TESIS DEFENDIDA POR

Marcel Isabel Juárez Valdez

Y aprobada por el siguiente comité:

Dr. José Antonio García Macías
Director del Comité

Dr. Oscar Iván Lepe Aldama
Codirector del Comité

Dr. Jesús Favela Vara
Miembro del Comité

Dr. César Cruz Hernández
Miembro del Comité

Dr. Pedro Gilberto López Mariscal
Coordinador del programa en Ciencias de la Computación

Dr. Raúl Ramón Castro Escamilla
Director de Estudios de Posgrado

23 de Marzo del 2006.

CENTRO DE INVESTIGACIÓN CIENTÍFICA Y DE EDUCACIÓN
SUPERIOR DE ENSENADA

PROGRAMA DE POSGRADO EN CIENCIAS
EN CIENCIAS DE LA COMPUTACIÓN

Seguridad en Sistemas Multiagentes

TESIS

que para cubrir parcialmente los requisitos necesarios para obtener el grado de

MAESTRO EN CIENCIAS

Presenta:

Marcel Isabel Juárez Valdez

Ensenada, Baja California, México. Marzo del 2006.

RESUMEN de la tesis que presenta **Marcel Isabel Juárez Valdez**, como requisito parcial para obtener el grado de MAESTRO EN CIENCIAS en CIENCIAS DE LA COMPUTACIÓN. Ensenada, B. C. Marzo del 2006.

Seguridad en Sistemas Multiagentes

Resumen aprobado por:

Dr. José Antonio García Macías	Dr. Oscar Iván Lepe Aldama
Director de Tesis	*Codirector de Tesis*

Es innegable que la computación distribuida es la arquitectura fundamental hoy día y en el futuro previsible para todo sistema de información de importancia. Al mismo tiempo, el crecimiento de la Internet y el mejoramiento en las comunicaciones inalámbricas están propiciando el advenimiento de sistemas de cómputo distribuido cada vez más complejos tanto por el número de nodos y de enlaces de comunicaciones simultáneamente activos, como por la necesidad de percibir cambios en el entorno y adaptar el modelo de cómputo a estos cambios. Para enfrentar la creciente complejidad de estos sistemas de cómputo distribuido se ha propuesto el uso de la tecnología de agentes, originada del ámbito de la inteligencia artificial. Un agente es un pedazo de software autónomos, proactivos y social que interactúa con otros agentes en la solución de un conjunto de problemas o en la consecución de una serie de objetivos individuales o colectivos. Aunque el uso de agentes facilita la concepción de los sistemas de cómputo distribuido móviles o ubicuos, actualmente esta tecnología presenta deficiencias fundamentales en sus modelos de implementación que inhibe su uso a gran escala en sistemas de producción. Entre estas deficiencias se encuentra el sustento a la autenticación de agentes que se mueven entre los nodos que integran una plataforma de agentes o sistema multiagente.

El presente trabajo aborda el sustento en middleware a la autenticación de agentes móviles desarrollados en la plataforma JADE mediante el uso de certificados digitales X509 y la integración de nuevos actores a la plataforma JADE en su agregado JADE-S. El trabajo es original pues otros trabajos relacionados no abordan conjuntamente autenticación y movilidad en agentes. Por otro lado, el trabajo es pertinente pues un sistema basado en agentes móviles sin autenticación tiene un riesgo de seguridad no despreciable; asimismo, JADE se esta posicionando como un estándar de facto en middleware para sistemas multiagentes.

Palabras clave: middleware, agentes, seguridad, movilidad, FIPA, JADE/JADE-S.

ABSTRACT of the thesis presented by **Marcel Isabel Juárez Valdez**, as a partial requirement to obtain the MASTER SCIENCE degree in COMPUTER SCIENCES. Ensenada, B. C. March 2006.

Security in Multiagents Systems

Abstract approved by:

Dr. José Antonio García Macías
Thesis director

Dr. Oscar Iván Lepe Aldama
Thesis codirector

He is undeniable that the distributed computation is the fundamental architecture nowadays and in the future foreseeable for all information system of importance. At the same time, the growth of the Internet and the improvement in the wireless communications are causing the coming of distributed computing systems more and more complex as much by the number of nodes and simultaneously active connections of communications, like by the necessity to perceive changes in the surroundings and to adapt the model of computing to these changes. In order to face the increasing complexity of these distributed computing systems the use of the technology of agents has seted out, originated of the scope of the artificial intelligence. An agent is a piece of software independent, proactive and social that interacts with other agents in the solution of a set of problems or in the attainment of a series of individual or collective objectives. Although the use of agents facilitates the conception of the systems of distributed computing systems movable or ubicuos, at the moment this technology presents fundamental deficiencies in its models of implementation that inhibits its use on great scale in production systems. Between these deficiencies is the sustenance to the authentication of agents who move between the nodes that integrate a platform of agents or system multiagent.

The present work approaches the sustenance in middleware to the authentication of movable agents developed in platform JADE by means of the digital certificate use X509 and the integration of new actors to platform JADE in its aggregate JADE-S. The work is original then other related works jointly do not approach authentication and mobility in agents. On the other hand, the work is pertinent because a system based on movable agents without authentication has a risk of nondespicable security; also, JADE this positioning like a standard de facto in middleware for systems multiagents.

Keywords: middleware, agents, security, mobility, FIPA, JADE/JADE-S.

Dedicatoria

Especialmente para los seres que me han dado la vida, confianza y amor de forma incondicional.
Mami y Papi: Elvia Nanci y Sabino
Los amo.

A mis hermanos:
Elvia Nancy del Rosario y Erik Guadalupe,
quienes me dan la fuerza para seguir adelante y enseñarles
que no hay nada que detenga nuestros sueños si realmente los deseamos.
Los quiero con toda el alma.

A mí:
Por aquellos sueños e ilusiones que se quedaron en el pasado. Por la superación de todos los obstáculos y lecciones que viví en el transcurso del desarrollo de esta tesis. Y por continuar creyendo que el camino apenas empieza, por esas esperanzas, sueños e ilusiones que aún están conmigo.

A mi tío Titi,
sin usted presente en mi vida simplemente no me hubiera sentido como una princesa,
gracias por estar apoyándome en todo momento,
lo quiero mucho.

A mi mamá Chabela y papá Venado,
aún recuerdo lo orgullosos que se ponían ante mis logros,
me hicieron sentir con sus consejos y pláticas la mejor de las nietas,
los extraño mucho.

A mi papá Gabriel y mamá Chelina,
por estar al pendiente de lo que pasa en mi vida,
los quiero mucho.

A mi tía Lidia, tía Ermy, tía Cheta,
gracias.

A toda mi familia,
quienes me han apoyado, han creído en mí,
me han brindado su cariño y respeto a lo largo de toda mi vida,
muchas gracias.

Agradecimientos

A Dios por ayudarme a salir adelante y estar siempre cuando lo necesito.

En especial, quiero agradecer al Dr. Oscar Iván Lepe Aldama mi codirector de tesis, por confiar en mi y brindarme su apoyo en momentos que no hubiera podido enfrentar sola. Gracias Dr. Lepe, espero no haberlo defraudado.

A mi director de tesis Dr. José Antonio García Macías, por su paciencia y colaboración en este trabajo.

A los miembros del comité de tesis Dr. Jesús Favela Vara y Dr. César Cruz Hernández, por sus acertados comentarios y ánimos para que terminara mi tesis.

Abel el haber compartido parte de mi vida contigo ha sido sensacional, gracias por todas tus enseñanzas, cariño y sobre todo tu presencia en ocasiones donde parecía no estaba nadie, volteaba y estabas tú.

Juan8, Saúl, Zeus, Jburci, César, Rafa, Elizabeth Violeta las grandes personas que he tenido el placer de conocer y compartir divertidos momentos.

A mi amiga Ana Luisa por estar siempre al pendiente, a Jesús Horacio por brindarme siempre sus consejos, sus palabras de aliento y apoyo incondicional.

A mis compañeros de generación 2003-2005, sin su presencia estos dos años no hubieran tenido tanto aprendizaje.

Mara, Paty, Ariel, Araceli, Edgar, Carlos (Wolverine), por esos ratos de risas, juegos de cartas, películas, carnes asadas y bastantes aventuras.

Al departamento de Telemática dónde tuve mi espacio para realizar mi tesis. Fue agradable compartir más de un año con ustedes, gracias por haber estado en la mejor disposición y al pendiente de lo que necesitaba.

Al Centro de Investigación Científica y de Educación Superior de Ensenada (CICESE), por enseñarme el verdadero significado de estudiar.

Al Consejo Nacional de Ciencia y Tecnología (Conacyt),
por su apoyo económico sin el cual no hubiera sido posible este trabajo de investigación.

A todas aquellas personas que en algún momento compartimos una sonrisa.

Ensenada, México — Marcel Isabel Juárez Valdez
23 de Marzo del 2006.

Tabla de Contenido

Capítulo Página

Tabla de Contenido (Continuación)

Lista de Figuras

Lista de Figuras (Continuación)

Lista de Tablas

Capítulo I

Introducción

"Aprende a amar el viaje, no el destino".
Anónimo

I.1 Antecedentes

Existe la tendencia de construir sistemas de información llamados sistemas de cómputo ubicuo (Weiser, 1991), en base a comunicaciones inalámbricas y modelos de cómputo móvil que permitan el acceso irrestricto a información particularizada a la situación en la cual se encuentra inmersa una persona. Asimismo, existe la tendencia de utilizar la tecnología de agentes, surgida inicialmente del ámbito de la inteligencia artificial, para la construcción de estos sistemas de cómputo ubicuo. Bajo la visión de estas tendencias, los sistemas de cómputo ubicuo se implementan mediante componentes de software autónomos, proactivos y sociales, llamados agentes, que interactúan entre si en la solución de un conjunto de problemas o en la consecución de una serie de objetivos individuales o colectivos. Un sistema de información constituido por varios agentes es comúnmente llamado un sistema multiagente. Varios son los retos tecnológicos para dotar a los agentes de sus características. Para dotarlos de autonomía es necesario instrumentar algoritmos de inteligencia artificial. Para dotarlos de proactividad hay que proveerlos de mecanismos para percibir el entorno. Para dotarlos de sociabilidad hay que integrarles capacidades para comunicarse. Este trabajo de investigación se enfoca hacia uno de los problemas que surge al dotar de sociabilidad a los agentes.

La Internet esta surgiendo como la infraestructura de telecomunicaciones ubicua que sustenta todo tipo de sistema de información distribuido, ya sea de datos, de voz o de video. En este sentido la construcción de los sistemas de cómputo ubicuo hoy día se contempla usando a la Internet. Asimismo, los sistemas multiagentes utilizan la Internet como su

mecanismo de comunicación que sustenta la sociabilidad de los agentes. Desafortunadamente hoy día la Internet se considera todavía un medio de comunicación inseguro no obstante que existan trabajos de investigación orientados a mejorar la privacidad y secrecía de los datos intercambiados. Este trabajo de investigación se enfoca hacia uno de los problemas que surge de utilizar un medio inseguro de comunicación para las comunicaciones entre agentes.

Java es un lenguaje de programación y una plataforma de ejecución que promete la interoperabilidad universal. Mediante la tecnología de máquina virtual es posible recibir y ejecutar en una plataforma X, componentes de Java desarrollados en otra plataforma Y. Para sistemas de cómputo ubicuo basados en tecnología multiagentes esto representa una infraestructura universal para la ejecución de componentes de software. Obsérvese que esta facilidad para descargar componentes de software desde una plataforma hacia otra trae consigo un riesgo de seguridad, si acaso el componente descargado contiene información ó instrucciones maliciosas. Hoy día Java contempla mecanismos para facilitar la implementación de seguridad. Sin embargo, no todas las plataformas para sistemas multiagentes exportan apropiadamente o completamente estos mecanismos de seguridad para la implementación de los sistemas basados en agentes. Este trabajo de investigación se enfoca hacia uno de los problemas que surgen de utilizar plataformas de sistemas multiagentes basados en Java como la infraestructura universal de ejecución.

La movilidad contemplada en los sistemas multiagentes se refiere a la movilidad de código (agentes) y la movilidad de dispositivos. En movilidad de dispositivos los agentes se encuentran de manera constante censando el ambiente en busca de nuevos dispositivos, tales como teléfonos celulares y PDA´s. En la movilidad de código, los agentes para realizar las tareas pueden permanecer en el entorno donde inician su ejecución (agente estacionario), ó migrar de un entorno de ejecución a otro para el cumplimiento de sus metas u objetivos (agentes móviles). La movilidad de código trae consigo algunas ventajas como la de reducción de tráfico en la red, capacidades de procesamiento mejoradas para dispositivos móviles, encapsulado de los protocolos de comunicación, entre otras. Y algunas desventajas como la necesidad de que los sistemas sean robustos y tolerantes a fallas, y que deban adaptarse

dinámicamente a los cambios en el sistema. Este trabajo de investigación se enfoca hacia uno de los problemas que surge de utilizar agentes móviles para llevar a cabo el cumplimiento de las tareas u objetivos de los sistemas multiagentes.

Inspirados por la visión de que los agentes seguirían siendo solamente un sueño si no se preserva la interoperabilidad punto a punto a través de los diferentes fabricantes y desarrolladores, en 1996 TILAB [1] (anteriormente CSELT) promueve la creación de FIPA (Foundation for Intelligent Phisical Agents), una asociación internacional sin fines de lucro, de compañías y organizaciones, que comparten el objetivo y el esfuerzo para producir las especificaciones estándar para la tecnología de agentes. TILAB, y en particular el equipo JADE, apoyaron y reforzaron esta iniciativa participando en la edición de especificaciones técnicas y la arquitectura FIPA. JADE también participó exitosamente en las pruebas de interoperabilidad de FIPA en los años 1999 y 2001, en nuestros días, JADE es la única plataforma que cumple totalmente con el estándar FIPA.

Durante los últimos años, el equipo JADE se ha enfocado en el diseño y desarrollo del middleware adicionandole funcionalidades, documentación y consolidación en las funcionalidades ya existentes. En la plataforma JADE se pone de manifiesto que la seguridad es multifacética ya que en el mundo real involucra variables no conocidas e implementaciones imperfectas. Por otro lado la movilidad en sistemas multiagentes en la plataforma JADE se encuentra en estudio y desarrollo.

En este trabajo de investigación se presenta una propuesta bajo la plataforma de JADE y la extensión de seguridad JADE-S, con el objetivo de brindar seguridad a los agentes móviles durante el cumplimiento de sus tareas y su migración inter-plataforma, mediante el uso de certificados y el protocolo de transporte HTTP-S. Considerando el apego a las especificaciones de FIPA, lo cual asegura que la plataforma JADE siga siendo interoperable.

[1] http://www.telecomitalialab.com

I.2 Objetivos general y específicos

I.2.1 Objetivo general

Analizar la seguridad informática y movilidad actual en los sistemas multiagentes, con la finalidad de identificar oportunidades de mejora y proponer una solución.

I.2.2 Objetivo específico

1. Estudiar los fundamentos teóricos de la seguridad informática y movilidad en los sistemas multiagentes.

2. Estudiar el estado del arte de la seguridad informática y la movilidad en sistemas multiagentes.

3. Hacer un estudio comparativo de los middlewares existentes para sistemas multiagentes y elegir uno de acuerdo al interés sobre seguridad y movilidad de agentes.

4. Realizar una aplicación para dirigir el análisis del estado actual de la seguridad informática y la movilidad en el middleware seleccionado.

5. Identificar oportunidades de mejora en el sustento de agentes móviles seguros en el middleware seleccionado.

6. Proponer un prototipo de solución para alguna de las oportunidades de mejora identificadas.

7. Implementar el prototipo de solución para la oportunidad de mejora identificada.

8. Presentar conclusiones e ideas para trabajo futuro en la misma línea de investigación.

I.3 Organización de la tesis

El resto de este documento está organizado de la siguiente manera. En el capítulo II se muestran los fundamentos y el marco teórico de la seguridad y movilidad en sistemas multiagentes. Asimismo, se presenta un estudio comparativo de los principales middlewares

para sistemas multiagentes desde el punto de vista de movilidad y seguridad, cuyo resultado es la selección del middleware JADE en su agregado JADE-S para la realización de esta tesis. Los capítulos III y IV muestran el análisis realizado en la plataforma JADE en su agregado JADE-S con el objetivo de identificar una oportunidad de mejora. El capítulo V describe la propuesta del prototipo solución implementada en JADE-S. Y por último, las conclusiones y trabajo futuro se encuentran en el capítulo VI.

Capítulo II

Fundamento y marco teóricos

"Ve a donde quieras ir, sé lo que quieras ser, porque tienes tan solo una vida y una oportunidad para hacer todo lo que quieras hacer".
Anónimo

II.1 Fundamento teórico

"The most profound technologies are those that disappear. They wave themselves into the fabric of everyday life until they are indistinguishable from it". Así inicia Mark Weiser su artículo seminal: *The Computer for the 21st Century* (Weiser, 1991); en donde describe su visión acerca del cómputo ubicuo, teniendo como esencia la saturación del ambiente con sistemas de cómputo e información, con los cuales se interactúe de manera casi transparente.

II.1.1 Cómputo Ubicuo

Mark Weiser, el padre del Cómputo Ubicuo, visualiza a éste como "la tercera ola", el nuevo paradigma de computación en el siglo XXI.

La tercera ola en computación nace de la combinación de los sistemas distribuidos y el cómputo móvil, como podemos observar en la Figura 1, las características de los sistemas distribuidos y el cómputo móvil se solidifican bajo el concepto de cómputo ubicuo, el cual presenta las características de: escalabilidad de localización, para encontrarnos en el punto donde esté el usuario; transparencia, para pasar desapercibidos en el medio físico, la tecnología estará tan incorporada en la vida cotidiana, que será invisible para la gente en general; y espacios inteligentes, para adaptarse a las preferencias de cada individuo (Satyanarayan, 2001). En la actualidad el cómputo ubicuo está en fase de investigación y desarrollo, pero se cree que en un futuro modificará el entorno doméstico, industrial, médico e incluso el ocio personal.

Los avances tecnológicos necesarios para construir un ambiente de cómputo ubicuo se pueden clasificar en cuatro áreas: dispositivos, redes, middleware y aplicaciones (Saha y Mukherjee, 2003; Lyytinen y Yoo, 2002).

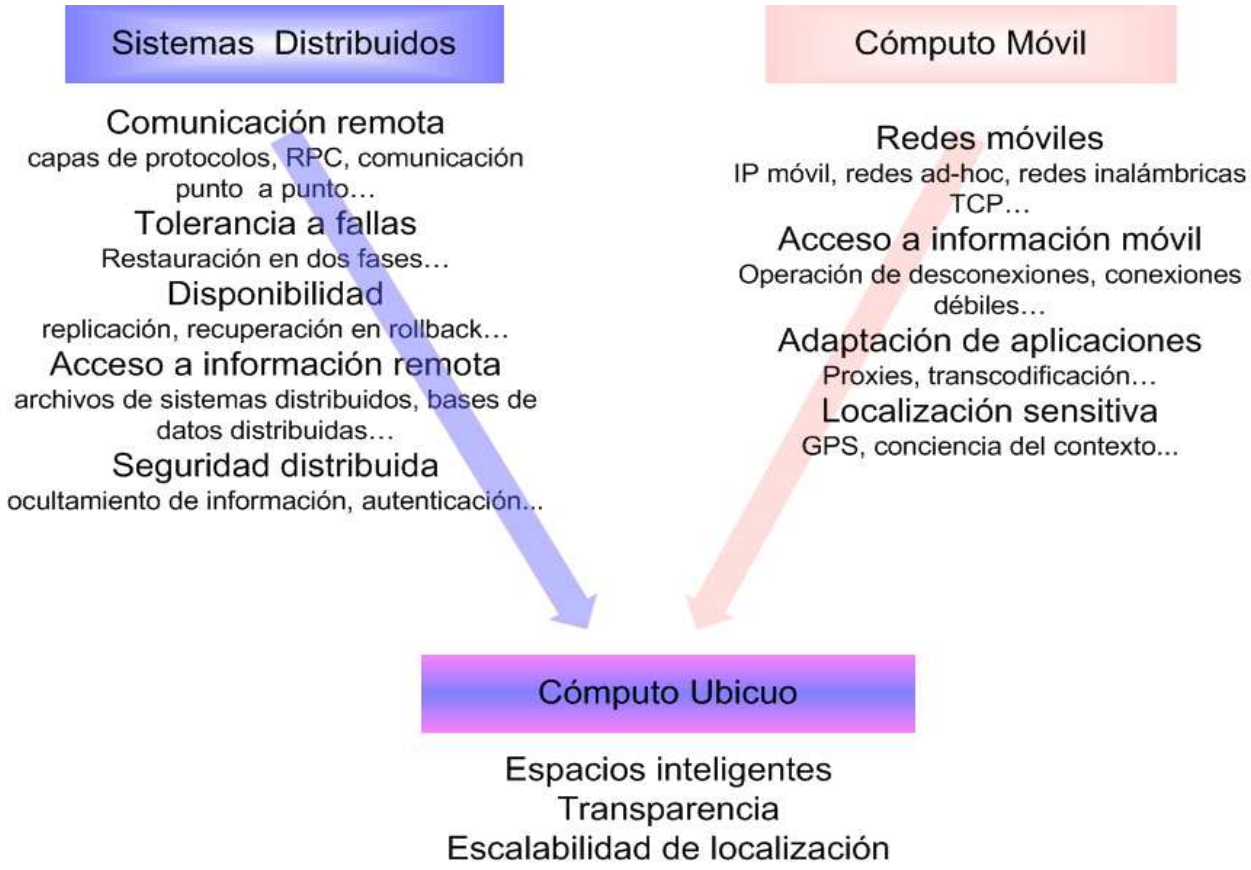

Figura 1. Cómputo ubicuo, nace de la combinación de los sistemas distribuidos y el cómputo móvil.

La computación ubicua pretende construir dispositivos capaces de percibir cambios en el entorno, de igual forma que lo haría una persona. El cómputo ubicuo comparte la idea de convertir los objetos inanimados que nos rodean, como coches, oficinas y ropa, en sensores y actuadores con capacidades de procesamiento y comunicación creando una gran red de dispositivos interconectados. Esto representa el cambio de paradigma desde Interacción Humano-Computadora, por sus siglas en inglés (HCI ó CHI) hacia Interacción Humano-Información (HII). Un tratado más amplio sobre el desarrollo tecnológico de dispositivos para cómputo ubicuo se considera fuera del ámbito de este trabajo. Al lector interesado en

este tema se le invita a leer Pantic *et al.*(2005).

El software social es básicamente el conjunto de herramientas de comunicación que facilitan la interacción y colaboración por medio de convenciones sociales, estas herramientas engloban correo electrónico, mensajes instantáneos, así como cualquier otro tipo de comunidad virtual en red (Lepe *et al.*, 2004).

Los aspectos de middlewares se abordan en secciones posteriores de este capítulo, así como en capítulos más adelante.

A partir de la descripción anterior debe ser evidente que dentro del cómputo ubicuo existen retos como la necesidad de proveer de seguridad informática a los sistemas de cómputo distribuido en general, el análisis y diseño de nuevas arquitecturas, ontologías, las cuales se adapten a los distintos escenarios, que faciliten el tener una visión amplia sobre las nuevas configuraciones y formas de adaptación que se requieren en el desarrollo de aplicaciones sobre el paradigma de sistemas multiagentes.

El cómputo ubicuo como tal (Saha y Mukherjee, 2003), presenta los siguientes puntos clave:

1. Invisibilidad: abarca dos conceptos, el hecho de desaparecer físicamente y desaparecer de la conciencia. El primero pretende hacer dispositivos suficientemente pequeños como para esconderlos dentro o detrás de objetos cotidianos y convertir estos objetos en portales de información particularizada. En el segundo, se pretende saturar el entorno con dispositivos interactuando con el usuario, quien tiene acceso a la información proveída por todos y cada uno de ellos sin tener que explícitamente dirigirse a ninguno en particular.

2. Escalabilidad: los sistemas de cómputo ubicuo consideran el acceso masivo a servicios de información, donde todo individuo en el mundo y artefacto fabricado por el hombre usará y ofrecerá estos servicios. Los mecanismos, algoritmos y elementos integrados en estos sistemas deben de ser capaces de manejar efectivamente desde un puño de instancias hasta millones o cientos de millones.

3. Interoperabilidad: las aplicaciones de sistemas multiagentes comparten información independientemente de las herramientas utilizadas en su desarrollo, lo que conduce a la necesidad de estandarización de interfaces de aplicaciones. Como por ejemplo, los servicios web han integrado dos mecanismos, XML-RPC y SOAP, los cuales especifican todas las reglas necesarias para ubicar servicios Web XML, integrarlos en aplicaciones y establecer comunicación entre ellos.

4. Confiabilidad: la información dentro de los sistemas de cómputo ubicuo, debe ser verídica y ofrecer disponibilidad en cualquier momento para todos los usuarios.

5. Seguridad: como se mencionó arriba, los sistemas de cómputo ubicuo consideran el acceso masivo a servicios de información. Evidentemente, ésto demanda el uso de mecanismos para asegurar la confiabilidad, integridad, autorización, y no-repudio de la información y las transacciones entre usuarios.

6. Privacidad: es un problema de enfoque social al que se enfrenta el cómputo ubicuo. Los usuarios utilizan muchos dispositivos computacionales personales y al mismo tiempo comparten dispositivos públicos con varios usuarios, por lo que existe una discrepancia entre el beneficio que la tecnología de cómputo ubicuo proporciona y los aspectos de privacidad a los que estamos dispuestos a prescindir.

El siglo XXI es de grandes cambios, el cómputo ubicuo busca simplificar nuestras vidas, a través de un ambiente digital, que se adapte a nuestras necesidades y cambios constantes, que sean conscientes de la vida diaria, que interactúen ambos: el mundo físico y el mundo virtual. Para comprender más este nuevo paradigma es necesario conocer algunos conceptos como agentes, middleware, sistemas multiagentes, que hay detrás de ellos y cómo están influyendo para que el cómputo ubicuo sea una realidad.

