centralizados multiusuario por los años 60 y 70, el desarrollo de los potentes microprocesadores que dieron lugar a la aparición de la PC o Computadora Personal por los años 80; estos siempre han presentado ciertos problemas, ya sean los relacionados a los costos elevados, o su poco aprovechamiento en términos de rendimiento, o la posibilidad de administrar otro tipo de recursos (como por ejemplo impresoras o procesadores especializados). En ese contexto aparecen la redes de computadoras de alta velocidad, y la solución a esto ultimo con la unión de los computadores personales y las redes LAN, que permiten la aparición de los *Sistemas de Red.* El siguiente paso fue el de los *Sistemas Distribuidos*. Con estos sistemas, los usuarios pueden saber que hay multiplicidad de estaciones y equipos en la red, pero no necesitan conocerlos explícitamente (ni cuántos, ni cuáles), ellos solamente saben que hay recursos en la red y conocen su nombre o identificador, de tal forma que pueden acceder a ellos de igual manera que acceden a los recursos locales (por su nombre), sin tener que conectarse (explícitamente) en ningún caso a la máquina propietaria para utilizar el recurso. Es más, ni siquiera necesita conocer el nombre de la máquina que alberga el recurso. Como podemos ver, la diferencia fundamental entre los sistemas en red y los sistemas distribuidos es la *transparencia*. En el sistema distribuido, la composición y estructura de la red le pasa desapercibida al usuario, es decir, ni la ve ni le importa. Solamente le importan los recursos disponibles o, a veces, simplemente el tipo de los recursos disponibles, sin tener en cuenta en qué máquina están realmente ubicados.

2.1 Definición

Aunque el concepto ya tiene bastantes años, todavía está en completo desarrollo, y hay distintas ideas y opiniones sobre lo que debe ser un sistema distribuido. Una de las definiciones más populares es la de Lamport: *"Un Sistema Distribuido es aquel que deja de funcionar cuando se estropea una máquina de la que ni siquiera habíamos oído hablar"*.

Más formal es la definición que cita Tanenbaum: *"Se trata de un conjunto de ordenadores independientes e intercomunicados por una red y provistos de software distribuido, tal que el usuario lo percibe como si se tratara de un único ordenador"*. Como ejemplo se tiene la red de redes como lo es Internet, ver la siguiente figura.

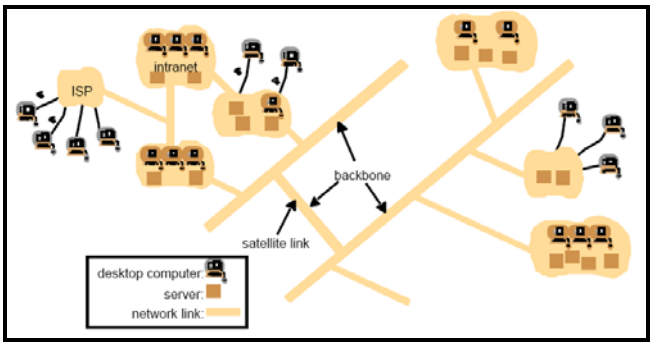

Figura 1. Estructura Típica de Internet

También tenemos a Coulouris [5] con la siguiente definición: *"Un sistema distribuido es aquel en el que los componentes localizados en computadores, conectados en red, comunican y coordinan sus acciones únicamente mediante el paso de mensajes"*.

Lo que debe quedar claro es que el *objetivo primordial* de los sistemas distribuidos es el *compartimiento fácil y eficiente de los recursos* entre múltiples usuarios.

Un sistema distribuido se puede definir como un conjunto cooperante y transparente de componentes de hardware y software remotos que se comunican entre sí mediante el intercambio de mensajes y que presenta dos características fundamentales:

- Está basado en una arquitectura en dos y tres niveles.

- Define un servicio particular ofrecido en una red de computadoras.

