TABLA DE CONTENIDO

INDICE DE FIGURAS

ANEXO A. HERRAMIENTAS DE DESARROLLO

INTRODUCCIÓN

Existe una gran variedad de plataformas que permiten el desarrollo de aplicaciones con agentes, en este anexo se da una breve descripción de aquellas que soportan desarrollos para dispositivos de baja capacidad tales como PDA's y teléfonos celulares. En primer lugar esta la descripción de la plataforma Jade por que con esta se desarrollo el prototipo, ya que ofrece varias ventajas, como por ejemplo: documentación completa sobre su manejo, disponibilidad de código, además de estar actualizada y mejorando continuamente.

Al final se describen brevemente otras plataformas que fueron analizadas con el fin de explotar su viabilidad y que de acuerdo con sus características pueden ser útiles en proyectos diferentes a este.

1 JADE

1.1 INTRODUCCIÓN

JADE (Java Agent DEvelopment Framework) es un paquete compuesto por un conjunto de herramientas y aplicaciones escritas totalmente en lenguaje Java, este *framework* simplifica el desarrollo de sistemas multiagentes gracias a la introducción de una capa de mediación acorde con las especificaciones FIPA y a través de un conjunto de herramientas gráficas que facilitan la depuración e implantación.

Adicionalmente con las librerías JADE se incluye el entorno de ejecución de agentes que puede instalarse y ejecutarse simultáneamente en varias estaciones de trabajo o computadoras, sin que sea necesario tener instalado el mismo sistema operativo en ellas, lo único que se requiere es que la maquina virtual de Java este funcionando correctamente. La configuración de este entorno se puede realizar de varias formas la primera es por medio de la línea de comandos cuando se invoca la ejecución de la plataforma, estos comandos establecen los valores iniciales con que comienza a funcionar el sistema; la segunda forma es utilizar una interfaz gráfica, que ejecutada localmente o remotamente permite modificar dinámicamente estos valores; una última forma que es indirecta es producida por el movimiento u operación de los gentes sobre una plataforma durante el ciclo de ejecución de las distintas aplicaciones.

La interacción cooperativa entre JADE y las librerías LEAP permiten obtener una plataforma de agentes fiel a las especificaciones FIPA con reducido consumo de memoria y compatible con dispositivos de baja capacidad como los caracterizados por la configuración J2ME - CLDC MIDP 1.0; estas librerías han sido desarrolladas con la colaboración del proyecto LEAP y se pueden conseguir como un componente mas de JADE.

JADE es distribuido gratuitamente como código abierto bajo los términos de LGPL por TILAB. Actualmente, la lista de empresas miembros del proyecto son: Telecom Italia (Tilab), Motorola, Whitestein, Profactor GMBH y France Telecom.

1.2 DESCRIPCIÓN TÉCNICA

Como se menciono anteriormente el principal objetivo de JADE es simplificar el desarrollo de sistemas multiagente, mientras que se asegura el cumplimiento de las especificaciones FIPA a través de un conjunto de servicios y agentes allí descritos, tales como: el servicio de nombrado, páginas amarillas, transporte de mensajes, análisis gramatical y los protocolos de interacción.

La plataforma de agentes JADE cumple con las especificaciones propuestas por FIPA incluyendo todos los componentes mínimos requeridos, como el ACC, el AMS y el DF.

Toda la comunicación entre agentes es desempeñada a través de mensajes, donde FIPA ACL es el lenguaje por defecto.

La plataforma opera sobre un equipo en el cual se crea un contenedor principal sobre el cual se generan automáticamente el AMS y el DF en el momento en que esta es iniciada. Además del contenedor principal el cual es indispensable dentro de la plataforma, existen otros contenedores que son creados a medida que se necesitan, y que se encargan de contener a los agentes que pertenecen a una aplicación determinada. Cada uno de estos puede operar en un equipo remoto diferente a aquél donde se ejecuta la plataforma, permitiendo la distribución, de tal manera que es posible que los agentes migren entre estos.

Dentro de las características principales de la plataforma se encuentran los Protocolos de comunicación, los Protocolos de interacción, Ontologías, Comportamientos (Behaviours) y las Plantillas.