II.1.2 Agentes

El paradigma de agentes aplica conceptos de la inteligencia artificial y la teoría del acto de conversar en la tecnología de objetos distribuidos (Jennings y Wooldridge, 1998). Un **agente**

es un componente de software que presenta la característica de ser autónomo, proactivo y social.

Autónomo: los agentes tienen un grado de control sobre sus acciones y bajo algunas circunstancias, son capaces de tomar decisiones.

Proactivo: los agentes no sólo reaccionan en respuesta a eventos externos, ellos también presentan un comportamiento en dirección a sus objetivos y son capaces de tomar la iniciativa.

Social: los agentes son capaces y necesitan interactuar con otros agentes en función de cumplir ciertas tareas individuales con el objetivo de alcanzar la meta del sistema.

A diferencia de los sistemas distribuidos típicos, dónde se utiliza una arquitectura cliente/servidor, la arquitectura de interacción entre agentes en sistemas multiagentes es del tipo punto a punto o de igual a igual; esto es, un agente puede ser simultáneamente tanto un ofertador de servicio como un cliente de otro servicio.

Según Jennings (2001), los agentes surgen de la necesidad de ayudar a los usuarios a coordinar actividades y manejar datos en aplicaciones cuya naturaleza dinámica y distribuida exige que el software no sólo responda a requerimientos simples de los usuarios, sino que además se adapte a las circunstancias, se anticipe y busque formas de dar solución a los problemas que se presentan, aún sin la intervención del usuario.

II.1.3 Middleware

Un **middleware** es la capa intermedia entre las aplicaciones y los sistemas operativos (Bellifemine *et al.*, 2003), ver Fig.2.

Un servicio de middleware está situado entre las plataformas y las aplicaciones. El concepto de plataforma involucra todos los servicios y procesos definidos en la arquitectura de los sistemas operativos (Bernstein, 1996).

Los middlewares vienen a simplificar la implementación de los sistemas operativos y hacen que los servicios estén habilitados para múltiples plataformas; éstos deben ser portables, distribuidos y transparentes con respecto a las demás APIs, resolviendo así los problemas

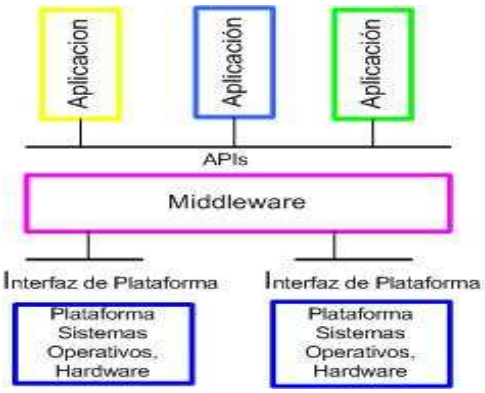

Figura 2. Middleware

de distribución, heterogeneidad e interoperabilidad.

II.1.4 Estándares de middlewares de agentes

La rápida adopción de las tecnologías de agentes por parte de la comunidad computacional, trae consigo algunos problemas de heterogeneidad entre las aplicaciones desarrolladas bajo el paradigma de agentes, por lo que ha sido necesario pensar en la manera de hacer que las aplicaciones cumplan con la característica de ser interoperables. Desde el origen de los agentes han existido esfuerzos por lograr estandarizar su desarrollo y con ello asegurar una larga vida a las aplicaciones, estos esfuerzos se han materializado en estándares, algunos de los cuales se mencionan a continuación:

KQML (Knowledge Query Meta-Language): fue una de las primeras iniciativas en donde se especificaba como brindar un soporte en las interacciones de los agentes usando un protocolo basado en sus conversaciones. No prosperó debido a que no todos los agentes se comunican con el mismo dialecto (Finin *et al.*, 1994).

MASIF (Mobile Agent System Interoperability Facility): define las características para un agente en lo que respecta a la movilidad de una localidad a otra. MASIF no cuenta con una estandarización para la comunicación entre agentes o entre diferentes plataformas. Restringe su operación a los agentes que han sido desarrollados en CORBA (Schoeman y Cloete, 2003).

FIPA (The Foundation of Intelligent Physical Agents): está enfocado al soporte de la interoperabilidad de agentes desarrollados sobre marcos de trabajo en los cuales se soporten transportes heterogéneos. Se busca la fácil interoperabilidad entre los sistemas de agentes, en particular, define al modelo referencial de una plataforma de agente y un conjunto de servicios que deben proporcionarse. La colección de éstos servicios y su interfaz estándar, representan las reglas normativas que permiten a una sociedad de agentes existir, operar y ser manejados. En la actualidad se puede considerar el estándar más ampliamente reconocido y extendido internacionalmente, y se ha convertido en un referente a seguir para el desarrollo de sistemas basados en agentes (Georgousopoulos y Rana, 2002).

Una de las motivaciones del presente trabajo es, el desarrollo de sistemas multiagentes con capacidades móviles, los cuales día a día toman fuerza, por lo que será necesario el fortalecer los estándares como FIPA o MASIF, ó tomar la alternativa de creación de nuevos estándares que cumplan con las nuevas necesidades de los sistemas multiagentes.

II.1.5 Lenguajes de comunicación de agentes (ACL)

La visión sobre el comportamiento social, cooperativo y comunicativo entre los sistemas multiagentes, condujo a la necesidad de establecer un estándar para el acto comunicativo entre agentes.

Cualquier acto comunicativo involucra tres aspectos: 1) el método de paso de mensajes, 2) formato o sintaxis de la información que será transferida y 3) el significado o semántica de la información (en este caso del mensaje).

Las conversaciones entre agentes siguen en algunas ocasiones unos patrones determinados, que se repiten en muchos casos; aprovechando estos patrones y su repetición se han definido protocolos. Un protocolo es un patrón que se usa para llevar por unos cauces concretos una conversación, son conversaciones guiadas, en las que cada agente sabe qué mensaje enviar y cuales puede recibir. Es necesario definir performativas (actos comunicativos) y un lenguaje para que los agentes puedan comunicarse y entenderse. Por último, se requiere de entender el significado del contenido de los mensajes, esto se provee con un componente

dentro del ACL llamado ontología donde se define el vocabulario a usar en los mensajes, este vocabulario cambia dependiendo de la aplicación en que trabajen los agentes. Ahora, agentes que comparten un lenguaje común y una ontología pueden comunicarse de manera efectiva tanto sintáctica como semánticamente.

A continuación se mencionan dos estándares que se consideran relevantes, primero el que representa las bases y segundo el estándar de facto adoptado para el desarrollo de sistemas bajo el paradigma de agentes.

Knowledge Query Meta-Language o KQML (Poslad *et al.*,), fue una de las primeras iniciativas para el soporte de la interacción social entre los agentes usando un protocolo basado en actos comunicativos. Sin embargo, KQML no fue adoptado como el estándar de facto debido a la falta de bases y solidez en sus especificaciones.

Uno de los principales problemas con KQML, fue el no contar con una semántica definida, ya que el uso de performativas(actos comunicativos) es insuficiente para garantizar que los mensajes vayan a ser correctamente interpretados por otros agentes.

Foundation for Intelligent Physical Agents - Agent Communication Lenguaje o FIPA-ACL (FIPA, 2002a)(FIPA, 2003)(FIPA, 2001)(FIPA, 2002c), su objetivo fue el de estandarizar los mensajes intercambiados entre los agentes con el fin de lograr la interoperabilidad y proveer a los agentes de una buena definición de procesos en sus conversaciones. FIPA define los tipos de performativas denotando el tipo de conversación, quienes son los participantes en la comunicación, contenido del mensaje, descripción del contenido del mensaje, mediante el uso de lenguajes (SL - Semantic Language) y ontologías, además del control de la conversación a través de protocolos.

II.1.6 Movilidad en agentes

En sistemas de cómputo móvil y/o ubicuo, el modelo de movilidad puede ser implementado ya sea facilitando el desplazamiento físico de dispositivos que alojan aplicaciones, o permitiendo que las aplicaciones se trasladen entre dispositivos físicamente distribuidos, o

ambos simultáneamente. Esto es, por un lado es necesario proveer a los dispositivos de mecanismos para detectar el desplazamiento físico y el cambio de entorno de conectividad, así como de mecanismos en el sistema operativo del dispositivo para reconfigurar las sesiones de comunicación entre las aplicaciones locales y remotas. Estos problemas son abordados por tecnologías como Mobile-IP y otros (Ghosh, 2000), y no es el enfoque de este trabajo de tesis. En este trabajo se abordará la problemática de permitir que las aplicaciones se trasladen entre dispositivos físicamente distribuidos, en particular cuando las aplicaciones son sistemas multiagentes.

En sistemas multiagentes, el modelo de movilidad para la aplicación se implementa permitiendo que un agente, el cual es llamado agente móvil, abandone el entorno de ejecución dónde fue creado y se incorpore a un nuevo entorno de ejecución. Los entornos de ejecución, origen y destino, pudieran estar en dispositivos físicamente distribuidos.

Según Lange y Oshima (1999), el interés por el manejo de agentes móviles no es motivado por la tecnología, pero si por los beneficios que éstos ofrecen para los sistemas distribuidos y algunas necesidades que se han presentado. Algunos de estos beneficios son:

1. Los agentes móviles reducen el tráfico en la red: ya que se ejecutan asíncrona y autónomamente, las tareas pueden ser embebidas en los agentes móviles, los cuales pueden ser enviados por la red. Un agente móvil también es homogéneo ya que se ejecuta y transporta, independientemente del ambiente de ejecución, por lo que proveen las condiciones óptimas para la integración de sistemas.

2. Ocultar latencia de red en aplicaciones de tiempo real: en sistemas de tiempo real críticos, es importante que las instrucciones de control se ejecuten sin retraso. Los agentes pueden resolver el problema al ser despachados desde un controlador central e instalarse en los componentes controlados.

3. Encapsulado de los protocolos de comunicación: se utilizan para manejar nuevos requerimientos como la autenticación, el cifrado, el control de acceso e instalación y actualización de nuevos protocolos en todos los nodos.

4. Capacidades de procesamiento mejoradas para dispositivos móviles: se espera que la tecnología de agentes móviles se apliquen a la computación móvil, por ejemplo en los teléfonos celulares y en los organizadores personales (PDA´s). Debido a que los dispositivos móviles tienen baja capacidad de procesamiento, será mejor para estos enviar un agente al servidor que procese ahí las aplicaciones.

Algunas necesidades (o requerimientos) de los sistemas multiagentes son:

1. Deben ser robustos y tolerantes a fallas: si un nodo se apaga, todos los agentes que se están ejecutando en ese nodo son avisados y toman tiempo para detener sus tareas y continuar sus operaciones en otra máquina disponible que se encuentre en la red.

2. Deben adaptarse dinámicamente: los agentes móviles están constantemente sensando su ambiente de ejecución y reaccionan autónomamente a los cambios. Se pueden encontrar múltiples agentes móviles, cuya única actividad es la de distribuirse a través de la red con el objetivo de mantener una configuración óptima para resolver algún problema en particular.

Actualmente las aplicaciones de agentes han ido incrementando, con lo que se ha demostrado su gran versatibilidad en aplicaciones como: comercio electrónico, recuperación de información distribuida, turismo, servicios de redes de telecomunicaciones, aplicaciones de workflow y groupware, monitoreo y notificación, procesamiento paralelo, mar de datos, entre otros.

Movilidad en Sistemas Multiagentes

La movilidad en los sistemas multiagentes se basa en el concepto de que los agentes se componen de código, datos y estado. Un agente puede clonarse o migrarse de un entorno de ejecución de agentes a otro, de aquí en adelante llamados contenedores, o de una plataforma de agentes a otra, la cual se conforma de un conjunto de contenedores, según lo requieran para cumplir sus tareas y objetivos.

Clonar un agente: significa realizar una copia del agente ya sea en el mismo contenedor donde ha sido creado o en otro distinto.

Migrar un agente: involucra enviar el código y estado del agente a través de la red, ya sea a un contenedor de la misma u otra plataforma de agentes.

En el desarrollo de agentes móviles, deben considerarse algunas diferencias con respecto a los agentes en general, las cuales se mencionan a continuación:

Ciclo de vida: representa los estados en los que se encuentra en determinado momento el agente, los estados del ciclo de vida son: inicio, activo, suspendido, esperando, eliminado, en tránsito. El estado *en tránsito* es exclusivo para los agentes móviles.

Ontología: define el vocabulario a usar en la comunicación entre los agentes. Este vocabulario cambia dependiendo de la aplicación en que trabajen los agentes, agentes que comparten un lenguaje y una ontología pueden comunicarse de manera efectiva tanto sintáctica como semánticamente. Las aplicaciones desarrolladas con agentes móviles, deben definir su ontología.

Protocolos de interacción: un protocolo de interacción es una conversación, una secuencia predeterminada de mensajes intercambiados entre agentes. Con un protocolo de interacción, un agente sabe que mensajes puede enviar a otro agente y los mensajes que puede recibir. Actualmente, no existe un protocolo que se llame exclusivo para la movilidad de agentes en sistemas multiagentes.

Otro tipo de movilidad contemplada en los sistemas multiagentes es la movilidad de dispositivos, tales como teléfonos celulares y organizadores personales, en donde los agentes se encuentran de manera constante sensando el ambiente en busca de nuevos dispositivos.

II.1.7 Seguridad en agentes

Seguridad es toda aquella acción que tiende a garantizar el cumplimiento de algunos o al menos uno de seis objetivos primordiales: autenticación, autorización, confidencialidad, integridad, disponibilidad y no repudio (Johner *et al.*, 2000).

1. Autenticación: es el proceso de verificar que alguien o algo es quien o lo que dice ser. En redes de equipos públicos y privados (incluyendo Internet), la autenticación se lleva a cabo comúnmente a través de contraseñas de inicio de sesión, también se suele hacer

uso de sistemas criptográficos basados en claves asimétricas, como firmas digitales o certificados digitales.

2. Autorización: es el derecho otorgado a un individuo o proceso, para utilizar el sistema y la información almacenada en éste. Típicamente la autorización es definida por un administrador de sistemas y verificado por el equipo basado en alguna identificación del usuario, como son un nombre de usuario o una contraseña.

3. Confidencialidad: la información sólo ha de poder ser accedida por las personas autorizadas. Puede ser confidencialidad de datos, protegiéndolos por medio de ocultamiento de información, o confidencialidad de flujo en la que se trata de ocultar quien es el destinatario, cuál es el contenido y el momento del tráfico.

4. Integridad: es necesario saber que los datos recibidos no han sido modificados por alguien distinto del emisor. Además la integridad de secuencia, asegura que los bloques de información recibida han llegado en el orden en que fueron enviados.

5. Disponibilidad: la información siempre debe de estar accesible para las entidades autorizadas, se asocia con el momento en que ocurre un ataque de negación de servicio, el cual puede poner lento o paralizar totalmente al sistema.

6. No repudio: es la habilidad de identificar quien ha llevado a cabo varias acciones en un equipo de cómputo, para que los usuarios no puedan negar las responsabilidades de las acciones que ellos llevan a cabo. Generalmente utilizado en el sentido de crear una huella de auditoría indiscutible para identificar la fuente de una transacción comercial o acciones maliciosas.

Seguridad en Sistemas Multiagentes (MAS)

En la Figura 3, se muestra el flujo normal de la información entre dos agentes que se están comunicando y los tipos de ataques a los cuales se enfrentan los agentes en el momento de su comunicación.

Ataques pasivos:

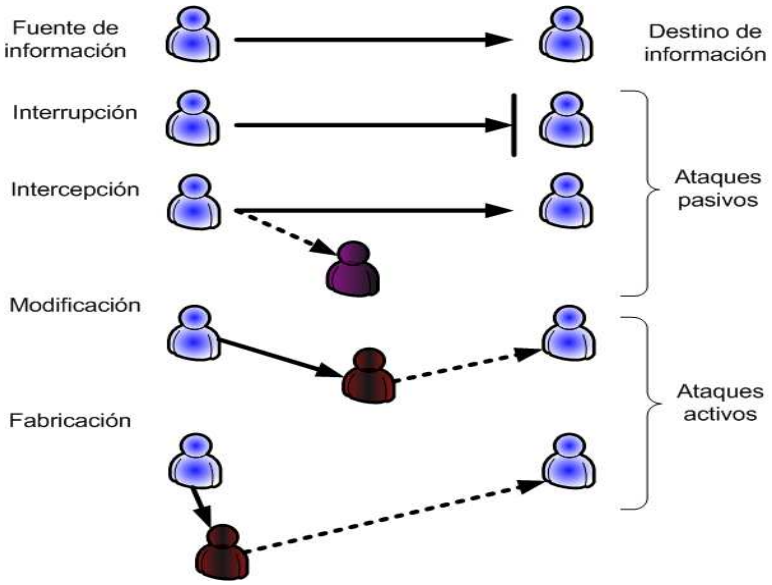

Figura 3. Tipos de ataques a sistemas multiagentes

Interrupción: un recurso del sistema es destruido o se vuelve no disponible. También es conocido como Negación de Servicio.

Intercepción: una entidad no autorizada consigue acceso a un recurso.

Ataques activos:

Modificación: una entidad no autorizada no sólo consigue acceder a un recurso, sino que es capaz de manipularlo.

Fabricación: una entidad no autorizada inserta objetos falsificados en el sistema; esto es, un agente se hace pasar por otro.

II.1.8 Agentes móviles seguros

Según Bierman y Cloete (2002), los agentes móviles se enfrentan a nodos, agentes y entidades maliciosas, las cuales tratan que el agente móvil no cumpla con sus objetivos, utilizando cualquiera de los ataques antes mencionados.

En la Tabla I, se muestran los tipos de ataques a los que se enfrenta un agente móvil, la clase de subataque y como combatirlo.

Ataques a la integridad: implica que la integridad de un agente móvil pueda ser

Tabla I. Tipos de ataques a un agente móvil

Clase de amenaza	Suclase de amenaza	Combate
Ataques a la integridad	Integridad de código Integridad de información	Computación basada en la verdad
Habilidad de refutar recursos	Denegación de servicios Retardos en el servicio No transmisión	Combate basado en el registro de trayectoria
Ataques a la confidencialidad	Invasión a la privacidad Robo Ingeniería inversa	Técnicas de criptografía
Riesgos de autentificación	Clonación Enmascaramiento	Técnicas basadas en tiempo

violada, ya sea por motivos maliciosos o accidentales. En general, se pueden encontrar dos formas en las que la integridad de un agente móvil se ve comprometida. a) Integridad de código, sucede cuando un nodo interfiere en la ejecución de un agente móvil, aún sin alterar el estado del agente o su información; esto es, el nodo manipula o altera el código del agente al transmitirlo o recibirlo, o no deja que el agente se ejecute de manera completa. b) Integridad de información, ocurre cuando un nodo interfiere las comunicaciones entre agentes mediante alguno de los ataques arriba descritos. También se puede hablar de alteración, corrupción, manipulación, borrado, o una incorrecta ejecución de un agente.

Habilidad de refutar recursos: cuando un agente móvil llega a un nodo, debe de contar con permisos para acceder y utilizar los recursos necesarios para llevar a cabo sus tareas. Un nodo malicioso puede interferir con las tareas de un agente móvil que recién llega mediante ataques pasivos, ya sea negándole o retrasando de manera indefinida el acceso a uno o más recursos necesarios, ó no transmitiendo al agente hacia otro nodo, independientemente de su itinerario, lo que puede ser negativo para los propósitos del agente móvil.

Ataques a la confidencialidad: ocurre cuando un agente móvil es accesado ilegalmente por un agente malicioso, aplicándole ingeniería en reversa con el propósito de analizar sus datos y su código, e incluso quedarse con una copia de los mismos sin autorización, y poderlo manipular.

Riesgos de autenticación: un agente móvil debe ser capaz de identificar las credenciales tanto de los nodos donde se va a ejecutar, como de los agentes con los que se va a comunicar. Evitando con esto, que un nodo se haga pasar por uno de los que se encuentran

en el itinerario del agente móvil, o que un agente clonado con maliciosas intenciones logre interactuar con el agente móvil, evadiendo los ataques de fabricación y modificación a los que se ven expuestos los agentes móviles.

Para resolver estos problemas, se han identificado 4 tipos de combates:

Computación basada en la verdad: crear un ambiente seguro para los agentes móviles, usando políticas de seguridad, adicionar software o hardware para ofrecer mayor seguridad.

Combate basado en el registro de trayectoria: un agente lleva consigo un registro de los nodos visitados en los cuales le es permitido ejecutarse, permitiéndole llevar un historial de sus acciones e iteracciones con los demás agentes.

Técnicas de criptografía: se utilizan algoritmos de cifrado/descifrado, llave pública y privada, firmas digitales, funciones hash y certificados.

Técnicas basadas en tiempo: lo básico de esta técnica es adherir restricciones sobre el tiempo de vida de los agentes móviles.

La implementación de alguna técnica de combate con la cual ofrecer seguridad aceptable en agentes móviles, es una verdadera preocupación, por lo que es muy recomendable seleccionar y combinar algunas técnicas de acuerdo a las necesidades.

II.2 Marco teórico

II.2.1 Alcances

Existe una gama de middlewares para sistemas multiagentes que varían en sus características, prestaciones e implementaciones. Trabajar con cualquiera de ellos para extenderlo y mejorarlo requiere de no poco tiempo y esfuerzo. Por lo tanto, con el objetivo de acotar el tiempo de desarrollo de este trabajo de tesis, nos dimos a la tarea de hacer un análisis comparativo de los diferentes middlewares para seleccionar uno. La selección se basó en un middleware que estuviera siendo activamente desarrollado, que se apegara a alguna norma internacional vigente, que fuera del interés de la industria y que no tuviera resuelto de manera efectiva las cuestiones de agentes móviles seguros.

A continuación se presenta el análisis realizado en los middlewares más difundidos, y se muestra el cuadro comparativo en la Tabla II.

1. JADE, se especifica para el desarrollo de agentes móviles y estacionarios, basado en el lenguaje Java, su código fuente es abierto y ha sido adoptado tanto por la academia como por la industria. Su arquitectura base se apega a las especificaciones del estándar FIPA, además presenta propuestas para el desarrollo de agentes móviles y seguridad para agentes tanto móviles como estacionarios.

2. FIPA-OS, se apega a las especificaciones del estándar FIPA, desarrollado bajo el lenguaje Java, su código fuente es abierto y se limita al desarrollo de agentes estacionarios sin especificaciones de seguridad, debido a que el estándar FIPA no presenta especificaciones vigentes para seguridad y movilidad de agentes.

3. Aglets, se especifica para el desarrollo de agentes móviles, basado en el lenguaje Java, es de libre distribución, no especifica un apartado referente a seguridad de agentes y se basa en el estándar MASIF, donde se definen las características para un agente con respecto a la movilidad de una localidad a otra. MASIF no cuenta con una estandarización para la comunicación entre agentes o entre diferentes plataformas, restringe su operación a los agentes que han sido desarrollados en CORBA.

4. Aglets y Grasshopper, se especifican para el desarrollo de agentes móviles, basados en el estándar FIPA y el lenguaje Java, cuentan con un prototipo, un agente llamado GUEST, con el cual se busca la interoperabilidad en plataformas heterogéneas y se busca que también sea soportado por JADE.

5. Ajanta, se especifica para el desarrollo de aplicaciones de agentes móviles, basado en los lenguajes Java y Perl, su utilización sólo ha sido a nivel académico presentando propuestas de solución en problemas de seguridad a los que se enfrentan los agentes móviles, aunque con el inconveniente de no basarse en un estándar en el cual sustentar

Tabla II. Comparaciones de Middlewares

Nombre, Compañia y URL	Tipo de agentes Estacionario/Móvil	Estándar	Lenguaje	Disponibilidad	Modulo de seguridad	Aplicaciones
JADE (CSELT) http://jade.tilab.com/	Estacionarios Móviles (intra-plataformas)	FIPA	Java	Open Source	si	Asistente de viajes aplicaciones académicas industria
FIPA-OS (Nortel Networks) http://www.nortelnetworks.com/fipa-os	Estacionarios	FIPA	Java	Open Source	no	Personalización de servicios VPN,VHE planificador de tareas
AGLETS (IBM Japón) http://aglets.sourceforge.net/	Móviles	MASIF	Java	Open Source	no	Monitoreo remoto (ej. Web site planificador de tareas
GRASSHOPPER (ikv++) http://www.grasshopper.de/	Móviles	MASIF/FIPA	Java	Código binario libre /sobre demanda	no	Comercio electrónico recuperación de datos
AJANTA (University of Minnesota) http://www.cs.umn.edu/Ajanta/	Móviles	-	Java,Perl	Académico	si	Aplicaciones a través del web.
ZEUS (BT UK) http://zeus.objectweb.org/	Estacionarios	FIPA	Java KQML	Open Source	no	

su arquitectura, lo que lo hace no interoperable con aplicaciones desarrolladas bajo este mismo middleware.

6. Zeus, se especifica para el desarrollo de agente estacionarios, toma como referencia el estándar FIPA, desarrollado bajo el lenguaje Java y su código fuente es abierto. Este middleware no presenta en sus especificaciones módulos referentes a seguridad de agentes.

Una vez analizados los diversos middlewares, en base al interés en seguridad y movilidad de agentes, el middleware de nuestra elección es JADE. En las siguientes secciones se presentan los detalles del estándar FIPA y JADE que nos servirán como fundamento para desarrollar la propuesta de agentes móviles seguros en JADE.