2.2 Características de los Sistemas Distribuidos

Un sistema distribuido puede verse como un sistema formado por varios ordenadores haciendo algo conjuntamente, de lo que se desprenden tres características inmediatas:

- Compuesto por múltiples ordenadores.
- Hay interconexión entre ellos.
- Tienen un estado compartido.

Entre algunas otras características que se consideran fundamentales tenemos [7]:

a) Consistencia. Un sistema distribuido es un único sistema formado por múltiples máquinas independientes. Ya que es un único sistema, debe tener un único estado global compartido por todos los equipos que lo componen. El estado global se refiere a datos: como tablas de mantenimiento del sistema, la hora actual o datos que estén compartidos por procesos de distintas máquinas.

b) Transparencia. La construcción de un sistema distribuido que se comporte según esperan los usuarios, requiere la consideración de otras características, que en principio diremos que son deseables tales como: Compartimiento de Recursos, Sistema Abierto, Escalabilidad, Tolerancia a Fallos y Seguridad. Un sistema que reúna todas estas propiedades conseguirá entonces la principal característica deseada por el usuario: la *TRANSPARENCIA, e*s decir, su sistema distribuido va a comportarse de manera ideal sin que el usuario se aperciba en ningún momento de los posibles y normales problemas que pueden producirse en el sistema (en las máquinas que lo componen).

2.3 Modelos de Cliente/Servidor para el desarrollo de Aplicaciones Distribuidas

Para el desarrollo de aplicaciones distribuidas o en red según el modelo cliente-servidor, existen los siguientes modelos [4]:

- Modelos basados en llamadas a funciones, Sockets.
- Modelos basados en llamadas a procedimientos remotos, RPC (Remote Procedure Call).
- Modelos basados en llamadas a objetos distribuidos, RMI (Remote Method Invocation), JINI, CORBA, COM+ (Common Object Model Plus), Enterprise Java Beans.
- Modelo basado en servicios Web distribuidos, es una tecnología que, a su vez, utiliza los siguientes estándares, XML, SOAP, WSDL, UDDI.
- Modelos basados en agentes móviles.

3. AGENTES DE SOFTWARE

3.1 Definición de agentes

La palabra agente deriva de otra, robota [9] (o robot), de origen checo cuyo creador fue el escritor checo Karel Kapec que usó dicha palabra en una novela (Opilek) y posteriormente en una obra de teatro (Rossum Universal Robots). Aunque durante siglos han existido autómatas de muy diversos tipos, especialmente como mecanismos de relojería y ha habido alguna obra influyente al respecto (Julián La Mettrie: L´Homme Machine) no es hasta la segunda guerra mundial y el desarrollo de las computadoras cuando empieza a cobrar importancia todo lo relacionado con los agentes o robots, especialmente de la mano del padre de la cibernética, Norbert Wiener. Sin embargo, el término agente, tal y como se entiende actualmente, procede de los trabajos comenzados en 1950, fundamentalmente, por parte del padre de la inteligencia artificial, John McCarthy, el cual definió un agente como un programa robot viviendo y trabajando en una computadora e interactuando con el exterior en términos próximos al modelo humano. Quiere decir esto, que la primera disciplina informática en estudiar el mundo de los agentes, o los sistemas compuestos por múltiples agentes, fue la inteligencia artificial y más en concreto la inteligencia artificial distribuida [6] (DAI); la cual se divide, a su vez, en dos grandes áreas: Resolución distribuida de problemas (DPS) y sistemas multiagente (MAS). Conviene resaltar que DAI se centra principalmente en la investigación de agentes estáticos o fijos en una máquina sin capacidad de moverse por la red.

Aunque el término agente no es un término nuevo, no hay todavía una definición estandarizada y aceptada por todos. A la hora de definir un agente, existe una clara controversia entre los investigadores en función de sus características.

3.2 Características de los Agentes

Podemos reconocer dos tipos de características: las fundamentales y las adicionales.