1.2.1 Protocolos de comunicación

En cuanto a los *Protocolos de comunicación*, utilizados para comunicarse con otras plataformas, JADE se comporta como un camaleón porque es capaz de adaptarse a cada situación, escogiendo transparentemente el mejor de los disponibles: Java RMI, notificación de eventos, HTTP e IIOP; además de estos, pueden adicionarse otros protocolos a través de interfaces MTP e IMTP JADE (estas serán descritas más adelante).

1.2.2 Protocolos de interacción

Este tipo de protocolos es el utilizado por los agentes para comunicarse entre sí. FIPA ha definido un conjunto de protocolos de interacción de los cuales Jade implementa la mayoría dentro de un paquete llamado jade.proto, estos son: FIPA-Request, FIPA-query, FIPA-Request-When, FIPA-recruiting, FIPA-brokering. Todos ellos presentan la misma estructura, razón por la cual Jade proporciona un par de clases llamadas AchieveREInitiator y Responder, las cuales son una representación simple de estos.

1.2.3 Comportamientos

Un agente ejecuta tareas en respuesta a eventos externos o que el mismo genera. Cada agente está compuesto por un hilo de ejecución sobre el cual pueden ser realizadas todas las tareas que debe efectuar para lograr un estado deseado; para ello Jade proporciona una clase especial llamada *Behaviour* con la cual es posible implementarlas.

1.2.4 Ontologías

Para proporcionar entendimiento entre agentes es necesario que tanto el emisor como el receptor de los mensajes estén de acuerdo en el significado que estos tienen. Jade incluye características que permiten manipular la información de los mensajes ACL que generalmente poseen un formato String o Bytes, para representarlos en forma de objetos y hacerlos más fáciles de manejar. La siguiente figura muestra como se realiza tal conversión:

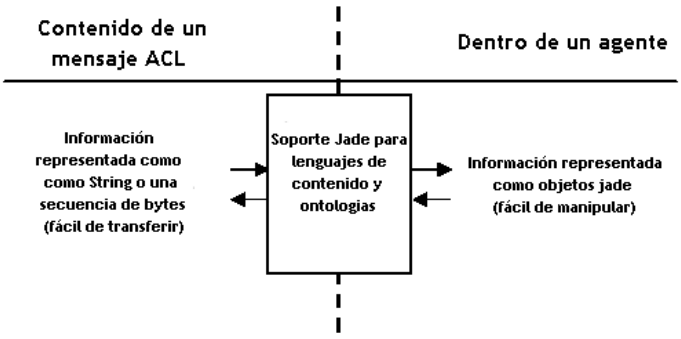

Figura 1. Lenguajes de contenido y ontologías.

En este punto es necesario aclarar la diferencia entre el lenguaje ACL y un lenguaje de contenido.

1.2.5 Lenguaje ACL y lenguajes de contenido

Es el lenguaje definido por FIPA que define una serie de sentencias que son utilizadas por los agentes para comunicarse. Cada mensaje ACL es codificado por medio de un codec que puede ser SL o LEAP dependiendo de la necesidad para ser transmitido; El codec SL codifica al mensaje dándole un formato de Strings y el codec LEAP lo transforma en una secuencia de bytes. Este último es utilizado en aplicaciones para dispositivos móviles ya que hace que los mensajes sean menos densos y más manejables por dispositivos con bajas capacidades tales como teléfonos celulares y PDA's. El resultado de estas dos formas de codificación es lo que se conoce como un *Lenguaje de contenido*.

Tal como mencionamos anteriormente cada agente es ejecutado en un hilo propio de tal manera que dentro de cada uno de ellos se desempeñan varias tareas paralelamente. Si tenemos en cuenta que no solo un agente está actuando en un determinado momento sino varios, podemos concluir que se pueden generar serios problemas cuando estos acceden a recursos compartidos como locaciones de memoria, periféricos e información en general. Para dar solución a esta dificultad JADE incluye una herramienta denominada ordenación cronológica de comportamientos colaborativos que permite solucionar y controlar la concurrencia en este tipo de sistemas, de manera que es posible programar tareas en forma clara y efectiva.