II.2.2 Estándar FIPA

FIPA [1] comenzó sus actividades en 1995 con el objetivo de estandarizar aspectos relacionados con la tecnología de agentes y sistemas multiagentes. En la actualidad se puede considerar el estándar más ampliamente reconocido y extendido internacionalmente, y se ha convertido en un referente a seguir para el desarrollo de aplicaciones basados en agentes. FIPA define un entorno de ejecución para agentes llamado plataforma (ver Fig. 4), la cual provee sustento al ciclo de vida del agente y las comunicaciones entre agentes, fundamentalmente. Para facilitar el despliegue de una plataforma a través de varias computadoras se definen contenedores como secciones del entorno de ejecución, existiendo siempre un contenedor principal y cero, uno o varios contenedores secundarios. El contenedor principal aloja agentes especializados para la administración de la plataforma. El Agent Management System (AMS) representa la autoridad de la plataforma, es el encargado de proporcionar los servicios de páginas blancas, administrar el ciclo de vida de los agentes, mantener el directorio de los identificadores de agentes (AID Agent Identifier) y su estado. El Directory Facilitator (DF) provee las páginas amarillas, lo que significa que a través de él un agente puede localizar y solicitar servicios

[1] http://www.fipa.org/

proporcionados por otros agentes de la plataforma (FIPA, 2004).

El middleware JADE toma como base las especificaciones de este éstandar para su desarrollo.

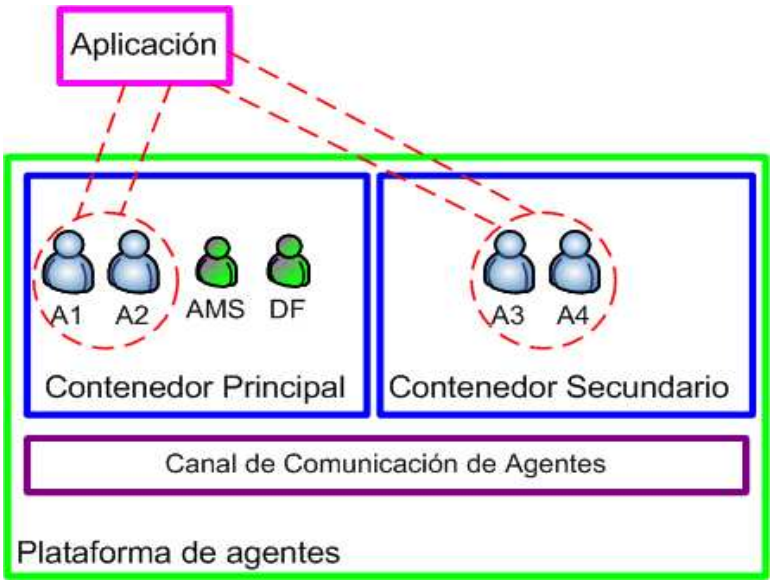

Figura 4. Plataforma definida por FIPA y adoptada por JADE

II.2.3 Middleware JADE

De acuerdo con Bellifemine *et al.* (2003), JADE (Java Agent DEvelopment Framework) es un middleware desarrollado por TILAB para aplicaciones distribuidas multiagentes basados en la arquitectura de comunicación igual-a-igual (peer-to-peer). Se ha convertido en el estándar de facto en su tipo utilizado tanto por la comunidad académica y la industria. JADE esta totalmente desarrollado en Java, es de libre distribución, su código fuente es abierto y cumple con las especificaciones de FIPA (ver Fig.5).

JADE es un middleware que facilita el desarrollo de sistemas multiagentes (Caire, 2003) e incluye un ambiente de ejecución, una librería de clases y herramientas gráficas. El ambiente de ejecución es dónde los agentes de JADE "viven", mismo que debe ser activado antes de que uno o más agentes puedan ser ejecutados en el nodo. La librería de clases es para el desarrollo de los agentes y está pensada para ser usada por los programadores de

aplicaciones. Finalmente, las herramientas gráficas permiten la administración y monitoreo de las actividades que los agentes realizan durante su ejecución, incluidos el AMS y DF.

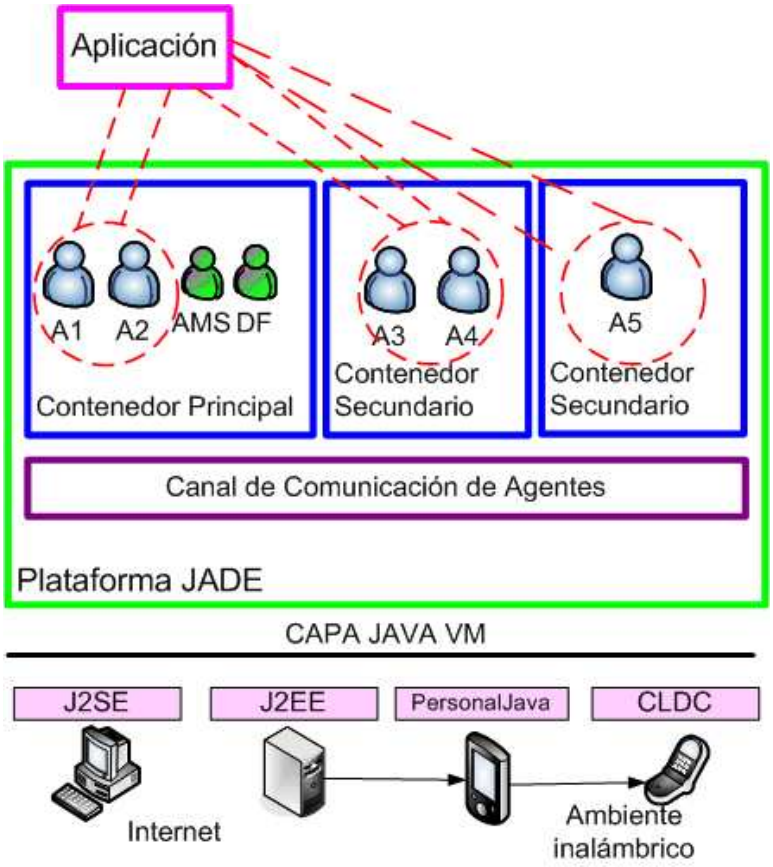

Figura 5. Arquitectura JADE

El ambiente de ejecución de JADE se apega a la especificación FIPA y por lo tanto se organiza en plataformas y contenedores. Asimismo, el contenedor principal de una plataforma implementa los agentes AMS y DF. Adicionalmente, JADE implementa un tercer agente administrativo en el contenedor principal llamado ACC (Canal de Comunicación de Agentes), que es el encargado de controlar el intercambio de mensajes. JADE está diseñado de tal manera que una máquina virtual de Java puede soportar en un solo nodo uno o más contenedores, cada uno con su propio ambiente de ejecución y servicios para uno o mas agentes. En la Figura 6, se ilustra un escenario con 2 plataformas JADE, compuestas de un contenedor principal y uno secundario respectivamente. Los agentes de JADE son

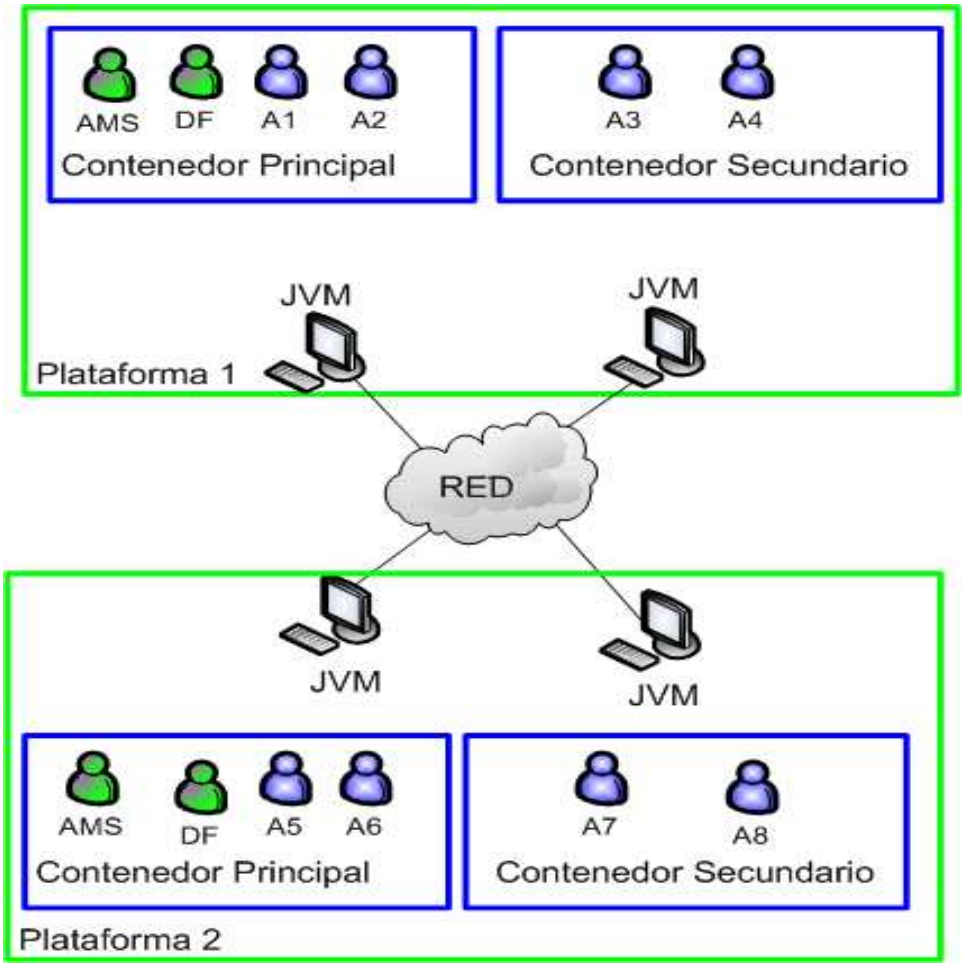

Figura 6. Dos plataformas JADE

identificados por un único nombre, y ellos conocen el de los otros agentes, por lo que pueden comunicarse independientemente de su localización; esto es, en el mismo contenedor, diferentes contenedores o diferentes plataformas.

El modelo de comunicación entre agentes es asíncrono. Los agentes envían/reciben objetos Java que representan mensajes ACL (Lenguaje de Comunicación de Agentes) , por medio del lenguaje SL0 (Lenguaje Semántico 0) definido por FIPA y se manejan colas de mensajes, además pueden añadirse otros protocolos a través de las interfaces MTP (Protocolo de Transporte de Mensajes) e IMTP(Protocolo de Transporte de Mensajes Interno) de JADE. Desde el punto de vista de comunicación se distinguen dos escenarios: comunicación intra-plataforma y comunicación inter-plataforma (Cortese *et al.*, 2003). En la primera, los

agentes que desean comunicarse residen en una misma plataforma ya sea en el mismo o en diferentes contenedores. En la segunda, los agentes residen en diferentes plataformas (ver Fig.7 y Fig.8).

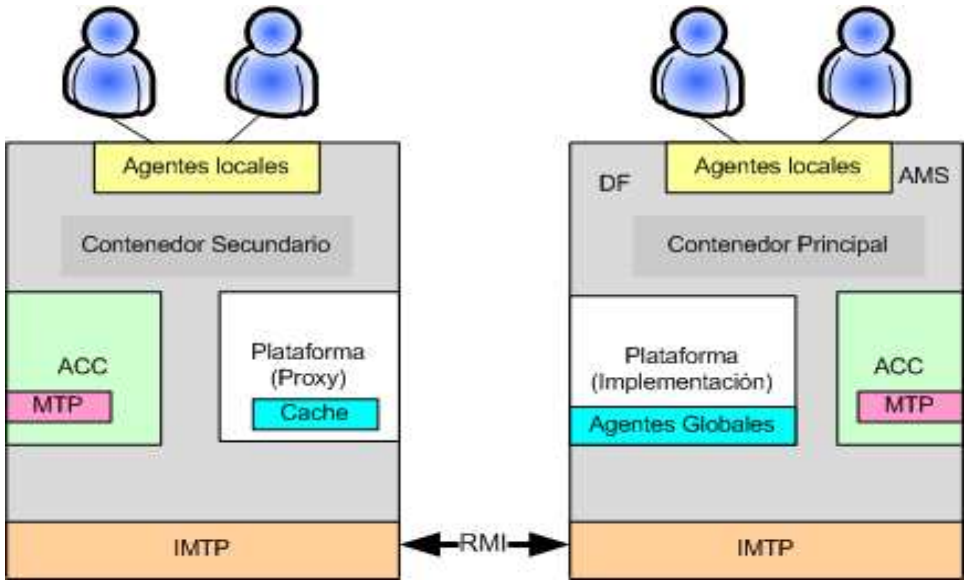

Figura 7. Diagrama de implementación de comunicación intra-plataforma

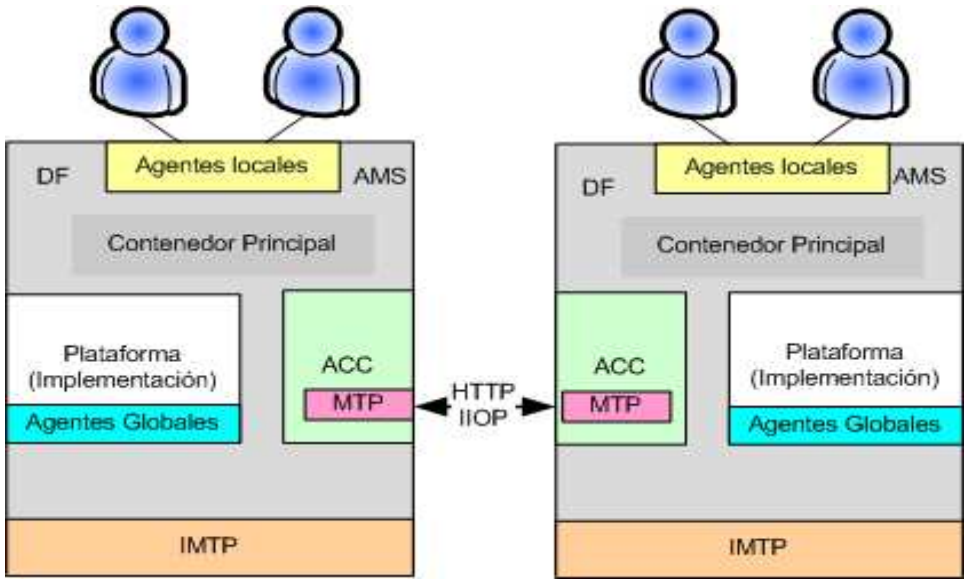

Figura 8. Diagrama de implementación de comunicación inter-plataforma

De acuerdo a las especificaciones de FIPA se han definido algunos MTPs, particularmente

basados en IIOP, HTTP y WAP. JADE adopta estos últimos, donde la comunicación inter-plataforma se realiza a través del ACC (Canal de Comunicaciones de Agentes), el cual está distribuido a través de todos los contenedores. Cada contenedor tiene su propio MTP y el ACC tiene la tarea de asegurarse que los mensajes lleguen al contenedor apropiado.

Para el IMTP, las especificaciones de FIPA no tienen una definición particular. En este caso, JADE adopta RMI(Remote Method Invocation) de Java y event passing.

La plataforma JADE, además nos permite extender la funcionalidad de cualquier aplicación desarrollada con "agregados", como por ejemplo Jade-Leap, éste es utilizado para el desarrollo de aplicaciones en dispositivos móviles. El agregado base estudiado en la presente investigación es JADE-S, el cual proporciona a las aplicaciones seguridad. Existen otros agregados para JADE, los cuales se pueden aplicar a nuestros sistemas de acuerdo a las necesidades, la información se encuentra en la página principal de JADE [2].

II.3 Trabajos relacionados

En esta sección se presentan algunos trabajos previos relacionados con la seguridad de sistemas multiagentes y movilidad.

En Novák *et al.* (2003), se propone una arquitectura de seguridad y su implementación mediante el sistema llamado X-Security para sistemas multiagentes. Este trabajo fue desarrollado bajo el middleware JADE en versiones anteriores a la utilizada en éste trabajo de tesis; se introduce el concepto de agentes seguros e inseguros, debido a que parte del concepto de seguridad lo concentran en el núcleo del agente; el cual, es independiente de la forma de comunicación entre ellos, solucionando los problemas de autenticación y comunicación segura entre agentes ya sea móviles o estacionarios ubicados en una misma plataforma. En X-Security proponen agregar a la plataforma una Autoridad Central de Seguridad (SCA), X-Security no se apega al estádar FIPA aunque si presenta una alternativa de solución a la seguridad de sistemas multiagentes. El presente trabajo de tesis es diferente debido a que la seguridad es otorgada por los servicios ofrecidos por la plataforma y el apego al estándar

[2] http://jade.tilab.com/

FIPA.

En Ametller *et al.* (2003a), toman en consideración el escenario de MdD (Mar de Datos) donde proponen la implementación de un mecanismo de movilidad inter-plataforma para agentes basados en FIPA-ACL; esto es, los agentes migran utilizando el sistema de mensajes entre agentes dirigido por la plataforma mediante el AMS. Diseñaron un gestor de migración, su propio cargador de clases, los módulos de conversación; es decir, se implementó un protocolo para la comunicación inter-plataformas, donde la migración esta dirigida por la plataforma; esto es, el agente viajará en un mensaje entre los AMS de cada una de las plataformas implicadas en la migración. El presente trabajo de tesis no implementa un nuevo protocolo, sino utiliza los ya establecidos en las interacciones de agentes en la plataforma JADE.

En los trabajos de Ametller *et al.* (2004, 2003b) proponen un mecanismo de protección flexible en el contexto de agentes móviles a nivel intra-plataforma mediante la utilización de criptografía de llave pública. En este esquema los agentes protegen su propio código y datos, debido a que llevan consigo su propio mecanismo de protección, esta solución afecta tanto la estructura del agente como la de la plataforma. En el presente trabajo de tesis se aborda el tema de autenticación mediante el uso de certificados, proporcionando las bases para que los agentes puedan usar la criptografía de llave pública, con el fin de tener un eslabón mas en cuestiones de seguridad.

En Robles *et al.* (2002) proponen una plataforma llamada MARISM-A, desarrollada para agentes móviles seguros que provee mecanismos para proteger la migración, confidencialidad e integridad de los agentes. Basada en el estándar MASIF, en este trabajo definieron su propio protocolo de comunicación, utilizaron la infraestructura de llave pública, el protocolo de migración SSL para dotar de confidencialidad, autenticidad y no-repudio a la plataforma de agentes. El presente trabajo de tesis se basa en el estándar FIPA y el uso de certificados.

Existen otros trabajos relacionados como RETSINA (Shehory y Sycara, 2000) y SeMoA (Roth y Jalali-Sohi, 2001); los cuales, se han tomado como base para las actuales propuestas de protección de mensajes y agentes. La mayoría de las propuestas son a nivel

intra-plataforma, las soluciones son a través de la utilización de técnicas criptográficas y el empleo del concepto PKI (Public Key Infrastructure). Sin embargo, estos trabajos no se desarrollan bajo algún estándar como FIPA o MASIF, ni bajo la plataforma JADE. Lo cual enfatiza la importancia de estandarizar o proporcionar bases para la seguridad en sistemas multiagentes.

II.4 Conclusiones

Es verdad que la convergencia tecnológica de las comunicaciones inalámbricas, la miniaturización electrónica y el robustecimiento del software, permite pensar en la materialización del concepto del cómputo ubicuo. En esto, los sistemas multiagentes, gracias a sus propiedades, jugarán un papel importante. Entre estas propiedades el presente trabajo aborda la seguridad y la movilidad de agentes. Los agentes móviles presentan beneficios para los desarrolladores de sistemas multiagentes, sin embargo también presentan nuevas vulnerabilidades en seguridad. La definición de estándares es fundamental en todos los ámbitos de la industria y en los sistemas multiagentes no es la excepción. El concepto de middlewares facilitará la construcción de sistemas de cómputo ubicuo basado en sistemas multiagentes. En este sentido, el estándar de FIPA y su implementación en Java, el middleware JADE, se presentan como una buena opción.

En este capítulo se estudiaron los fundamentos teóricos de la seguridad informática y movilidad en los sistemas multiagentes; en el siguiente capítulo, se analizarán los aspectos de seguridad en sistemas multiagentes.

Capítulo III

Análisis de los mecanismos y protocolos de seguridad para sistemas multiagentes

"No desesperes, ni siquiera por el hecho de que no desesperas. Cuando todo parece terminado, surgen nuevas fuerzas. Esto significa que vives".
Franz Kafka

III.1 Introducción

El middleware JADE en su agregado JADE-S, toma las bondades que ofrecen los mecanismos de seguridad del lenguaje Java, aplicándolos a su filosofía de dotar de seguridad a las aplicaciones que se desarrollen bajo éste middleware. JADE, propone la utilización de protocolos seguros para implementar la comunicación a nivel intra-plataforma e inter-plataforma, tales como https, rmi-ssl y jicp-ssl.

En este capítulo se presenta el análisis realizado al agregado JADE-S, los mecanismos de seguridad y protocolos que resultan de interés para el desarrollo del presente trabajo, así como el estado actual de la seguridad en el estándar FIPA y el middleware JADE.

III.2 Mecanismos de seguridad

III.2.1 Seguridad en Java

Desde luego, estando JADE escrito en Java, para este trabajo es importante tener por lo menos una idea básica de los mecanismos y las herramientas de seguridad del lenguaje Java.

En Java 1.2 se introducen nuevas características que mejoran el soporte y el control de la seguridad, tales como control de acceso de grano fino, control de acceso aplicado a todo el código, facilidad de configuración de políticas de seguridad y una estructura de control

de acceso extensible. El control de acceso de grano fino introduce un sistema de control de permisos de grano fino, que permite dar permisos a trozos de código específicos para acceder a ciertos recursos en el cliente, dependiendo de la firma del código y/o el URL del que este se obtuvo; control de acceso aplicado a todo el código, el concepto de código firmado es ahora aplicable a todo el código independientemente de su procedencia (local o remoto); facilidad de configuración de políticas de seguridad, la nueva arquitectura de seguridad permite el ajuste sencillo de los permisos de acceso usando un archivo de políticas (policy file), en el que se definen los permisos para acceder a los recursos del sistema para todo el código (local o remoto, firmado o sin firmar). Gracias a ello, el usuario puede bajar aplicaciones de la red, instalarlas y ejecutarlas asignándoles sólo los permisos que necesiten; por último la estructura de control de acceso extensible, permite definir permisos que representan un acceso a recursos del sistema y el control automático de todos los permisos de un tipo correcto, lo que repercute en que en la mayoría de los casos, se hace innecesario añadir métodos al gestor de seguridad.

En Java 1.2 se incluyen tres herramientas de seguridad:

1. keytool. Es una herramienta para la generación de pares de llaves (públicas y privadas), importar y exportar certificados X.509 (versiones 1, 2 y 3), generar certificados X.509 V1 autofirmados y gestionar almacenes de llaves (keystores).

2. jarsigner. Se emplea para firmar archivos JAR y verificar las firmas de archivos JAR ya firmados. La herramienta emplea los almacenes de llaves para buscar los datos que necesita para firmar archivos (una llave privada), verificar firmas (llave pública) o verificar llaves públicas (certificados). Esta herramienta junto a la anterior reemplazan al antiguo javakey.

3. policytool. Se emplea para crear y modificar los archivos de configuración de políticas de seguridad del cliente de modo sencillo. El API de seguridad se ha incrementado con nuevos subpaquetes que mejoran el soporte de certificados X.509 y permiten crear Listas de Revocación de Certificados (CRLs) y Peticiones de Firmado de Certificados

(CSRs).

Desde el punto de vista de la seguridad, el conjunto de clases de seguridad distribuidas con el Java 1.2 pueden dividirse en dos subconjuntos: clases relacionadas con el control de acceso y la gestión de permisos, y clases relacionadas con la criptografía.

Java incluye APIs de acceso a funciones criptográficas de propósito general, conocidas colectivamente como la Arquitectura Criptográfica de Java (JCA) y la Extensión Criptográfica de Java (JCE). La separación de JCA y JCE está motivada por las reglas de exportación de los EEUU para el ocultamiento de la información. El JCA es un marco de trabajo para acceder y desarrollar funciones criptográficas en la plataforma Java, se basa en dos principios básicos, independencia e interoperabilidad de las implementaciones, e independencia y extensibilidad de los algoritmos. El JCE se introduce como una extensión de la plataforma Java y proporciona implementaciones de algoritmos que permiten el ocultamiento de información, generar llaves, intercambiar llaves y autentificar mensajes.

Adicionalmente Java incluye una infraestructura de gestión de certificados para sustentar los certificados X.509 V3, el manejo de conexiones seguras por medio de la Extensión de Sockets Seguros en Java (JSSE), y el manejo de políticas con el Servicio de Autentificación y Autorización de Java (JAAS) (ver Fig.9).

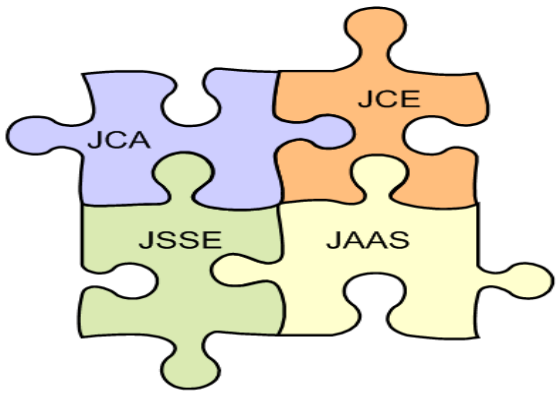

Figura 9. Seguridad en JDK 1.2

III.2.2 Certificados digitales e Infraestructura de Llave Pública (PKI)

El estudio de certificados digitales y la infraestructura de llave pública se debe a que representan la parte medular de la propuesta de solución en el presente trabajo de investigación.

Los certificados digitales proporcionan un mecanismo criptográfico para implementar la autentificación, y un mecanismo seguro y escalable para distribuir llaves públicas en comunidades grandes. Permiten que un individuo demuestre que es quien dice ser; es decir, que está en posesión de la llave secreta asociada a su certificado.