- Características fundamentales:
  - ✓ Autonomía: Independencia de actuación, es decir, sin sufrir la intervención directa de nadie.
  - ✓ Flexibilidad:
    - Reactividad-Adaptabilidad: Perciben, se adaptan al entorno y actúan en él.
    - Proactividad: Capacidad de tomar la iniciativa.
    - Comunicabilidad – Sociabilidad: Acceso a recursos y capacidad de interacción con otras entidades.
  - ✓ Continuidad Temporal: Proceso en continua ejecución.
- Atributos adicionales:
  - ✓ Movilidad (Agentes móviles): Migrar, interactuar y regresar bajo su propio control.
  - ✓ Capacidad de Razonamiento/Aprendizaje: Inteligencia.
  - ✓ Seguridad/Confiabilidad: Mecanismos y servicios de seguridad para evitar sorpresas por parte del agente representante y/o de otras entidades.

En función de lo anterior, cuando se define un agente se establecen dos tipos de visiones:

- Visión amplia - Atributos mínimos:
  - ✓ Autonomía.
  - ✓ Flexibilidad:
    - Reactividad-Adaptabilidad,
    - Proactividad
    - Comunicabilidad-Sociabilidad
  - ✓ Continuidad Temporal
- Visión estricta.- Atributos opcionales añadidos a los mínimos o básicos.
  - ✓ Movilidad
  - ✓ Capacidad de Razonamiento-Aprendizaje
  - ✓ Seguridad-Confiabilidad

Según se muestra en la siguiente figura, la definición más completa de un agente sería aquélla que junte los atributos básicos de la visión amplia con los atributos opcionales de la visión

estricta. En definitiva, esta definición se correspondería con todos los atributos que en teoría debe disponer un agente móvil, inteligente y seguro.

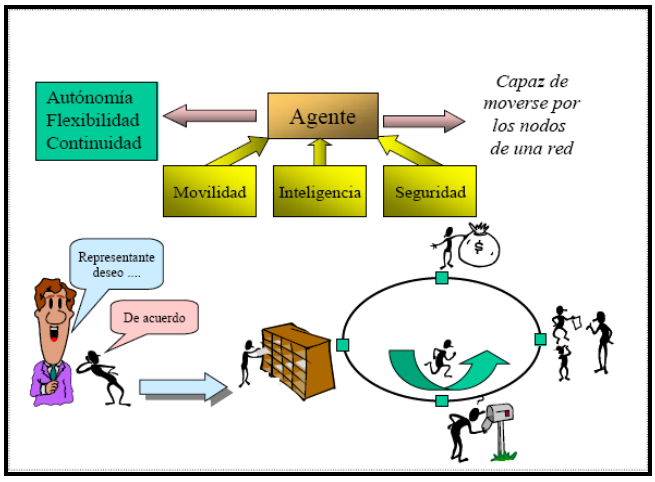

Figura 2. Un Agente móvil, inteligente y seguro.

3.3 Tipos de Agentes

Si la definición de un agente es algo compleja, su clasificación también lo es ya que la taxonomía de los agentes es muy amplia y variada. Para una mayor comprensión, este artículo clasifica los agentes de software desde la perspectiva de los sistemas distribuidos, en dos grandes tipos y con las siguientes características:

- ✓ Agentes estáticos:
  - Ejecución limitada al sistema donde se inicia.
  - Interactúan con entidades locales: agentes, programas, usuarios.
  - Accesos a recursos remotos vía sockets, RPC, RMI, CORBA, etc.
- ✓ Agentes móviles:
  - Capacidad de movimiento: Migración.
  - Interacción directa con entidades remotas: Programación Remota.
  - Transportan un mensaje: Estado (datos) y código (proceso o clase).

4. AGENTES MÓVILES

Un agente móvil es un proceso situado en una plataforma o entorno de ejecución [1] para actuar en representación básicamente de una persona u organización, y llevar a cabo una tarea para la cual ha sido especialmente designado. Asimismo, dispone de un conjunto más o menos amplio de atributos entre los que destaca la movilidad en el sentido de que no está limitado al sistema donde comienza la ejecución sino que tiene la completa libertad para viajar, es decir, migrar, interactuar y regresar bajo su propio control.