1.3. INTERFAZ GRÁFICA

La plataforma de agentes provee una interfaz gráfica de usuario para el monitoreo y control del estado de ejecución de los agentes permitiendo gestionarlos de forma local y remota.

La siguiente figura muestra los elementos que se crean al iniciar la ejecución de la plataforma, estos son: El contenedor principal y los agentes *RMA*, AMS y *DF*.

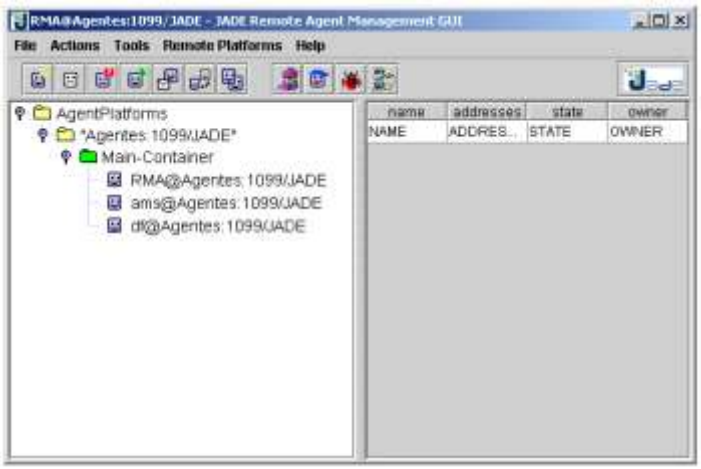

Figura 2. Interfaz grafica JADE

1.3.1 El Facilitador de Directorio (DF)

Ofrece el servicio de páginas amarillas a los agentes y permite tanto al usuario como a los demás componentes de un sistema multiagente observar, eliminar, modificar y buscar la descripción de un determinado agente previamente registrado. La interfaz también permite federar este directorio con otros y crear un dominio y subdominios de páginas amarillas. Un DF federado que pertenezca una plataforma de un tipo diferente a JADE, puede ser controlado por la misma interfaz como si se tratase de una plataforma JADE, esto siempre y cuando la otra plataforma cumpla con las especificaciones de FIPA.

1.3.2 El Sistema de Gestión de Agentes (Agent Management System-AMS)

Es un componente obligatorio en una plataforma de agentes. Este componente se encarga de registrar a cada agente que es creado, asignándole su respectivo AID. Ofrece un servicio de páginas blancas a un sistema de agentes. El AMS almacena entre otras cosas, la información de las direcciones con las cuales tiene contacto cada AID registrado.

1.3.3 El Agente de Monitoreo Remoto (Remote Monitoring Agent - RMA)

Permite controlar el ciclo de vida de la plataforma y todos los agentes registrados en ella. La arquitectura distribuida de JADE también permite un monitoreo remoto a través de la GUI (interfaz de usuario).

La siguiente figura muestra la interfaz grafica del DF.

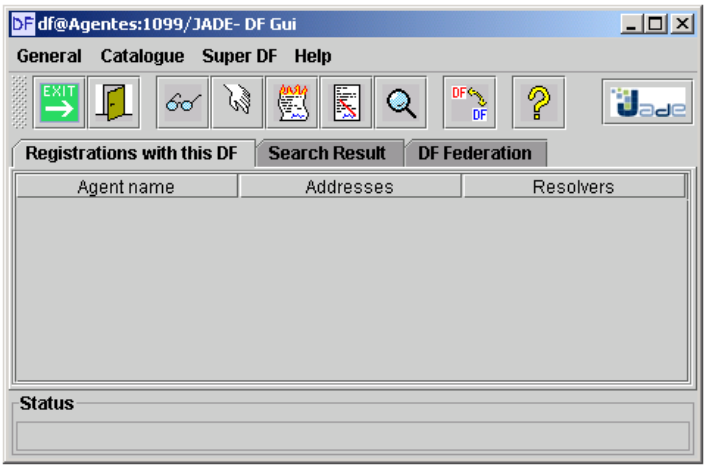

Figura 3. Interfaz grafica DF

El agente *Dummy* es muy simple pero útil para inspeccionar el intercambio de mensajes entre agentes. Los agentes de prueba como el Dummy facilitan la validación de una interfaz de agentes antes de integrarla al sistema multiagente y la ejecución de consultas de prueba en el caso de que uno de estos falle. La interfaz grafica también provee soporte para edición, composición, recepción y envió de mensajes ACL desde y hacia cualquier agente además de permitir almacenar sus transacciones para su posterior análisis.