Un certificado de llave pública es un punto de unión entre la llave pública de una entidad y uno o más atributos referidos a su identidad. El certificado garantiza que la llave pública pertenece a la entidad identificada y que la entidad posee la correspondiente llave privada. Los certificados digitales sólo son útiles si existe alguna Autoridad Certificadora (Certification Authority o CA) que los valide, ya que si uno se certifica asimismo no hay ninguna garantía de que su identidad sea la que anuncia, y por lo tanto, no debe ser aceptada por un tercero que no lo conozca. Es importante ser capaz de verificar que una autoridad certificadora ha emitido un certificado y detectar si un certificado es válido. Para evitar la falsificación de certificados, la entidad certificadora después de autentificar la identidad de un sujeto (entidad identificada), firma el certificado digitalmente.

Certificados X509

El formato de certificados X.509 es un estándar del ITU-T (International Telecommunication Union-Telecommunication Standarization Sector) y el ISO/IEC (International Standards Organization/ International Electrotechnical Commission) que se publicó por primera vez en 1998. El formato de la versión 1 fue extendido en 1993 para incluir nuevos campos que permiten soportar el control de acceso a directorios. Después de emplear el X.509 v2 para intentar desarrollar un estándar de correo electrónico seguro, el formato fue revisado para permitir la extensión con campos adicionales, dando lugar al X.509 v3, publicado en 1996. Los elementos del formato de un certificado X.509 v3 son: versión, número de serie del certificado, identificador del algoritmo de firmado, nombre del emisor, periodo de validez,

nombre del sujeto, información de llave pública del sujeto, identificador único del emisor, identificador único del sujeto y extensiones.

El formato de certificados X.509 se especifica en un sistema denominado de notación de sintaxis abstracta uno (Abstract Syntax Notation One o ASN-1). Para la transmisión de los datos se aplica el DER (Distinguished Encoding Rules o reglas de codificación distinguible), que transforma el certificado en formato ASN-1 en una secuencia de octetos apropiada para la transmisión.

Autoridades Certificadoras

Una autoridad certificadora es una organización fiable que acepta solicitudes de certificados de entidades, las valida, genera certificados y mantiene la información de su estado.

Una CA debe proporcionar una Declaración de Prácticas de Certificación (Certification Practice Statement o CPS) que indique claramente sus políticas y prácticas relativas a la seguridad y mantenimiento de certificados, las responsabilidades de la CA respecto a los sistemas que emplean sus certificados y las obligaciones de los subscriptores respecto de la misma. Las labores de una CA son admisión de solicitudes, autentificación del sujeto, generación de certificados, distribución de certificados, anulación de certificados y almacenes de datos.

Infraestructuras de Llave Pública

La aplicación de la seguridad a sistemas con agentes móviles, como a cualquier otro sistema que implique la descarga de código ejecutable desde un servidor hacia un cliente, requiere de un servicio de soporte que permita asegurar tanto la identidad como la confiabilidad del código descargado desde el servidor, o el agente móvil que llega a una plataforma. Una infraestructura de Llave Pública (PKI) es el conjunto de protocolos, servicios y estándares que permiten hacer lo anteriormente descrito. En la Figura 10, se muestran los componentes básicos de la Infraestructura de Llave Pública (Thompson *et al.*, 2003).

La pieza central que proporciona la base de confianza en la PKI es la autoridad de certificación (CA). Con el objetivo de que los usuarios operen entre ellos es necesario un almacén de certificados que se encuentre en total disponibilidad de acceso, para ello se hace

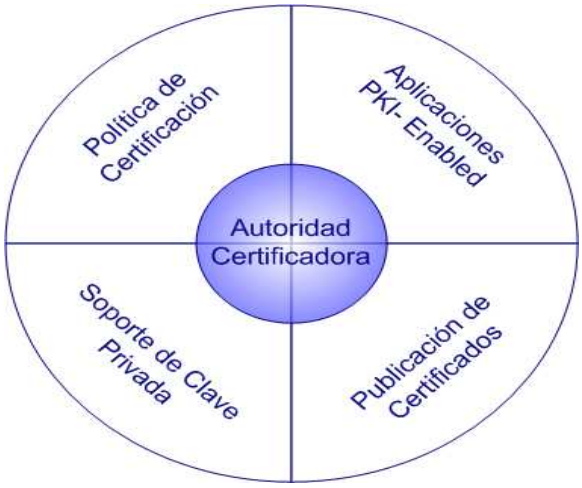

Figura 10. Componentes de la Infraestructura de Llave Pública (PKI)

uso de la publicación de certificados. El tener un soporte de llave privada, permite que los usuarios custodien su llave privada; por ejemplo, si la llave está en una SmartCard, es necesario diseñar el sistema de gestión de SmartCards que permita la emisión y distribución de las tarjetas a los usuarios. Una infraestructura de llave pública no tendría ningún valor de cara al usuario sin las aplicaciones software capaces de operar con certificados digitales a las que se les denomina Aplicaciones "PKI-Enabled". El proceso de construcción de una PKI deberá siempre partir de la definición de las políticas operativas y contemplar como requerimiento esencial el asegurar la calidad y seguridad de las operaciones que los usuarios finales realizan con sus llaves privadas; por ejemplo, un agente móvil debe apegarse a las políticas que establezca el contenedor donde se este ejecutando de acuerdo al certificado que éste presente; en las cuales, se establecen los recursos de la plataforma disponibles para el agente móvil, los agentes con los que le es permitido establecer comunicación, entre otras acciones que el agente móvil puede realizar en la plataforma.

III.3 Protocolos

III.3.1 Protocolo seguro SSL

SSL o Secure Sockets Layer es un protocolo diseñado por la empresa Netscape Communications, que permite cifrar la conexión, incluso garantiza la autenticación. Se basa en la criptografía asimétrica y en el concepto de los certificados. La versión estandarizada por el IETF se conoce como TLS.

Su mayor ventaja es que funciona entre la capa TCP y la capa de aplicación, por esto es muy fácil usarlo para proteger los protocolos de la capa de aplicación (por ejemplo FTP, gopher, HTTP...) sin tener que realizar cambios importantes en los mismos.

SSL intenta resolver tres problemas principales, seguridad de la información, integridad de los datos y autenticidad de los datos. En seguridad de la información SSL, garantiza que terceros no tengan acceso a la información mientras viaja por los canales de comunicación al aplicarle técnicas criptográficas; integridad de los datos se refiere a que la información recibida desde un servidor por SSL puede ser "validada" para comprobar que no ha sido alterada en la trayectoria; y autenticidad de los datos, la cual se implementa mediante los algoritmos de ocultamiento de información, es posible comprobar que los datos realmente han llegado del servidor que el cliente espera. Esto evita que alguien se haga pasar por un sitio para cometer fraudes (evitando ataques como Phishing, Man in the Middle, etc).

Mediante el uso de certificados digitales avalados por autoridades certificadoras SSL incorpora una eslabón más en la cadena de confianza (Persiano y Visconti, 2003).

III.3.2 Protocolo HTTPS

La aplicación de protocolos seguros para los sistemas multiagentes, en especial cuando las aplicaciones utilizan agentes móviles juegan un papel importante, debido a la transferencia de información que estas involucran. El sistema HTTPS utiliza el ocultamiento de información basado en Secure Sockets Layer (SSL) para crear un canal oculto más apropiado para el tráfico de información sensible que el protocolo HTTP. Es utilizado principalmente por entidades bancarias, tiendas en línea, y cualquier tipo de servicio que requiera el envío

de datos personales o contraseñas.

En el presente trabajo de investigación el protocolo HTTPS interviene como un elemento fundamental en la propuesta inicial de solución en sistemas multiagentes con agentes móviles seguros.

Para una mayor información acerca de temas de seguridad como: seguridad de plataforma, ocultamiento de información, autenticación y control de acceso, comunicaciones seguras e infraestructura de llave pública, se recomienda consultar la bibliografía disponible en (Johner *et al.*, 2000), (Sun-Developer-Network, 2005a) y (Sun-Developer-Network, 2005b).

III.4 Seguridad en FIPA

En 1998, FIPA realizó la primera especificación para la seguridad (FIPA, 1998), basando su modelo en el supuesto de que la seguridad debe ser un soporte a la infraestructura y no al agente mismo (ver Fig.11). El estado actual de esta especificación es obsoleto, no obstante es de gran utilidad ya que provee algunas ideas para el modelado de una arquitectura de seguridad dentro de una plataforma de agentes basada en FIPA.

El modelo de seguridad ofrecido por FIPA citado anteriormente, extiende la funcionalidad de la especificación base establecida desde 1995 (FIPA Agent Management Specification (FIPA, 2004)), agregando el Agent Platform Security Manager (APSM) a la plataforma, a través del cual pasa toda la información de cualquier otro agente que interactúe con la plataforma. Asimismo, se modificaron algunas funciones del AMS y del DF, y se propusieron extensiones en los metadatos de los mensajes intercambiados entre los agentes para separar niveles de confidencialidad e integridad; es decir, el agente podía configurar el nivel de privacidad e integridad que necesitaba según los requerimientos de la aplicación y las políticas manejadas en ese momento, que podían ser alto, medio o bajo.

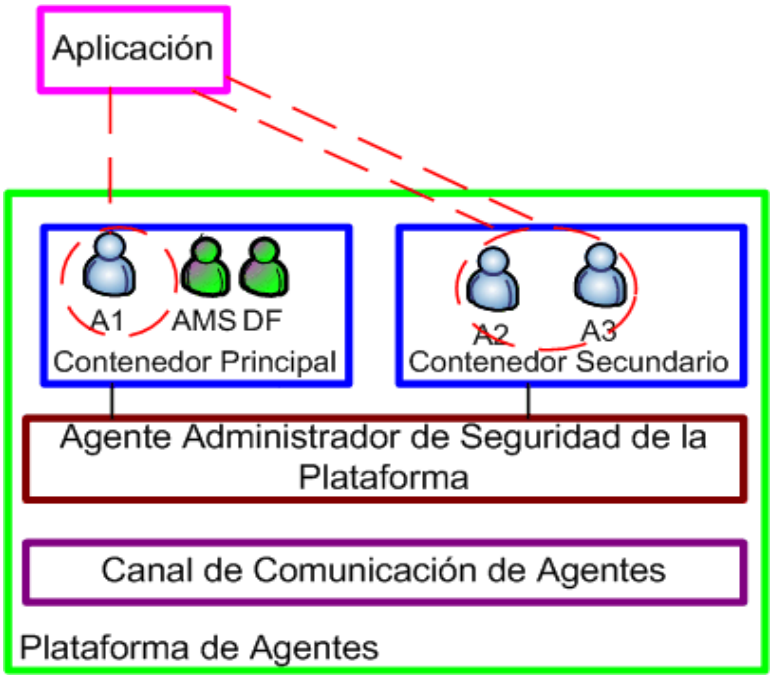

Figura 11. Especificación de FIPA 1998 para seguridad

III.5 Seguridad en JADE

JADE-S (Board, 2005) es la base para la protección de los sistemas multiagentes desarrollados en JADE. Su objetivo es brindar seguridad y cubrir algunos de los requerimientos básicos como: confidencialidad, integridad, disponibilidad, autenticación, no-repudio, autorización y privacidad. La filosofía de JADE-S es que para todos los componentes, agentes y contenedores pertenecientes a una plataforma sus dueños sean usuarios autentificados, los cuales sean autorizados por un administrador de la plataforma y se les otorgue ciertos permisos y privilegios en la ejecución de acciones. Cada agente debe contar con su respectiva llave pública y privada con las cuales puedan firmar y ocultar mensajes.

JADE en su página principal presenta el estatus de diseño e implementación de JADE-S; el cual, es tomado como base inicial para la primer concepción de las limitantes y posibles aportaciones a la plataforma JADE en su agregado JADE-S (ver Tabla III).

Para enfatizar la importancia de la seguridad en los sistemas multiagentes distribuidos, se muestran a continuación distintos escenarios en los que se manifiesta que el grado de

Tabla III. Estatus de diseño e implementación de JADE-S

Características	Estatus de diseño	Estatus de implementación
Arquitectura total	Mediano	Bajo
Autenticación de usuarios	Mediano	Bajo
Firmas y ocultamiento de información	Alto	Mediano
Permisos y delegación	Mediano	Mediano
Seguridad de comunicación inter-contenedores	Bajo	Mediano
Limitaciones MIDP	Mediano	Bajo
Autenticación de agentes	Bajo	Bajo
Servicios en la plataforma de seguridad	Bajo	Bajo

seguridad, depende del ambiente en donde se desempeñen las aplicaciones:

1) Consideremos un sistema de una aplicación de agentes donde cada participante es representado por un agente. En este escenario, los agentes van a competir por hacer la mejor compra de la mejor oferta, lo que quiere decir que van a competir entre ellos mismos; lo cual, implica que un agente malicioso entre en acción e intente sabotear las compras que realizan los agentes modificando la información o utilizar esta información para fines propios.

2) Consideremos ahora una aplicación de negocios, en donde se maneja información confidencial que debe ser intercambiada entre distintos agentes. Un agente malicioso puede husmear la información contenida en los mensajes o modificarla, y el agente que recibe la información puede ejecutar una información diferente a la que le fue solicitada.

3) Finalmente, consideremos una aplicación mínima de datos basada en agentes móviles, los cuales presentan una alta movilidad entre las computadoras donde realizan algunas consultas a bases de datos locales. Un agente malicioso puede situarse en un nodo remoto, instalarle virus, borrar información, o corromperla.

JADE-S presenta una arquitectura en tres niveles: autenticación, permisos e integridad de mensajes y confidencialidad (ver Fig.12). Asimismo, JADE-S se apega a la nueva arquitectura del kernel JADE la cual ha sido basada en el concepto de servicios. Cuatro servicios implementan los tres niveles de seguridad en JADE-S.

1. jade.core.security.SecurityService: es el encargado del mecanismo de autenticación y

también provee la funcionalidad de los servicios relacionados de ocultamiento de información y administración de llaves.

2. jade.core.security.permission.PermissionService: es el encargado de corroborar que las acciones que desean realizar los agentes sean permitidas, por ejemplo enviar mensajes, moverse entre contenedores, crear o matar agentes, entre otras acciones.

3. jade.core.security.signature.SignatureService: es el encargado de la firma de los mensajes cuando se recibe una petición por parte de un agente que envía, se verifica la validez del mensaje firmado.

4. jade.core.security.encryption.EncryptionService: es el encargado de cifrar los mensajes cuando se recibe una petición por parte de un agente que envía, y se descifra el mensaje cifrado.

En el orden como se inicia la plataforma de JADE-S es necesario activar los servicios de seguridad (SecurityService), y dependiendo de las necesidades de la aplicación se habilitan PermissionService, SignatureService o EncryptionService.

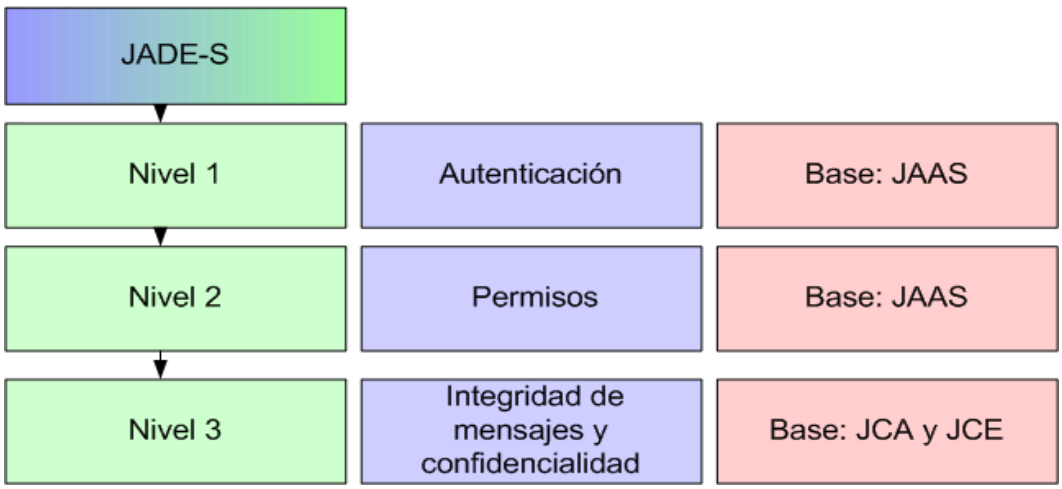

Figura 12. Niveles de seguridad ofrecido por JADE-S

III.5.1 Análisis de los mecanismos de seguridad de JADE-S

Con el objetivo de identificar oportunidades de mejora en los mecanismos de seguridad en la plataforma JADE en su agregado JADE-S, se realizó un estudio y análisis minucioso en los diferentes niveles que ésta ofrece; utilizando una aplicación desarrollada bajo la plataforma JADE y el estándar FIPA con agentes estacionarios (ver Fig.13), simulando el funcionamiento de un cajero de un banco donde se pueden hacer consultas del estado de cuenta, depósitos, retiros y consulta de operaciones. Dicha aplicación se tomó como base para realizar las pruebas en los distintos niveles de seguridad que JADE-S ofrece. A continuación se menciona el middleware, lenguaje, entorno de desarrollo y marco de trabajo utilizados para el desarrollo de dicha aplicación: middleware JADE versión 3.3, agregado JADE-S versión 2, lenguaje Java 1.2, Borland JBuilder versión 2005 y Apache Ant versión 1.6.5.

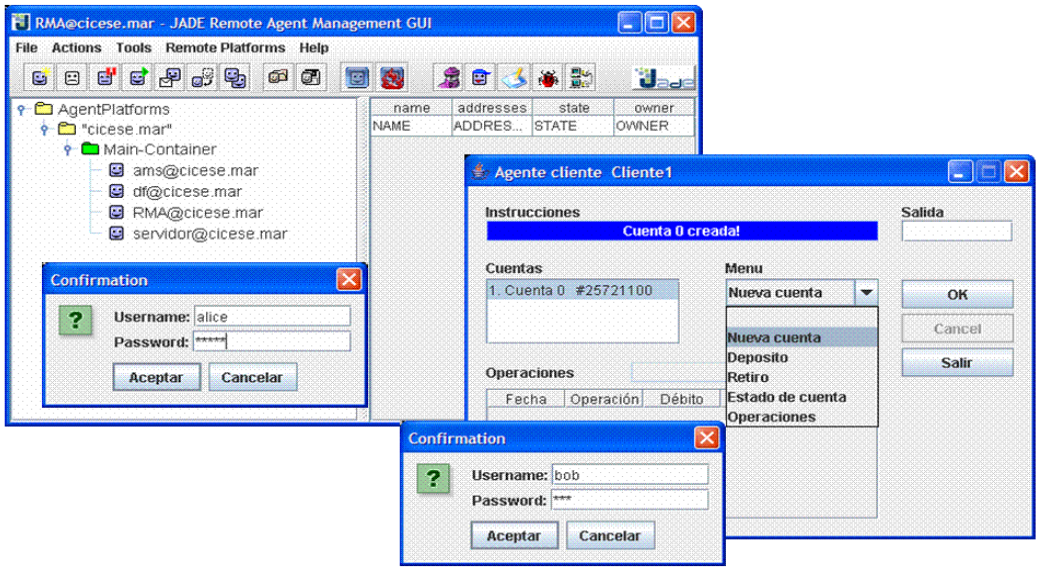

Figura 13. Aplicación del banquero

Análisis de Nivel 1: Autenticación.

Garantiza que los usuarios que se inician en la plataforma JADE, sean considerados seguros de poder ejecutar agentes y levantar contenedores en la plataforma. En JADE actualmente este punto se combate mediante la utilización de contraseñas y nombres de

usuarios, estos deben estar registrados como usuarios válidos en la plataforma.

El mecanismo de autenticación de JADE está basado en JAAS (Java Authentication and Authorization Service); el cual, es un API que maneja los diferentes niveles de control de acceso de los usuarios. Existen diferentes tipos de configuraciones para JAAS y JADE ha adoptado una de las más simples. Bajo esta configuración, la autentificación se lleva a cabo cuando un contenedor de JADE, usando la clase CallbackHandler, presenta un identificador de usuario y una contraseña ante la clase LoginModule. Estos datos son presentados en texto libre; esto es, sin cifrar. Si ocurre algún problema durante la autenticación, el contenedor no se levanta y se genera un mensaje de error. La Figura 14 muestra la implementación de la arquitectura de JADE-S.

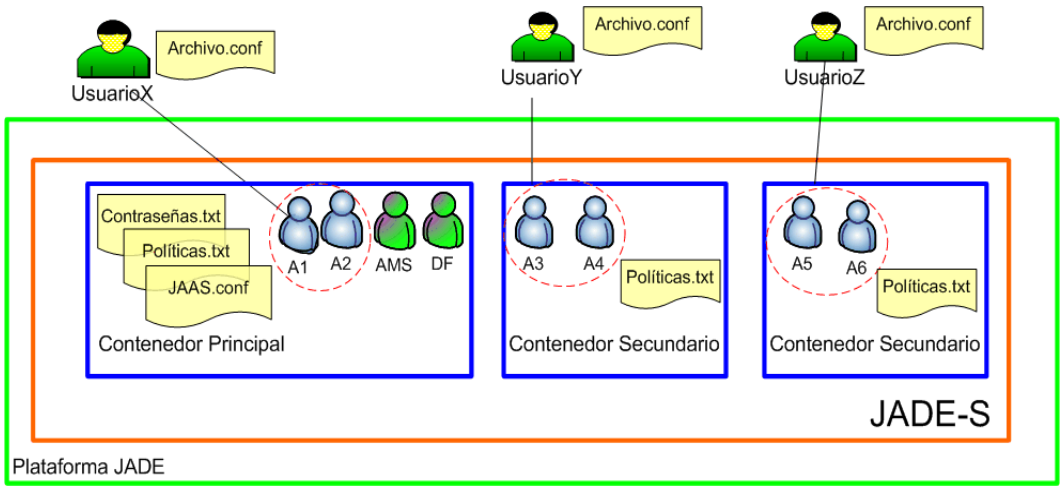

Figura 14. Implementación de la arquitectura actual de JADE-S. Se consigue mediante la habilitación de LoginModule en su configuración Simple.

JADE actualmente ofrece tres mecanismos para implementar la clase de CallbackHandler: a) cmdline: el nombre de usuario y contraseña se proveen por medio de algunos parámetros de configuración de JADE, ya sea por la línea de comandos con -owner user:pass, o dentro de un archivo de configuración (owner= user:pass). b) text: cuando se inicia un contenedor mediante el prompts se introduce el nombre de usuario y contraseña. c) dialog: cuando se inicia un contenedor aparece un cuadro de dialogo donde los usuarios introducen su nombre de usuario y contraseña.

Análisis de Nivel 2: Permisos.

Gracias al mecanismo de autenticación JAAS, la plataforma de JADE ofrece a sus componentes (agentes y contenedores) seguridad de que éstos tienen un usuario autentificado como dueño. Todas las acciones que los agentes realizan sobre la plataforma, pueden ser permitidas o negadas de acuerdo a algunas reglas impuestas por la plataforma. El conjunto de reglas de la plataforma generalmente residen en un archivo llamado policy.txt apegado a la sintaxis por default de JAAS.

Existen dos tipos de archivos de políticas que son usados para otorgar permisos a los agentes, políticas de contenedor principal y políticas de contenedor secundario. La política del contenedor principal especifica las acciones permitidas globalmente en la plataforma para el usuario autentificado. Este usuario crea sus agentes en el contenedor principal y cuenta con los permisos de crear, matar, suspender y reanudar agentes, ya sea especificando el nombre del dueño del agente, nombre del agente, nombre del contenedor o la clase con la que se crea el agente (sólo en el contenedor local); enviar mensajes a los agentes registrados en la plataforma, especificando el nombre del dueño del agente o el nombre del agente; crear o matar agentes en la plataforma; y crear o matar a los agentes que pertenezcan a cierto usuario.

La política del contenedor secundario especifica los permisos dentro de ese contenedor. Un usuario autentificado puede crear agentes en un contenedor secundario, y cuenta con los permisos de crear, matar, suspender y reanudar agentes, sólo en el contenedor local; y enviar mensajes a agentes registrados en la plataforma, especificando el nombre del dueño del agente o el nombre del agente.

El agregado JADE-S se encuentra en sus inicios de desarrollo, debido a esto, una vez analizado este nivel de seguridad mediante la aplicación desarrollada para este fin, se detectaron algunos puntos que deben ser considerados:

- Los permisos dentro de la plataforma JADE, en su agregado JADE-S, sólo están especificados para agentes estacionarios, lo que resulta un punto débil dentro de la plataforma si se piensa desarrollar aplicaciones que impliquen movilidad en software.

- Los agentes AMS, DF pueden ser eliminados por cualquier usuario, lo que implica que la plataforma sea eliminada por cualquier usuario independientemente de los permisos otorgados.

Estos detalles son indicativos que aún existe trabajo por hacer en éste nivel, no sin antes analizar el nivel 3 de seguridad propuesto por JADE-S.

Análisis de Nivel 3: Integridad de mensajes y confidencialidad.

Las firmas y el ocultamiento de la información garantizan cierto nivel de seguridad cuando enviamos un mensaje ACL (Lenguaje de Comunicación de Agentes), ya sea a un agente que se encuentre dentro de la plataforma o fuera de ésta. Las firmas nos aseguran la integridad de los mensajes y la identidad de quien envía el mensaje. El ocultamiento de información nos asegura la confidencialidad de los mensajes; es decir, protege a los datos del mensaje de algún sniffer o ataque a la información.

Los mensajes ACL en JADE están compuestos de dos partes envelope y payload; el envelope contiene la información relacionada con el agente que envía y el agente que recibe, y el payload contiene los datos importantes del mensaje. En JADE-S se propone que las firmas y el ocultamiento de información, se apliquen en todo el payload con el fin de proteger toda la información importante contenida en el mensaje ACL.

Para el análisis de este nivel de seguridad se utilizó la herramienta *sniffer* propia del middleware JADE, con el objetivo de observar una representación gráfica del envío de mensajes entre los agentes, además de ayudarnos de la herramienta *Ethereal* con el fin de capturar la información que viajaba por la red en el momento de la comunicación de los agentes en la aplicación. Los resultados obtenidos en éste análisis fueron los siguientes (ver Fig.15).