Como se muestra en la siguiente figura, los agentes móviles son funcionalmente objetos de software [3] que transportan en un mensaje su estado y código para poder interactuar remotamente con otros agentes o recursos. Para que el agente móvil pueda interactuar con otros agentes o recursos afines es necesaria la definición de un protocolo que especifique el formato de los mensajes y las acciones que se han de efectuar. En el mensaje transportado por el agente móvil se incluye un:

a) *Estado*: Como su nombre indica, contiene el estado del objeto o los valores de los atributos del agente y el estado de la máquina o estado de ejecución (contador de programa, punteros de pila, registros, etc.).
b) *Código*: Incluye la clase con los métodos de interacción permitidos.

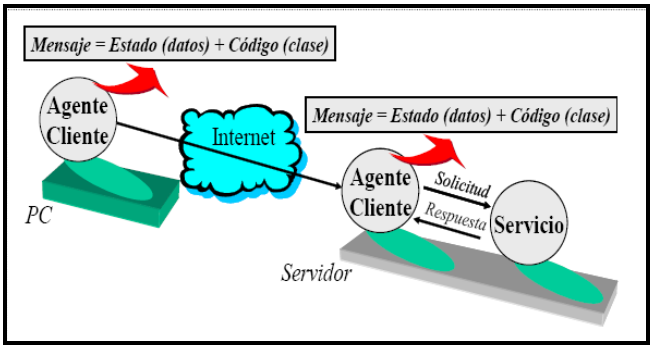

Figura 3. Agentes como Objetos

Como ya se ha indicado, cuando un agente se mueve, realiza una migración suspendiendo la ejecución, transportándose a otra plataforma en la red y reanudando la ejecución en el punto en que se suspendió. En este contexto, existen dos tipos de migración que se diferencian por el transporte de más o menos información o datos de estado:

- Migración fuerte: Transporta el estado del objeto (valores de los atributos del agente) y estado de la máquina o estado de ejecución (contador de programa, punteros de pila, registros, ...). Es la "foto completa", justo cuando se suspende la ejecución para su posterior reanudación en el entorno de otra máquina.
- Migración débil: Transporta sólo el estado del objeto (valores de los atributos del agente). Es la "foto incompleta" que permite sólo reiniciar la ejecución con los valores actuales de los atributos.

Con respecto al transporte de la otra parte del mensaje de un agente móvil, es decir, el código de la clase, se presentan tres posibilidades:

A. El código se encuentre disponible en la plataforma de destino ya sea en la memoria caché de clases o en el sistema local de ficheros. Consecuentemente, no es necesario transportar dicho código en el mensaje del agente.
B. El código se encuentre disponible sólo en la plataforma de origen. Este es el caso más frecuente y, por tanto, es necesario su transferencia en el mensaje del agente con la lógica penalización del tráfico de la red.
C. El código se encuentre en un servidor de clases. Se procede a recuperarlo bajo demanda desde la plataforma de destino.

Llegados a este punto, es importante distinguir ahora entre objetos y agentes móviles [2]. Aunque los agentes son funcionalmente objetos, existen, entre otras, las siguientes diferencias:

- Un objeto no es un agente móvil ya que un agente móvil es más que un objeto, es decir, es autónomo, flexible, etc.
- Un objeto no tiene autonomía sobre su comportamiento a la hora de ejecutar o no un determinado método.

- Un objeto no es flexible, es decir, no combina un comportamiento reactivo, proactivo y sociable.
- Desde un agente móvil no se invoca un método sino que se realiza una solicitud de servicio que el agente llevará a cabo o no.