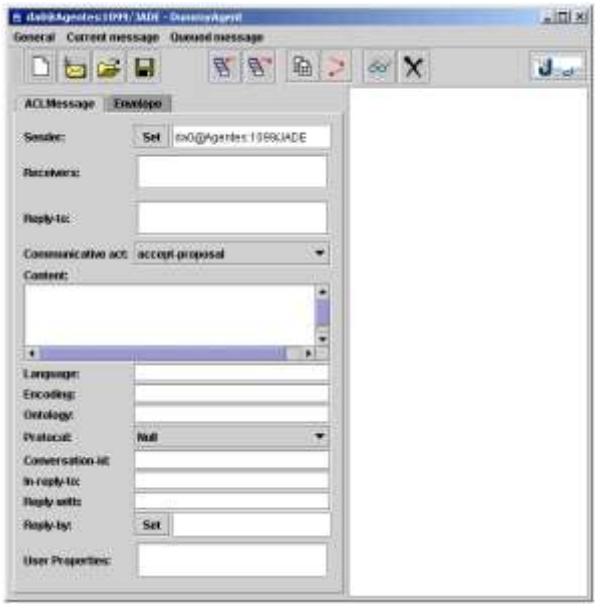

Figura 4. Interfaz grafica del DummyAgent

El agente *Sniffer*, como su nombre lo indica, permite rastrear mensajes intercambiados en la plataforma JADE. Cuando el usuario decide husmear a un agente, o a un grupo de agentes, cada mensaje dirigido a o desde tal agente o grupo es rastreado y mostrado en la ventana.

El agente inspector permite monitorear y controlar el ciclo de vida de los agentes que se ejecutan, además de interceptar los mensajes, tanto los enviados como los recibidos.

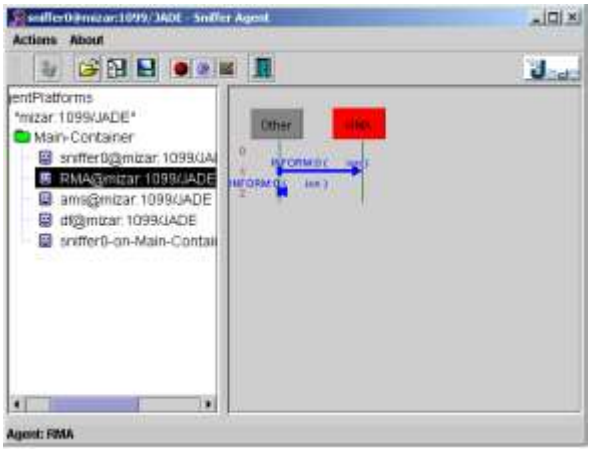

Figura 5. Interfaz grafica agente inspector

1.4. APLICACIONES

1.4.1 Introducción

JADE es un entorno que permite el desarrollo de aplicaciones tanto para dispositivos móviles como fijos cubriendo una amplia gama de equipos que van desde estaciones de trabajo, computadoras, hasta dispositivos portátiles como: PDA, teléfonos celulares, pager y demás. Todo soportado por una plataforma que permite desarrollar y poner en marcha sistemas multiagentes con cierto grado de inteligencia.

JADE provee un modelo en el que todos los elementos que componen un sistema son autónomos e independientes, es decir, que los agentes tienen la absoluta libertad de comunicarse y colaborar con otros hasta alcanzar sus objetivos, por lo cual, metafóricamente hablando este marco para la construcción de agentes intenta abstraer el comportamiento de la sociedad humana, en donde cada individuo interacciona con otros principalmente orientado por sus creencias, deseos y objetivos.