- El contenido de los mensajes viaja en texto libre por la red.
- Cuando un usuario inicia el proceso de autentificación, el nombre de usuario y contraseña viajan en texto libre.
- Los algoritmos que proponen para el ocultamiento de mensajes son RSA y SHA1, ya que se puede observar que estos nombres viajan por la red como parámetros fijos.

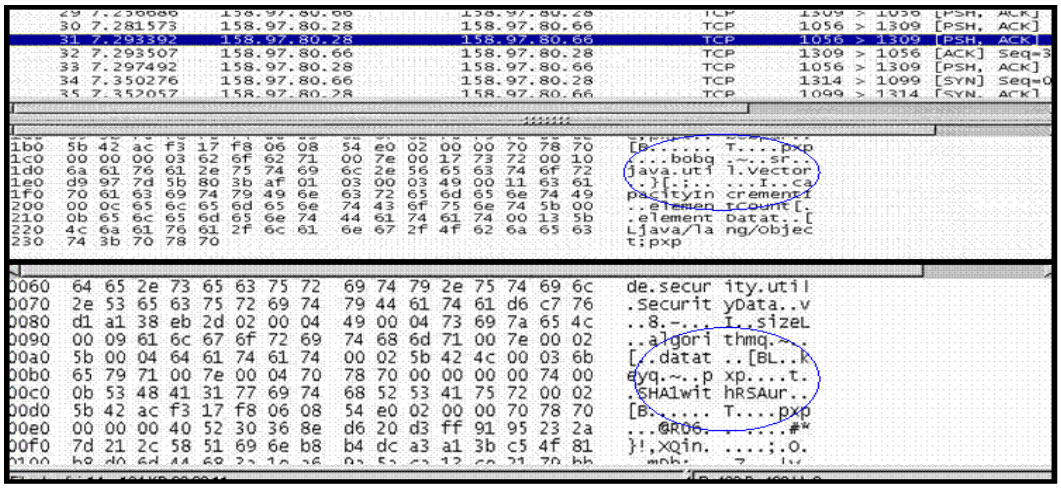

Figura 15. Análisis de paquetes en JADE-S

Dado los resultados anteriormente descritos, se consideró necesario hacer una inspección de código en donde se encontró lo siguiente:

- En el módulo correspondiente a firmas y ocultamiento de información se puede encontrar con frecuencia las palabras *FIXME* y *TOFIX* , esto significa que aún no está funcionando, que el código se encuentra en fase de propuesta o prototipo esperando tener mejores resultados con posibles aportaciones (ver Fig.16).

La implementación del cifrado de mensajes aún está en proceso, se debe destacar el gran esfuerzo que el equipo de JADE está realizando para dotar de seguridad a la plataforma. Un punto clave a resaltar es que el uso de certificados en este módulo de seguridad se encuentra en su fase de prototipo.

III.5.2 Análisis de HTTPS en JADE-S

JADE con el propósito de dotar de seguridad a las aplicaciones a nivel inter-plataforma, propone la utilización del protocolo HTTPS en la comunicación entre los agentes de distintas plataformas JADE, este protocolo aún no puede ser utilizado para el desarrollo de aplicaciones basadas en este middleware debido a la falta de la implementación completa dentro del entorno. En el mismo estado se encuentra el protocolo RMI-SSL, el cual se propone

```
C:\jade\add-ons\security\src\jade\core\security\signature
/** Verify signature with a given public key*/

    public boolean verifiedSignature(ACLMessage msg, JADEPrincipal p) {
    // - get the envelope from the msg  (if doesn't have one, return false)
    // - get the SecurityObject (for now serialized into x-security user-def property)
    // - return the value of the "verified" field into the SecurityObject
    // @@ Fixme
    System.out.println(
       " SecurityManager.verifiedSignature() Not Implemented yet.");
```

```
C:\jade\add-ons\security\src\jade\core\security\authentication
        /** Extract credentials data from the user's Security Store
         *
         * @throws SecurityException
         */
    private void getDataFromUserSecurityStore() throws SecurityException {
      myLogger.log(Logger.FINE, "Retrieving data from User Security Store...");
      // open the user's security store
      SecurityStore ss = new SecurityStore( user );
      //ss.open( passwd ); // TOFIX: this should throw SecurityException or other
     // ss.open( "secret".getBytes() ); // TOFIX: real user password should be used (?)
      ss.open( pass );
      myLogger.log(Logger.FINER, " ss.hasMyKeyPair() = "+ss.hasMyKeyPair() );
      myLogger.log(Logger.FINER, " ss.getMyKeyPair() = "+ss.getMyKeyPair() );
      //Logger.println( " ss.getPrincipal(\"snoncemale\")=\n"+ss.getPrincipal("noncemale") );

      // retrieve principal and credentials
      principal = ss.getMyPrincipal();
      myLogger.log(Logger.FINE, " ss.getMyPrincipal()=\n"+ principal );
      creds = null; // TOFIX: Credentials storage not yet supported
    }
```

Figura 16. Código dentro de los módulos de firmas y ocultamiento de información

para la comunicación intra-plataforma, JADE para ambientes de JADE/LEAP introduce el protocolo JICP/SSL, el cual está en funcionamiento (JADE, 2005).

III.6 Conclusiones

Durante el estudio de JADE-S se detectaron que existen algunas limitaciones:

1. Los permisos relacionados con la movilidad aún no están habilitados. Por lo que el desarrollo de aplicaciones que involucren movilidad se ven limitadas.

2. Las piezas de información que son intercambiadas entre contenedores son transferidas a través del protocolo de comunicación SSL pero no son firmados. Un agente malicioso o hacker podría penetrar en la comunicación entre contenedores y modificar la información.

3. Muchas de las ventajas de seguridad no están accesibles para dispositivos como MIDP (Mobile Information Device Profile).

De acuerdo a estas limitaciones se ha tomado como base para el curso de la presente investigación: los permisos con la movilidad aún no están habilitados. Como consecuencia de ello JADE-S no soporta movilidad del todo. Hasta este punto se ha estudiado el estado del arte de la seguridad informática y la movilidad en sistemas multiagentes; se ha realizado el estudio comparativo de los middlewares existentes eligiendo JADE de acuerdo al interés sobre seguridad y movilidad de agentes; y se ha desarrollado una aplicación para el análisis del estado actual de la seguridad informática y la movilidad en JADE. Por lo que el siguiente paso es el bosquejo sobre la movilidad permitida en JADE, con el objetivo de acercarnos a la propuesta de mejora como fruto de este trabajo de investigación.

Capítulo IV

Análisis de los mecanismos y protocolos de movilidad para Sistemas Multiagentes

"Nuestra recompensa se encuentra en el esfuerzo y no en el resultado. Un esfuerzo total es una victoria completa".
Mahatma Gandhi

IV.1 Introducción

Como se mencionó en los antecedentes el diseño e implementación de sistemas multiagentes pueden obtener beneficios del uso de mecanismos y protocolos de movilidad de agentes. Existen muchas maneras de expresar movilidad, tales como movilidad del código, migración del agente y clonación del agente. Los desarrolladores de JADE, en su afán por sustentar el desarrollo de aplicaciones con agentes móviles, han encontrado en el mecanismo de serialización de objetos del lenguaje Java una solución para apoyar a la movilidad de los agentes JADE. Esto aunado a los protocolos de movilidad de agentes que están en proceso de estandarización dentro de FIPA.

En este capítulo se presentan los mecanismos y protocolos de movilidad en los cuales el middleware JADE basa la movilidad de sistemas multiagentes.

IV.2 Mecanismos de movilidad

La movilidad de un agente implica enviar su código y estado a través de la red (proceso de serialización y deserialización), algunos de los recursos usados por el agente móvil se moverán con él, mientras que otros serán desconectados antes de salir y eventualmente reconectados en el destino una vez seguros de que el agente se ha movido de manera satisfactoria. La

operación de movilidad es considerada completa cuando el estado, código y la identidad del agente se hayan transferido con éxito.

El lenguaje Java ha añadido una interesante faceta denominada serialización de objetos, que permite convertir cualquier objeto cuya clase implemente el interface Serializable en una secuencia de bytes que pueden ser posteriormente leídos para restaurar el objeto original (ver Fig.17). Esta característica se mantiene incluso a través de la red, por lo que se puede crear un objeto en una computadora que corra bajo Windows, serializarlo y enviarlo a través de la red a una estación de trabajo que corra bajo UNIX donde será correctamente reconstruido. La serialización es una característica del lenguaje Java para dar persistencia y soporte al intercambio de objetos con estructuras de datos complejas. JADE utiliza la serialización de objetos para el proceso de migración de agentes móviles a nivel intra-plataforma.

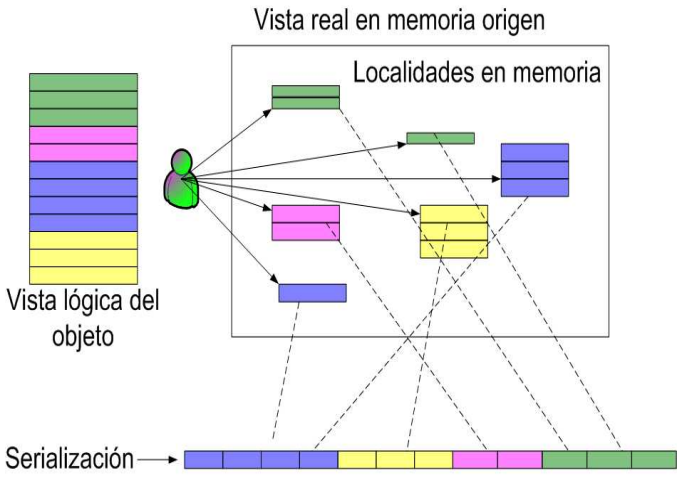

Figura 17. Serialización. Un objeto almacenado en memoria en el momento de la serialización es una secuencia de bytes que representan el código y estado de dicho objeto; los cuales, son enviados a través de la red de un nodo origen a un nodo destino donde el objeto es restaurado.

IV.3 Protocolos de movilidad

FIPA propone dos protocolos base con el fin de dar sustento a la movilidad en sistemas multiagentes, estos son: protocolo de movilidad simple y protocolo de movilidad total (FIPA,

2002b).

Protocolo de movilidad simple

Se propone se aplique cuando un agente necesita realizar una sola acción; por ejemplo, moverse. Esta acción involucra que el agente se mueva a un destino dentro de la plataforma. La operación de movilidad es delegada a un agente dentro de la plataforma de agentes (ver fig18).

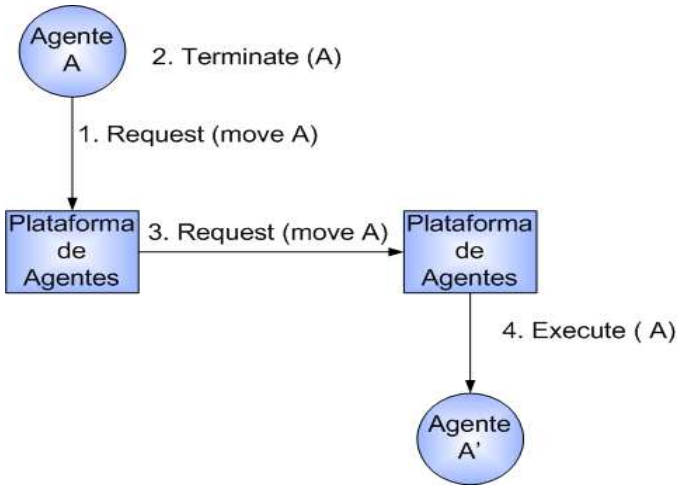

Figura 18. Protocolo de movilidad simple propuesto por FIPA

Protocolo de movilidad total

Propone que el mismo agente sea el encargado del manejo de la movilidad, evitando delegar a la plataforma esta función. Este protocolo permite llevar un control sobre las plataformas visitadas por el agente y conocer el contenedor actual de este mismo; lo cual, ayuda en cuestiones de seguridad de agentes móviles facilitando el control del histórico de trayectoria que es empleado como un combate a los ataques a que están expuestos los agentes móviles (ver Fig.19).

IV.4 Movilidad en FIPA y JADE

Movilidad en FIPA

La especificación de movilidad de FIPA (FIPA, 2002b) considera dos tipos de movilidad:

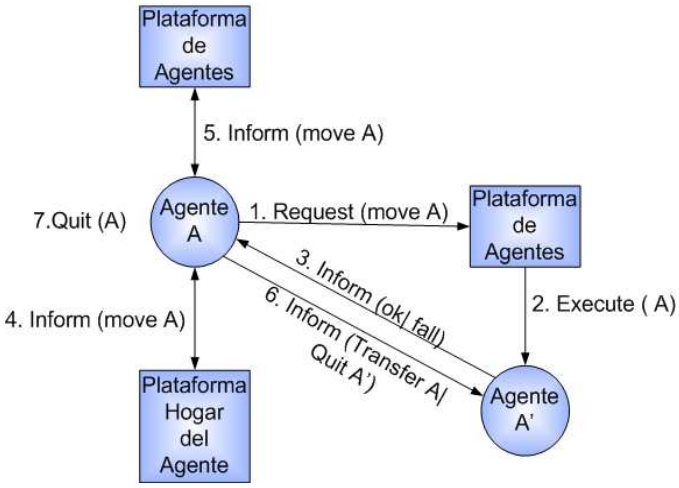

Figura 19. Protocolo de movilidad total propuesto por FIPA

movilidad de dispositivos, que puedan estar conectados de manera intermitente en la red; y movilidad en software como los agentes móviles; los cuales, pueden moverse entre nodos en la red.

Dentro de la especificación de movilidad en FIPA se proponen modificaciones respecto a los protocolos de comunicación entre agentes ya mencionados en la sección anterior, ontologías y ciclo de vida de los agentes. Estas modificaciones han sido utilizadas en la migración y clonación de los agentes a nivel intra-plataforma. Sin embargo, hasta el momento no hay implementaciones que soporten íntegramente la movilidad de código a nivel inter-plataforma.

La ontología propuesta por FIPA es la siguiente:

1. fipa-mobile-description: indica el nombre del agente, dirección origen-destino, protocolo utilizado, versión del agente.

2. fipa-mobile-profile: indica el sistema operativo, lenguaje y ambiente requerido por el agente móvil para poder ejecutarse.

3. fipa-mobile-agent-system: indica el nombre del sistema del agente móvil, versiones y dependencias.

4. fipa-mobile-agent-language: indica el nombre del lenguaje, versiones aceptables del

lenguaje, filtros que deben ejecutarse antes de que el agente móvil pueda ejecutarse, dependencias.

5. fipa-mobile-agent-os: indica el nombre del sistema operativo, versiones aceptables, especificaciones de hardware, dependencias.

Dentro de las especificaciones de las ontologías se consideran dos acciones Move y Transfer. Move es soportada por el AMS, cuando un agente desea moverse debe enviarle una petición al AMS local o remoto, el cual tiene la autoridad para aceptar o rechazar su petición de acuerdo al perfil del agente; dentro de la ontología de los agentes móviles se encuentra el mobile-agent-description, el cual es obligatorio que sea enviado entre los valores del agente que realiza la petición. Transfer es soportada por un agente, cuando un agente hace una petición de transferencia a otro agente, el agente destino consulta con el AMS si la petición es posible, además de revisar los valores obligatorios que deben ser enviados por el agente, el agente que recibe la petición puede aceptar o rechazar por cuestiones de seguridad de la plataforma o el agente mismo.

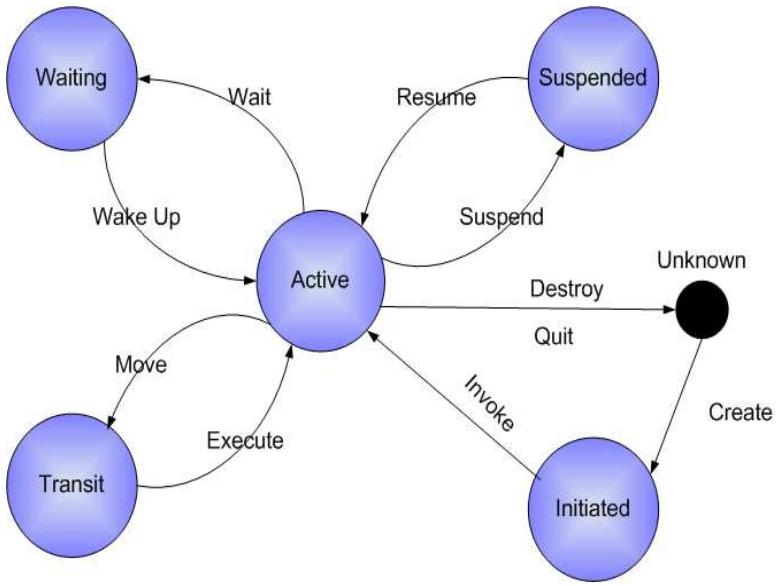

Figura 20. Ciclo de vida para el agente móvil propuesto por FIPA

Las especificaciones referentes al ciclo de vida(ver Fig.20) incorporan un estado exclusivo

para agentes móviles: *transit*, y dos nuevas acciones Move y Execute. Move pone al agente en un estado transitorio, sólo puede ser inicializado por un agente; y Execute quita al agente de un estado transitorio.

Movilidad en JADE

JADE propone especificaciones de movilidad para el desarrollo de aplicaciones de sistemas multiagentes móviles; las cuales, no forman parte del estándar FIPA, pero las sustenta en prototipos de prueba para una futura implementación de estas especificaciones como estándares de movilidad.

Según Bellifemine *et al.* (2005a, 2005b) JADE restringe el soporte de movilidad entre contenedores de una misma plataforma (intra-plataforma). JADE de acuerdo con el estándar FIPA solidifica el que un agente móvil tiene la capacidad de migración y clonación, por lo que proporciona en su API para la movilidad dos métodos: doMove() y doClone(). doMove representa el destino requerido donde debe migrar el agente; y doClone() contiene el nombre del nuevo agente que se creará con la copia del actual.

La migración de un agente involucra la necesidad de conocer su localización exacta para decidir cuándo y a dónde desplazarse. Para ésto se ha desarrollado una ontología integrada por cinco conceptos: mobile-agent-description, mobile-agent-profile, mobile-agent-system, mobile-agent-language, mobile-agent-os; y dos acciones: move-agent y clone agent. Estos conceptos y acciones son soportadas por el AMS de la plataforma, además de las acciones Where-is-agent que proporciona la información sobre la localización de algún agente y Query-platform-locations, que proporciona todos los contenedores o localidades habilitadas en la plataforma.

JADE proporciona dos métodos para la gestión de recursos dentro de la plataforma: beforeMove(), se invoca en la localización de partida antes de enviar el agente a través de la red; y afterMove(), se invoca en la localización de destino tan pronto como el agente llega y se identifica en el lugar. De la misma manera se utilizan los métodos: beforeClone() y afterClone().

La ontología utilizada en JADE presenta extensiones y modificaciones a la propuesta

por el estándar FIPA, debido a ésto, fue necesario explicarla de manera detallada; el ciclo de vida y los protocolos de interacción utilizados por JADE para agentes móviles, se apegan a las especificaciones del estándar FIPA descritos anteriormente.

IV.5 Conclusiones

A pesar de los esfuerzos de JADE por el apego al estándar FIPA como hasta ahora se ha presentado, la movilidad en sistemas multiagentes ha sido la excepción por la falta de estándares; las propuestas del equipo de desarrollo JADE en el manejo de ontologías, protocolos y ciclo de vida, se considera son acertadas como un inicio hacia la movilidad.

Una vez explorados los aspectos de seguridad y movilidad ofrecidos por la plataforma JADE, el sustento de los antecedentes presentados en las primeras secciones de este trabajo de tesis, y de acuerdo a las oportunidades de mejora, se presenta en el siguiente capítulo la propuesta de mejora al sustento de agentes móviles seguros en el middleware JADE.

Capítulo V

Extensión de la plataforma JADE-S para sustentar agentes móviles seguros

"Donde una puerta se cierra, otra se abre".
Miguel de Cervantes Saavedra

V.1 Introducción

El desarrollo de sistemas multiagentes para aplicaciones reales trae consigo la necesidad de la implementación de seguridad en la comunicación entre agentes y plataformas. En este capítulo se presenta una propuesta bajo la plataforma JADE y el agregado JADE-S en el nivel de Autenticación, con el objetivo de brindar seguridad a los agentes móviles durante su migración inter-plataforma, utilizando certificados X509 y el protocolo HTTP-S, no dejando atrás el apego a las especificaciones FIPA, lo cual asegura que la plataforma JADE siga siendo interoperable.

Durante el desarrollo de la propuesta inicial surgieron problemas derivados de limitantes encontradas en dos de las tecnologías utilizadas: 1) El uso del protocolo HTTP-S en JADE y 2) la interacción del proceso de serialización en Java con los certificados digitales X509. La necesidad de resolver estos problemas resultó en variaciones a la propuesta inicial. Como se mostrará mas adelante, la propuesta inicial considera el uso del protocolo HTTP-S y certificados digitales X509, por lo que es aplicable tanto a sistemas multiplataforma como a sistemas uniplataforma; lo cual la hace una solución general. Enseguida se tiene un acotamiento de la solución a una sola plataforma de agentes para evitar el uso del protocolo HTTP-S. Finalmente, se sustituye un manejo real de certificados X509 por un manejo simulado de estos certificados digitales. Se considera que la propuesta final aunque menos

general y completa que la propuesta inicial, si cumple con el objetivo de servir como prueba de concepto del sustento a la seguridad de agentes móviles en JADE.

V.2 Planteamiento del problema

Con base a lo expuesto en los capítulos anteriores, el precio de utilizar los agentes móviles es el enfrentamiento a los problemas inherentes de seguridad; los cuales, aún no tienen una solución satisfactoria para los agentes estacionarios mucho menos en los agentes móviles. Cuando se habla de agentes móviles se hace referencia a aquellas piezas de software que tienen la característica de ser autónomos, proactivos, sociales y son capaces de moverse a través de la red, migrar su código y estado con el fin de cumplir sus metas u objetivos retornando al nodo inicial con resultados. A partir de la revisión bibliográfica realizada y la experimentación conducida mediante la aplicación del banquero, los siguientes pueden ser identificados como los principales problemas de seguridad para sistemas multiagentes móviles:

1. Protección de los agentes y la información que manejan en su transferencia a través de una red insegura como Internet. Este es el problema clásico de transferir información en forma segura a través de un canal inseguro como los de Internet. La solución del problema puede lograrse enviando el agente a través de un mecanismo de cifrado y descifrado, un ejemplo es usar los protocolos como SSL/TLS o SSH (ver Fig. 21).

2. Protección de los entornos de ejecución o plataformas y agentes contra agentes móviles hostiles. Como ya se mencionó, un agente consiste de código, datos y estado. El código no debe cambiar conforme el agente se mueve de una plataforma a otra, todas las plataformas que intervienen en la aplicación deben poder verificar la autenticidad del agente y con esto decidir si se ejecuta o no dicho agente (ver Fig. 22).

3. Protección de las plataformas contra agentes que han sido saboteados por otras plataformas hostiles. Si una plataforma ha sido atacada y ésta se ha convertido en una plataforma hostil, los agentes que residen en ella serán agentes hostiles; una plataforma

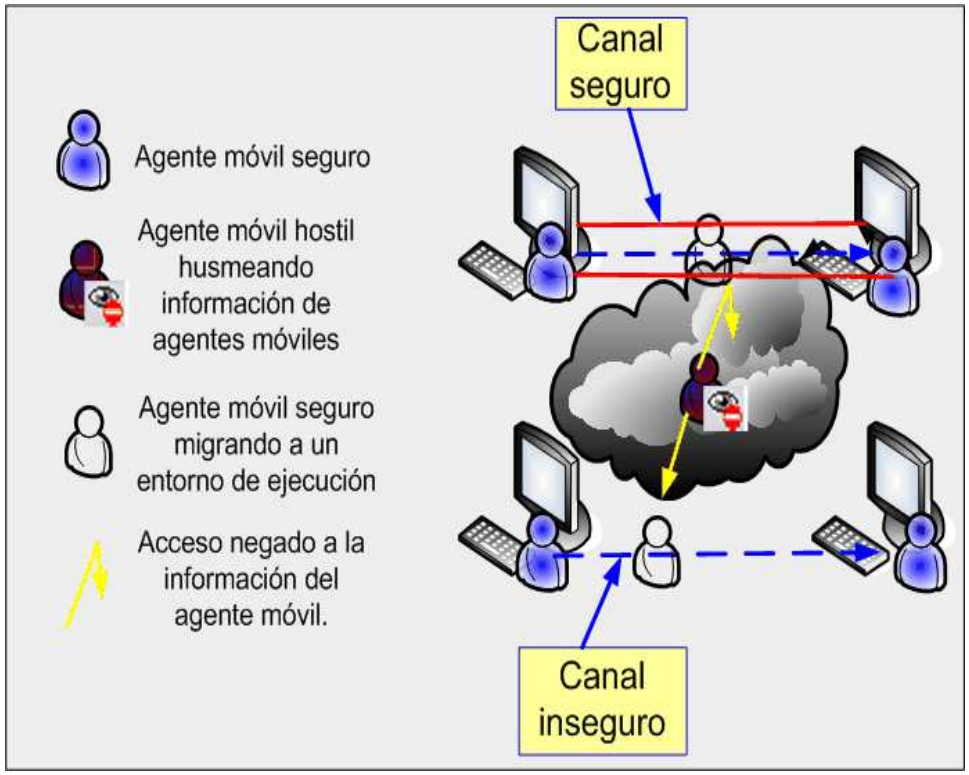

Figura 21. Los canales de comunicación seguros protegen a los agentes móviles de plataformas y agentes hostiles.

segura debe de contar con mecanismos que le permitan identificar agentes hostiles antes de permitir su ejecución (ver Fig. 23).