Se recuerda que la migración de un agente consiste básicamente en suspender la ejecución, transportarse a otra plataforma en la red y reanudar la ejecución en el punto en que se suspendió. Así en el lado del emisor (plataforma de origen) se suspende la ejecución y se serializa al agente, es decir, se serializa su estado y código en un determinado formato de serialización o de transferencia, es decir, en un flujo de octetos (stream) transferible capaz de ser transportado por la red y restaurado en el lado receptor (plataforma de destino). Seguidamente, se efectúa el marshalling, es decir, se toman los datos serializados y se codifican en un formato o sintaxis común de codificación, representación y transferencia. Finalmente, se transmite al agente, por ejemplo, vía sockets, RPC, Java RMI, CORBA IIOP, etc. A su vez, en el lado del receptor (plataforma de destino) se realiza el proceso contrario o unmarshalling, es decir, se decodifican los datos serializados para luego deserializar los octetos de dichos datos al formato nativo de la nueva máquina; con el objetivo final de poder reiniciar (migración débil) o reanudar (migración fuerte) la pertinente ejecución. Para finalizar con esta introducción y descripción de generalidades sobre agentes móviles sólo indicar que se pueden aplicar los agentes móviles en muchos escenarios como por ejemplo:

- Recopilación de información de estado:
  - ✓ El agente móvil recoge información diseminada por la red.
  - ✓ Se establece un itinerario para que un agente móvil viaje secuencialmente por una red de área local recopilando, por ejemplo, información sobre el estado de almacenamiento de los discos duros de las máquinas servidoras de la organización.
- Búsqueda y Filtrado de Información:
  - ✓ El agente móvil parte de un conocimiento de las preferencias del usuario y elimina la información irrelevante.
- Monitorización de Noticias:
  - ✓ El agente móvil se envía para que esté a la escucha de ciertos tipos de información y recoja una determinada información diseminada por la red a través del tiempo
- Copia o mirroring:
  - ✓ El agente móvil realiza una copia inteligente de una información que se actualiza a determinadas horas del día o la noche.
- Diseminación de la Información:
  - ✓ El agente móvil lleva a cabo una distribución interactiva de noticias o publicidad a grupos interesados.
- Negociación:
  - ✓ Los agentes móviles negociadores efectúan una planificación de reuniones en nombre de sus representados.
- Comercio Electrónico:
  - ✓ El agente móvil localiza productos y precios para, finalmente, llevar a cabo una comparación, negociación y compra.
- Entretenimiento:
  - ✓ El agente móvil se programa con una determinada estrategia para su posterior envío a un servidor de juegos. Opcionalmente, se puede establecer todo un itinerario de servidores de juegos.
- Subastas:
  - ✓ El agente móvil se programa con una determinada estrategia para su posterior envío a un servidor de subastas. Opcionalmente, se puede establecer todo un itinerario de servidores de subastas.

4.1 Sistemas de Agentes

Un sistema de agentes (SA) o agencia es una plataforma o entorno de ejecución de agentes que permite, fundamentalmente, la creación, suspensión, terminación y ejecución inicial de un agente o retomar su ejecución, posibilitando la comunicación entre agentes y sus transferencias.

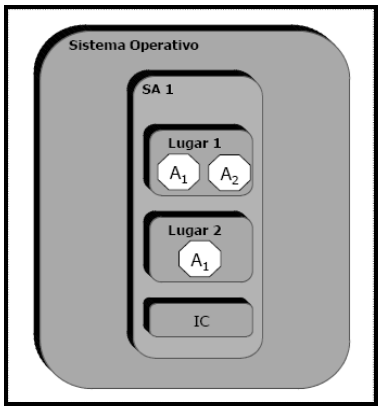

Figura 4. Sistema de Agentes en una Máquina

Por tanto, un sistema de agentes es responsable de efectuar las siguientes funciones:

- ✓ Coordinar todas las acciones entre agentes.
- ✓ Garantizar la seguridad del sistema interactuando con otras agencias y evitando la llegada de agentes procedentes de agencias maliciosas.
- ✓ Permitir la asociación con un representado (autoridad) que identifique a una persona u organización.
- ✓ Disponer de un tipo que, a su vez, describa parte del perfil de un agente. El perfil compleo de un agente queda definido por los tres siguientes parámetros: Tipo del SA, lenguaje y método de serialización.
- ✓ Disponer de uno o varios lugares (entornos específicos de ejecución) en función de los intereses y objetivos de los agentes).
- ✓ Disponer de un servicio de nombres que permita encontrar agentes y lugares locales. En este contexto, cada agente dispone de un identificador global y único formado por la concatenación de las siguientes informaciones: Autoridad representada, SA, lugar, un valor o identificador y la dirección IP de la máquina.
- ✓ Disponer de una infraestructura de comunicaciones (IC) que ofrezca los pertinentes servicios de transporte entre los SA.