En esta tecnología se obtienen los mejores resultados cuando las aplicaciones exhiben en gran medida un intercambio dinámico de información entre sus componentes, o sea entre agentes. Estas relaciones permiten modelar comportamientos muy complejos, que en una aplicación desarrolla a partir de técnicas convencionales de abstracción como es el caso de los objetos, conducirían a un sistema muy complicado y poco flexible; sin embargo, el modelo de agentes autónomos dificulta en gran medida el control de tareas y concurrencia de acceso a los recursos, por esta razón una implementación respetable de una plataforma de agentes debe impedir este tipo de inconvenientes y facilitar al desarrollador el dominio sobre la ejecución de los agentes, permitiendo de esta forma sacar el mayor provecho al paradigma de agentes; esta es precisamente una de las características entre otras, que hacen de JADE la plataforma más sobresaliente de todas las evaluadas

1.4.2 Aplicaciones móviles

Jade contiene un modulo adicional denominado LEAP. Este ofrece un ambiente de desarrollo modificado a partir del modulo básico de JADE para ser utilizado en un amplio rango de dispositivos que varía desde servidores JAVA hasta teléfonos celulares. Para logra esto JADE-LEAP puede ser configurado de varias maneras de acuerdo con los tres ambientes JAVA encontrados:

- *J2SE*: Se utiliza JADE-LEAP para operar sobre equipos o Servidores que pertenecen a una red fija sobre los cuales se está ejecutando el jdk1.2 o superior.
- *Pjava*: En este caso se utiliza sobre aquellos dispositivos que poseen el perfil personal java, tales como PDA's de alta capacidad.

- *Midp*: Para aquellos dispositivos que soportan el perfil MIDP1.0 o superior, tales como la mayoría de teléfonos celulares que existen actualmente con soporte java.

Las tres versiones de JADE-LEAP poseen el mismo conjunto de API's de tal forma que ofrecen una capa homogénea sobre una gran diversidad de dispositivos y tipos de red, tal como se muestra en la siguiente figura:

Algunas de las características disponibles en JADE-LEAP para j2se y pjava no están presentes en JADE LEAP para midp debido a que existen clases java que no están soportadas para este perfil, por estar restringido a equipos de baja capacidad de memoria, presentación gráfica, etc.

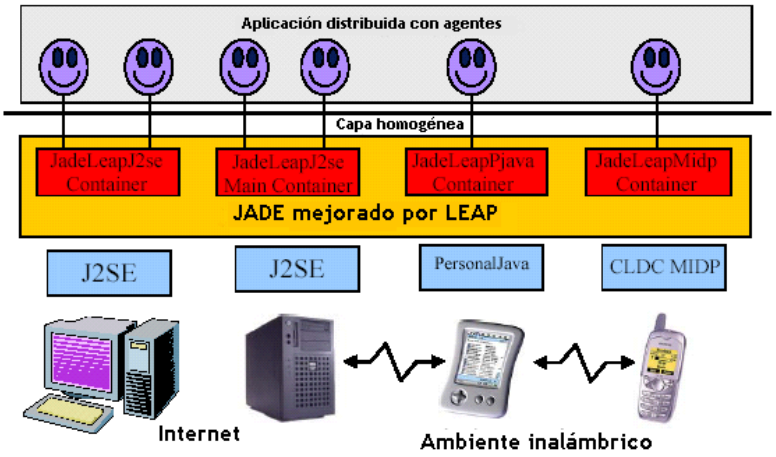

Figura 6. Ambiente de ejecución JADE-LEAP

1.5. OTRAS VENTAJAS OFRECIDAS POR JADE

Teniendo como base la tecnología de agentes aplicada en entornos móviles los desarrolladores pueden representar a los usuarios por medio de una serie de agentes gestores que tienen como objeto facilitar el descubrimiento y recuperación de información a través de la interacción con otros agentes, que indistintamente caracterizan a un usuario, proveedor de servicios o contenidos.

Los agentes desarrollados con JADE son extremadamente adaptables para actuar en un entorno móvil, sus habilidades en cuanto a proactividad y autonomía favorecen el surgimiento de aplicaciones útiles en las organizaciones, tales como agendas, información de búsqueda, servicios de negociación, servicios de información geográficos, entre otros.