4. Protección de los agentes contra agentes móviles hostiles. Si el sistema soporta movilidad, los datos y el estado del agente son transferidos junto con el código. Generalmente los datos y el estado cambian después de cada ejecución, esto implica un riesgo. Un agente amistoso puede transformarse en hostil si la plataforma sabotea sus datos y/o estado (ver Fig. 24).

La propuesta de solución consiste en una extensión de la plataforma JADE para sustentar agentes móviles seguros.

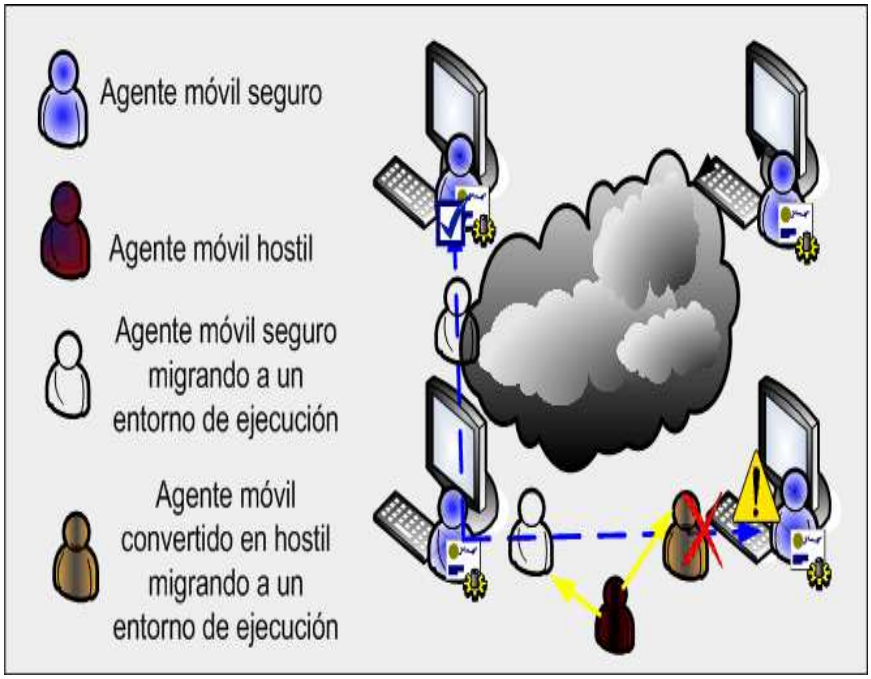

Figura 22. Los entornos de ejecución y agentes deben ser capaces de identificar agentes hostiles y evitar la ejecución de éstos.

V.3 Propuesta de solución a la seguridad de agentes móviles en sistemas multiagentes.

El objetivo es proporcionar un mecanismo de autenticación que se ajuste a las necesidades de agentes móviles desarrollados bajo la plataforma JADE en su agregado JADE-S; partiendo del conocimiento adquirido durante el estudio, análisis y experimentación conducidos hasta este punto del trabajo de investigación; y considerando el bajo grado de avance en el diseño y la implementación en la autenticación de agentes (ver Tabla III), la falta de definición e implementación en los permisos relacionados con la movilidad (ver sección III.6), y las preocupaciones 2 y 3 de lista de la sección anterior (protección de agentes móviles contra ataques de plataformas remotas hostiles y protección a plataformas remotas de agentes móviles hostiles).

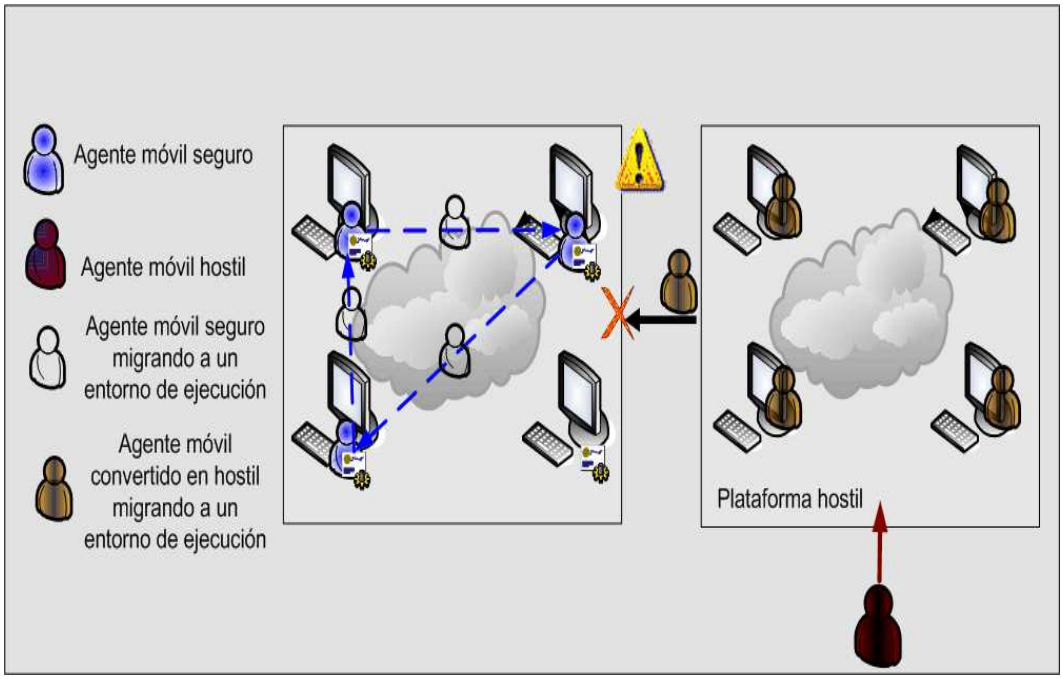

Figura 23. Una plataforma debe contar con mecanismos que le permitan la identificación de agentes hostiles.

V.3.1 Propuesta I

Se propone la implementación de la autenticación de agentes móviles mediante el uso de certificados X509 y la utilización del protocolo HTTP-S para la comunicación inter-plataforma (Juárez y Lepe, 2005). La Figura 25 ilustra el concepto de esta primera aproximación.

Cuando un usuario desarrolla una aplicación de agentes móviles debe dotar a estos agentes con un certificado que los identifique como seguros. El agente móvil seguro presenta su certificado ante el AMS y el DF de la plataforma con el objetivo de obtener los permisos necesarios para el cumplimiento de sus tareas y el registro de los servicios que ofrece. Cuando un agente móvil desea migrar hacia otra plataforma será enviado dentro de un mensaje ACL. Este agente móvil debe llevar consigo el certificado para presentarlo ante el AMS y el DF de la plataforma destino. Para la migración entre dos plataformas JADE se propone el uso del protocolo HTTP-S, con el que se busca prevenir que agentes maliciosos ataquen la información que se este intercambiando entre las plataformas; además, de proteger a los

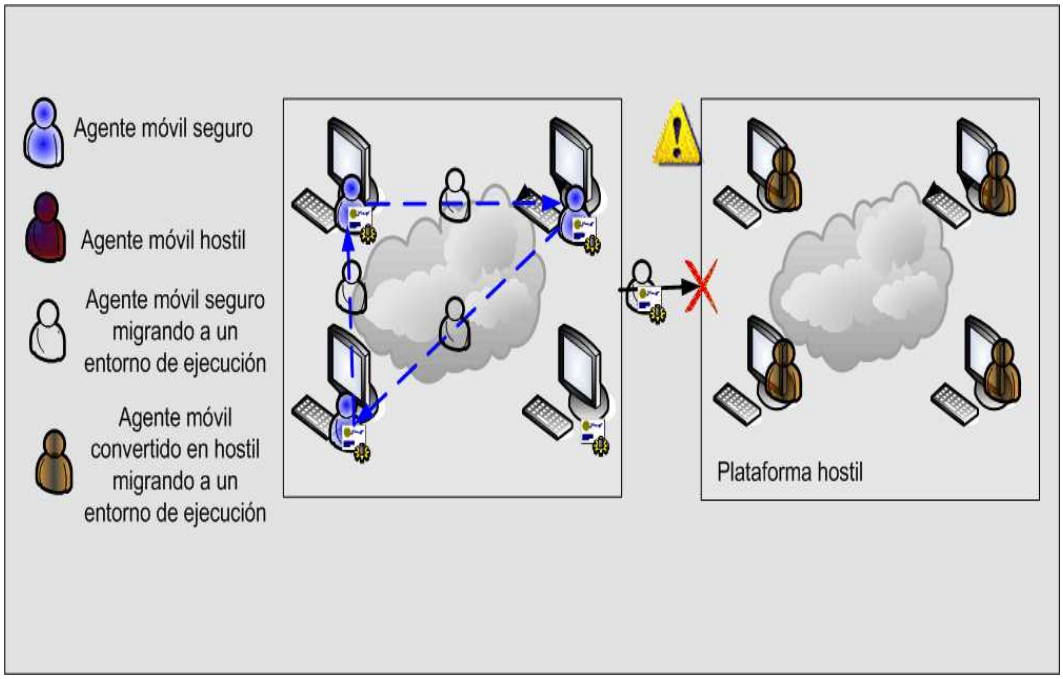

Figura 24. Un agente debe contar con mecanismos que le permitan la identificación de agentes y plataformas hostiles.

agentes móviles durante su migración inter-plataformas.

Con el objetivo de experimentar el uso del protocolo HTTP-S para sustentar comunicaciones seguras inter-plataforma, se montó el experimento de la Figura 26. En colaboración con Joan Amettler[1] localizado en Barcelona,España, Marco Antonio Calderón[2] localizado en La Paz,Bolivia y la presente colaboradora localizada en Ensenada,México; se tuvieron varias sesiones en el intento por lograr la comunicación inter-plataforma.

No fue posible debido a la falta de implementación del protocolo HTTP-S en JADE (ver Anexo 2). Por lo que considerando el apremio del tiempo para este trabajo de investigación, se decidió abandonar la idea de una propuesta basada en más de una plataforma JADE y enfocarse a una sola plataforma, lo que no influye de manera drástica en la idea central de la propuesta hacia agentes móviles seguros. En el Anexo I se presentan las conversaciones y correos intercambiados durante las pruebas de comunicación, con el objetivo de dejar prueba

[1]Doctor en Ciencias en la Universidad Autónoma de Barcelona.
[2]Estudiante de postgrado en la Universidad Mayor de San Andrés.

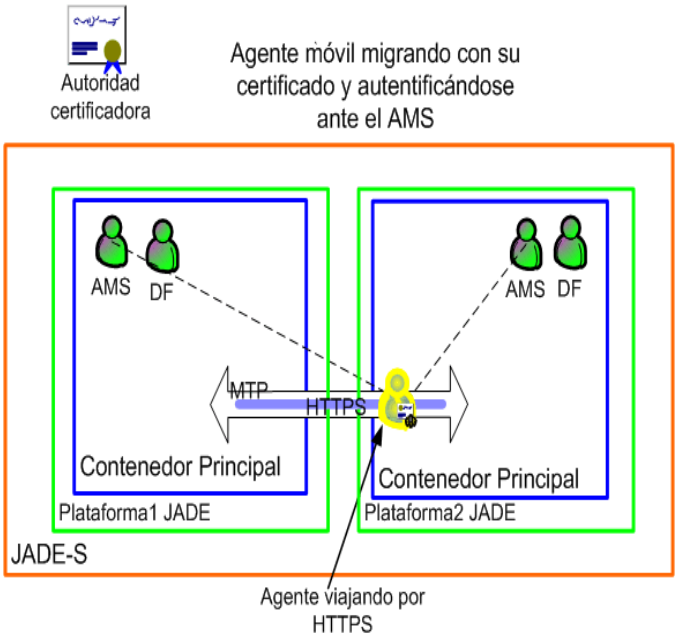

Figura 25. El agente móvil migra a la plataforma destino a través del protocolo HTTP-S y se autentifica ante el AMS de la plataforma destino por medio de su certificado X509.

testimonial del proceso de pruebas y facilitar un trabajo futuro en este sentido.

Mis contrapartes en Barcelona y Bolivia, Joan Amettler y Marco Antonio Calderón son colaboradores activos para el desarrollo del middleware JADE. Además son de las pocas personas que encontré que están investigando sobre seguridad y movilidad en JADE. En particular Joan pertenece a un grupo de investigación en la Universidad Autónoma de Barcelona, en donde tiene trabajos recientes acerca de la integración del protocolo HTTP-S en la plataforma JADE.

La implementación del protocolo HTTP-S en la plataforma JADE, queda ahora como una propuesta de investigación para trabajo futuro; además, de otros protocolos ya mencionados en capítulos anteriores.

V.3.2 Propuesta II

A consecuencia de lo anteriormente expuesto, fue necesario analizar y enfocar la propuesta de solución para la autenticación de agentes móviles mediante el uso de certificados X509

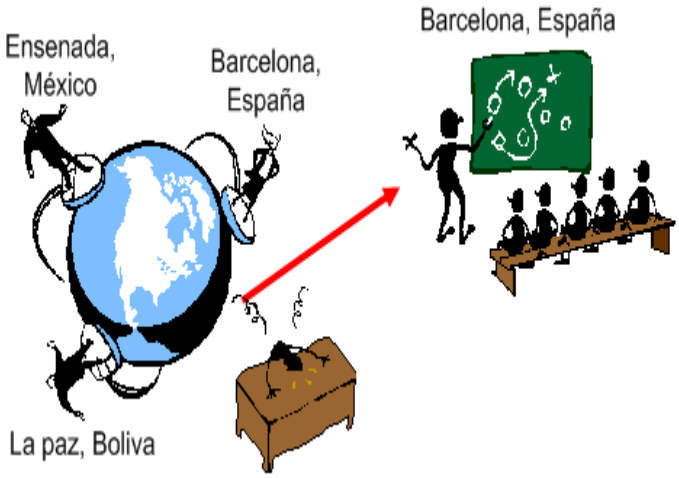

Figura 26. Colaboración de usuarios y desarrolladores JADE, para establecer la comunicación inter-plataforma a través del protocolo HTTP-S.

hacia una sola plataforma JADE. Por lo que se definieron nuevos actores que ayuden a la plataforma en la administración de certificados, usuarios válidos y agentes móviles (ver Fig.27). A continuación se presentan las definiciones de estos nuevos actores.

El Usuario X es un usuario registrado (válido) en la extensión JADE-S de la plataforma JADE; el cual, debe contar con un certificado digital X509 y la asignación de un nombre de usuario y contraseña para su autentificación en la plataforma.

El Usuario Oficial de Seguridad en la Plataforma (OSP) es una persona física que tiene la tarea de administración de los certificados digitales y la asignación de nombres de usuario y contraseñas en la plataforma.

El Almacén de Certificados es utilizado para el almacenamiento de certificados digitales y llaves privadas válidos en la plataforma. Este es administrado por el OSP.

El Agente Administrador de Seguridad (AAS) es un agente que es creado junto con el AMS y el DF en el contenedor principal cuando se levanta la plataforma segura JADE-S; el cual, es el encargado de administrar a los agentes móviles seguros creados por un Usuario X.

El Agente Contenedor de Seguridad (ACS) es un agente creado por el AAS en cada uno de los contenedores secundarios presentes en la plataforma, con el objetivo de que sean

los encargados de la autenticación de los agentes móviles cuando éstos migren hacia otros entornos de ejecución distintos a su origen de creación.

La Autoridad Certificadora (CA) no es parte de la extensión JADE-S, pero si es considerada para una mayor confianza y credibilidad en la administración de certificados digitales.

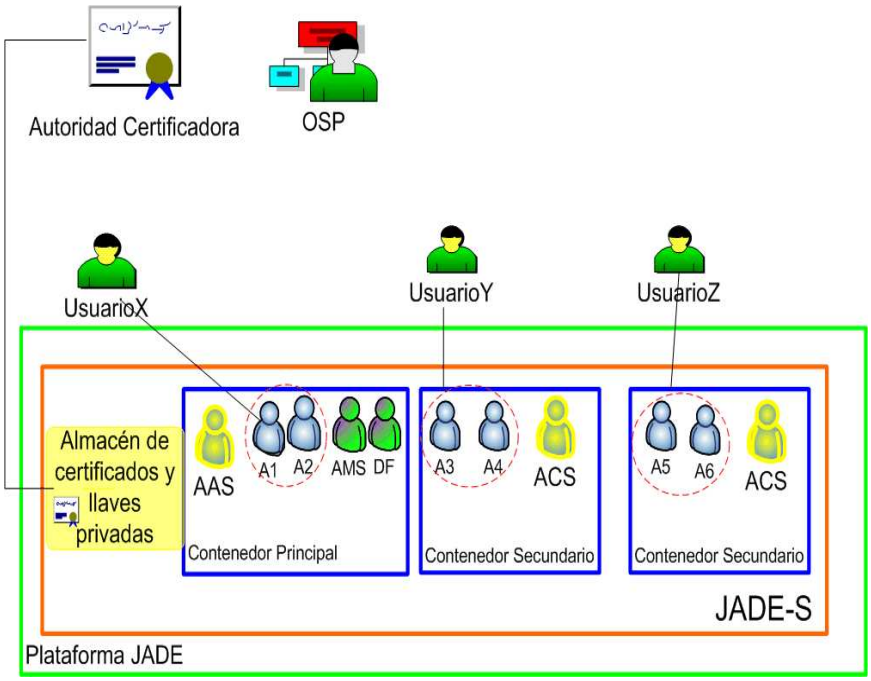

Figura 27. Actores propuestos en la plataforma JADE en su agregado JADE-S, para el soporte de agentes móviles seguros.

Una vez definidos los actores, a continuación se presenta el mecanismo de interacción entre ellos para conseguir la seguridad en agentes móviles:

El Usuario X presenta su certificado digital X509 ante el OSP (paso 1) para que manualmente lo importe[3] al almacén de certificados de la plataforma JADE en su agregado JADE-S (paso 2), y se le asigne al Usuario X su nombre de usuario y contraseña correspondiente (paso 3) (ver Fig. 28).

En la Figura 29 se presenta la continuación del mecanismo de interacción. Un Usuario X presenta ante el AAS un certificado digital X509 como primer fase de autenticación (paso

[3]Usando la herramienta keytool de Java la instrucción import se utiliza para almacenar los certificados en el keystore; el cual, es el almacén de certificados por default en Java.

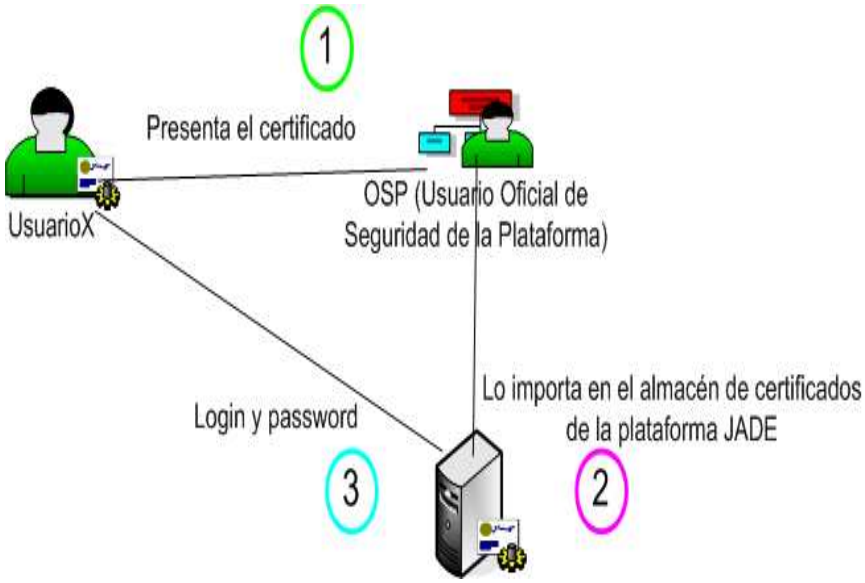

Figura 28. El OSP importa el certificado digital X509 presentado por el Usuario X, al almacén de certificados de la plataforma JADE y asigna el nombre de usuario y contraseña al Usuario X.

4). El AAS tiene la tarea de comprobar la validez del certificado, para esto consulta al almacén de certificados de la plataforma (paso 5); si el certificado es válido, el AAS solicita el nombre de usuario y contraseña del Usuario X (paso 6), los cuales están relacionados con el certificado; si los datos son incorrectos la plataforma termina su ejecución. Los ACS son creados por el AAS conforme se vayan iniciando los contenedores secundarios (paso 7).

Estando la plataforma JADE y su agregado JADE-S implementada con sus nuevos elementos, se puede iniciar la creación y ejecución de agentes móviles seguros (ver Fig.30). Los agentes móviles son creados por el AAS de la plataforma una vez autentificado el Usuario X (paso 8), durante su creación son dotados con sus respectivos certificados, los cuales se relacionan con el nombre de usuario y contraseñas asignados al Usuario X. Los agentes móviles tienen la capacidad de llevar un control de los contenedores visitados durante la ejecución de sus tareas mediante un histórico de trayectoria[4]; el cual, se implementa como un combate a los riesgos de seguridad a los que se enfrenta un agente móvil (paso 9). Cuando un agente móvil desea migrar hacia otro contenedor dentro de la plataforma, éste viaja

[4]Consultar Tabla I en capítulo II

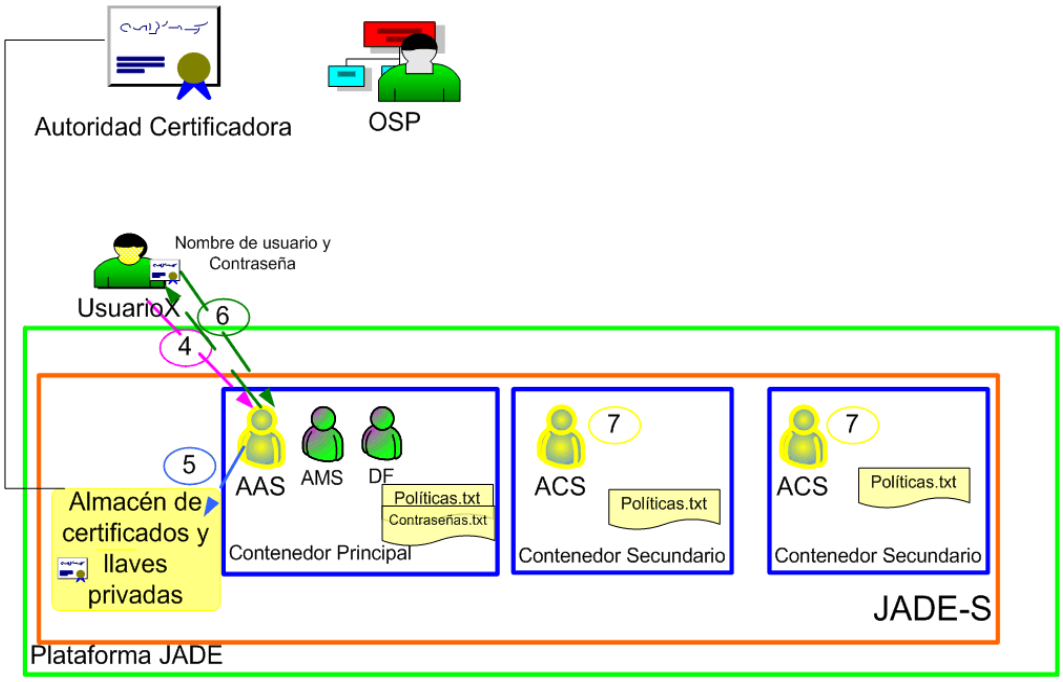

Figura 29. Dinámica propuesta para la extensión JADE-S para sustentar agentes móviles seguros.

hacia el contenedor destino junto con su certificado (paso 10) (se realiza el proceso de serialización), cuando llega al contenedor destino presenta ante el ACS correspondiente su certificado(proceso de deserialización), el ACS envía un mensaje ACL al agente AAS para verificar la validez del certificado (paso 11), el AAS verifica en el almacén de certificados y envía al ACS un mensaje sobre la validez del certificado (paso 12); si el certificado es válido el agente móvil puede ejecutar sus tareas en el contenedor.

Esta solución es la ideal para la extensión de la plataforma JADE para sustentar agentes móviles seguros, pero es aquí donde surge un imprevisto, un certificado digital al ser serializado se destruye (ver Fig.31); lo cual, involucra que el trabajo de investigación se vea obligado a tomar una decisión. El resolver el problema de serialización de certificados con la incertidumbre del tiempo que esto llevaría; ó tomar un paliativo a la propuesta de solución con el objetivo de probar el concepto del prototipo solución. La decisión fue tomar el paliativo, el cual se describe a continuación.

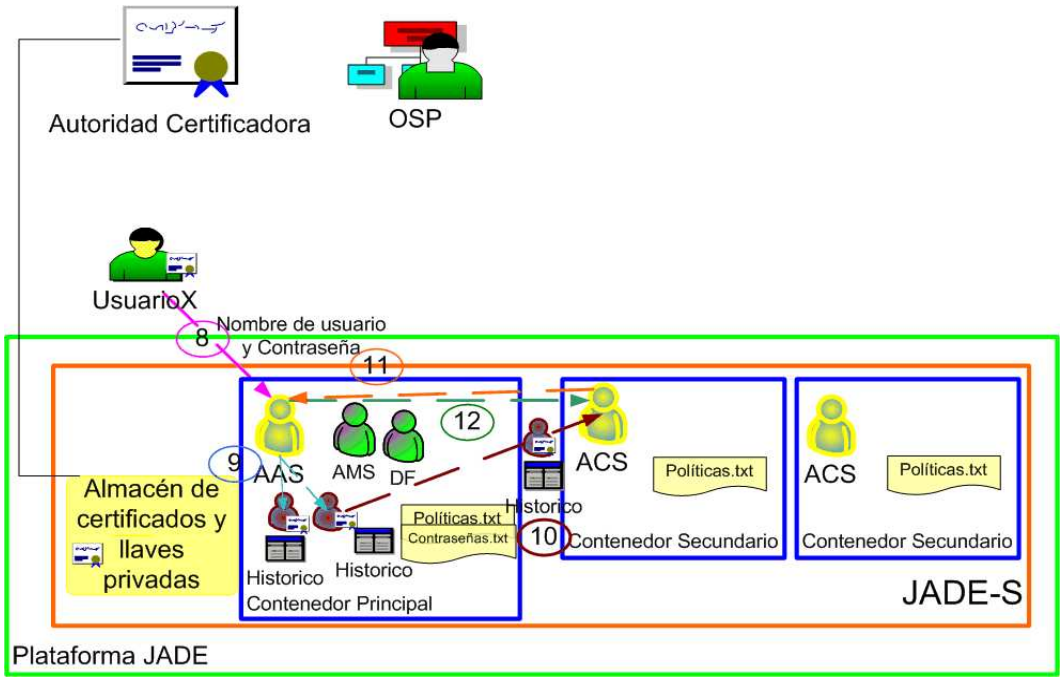

Figura 30. Agentes móviles creados en la plataforma JADE en su agregado JADE-S, migrando hacia otros contenedores llevando consigo su certificado e histórico de trayectoria.