El hecho de que los agentes móviles no actúan de forma aislada se puede ver todavía de una forma más clara a través del concepto de región que permite a una autoridad estar representada por uno o más SA. Una región contiene uno o más SA con la misma autoridad, pero no necesariamente con el mismo tipo de sistema de agente.

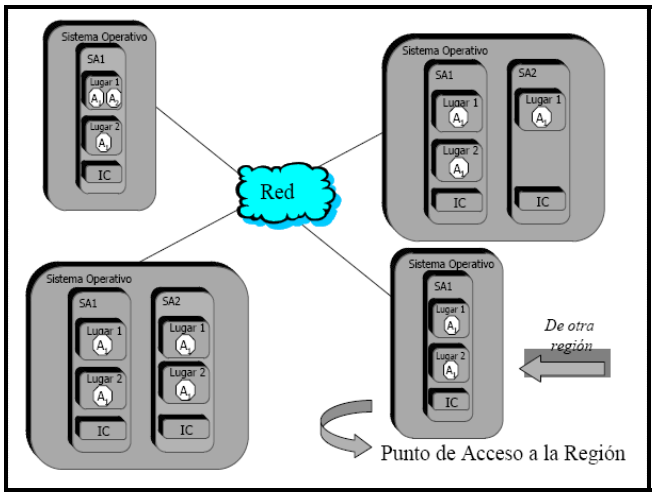

Figura 5. Relación Agentes y Regiones

Asimismo, una región es responsable de lleva a cabo las siguientes funciones:

- ✓ Facilitar la gestión de los componentes distribuidos (los SA, lugares, agentes).
- ✓ Mejorar la escalabilidad (mejor distribución de carga a través de múltiples SA).
- ✓ Mejorar el control de acceso y la seguridad.
- ✓ Disponer de un servicio de nombres al cual se conectan los servicios de nombres de cada SA.
- ✓ Interconectarse con otras regiones a través de los llamados puntos de acceso a la región (SA externas), los cuales pueden compartir un mismo servicio de nombres (previo acuerdo entre sus autoridades). Un punto de acceso a la región es otro SA que mantiene información de todos los SA de la región.

4.2 Lenguajes de Programación para el desarrollo de Sistemas de Agentes

En un principio se puede utilizar cualquier lenguaje de programación aunque no esté específicamente diseñado para el desarrollo de los agentes móviles y sus plataformas o entornos de ejecución (los SA). Este tipo de lenguajes obliga al programador a desarrollar al agente móvil desde cero teniendo que incluir aquellas funciones, operaciones y comandos que soporten las características de la teoría de agentes que desee añadir. En este contexto, se destacan los siguientes lenguajes:

- ✓ Lenguajes de propósito general: Java, C, C++, etc.
- ✓ Lenguajes interpretados y procedimentales basados en comandos (scripts): Tcl, Phyton, Perl, etc.

Si no se desea utilizar un lenguaje de propósito general o procedimental basado en scripts, se puede utilizar, como alternativa ideal, un lenguaje especialmente diseñado para el desarrollo de agentes móviles, es decir, que soporte la programación orientada a este tipo de agentes. Estos últimos lenguajes se caracterizan porque son interpretados y orientados a objetos y disponen de todas aquellas funcionalidades y capacidades integradas para dar soporte a las principales características de los agentes móviles como son la autonomía, migración, reactividad, proactividad, seguridad, etc. Un ejemplo de esto último, es el lenguaje Telescript similar a Java y que fue el primer lenguaje en ser especialmente diseñado y utilizado para desarrollar el primer sistema de agentes del mismo nombre. Sin embargo, en la actualidad la mayoría de los sistemas de agentes (Aglets, Mole, Odissey, Voyager, Grasshopper, ...) se basan en el lenguaje Java ampliado con nuevas clases para soportar características o atributos de la teoría de agentes móviles.