La intervención de los agentes proporciona ayuda en necesidades concretas: muchos ejemplos ilustran las fortalezas de los agentes JADE actuando como asistentes para viajes, soporte para sistemas de ventas, en los que las bondades de los agentes P2P permiten efectuar tareas dinámicas y compartir información.

Otro campo poco explorado es el sector del entretenimiento; se podría emplear JADE como plataforma para el desarrollo de juegos en red multijugador, en la que una se ofrezca una interacción real entre los jugadores, o puede ser la piedra angular para mejorar las comunidades del tipo móvil donde las comunicaciones peer-to-peer permiten más directamente una relación entre los miembros.

2 KSACI

KSACI es una plataforma para el desarrollo de software para dispositivos móviles, basado en una arquitectura llamada SACI (Simple Agent Communication Infrastructure) que proporciona una infraestructura para comunicación de agentes basada en KQML. kSACI hereda de la arquitectura SACI para permitir que sobre dispositivos con configuración CLDC tales como teléfonos celulares y PDA,s operen agentes.

3 FIPA-OS

FIPA-OS es una herramienta orientada a componentes para la construcción de Agentes que cumplen con el estándar FIPA utilizando todos sus componentes opcionales, variables y obligatorios, estos últimos, son todos los componentes que FIPA-OS necesita para que los agentes se ejecuten. La siguiente figura ilustra los componentes disponibles y la relación que existe entre ellos.

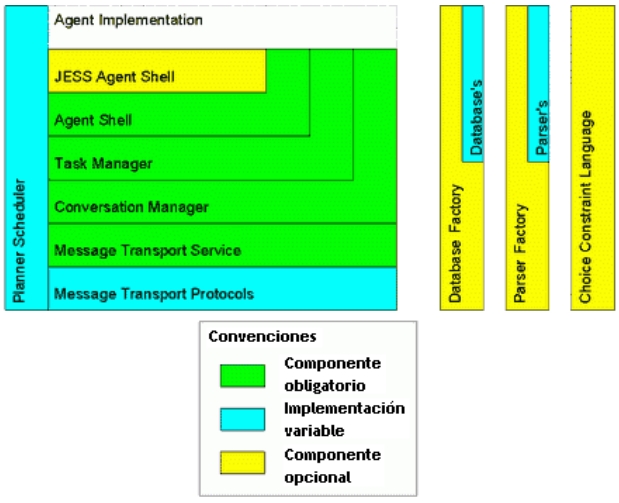

Figura 7. Arquitectura FIPA-OS

El modulo *Planner Scheduler* no está disponible actualmente.

Los componentes *Database Factory*, *Parser Factory* y *CCL* son opcionales y no tienen una relación explícita con otros componentes dentro de la herramienta de desarrollo. El *Planner Scheduler* generalmente tiene la habilidad de interactuar con todos los componentes de un Agente, aunque en el sentido opuesto no necesariamente ocurra lo mismo.

3.1 AGENT SHELL

La clase *FIPAOSAgent* suministra un shell para la implementación de agentes.

3.2 TASK MANAGER

El *Task Manager* provee la habilidad para dividir la funcionalidad de un agente en funcionalidades más pequeñas, separando unidades de trabajo llamadas tareas. El objetivo de esto es que las tareas contengan piezas de código que ejecutan ciertas operaciones y opcionalmente retornen un resultado, teniendo la habilidad de enviar y recibir mensajes y preferiblemente poca o ninguna dependencia del agente sobre el cual se están ejecutando.

3.3 CONVERSATION MANAGER

El *Conversation Manager* posee la habilidad de seguir el estado de una conversación a un nivel de desempeño, así como mecanismos para agrupar mensajes de la misma. Si una conversación es especificada bajo un protocolo particular (FIPA u otro definido por el usuario), el *Conversation Manager* se encargará de ver que el protocolo está siendo seguido por los participantes de la conversación.

3.4 MTS (MESSAGE TRANSPORT SERVICE)

El MTS envía y recibe mensajes hacia y desde un sistema de agentes. FIPA-OS proporciona una gran variedad de MTP's (Message Transport Protocols) para permitir que el MTS se comunique utilizando varios protocolos.

Los MTPs soportados son: RMI, SSL-RMI, Corbaname y http.