V.3.3 Propuesta III

Como una alternativa ante el problema de serialización de certificados digitales se limitará su uso a los UsuariosX. Ahora los agentes móviles estarán sujetos a un identificador de agentes (AID) y contraseñas, que se les asignará de acuerdo al certificado digital presentado por su Usuario X creador. La asignación de los AIDs y contraseñas correspondientes a los agentes móviles es trabajo del OSP de la plataforma; el cual, llevará la administración mediante un archivo XML. El archivo XML representa la definición de un nuevo actor que servirá como base de datos en la extensión de la plataforma segura JADE-S (ver Fig.32). Se espera que el problema de serialización de certificados digitales X509 sea resuelto en poco tiempo. Se tiene contemplado dentro del archivo XML la existencia de un certificado digital al que por ahora se le nombra pseudocertificado. El objetivo es que la presente propuesta pueda completarse con el trabajo futuro propuesto en capítulos posteriores.

De acuerdo a la nueva definición de actores, a continuación se presenta el mecanismo de

Figura 31. Un certificado digital al ser serializado se convierte en un certificado no válido.

interacción entre ellos para conseguir la seguridad en agentes móviles.

Tomando como base la explicación de la Figura 28. Una vez almacenado el certificado digital X509 y asignado un nombre de usuario y contraseña al Usuario X, el OSP tiene la tarea de asignar un AID y contraseña para los agentes móviles (paso 4) que serán creados por el Usuario X registrado en ese momento, éstos datos son almacenados en un archivo XML que funge como base de datos de la plataforma (ver Fig.33).

Continuando con el mecanismo de interacción (ver Fig.34), un Usuario X presenta ante el AAS un certificado digital X509 como primer fase de autenticación (paso 5). El AAS tiene la tarea de comprobar la validez del certificado, para esto consulta al almacén de certificados de la plataforma (paso 6); si el certificado es válido, el AAS solicita el nombre de usuario y contraseña del Usuario X (paso 7), los cuales están relacionados con el certificado; si los datos son incorrectos la plataforma termina su ejecución. Los ACS son creados por el AAS conforme se vayan iniciando los contenedores secundarios (paso 8).

Estando la plataforma JADE y su agregado JADE-S implementada con sus nuevos elementos, se puede iniciar la creación y ejecución de agentes móviles seguros (ver Fig.35).

Para que un usuario pueda ejecutar una aplicación en la extensión de seguridad en

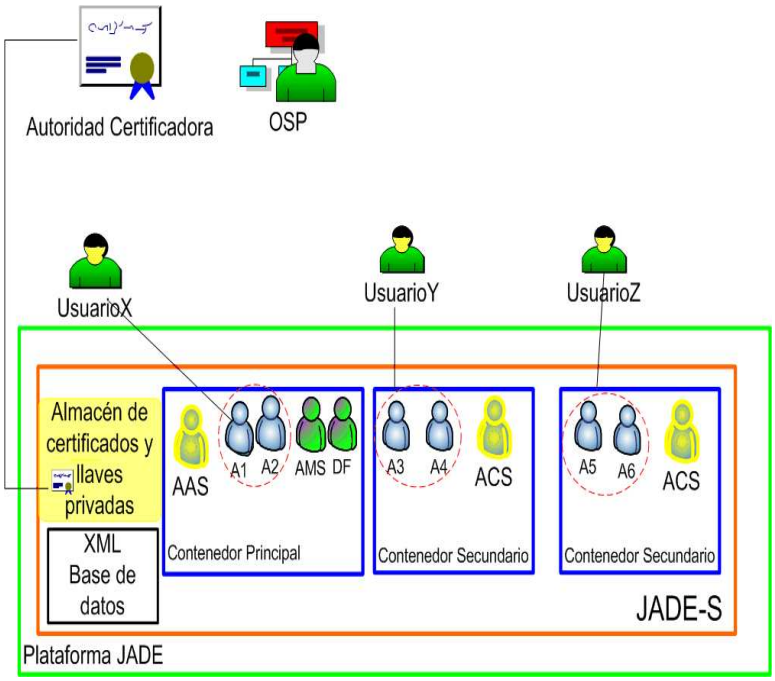

Figura 32. Actores propuestos en el paliativo de la plataforma JADE en su agregado JADE-S, para el soporte de agentes móviles seguros.

JADE, es necesario su autentificación ante el AAS por medio de su nombre de usuario y contraseña (paso 9). Los agentes móviles son creados por el AAS de la plataforma una vez autentificado el Usuario X. Si un Usuario X desea que sus agentes migren hacia otros entornos de ejecución, debe dotarlos de un mecanismo dónde por medio del nombre de usuario el AAS los acepte como agentes válidos (paso 10). Los agentes móviles tienen la capacidad de llevar un control de los contenedores visitados durante la ejecución de sus tareas mediante un histórico de trayectoria; el cual, se implementa como un combate a los riesgos de seguridad a los que se enfrenta un agente móvil. Cuando un agente móvil desea migrar hacia otro contenedor dentro de la plataforma, el agente móvil viaja hacia el contenedor destino llevando con el su pseudocertificado, AID y contraseña asignados (paso 11) (se realiza el proceso de serialización), cuando éste llega al contenedor destino presenta ante el ACS correspondiente su pseudocertificado, AID y contraseña (proceso de deserialización), el ACS envía un mensaje ACL al agente AAS para verificar la validez de los datos presentados por el agente móvil (paso 12), el AAS verifica en el archivo XML y

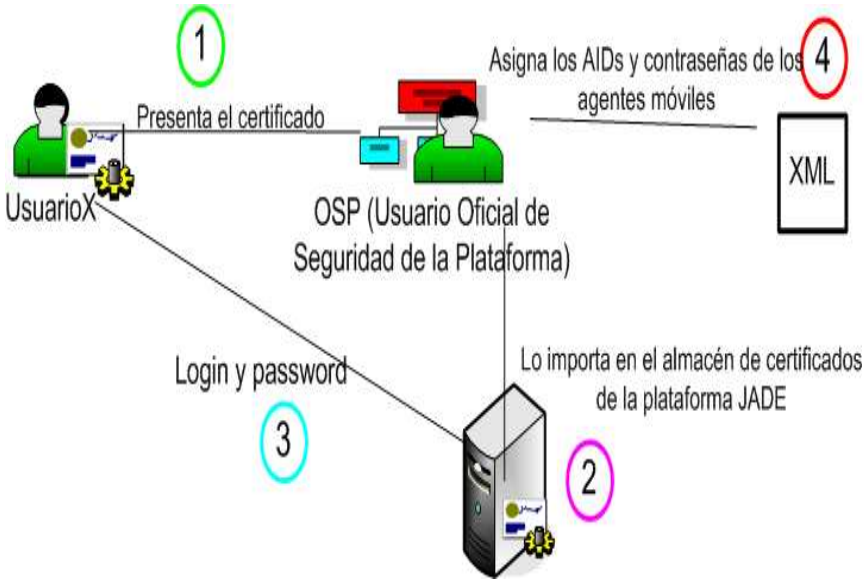

Figura 33. El OSP importa el certificado digital X509 presentado por el Usuario X, al almacén de certificados de la plataforma JADE; asigna los AIDs y contraseñas a los agentes móviles en relación al certificado digital presentado y almacena estos datos en un archivo XML.

envía al ACS un mensaje informándole si es o no un agente válido en la plataforma (paso 13).

V.3.4 Discusión

Esta propuesta de solución representa la implementación de la autenticación de agentes móviles seguros mediante el uso de certificados digitales X509, dando un paso adelante en seguridad a nivel de autenticación en la plataforma JADE en su agregado JADE-S. A continuación se discutirán sobre las ventajas y desventajas que proporciona esta propuesta de solución. En cuanto a ventajas se consideran las características de seguridad que se cubren por medio de la propuesta solución; y como desventaja se tiene el costo del aumento en tiempos de ejecución en la plataforma.

La seguridad actual ofrecida por JADE-S está basada en JAAS a nivel de autenticación; lo cual, nos provee de un mecanismo para la autenticación de usuarios y la autorización de acciones a realizar por los agentes en la plataforma JADE. Pero no provee a la plataforma

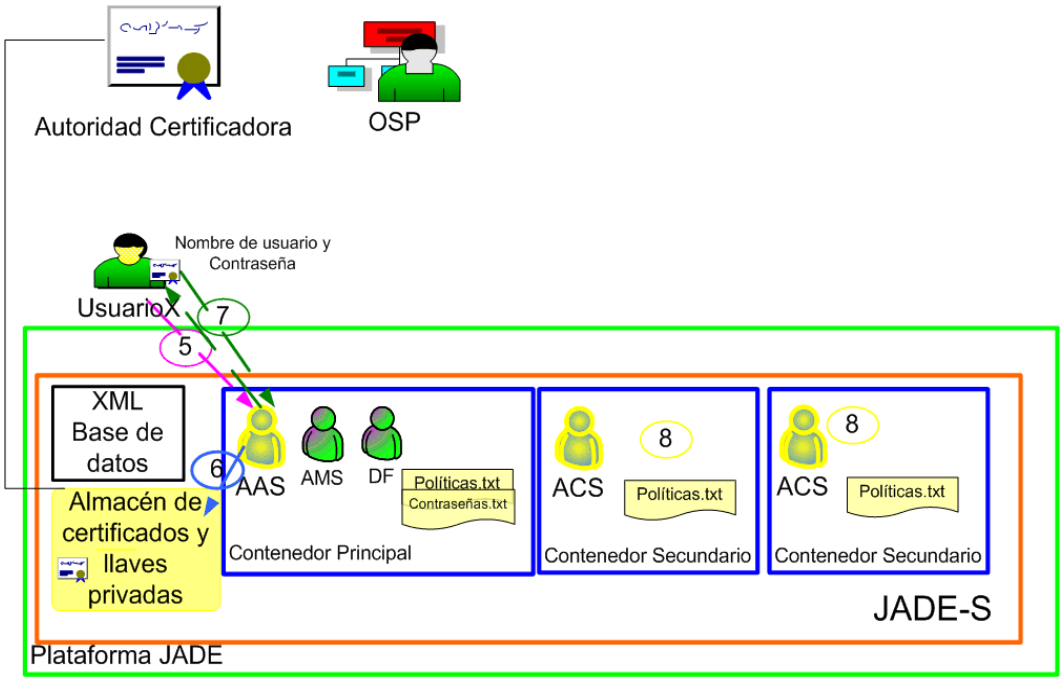

Figura 34. Dinámica propuesta en el paliativo para la extensión JADE-S para sustentar agentes móviles seguros.

de las características de confidencialidad y no-repudio; las cuales, son proporcionadas por los certificados digitales que además, son la base para el empleo de la Infraestructura de Llave Pública (PKI) en las aplicaciones que se desarrollen en un futuro bajo la plataforma JADE.

Por otro lado, la utilización del prototipo de JADE-S presentado en este trabajo de investigación, considera un aumento en los tiempos de ejecución en JADE-S; debido al intercambio de mensajes entre los nuevos actores en la plataforma con el fin de otorgar seguridad a ésta. En la Figura 36 se muestran las interacciones en la plataforma JADE-S durante la creación de un agente administrador de la seguridad, el cual usa certificados digitales, y se toma como ejemplo para comparar el incremento de mensajes intercambiados en relación a la Figura 37 dónde se muestra la creación de un agente X en el JADE-S actual. Otro ejemplo comparativo se ilustra en las Figuras 38 y 39, durante la creación de un agente móvil en el JADE-S actual y en el JADE-S propuesto. En las Figuras 40, 41, 42, 43 y 44 se presentan las interacciones de los actores propuestos en JADE-S, con el objetivo de

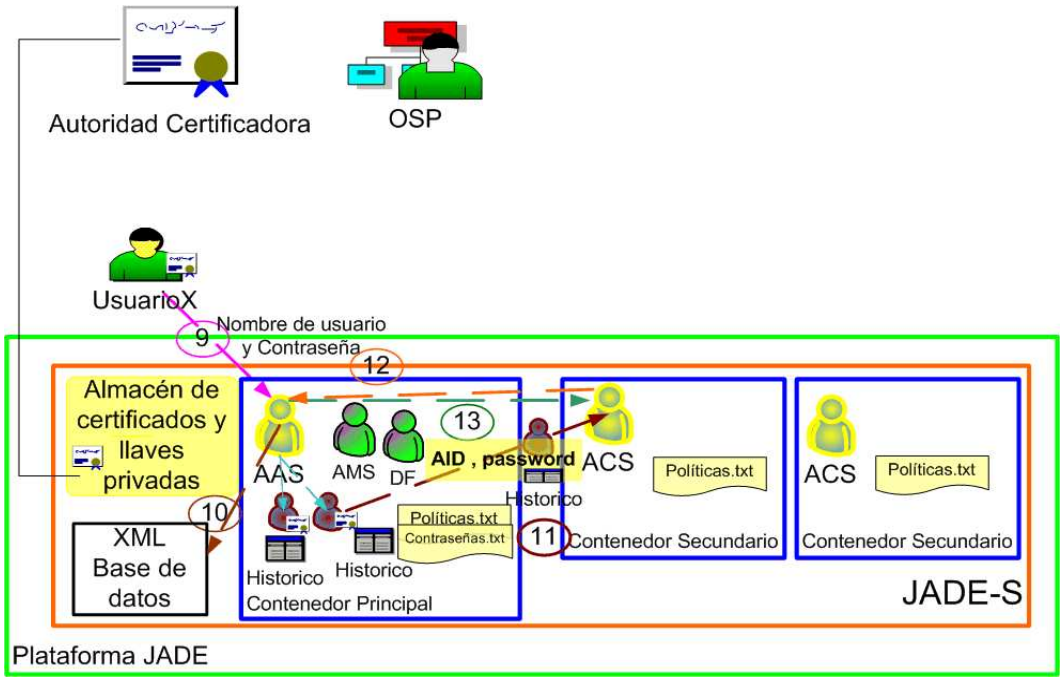

Figura 35. Paliativo para la extensión JADE-S para sustentar agentes móviles seguros.

representar el incremento de mensajes intercambiados en la plataforma JADE-S. En la Tabla IV, se muestra un condensado de las figuras anteriores expresando el incremento de mensajes en JADE-S propuesto en relación al JADE-S actual. Por otra parte, es necesario considerar una organización adicional por parte de los grupos que desarrollan sus aplicaciones bajo la plataforma JADE y JADE-S, en lo que respecta al manejo y administración del almacén de certificados digitales, el manejo de las bases de datos, los mismos certificados digitales y la elección de la Autoridad de Certificación[5] en quien confiar.

La Figura 45 representa las capas del middleware JADE; la cual, servirá de base en la explicación acerca de las clases que han sido incluidas para la funcionalidad de la propuesta en el agregado JADE-S. La clase medular se encuentra a nivel de plataforma; la cual, es la encargada del manejo de los certificados digitales X509 en la plataforma. A nivel de contenedor se agregó la clase correspondiente al histórico de trayectoria de los agentes móviles, y las clases encargadas de las peticiones de nombres de usuarios, contraseñas, AID´s; las

[5]Verisign es una de las más utilizadas junto con Thawte.

Tabla IV. Comparativa del número de mensajes intercambiados en el JADE-S actual y el JADE-S propuesto.

	Número de mensajes JADE-S actual	Número de mensajes JADE-S propuesto
Creación de un agente X	4	9
Creación de un agente móvil	4	6
Con los nuevos actores en JADE-S		
Autenticación de un Usuario X	No aplica	8
Creación de un agente móvil	No aplica	4
Migración de un agente móvil	No aplica	6
Interacción del AAS con el archivo XML	No aplica	4
Interacción del AAS con el almacén de certificados	No aplica	4

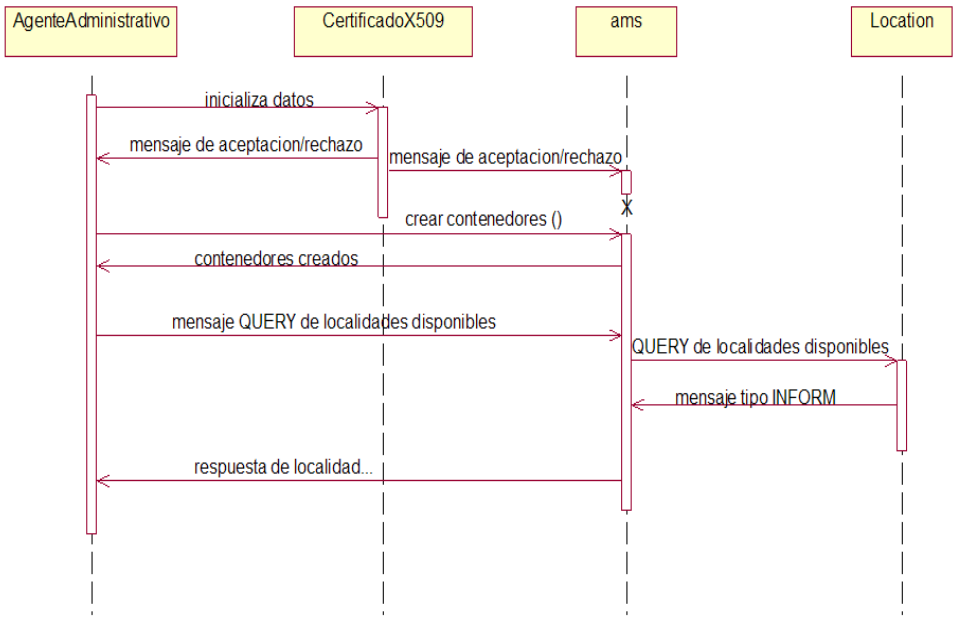

Figura 36. Interacciones en la plataforma JADE-S al crear un agente AAS.

cuales, ayudan en la autenticación tanto de usuarios como de agentes en la plataforma. A nivel de agentes administrativos se encuentran las clases que pertenecen al Agente Administrador de la Seguridad (AAS) y Agente Contenedor de Seguridad (ACS); los cuales, tienen la tarea de la nueva administración de seguridad en la plataforma. Y por último, a nivel de agentes comunes se encuentra la clase correspondiente a los agentes móviles que pueden ser creados en los contenedores. Esta clase de agentes móviles, representa una plantilla de todas las funcionalidades que JADE ofrece para éste tipo de agentes, además de presentar

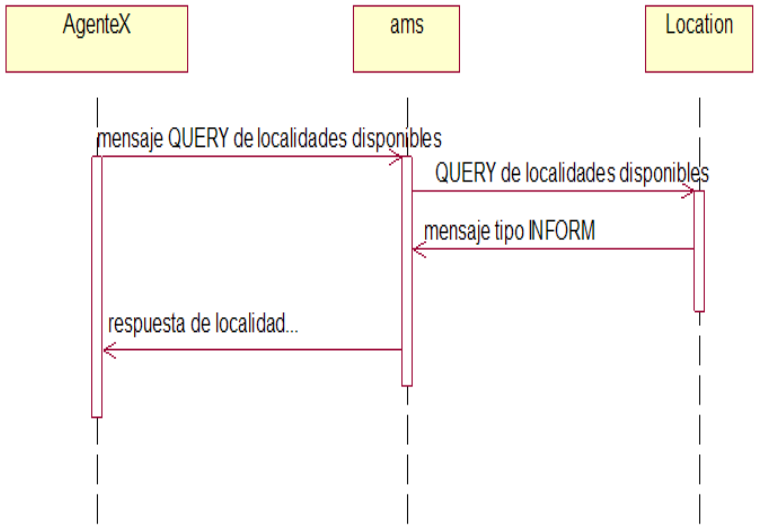

Figura 37. Interacciones en la plataforma JADE-S actual al crear un agente X.

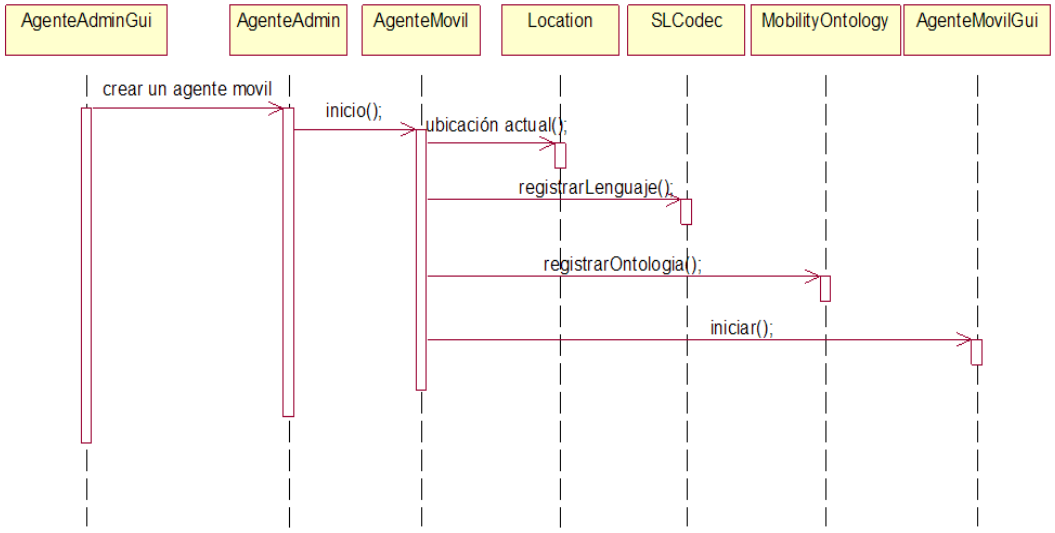

Figura 38. Interacciones en la plataforma JADE-S al crear un agente móvil.

una manera de utilizar la propuesta solución diseñada en este trabajo de investigación. En la Figura 46, se presenta el diagrama de clases del prototipo solución.

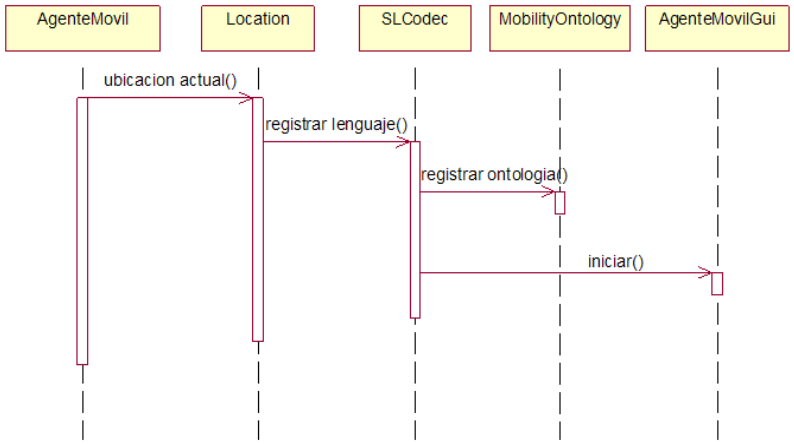

Figura 39. Interacciones en la plataforma JADE-S actual al crear un agente móvil.

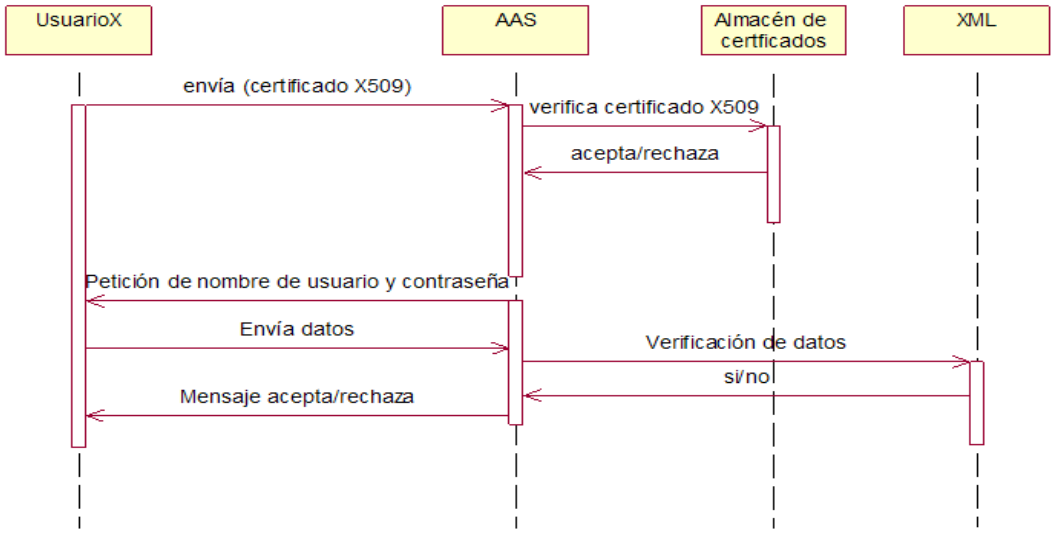

Figura 40. Interacciones de los nuevos actores en JADE-S durante la autenticación de un Usuario X.