4.3 Lenguajes para la comunicación entre Agentes Móviles

Igual que se ha dicho para los lenguajes de desarrollo de agentes móviles, lo mismo se puede decir para el caso de la comunicación entre éstos en los correspondientes entornos de ejecución. Teniendo en cuenta que el modelo de comunicación de los mensajes es por paso de mensajes; en principio, se puede utilizar cualquier lenguaje de programación aunque no esté específicamente diseñado ni para el desarrollo ni la comunicación entre los agentes móviles. Por consiguiente, el programador debe implementar aquellas clases, funciones, procedimientos, comandos, etc., que permita crear, enviar y recibir los datos de los correspondientes mensajes. En este contexto, se destacan los siguientes lenguajes:

- ✓ Lenguajes interpretados y procedimentales basados en scripts: Tcl, Pitón, Perl, etc.
- ✓ Lenguajes orientados a objetos: Java ampliado, Telescript, etc.

En este contexto, conviene reseñar que existen los siguientes tres esquemas para el paso de mensajes entre agentes móviles:

1. Paso síncrono de mensajes (parada y espera): El emisor bloquea su ejecución cada vez que trasmite un mensaje hasta que el receptor responde al mismo. Este esquema de parada y espera es el más popular y, por tanto, el más usado.
2. Paso asíncrono de mensajes basado en dos envíos: Ahora, el emisor no tiene que esperar hasta que el receptor responda y envíe la respuesta. Este esquema es más flexible y especialmente útil cuando múltiples agentes interactúan con uno. Para su correcta implementación es necesario mantener en el emisor un identificador del mensaje enviado para no tener que bloquear la ejecución actual e ir preguntando en determinados momentos si hay una respuesta para el mensaje transmitido.
3. Paso asíncrono de mensajes basado en un envío: Por ser un esquema asíncrono, al igual que en el esquema anterior, no se bloquea la actual ejecución. Ahora, el emisor no retiene el identificador del mensaje y el receptor no tiene que responder. Obviamente, este esquema es muy útil cuando el agente que transmite un mensaje no espera la pertinente respuesta.

Si no se desea utilizar un lenguaje orientado a objetos o procedimental basado en comandos (scripts), se puede usar, como alternativa ideal, un lenguaje especialmente diseñado para la comunicación entre los agentes móviles. Estos lenguajes se conocen como declarativos o próximos al modelo humano en el sentido de estar basados en sentencias declarativas. Entre dichos lenguajes se destacan los siguientes:

- ✓ KQML (Knowledge Query Management Language) de DARPA; sintaxis normalizada de comunicación entre agentes de sistemas heterogéneos.
- ✓ FIPA; sintaxis similar a KQML

Se destaca, que estos dos últimos lenguajes están más pensados para la comunicación entre agentes estáticos típicos en el contexto de la Inteligencia Artificial que para la propia comunicación entre agentes móviles.

5. SISTEMAS DE AGENTES

Seguidamente, se describen brevemente algunos, de los muchos, SA implementados en la actualidad:

1. TELESCRIPT . Diseñado por James White (1994) en la empresa General Magic, Incorporated (California): http://www.genmagic.com. Fue la primera plataforma comercial (SA) para el desarrollo y ejecución de agentes móviles, primera en el desarrollo de un comercio electrónico basado en agentes móviles. Basada en conceptos utilizados en MASIF (CORBA). El código fuente no esta disponible. Conceptos claves: Lugares y agentes. Componentes claves:
   - ✓ LENGUAJE TELESCRIPT: Basado en un lenguaje propietario orientado a objetos, similar a Java en cuanto a que es interpretado (máquina virtual Telescript) y dispone de un modelo de seguridad. Utiliza clases para crear y enviar mensajes. También cuenta con mecanismos de seguridad: autenticación y control de acceso
   - ✓ Motor Telescript: Intérprete del lenguaje
   - ✓ Protocolo de transporte Telescript.- Opera en dos niveles: Nivel más alto: Codificación y decodificación de los agentes. Nivel más bajo: Transporte de los agentes.