3.5 JESS AGENT SHELL

Esta extensión del Agent Shell estándar da soporte para escribir agentes FIPA-OS tomando ventaja del sistema JESS.

3.6 FIPA-OS FACTORY

Una fábrica de fábricas. FIPA-OS hace un uso extensivo de las fábricas (factories); FIPAOSFactory es una forma de bosquejar una instancia particular de la implementación de una fábrica con una interfaz de fábrica abstracta.

3.7 DATABASE FACTORY

EL módulo *Database Factory* proporciona un mecanismo simple para interactuar con un número de implementaciones de bases de datos de una manera orientada a objetos utilizando una interfaz consistente.

3.8 ABSTRACT DATABINDING IMPLEMENTATION

Este componente está diseñado para permitir expresar el contenido de un mensaje de manera que solo posea su información semántica, sin preocuparse por la sintaxis utilizada por los desarrolladores.

4 AGENTLIGHT

Agent Light fue un esfuerzo para tomar teorías de sistemas Multi Agentes y aplicarlas en el ambiente de los dispositivos móviles, motivado por el incremento del uso de potentes dispositivos de este tipo, y el subsecuente incremento de la demanda de servicios personalizados.

El objetivo del proyecto es unir la teoría con la práctica para construir la base del desarrollo de agentes autónomos para dispositivos de baja capacidad de procesamiento.

5 3APL-M

Es una plataforma para construir aplicaciones utilizando el Lenguaje de Programación de Agentes Autónomos Artificiales (3APL) que permite lógica para los ciclos de deliberación y representación interna de conocimiento. Es poco utilizada en aplicaciones de computación móvil. Su versión binaria es distribuida para J2ME y J2SE.

5.1 3APL

3APL es un lenguaje de programación para implementar agentes cognitivos. Ofrece la posibilidad de construir agentes con creencias, metas, capacidades básicas (tales como actualización de creencias, acciones

externas o acciones de comunicación) y un conjunto de reglas de razonamiento práctico a través de las cuales las metas de los agentes pueden ser actualizadas o revisadas. El programa 3APL es ejecutado por medio de un interprete que delibera basado en actitudes cognitivas del agente.

6 COUGAR

Cougar es una arquitectura basada en Java para la construcción de aplicaciones distribuidas de larga escala basadas en agentes. Este es un producto presentado de dos formas distintas, desarrollado a través de un esfuerzo e investigación hecho por DARPA a o largo de 8 años. El primer programa demostró la factibilidad de utilizar tecnología de agentes para manejar técnicas distribuidas de planeación y replaneación rápidas y de larga escala.

El Segundo desarrolló tecnologías de información para mejorar la supervivencia de los sistemas distribuidos basados en agentes operando en ambientes extremadamente caóticos.

La arquitectura resultante, Cougaar, provee un entorno de desarrollo para implementar aplicaciones distribuidas de larga escala basadas en agentes, con consideraciones mínimas para la arquitectura subyacente y la infraestructura. La arquitectura Cougaar utiliza lo último en diseño de sistemas basados en componentes y orientados a agentes y tiene una larga lista de características importantes.

Actualmente Cougaar está disponible en su sitio web como código abierto y su última versión es la v11.4. También están disponibles listas de correos de los desarrolladores interesados, papers sobre esta arquitectura y muchas cosas más.

7. REFERENCIAS

[AgentLight 2004]. http://home.wanadoo.nl/agentlight/

[Cougar 2005]. http://www.komotv.com/stories/35796.htm

[FIPA-OS 2004]. http://www.nortelnetworks.com/products/announcements/fipa/

[JADE ADM 2005]. BELLIFEMINE, Fabio. CAIRE, Giovanni. TRUCCO, Tiziana. RIMASSA, Giovanni. Jade Administrator's guide. TILAB. Enero 2005.

[JADE LEAP 2005]. CAIRE , Giovanni. Leap User guide. TILAB . Marzo 2005.

[JADE 2005] http://jade.tilab.com/

[KSACI 2004]. www.inf.furb.br/~jomi/ pubs/2001/Albuquerque-atal2001.pdf

[3APL-M 2005]. http://www.cs.uu.nl/3apl-m/