   Además presenta una migración fuerte. Sofisticado y relativamente costoso (96 Mbytes de memoria), como antecedente en julio de 1995, NTT, AT&T y Sony desarrollaron en Japón un servicio basado en Telescript. Actualmente General Magic ha suspendido el desarrollo de nuevas versiones de Telescript.
2. AGLETS (AGent+appLETS). Diseñado por Dany B. Lange (1996) en el laboratorio de investigación de IBM en Tokyo (IBM Tokyo Research Laboratory). Aglets Software Development Kit (ADSK o Aglets Workbench): http://sourceforge.net/projects/aglets. Muy documentado y con abundantes ejemplos. Es de libre distribución con un interfaz de programación basado en Java (Java Aglet API o J-AAPI) y un Servidor de aglets (Tahiti). Posiblemente, uno de los SA más conocidos y utilizados actualmente. No proporciona soporte (clases de agentes reactivos, proactivos, ...) para desarrollar agentes. Proporciona soporte para implementar objetos móviles. Un aglet es un objeto móvil escrito en Java. Migración débil. Componentes claves:
   - ✓ LENGUAJE JAVA: Método de serialización de objetos Java (JOS).
   - ✓ Proxy u objeto representante.
   - ✓ Paso de mensajes: Métodos específicos de Java.
   - ✓ Protocolo de transporte de agentes aglets (ATP).- Ubicado en el nivel de aplicación para construir un bloque de datos: Nombre del SA, identificador del agente y cadena de octetos con el estado y código procedente del nivel de ejecución de los aglets.
   - ✓ No contempla comunicaciones seguras dejando libertad a las plataformas para utilizar otros protocolos seguros (SSL).

6. CONCLUSIONES

La conjunción de dos áreas de la Informática en constante desarrollo como lo son la Inteligencia Artificial y los Sistemas de Software Distribuido, ha posibilitado la implementación de los Agentes Móviles como otro modelo que permite el acceso, la administración de Sistemas de Software Distribuido. Los agentes móviles basados en la programación remota y la orientada a objetos esta especialmente diseñada para aplicaciones distribuidas que requieren la vista de diversas plataformas o entornos de ejecución por la red.

7. BIBLIOGRAFÍA

[1] Agha G. ACTORS: A Model of Concurrent Computation in Distributed Systems. The MIT Press: Cambridge, MA, 1986.

[2] Agha G., Wegner P., y Yonezawa A., Editores. Research Directions in Concurrent Object-Oriented Programming. The MIT Press: Cambridge, MA, 1993.

[3] Booch G. Object-Oriented Analysis and Design (Segunda Edición). Addison-Wesley: Reading, MA, 1994.

[4] Comer D. E. Internetworking with TCP/IP. Principles, Protocols and Architectures. Vol. I, 4ª Edición, Prentice Hall, 2000.

[5] Coulouris G.; Dollimore J.; Kindberg T. Sistemas Distribuidos: Conceptos y Diseño, Ed. Addison Wesley, 2001.

[6] Lieberman H. "Leticia: An agent that assists web browsing". In Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence(IJCAI-95), Montréal, Québec, Canada, Agosto de 1995, pp.924-929.

[7] Liu M.L. Computación Distribuida Fundamentos y Aplicaciones. Ed. Addison Wesley. 2003.

[8] Maes P. Agents that reduce work and information overload. Communications of the ACM, vol. 37(7) pp. 31-40, Julio 1994.

[9] Russell S. and Norvig P. Artificial Intelligence: A Modern Approach. Prentice-Hall, 1995.

.