V.4 Conclusiones

El desarrollo del prototipo solución presentado para dotar de seguridad a los agentes móviles en la plataforma JADE en su agregado JADE-S, presentó variaciones debido a limitantes de tecnologías; como resultado se presentó una propuesta menos general y completa que la propuesta inicial, pero cumpliendo con el objetivo del sustento de agentes móviles seguros

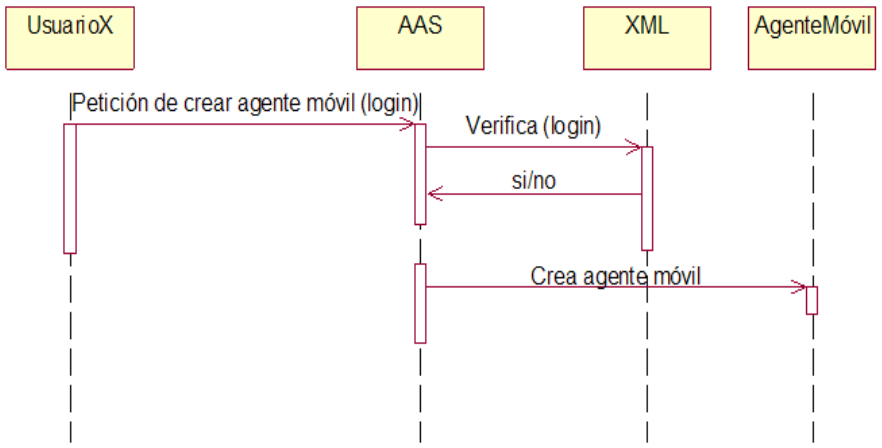

Figura 41. Interacciones de los nuevos agentes en JADE-S durante la creación de un agente móvil.

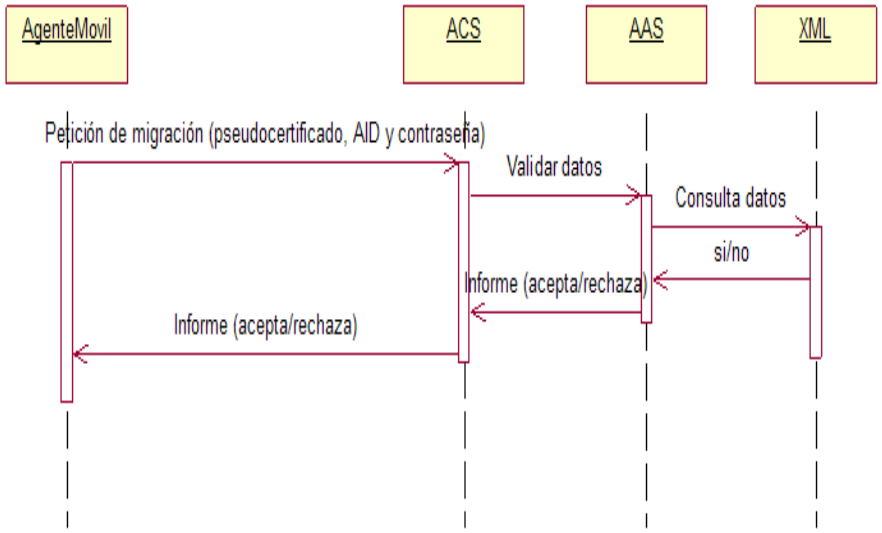

Figura 42. Interacción de los nuevos agentes en JADE-S durante la migración de un agente móvil.

en la plataforma JADE. La propuesta se desarrolla en el nivel 1 correspondiente a la autenticación dentro de JADE-S lo cual, logramos a través de los certificados digitales. El integrar un transporte seguro en la comunicación entre dos plataformas JADE con HTTP-S era con el fin de proteger a los agentes en el momento de su migración inter-plataforma, sin perder de vista nuestra atención central en la autenticación de agentes móviles. En el siguiente

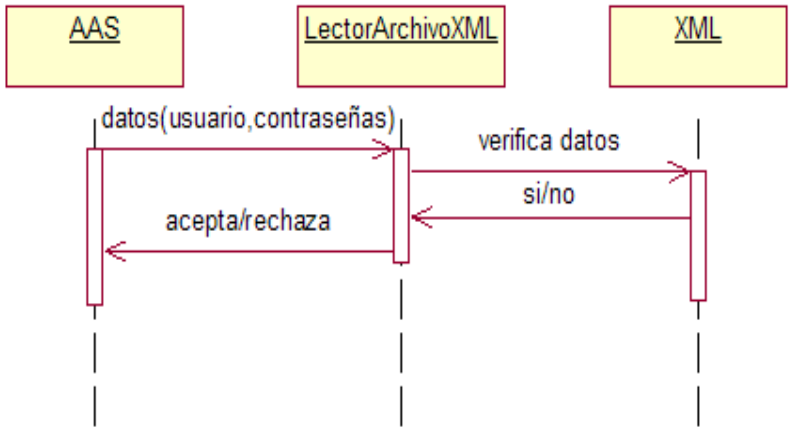

Figura 43. Interacción del AAS con el archivo XML de la plataforma JADE-S.

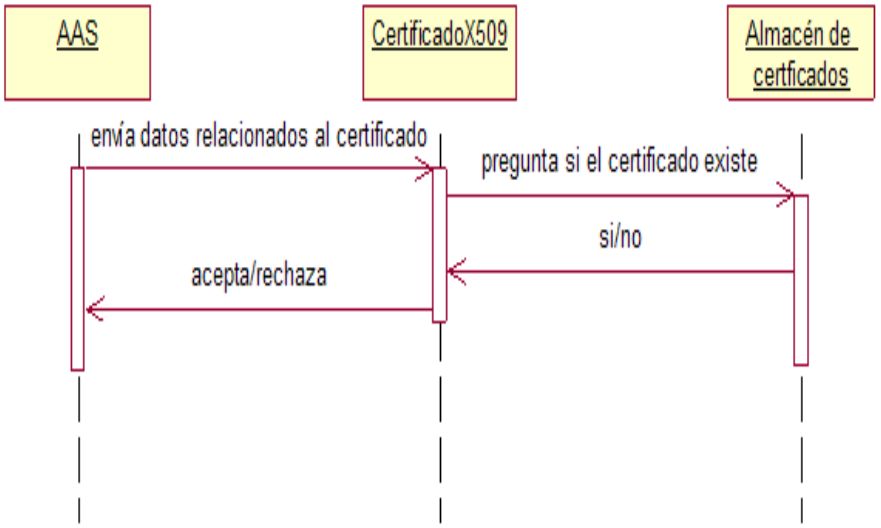

Figura 44. Interacción del AAS con el almacén de certificados de la plataforma JADE-S.

capítulo se presentan las conclusiones generales del presente trabajo de investigación así como las propuestas de trabajo futuro.

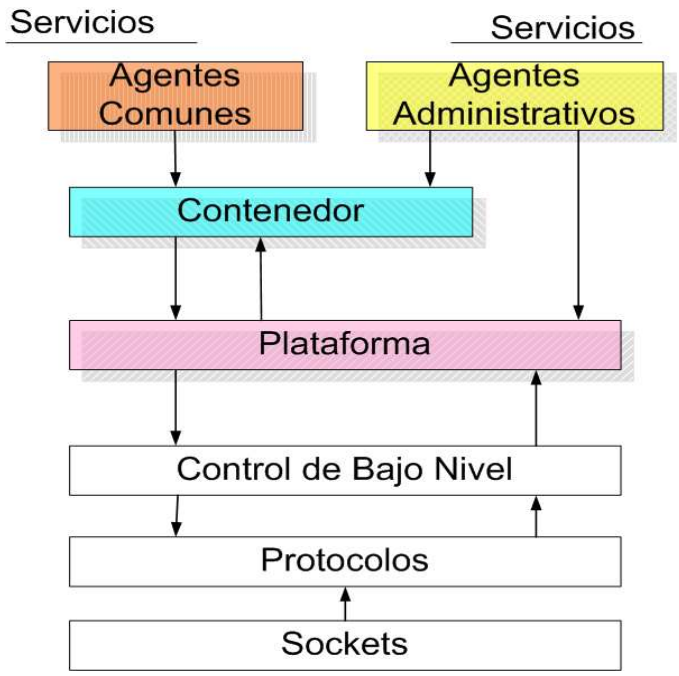

Figura 45. Representación de las capas de JADE.

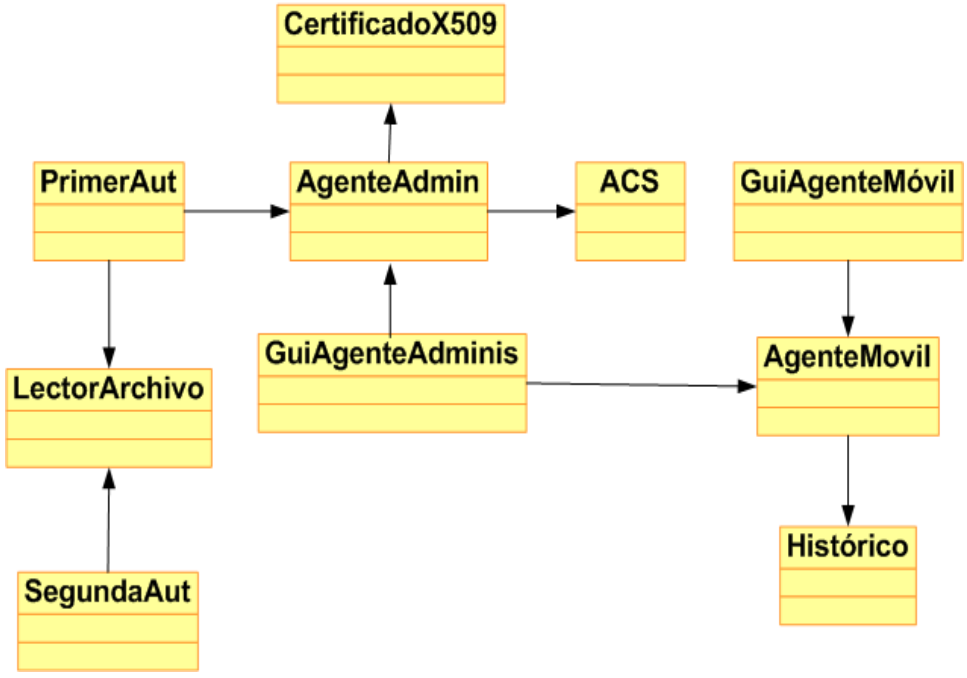

Figura 46. Diagrama de clases del prototipo solución de agentes móviles seguros.

Capítulo VI

Conclusiones y trabajo futuro

"La gota horada la roca, no por su fuerza sino por su constancia".
Ovidio

En este capítulo se presenta un resumen del trabajo realizado, se hacen algunas reflexiones y se proponen algunas ideas como trabajo futuro de investigación.

VI.1 Resúmen

El presente trabajo de investigación muestra un análisis realizado sobre la seguridad informática y movilidad en sistemas multiagentes. Para esto, como primer paso se realizó un estudio acerca de los fundamentos y marco teórico relacionado a la teoría de agentes, middlewares, seguridad y movilidad en sistemas multiagentes. Además de un estudio comparativo de los middlewares existentes para el desarrollo de sistemas multiagentes basados en los intereses de seguridad y movilidad. El middleware elegido fue JADE y su agregado JADE-S; el cual, sirvió como base para el análisis, identificación de oportunidades de mejora y la implementación de un prototipo de solución.

Durante el estudio y análisis de los mecanismos y protocolos de seguridad para sistemas multiagentes, se abordaron los mecanismos de seguridad ofrecidos por el lenguaje Java, la especificación de seguridad del estándar FIPA y el protocolo seguro HTTP-S. El lenguaje Java fue estudiado por ser el lenguaje bajo el cual se desarrolla el middleware JADE. La especificación de seguridad del estándar FIPA, fue estudiado con el objetivo de suponer el por qué ahora es una especificación obsoleta y tener una visión de posibles modelos arquitectónicos que pudieran establecerse como estándares de seguridad para el desarrollo de aplicaciones en sistemas multiagentes. Y el protocolo seguro HTTP-S, con el objetivo de dotar de seguridad a las comunicaciones inter-plataformas.

El análisis de la seguridad en JADE-S, se llevó a cabo a través de una aplicación a la que se llamó: aplicación del banquero; la cual, involucraba agentes estacionarios, comunicaciones entre agentes por medio de mensajes ACL, y autenticación de usuarios mediante nombre de usuarios y contraseñas (JAAS). Una vez analizados los tres niveles de seguridad ofrecidos por JADE-S, se detectaron algunas limitaciones y oportunidades de mejora; de las cuales, el trabajo se enfocó hacia los problemas de autenticación de agentes y movilidad.

Con base en lo anterior, se realizó un análisis sobre los mecanismos y protocolos de movilidad para sistemas multiagentes. Estudiando la serialización de objetos y los protocolos de movilidad del estándar FIPA; los cuales, dan sustento a la movilidad en sistemas multiagentes. JADE en lo que respecta a la movilidad no se basa en el estándar FIPA, pero propone extensiones y modificaciones a la propuesta por el estándar FIPA.

Con el objetivo de proporcionar seguridad a los agentes móviles desarrollados en la plataforma JADE, se propuso un prototipo solución para la implementación de agentes móviles seguros en la plataforma JADE en su agregado JADE-S, a nivel de autenticación basado en certificados digitales X509. Durante el desarrollo de la propuesta inicial surgieron problemas derivados de limitantes encontradas en dos de las tecnologías utilizadas: 1)El uso del protocolo HTTP-S en JADE y 2)la interacción del proceso de serialización en Java con los certificados digitales X509. Por lo que se obtuvo como resultado una propuesta final aunque menos general y completa que la propuesta inicial, cumpliendo con el objetivo de servir como prueba de concepto del sustento a la seguridad de agentes móviles en JADE.

Esta solución involucra la adición de dos nuevos agentes administradores de la seguridad en la plataforma JADE en su agregado JADE-S, el manejo de certificados digitales por parte de los usuarios, una organización adicional por parte de los administradores de la plataforma JADE y la implementación de bases de datos y almacenes de llaves en la plataforma JADE.

VI.2 Conclusiones

La implementación de seguridad en sistemas multiagentes exige la propuesta de soluciones que conlleven una visión futurista; la cual, se base en la visión del cómputo ubicuo, con el

objetivo de que apoyen la invisibilidad, escalabilidad, interoperabilidad, confiabilidad, privacidad y sobre todo seguridad para las aplicaciones que se desarrollen con agentes. Contar con un modelo de seguridad; el cual, sea aplicable de forma general a las aplicaciones desarrolladas para sistemas multiagentes, resulta difícil debido a que cada una de las aplicaciones presentan contextos y situaciones particulares.

Específicamente en la plataforma JADE, la cual basa su funcionamiento en el estándar FIPA, es necesario el continuar con la implementación de seguridad en su agregado JADE-S, con el objetivo de proveer a las futuras aplicaciones desarrolladas bajo esta plataforma de seguridad. El enfoque tomado en el presente trabajo de investigación hacia la seguridad en agentes móviles, implica el pensar en la necesidad de la utilización de protocolos seguros para la migración de agentes y el uso de certificados digitales para su autentificación. El poner un transporte seguro elimina solo el problema de los nodos maliciosos, es decir, el proteger a los agentes de las plataformas hostiles, sigue estando abierto. La utilización de certificados digitales otorga un grado más de confianza para la administración de la plataforma, más no garantiza el hecho de que un agente no pueda ser atacado por una plataforma o agente hostil.

El prototipo solución presentado en este trabajo de investigación, se enfoca al sustento de la seguridad de agentes móviles desarrollados en la plataforma JADE. Proponiendo una solución para enfrentar algunos de los ataques que pueden sufrir los agentes móviles durante la ejecución de sus tareas, así como el cumplimiento de algunos de los objetivos de seguridad, tales como autentificación y no-repudio. El cumplimiento de la autorización y disponiblidad se resuelve con una de las propuestas como trabajo futuro donde se habla de la integración de políticas basadas en certificados digitales. Confidencialidad e integridad pueden resolverse mediante el ocultamiento de información; lo cual, aún está en desarrollo en la plataforma JADE.

En conclusión, la implementación del prototipo solución presentado otorga a la plataforma JADE, una propuesta de seguridad que es útil tanto para agentes móviles como estacionarios, y tiene como propósito el que se tome como base para los trabajos futuros

relacionados a seguridad en la plataforma JADE.

VI.3 Trabajo futuro

Como resultado de este trabajo de tesis, las siguientes ideas se presentan como propuestas para trabajo futuro sobre la misma línea de investigación.

- Explorar la modificación de la ontología mobile-agent-description, para la integración de certificados a los agentes móviles. En la presente investigación este tema fue abordado como una alternativa de integración de certificados en los agentes móviles, mediante un análisis se detectó que la modificación de esta ontología involucra una modificación total en la plataforma, lo que llevaría un tiempo considerable para su implementación. Esta idea no está fuera del alcance a ser realizada, pero si involucra un conocimiento experto sobre la estructura JADE, además de una visión sobre futuras implementaciones.

- Implementación de políticas de plataforma basadas en certificados digitales. La utilización de certificados digitales cubre el primer nivel de seguridad que es la autenticación; el siguiente paso es tener el control de acceso a los recursos y control de las acciones que se realicen en la plataforma en base a certificados digitales.

- Implementación de un protocolo de migración para agentes móviles. El investigar las bondades de un protocolo de migración para JADE sería de gran provecho, debido a que se analizarían los protocolos actuales de la plataforma y se hiciera una propuesta para protocolos que soporten la migración de agentes junto con el soporte de comunicación de los protocolos actuales en JADE.

- Implementación de protocolos seguros en la plataforma JADE, en específico RMI-SSL y HTTP-S. Esto con el objetivo de que la plataforma JADE ofrezca seguridad en la transferencia de agentes a través de la red, y con esto, sea considerado para el desarrollo de aplicaciones reales de negocios o comercio electrónico por dar algún ejemplo.

- Analizar el comportamiento de los agentes que resultan de una clonación en la plataforma JADE. Un agente clonado es una copia de un agente residente en la plataforma JADE; el cual, se utiliza para la realización de una tarea en específico y al terminarla morir. Hasta ahora no se han explorado las ventajas y desventajas que los agentes clonados pudieran ofrecer a la plataforma JADE, o a las aplicaciones desarrolladas bajo esta plataforma.

Por algún motivo tuve un error al generar loa bibliografía y tuve que hacer algunos trucos, el siguiente párrafo no esta en la versión final impresa.

(Saha y Mukherjee, 2003) (Lyytinen y Yoo, 2002). (Pantic *et al.*, 2005). (Jennings, 2001) (Lange y Oshima, 1999) (Bierman y Cloete, 2002) (Bellifemine *et al.*, 2003) (Novák *et al.*, 2003) (Ametller *et al.*, 2003a) (Ametller *et al.*, 2004) y (Ametller *et al.*, 2003b) (Robles *et al.*, 2002)

(Bellifemine *et al.*, 2005a) (Bellifemine *et al.*, 2005b)

Bibliografía

Ametller, J., S. Robles, y J. Borrell 2003a. "Agent migration over fipa acl messages.". En: "Mobile Agents for Telecommunication Applications (MATA)", volume 2881 of Lecture Notes in Computer Science. 210-219 p.

Ametller, J., S. Robles, y J. Borrell 2003b. "Secure mobile agents applications: Implementation through platform services". En: "Mobile Agents for Telecommunication Applications (MATA)", volume 2881 of Lecture Notes in Computer Science. 265-274 p.

Ametller, J., S. Robles, y J. A. Ortega-Ruiz 2004. "Self-protected mobile agents". En: "3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004)", 19-23 August 2004, New York, NY, USA. IEEE Computer Society. 362–367 p.

Bellifemine, F., G. Caire, A. Poggi, y G. Rimassa 2003. "Jade a white paper". EXP in search of innovation, Telecom Italia Lab (TILAB), 3(3):6-19 p.

Bellifemine, F., G. Caire, T. Trucco, y G. Rimasa 2005a. "Jade programmer´s guide". EXP in search of innovation, Telecom Italia Lab (TILAB).

Bellifemine, F., G. Caire, T. Trucco, G. Rimasa, y R. Mungenast 2005b. "Jade administrator´s guide". EXP in search of innovation, Telecom Italia Lab (TILAB).

Bernstein 1996. "Middleware". Communication of the ACM, 39(2):87–98 p.

Bierman, E. y E. Cloete 2002. "Classification of malicious host threads in mobile agent computing". En: "SAICSIT '02: Proceedings of the 2002 annual research conference of the South African institute of computer scientists and information technologists on Enablement through technology", 16-18 September 2004, Republic of South Africa. Proceedings of SAICSIT. 141–148 p.

Board, J. 2005. “Jade security guide”. EXP in search of innovation, Telecom Italia Lab (TILAB).

Caire, G. 2003. “Jade tutorial, jade programming for beginners”. EXP in search of innovation, Telecom Italia Lab (TILAB).

Cortese, E., F. Quarta, G. Vitaglione, y P. Vrba 2003. “Scalability and performance of the jade message transport system. analysis of suitaility for holonic manufacturing systems”. EXP in search of innovation, Telecom Italia Lab (TILAB), 3(3):52-64 p.

Finin, T., R. Fritzson, D. McKay, y R. McEntire 1994. “Kqml as an agent communication language”. En: “CIKM ’94: Proceedings of the third international conference on Information and knowledge management”, November 29 - December 2, 1994, New York, NY, USA. ACM Press. 456–463 p.

FIPA 1998. “Fipa98 specification, agent security management”. http://www.fipa.org/specs/fipa00020/OC00020A.html. Foundation for Intelligent Physical Agents. Fecha de consulta: Marzo, 2004.

FIPA 2001. “Fipa interaction protocol library specification”. http://www.fipa.org/specs/fipa00025/XC00025E.html. Foundation for Intelligent Physical Agents.Fecha de consulta: Abril, 2004.

FIPA 2002a. “Fipa acl message structure especification”. http://www.fipa.org/specs/fipa00061/SC00061G.html. Foundation for Intelligent Physical Agents. Fecha de consulta: Abril, 2004.

FIPA 2002b. “Fipa agent management support for mobility specification”. http://www.fipa.org/specs/fipa00087/DC00087C.html. Foundation for Intelligent Physical Agents. Fecha de consulta: Abril, 2004.

FIPA 2002c. "Fipa communicative act library specification". http://www.fipa.org/specs/fipa00037/SC00037J.pdf. Foundation for Intelligent Physical Agents. Fecha de consulta: Abril, 2004.

FIPA 2003. "Fipa content languajes library specification". http://www.fipa.org/specs/fipa00007/DC00007C.html. Foundation for Intelligent Physical Agents. Fecha de consulta: Abril, 2004.

FIPA 2004. "Fipa agent management specification". http://www.fipa.org/specs/fipa00023/SC00023K.html. Foundation for Intelligent Physical Agents. Fecha de consulta: Marzo, 2004.

Georgousopoulos, C. y O. F. Rana 2002. "An approach to conforming a mas into a fipa-compliant system". En: "AAMAS '02: Proceedings of the first international joint conference on Autonomous agents and multiagent systems", 15-19 July, Bologna, Italy. ACM Press. 968–975 p.

Ghosh, D. 2000. "Mobile ip". Crossroads, 7(2):10–17 p.

JADE, J. A. D. F. 2005. "Uso de jicp-ssl en jade/leap". http://jade.tilab.com/doc/index.html. Telecom Italia Lab (TILAB). Fecha de consulta: Febrero,2005.

Jennings, N. R. 2001. "An agent-based approach for building complex software systems". Communication of the ACM, ACM Press, 44(4):35–41 p.

Jennings, N. R. y M. J. Wooldridge 1998. "Applications of intelligent agents". En: "Agent Technology: Foundations, Applications, and Markets", Springer-Verlag. 3-28 p.

Johner, H., S.Fujiwara, A.Yeung, A. Stephanou, y J. Whitmore 2000. "Deploying a pulic key infrastructure". http://www.redbooks.ibm.com. IBM. Fecha de consulta: Febrero, 2005.

Juárez, M. I. y O. I. Lepe 2005. "Consideraciones de seguridad en agentes móviles entre plataformas jade". En: "Avances en la Ciencia de la Computación ENC05, Sexto Encuentro Internacional de Computación, Taller de Cómputo Móvil.", 26-30 Septiembre, Puebla, Puebla, México. Sociedad Mexicana de Ciencias de la Computación, SMCC. 94–97 p.

Lange, D. B. y M. Oshima 1999. "Seven good reason for mobile agents". Communication of the ACM, 42(3):88–89 p.

Lepe, O. I., L. E. Vizcarra, E. Arriola, A. Galindo, y F. Pérez 2004. "Requirements analysis or ubiquitous computing system to be used during a congress". En: "Encuentro Internacional de Ciencias de la Computación ENC04, Quinto Encuentro Internacional de Computación, Taller de Cómputo Móvil.", 20-24 Septiembre, Colima, Colima, México. Sociedad Mexicana de Ciencias de la Computación, SMCC. 94–104 p.

Lyytinen, K. y Y. Yoo 2002. "Issues and challenges in ubiquitous computing". Communications of the ACM, 45(12):63–65 p.

Novák, P., M. Rollo, J. Hodík, T. Vlcek, y M. Pechoucek 2003. "X-security architecture in agent cities". EXP in search of innovation, Telecom Italia Lab (TILAB), 3(3):1-13 p.

Pantic, M., N. Sebe, J. F. Cohn, y T. Huang 2005. "Affective multimodal human-computer interaction". En: "MULTIMEDIA '05: Proceedings of the 13th annual ACM international conference on Multimedia", 6 - 11 November, Hilton, Singapore. ACM Press. 669–676 p.

Persiano, P. y I. Visconti 2003. "A secure and private system for subscription-based remote services". ACM Trans. Inf. Syst. Secur., 6(4):472–500 p.

Poslad, S., R. Bourne, A. Hayzelden, y P. Buckle. "Agent technology for communications infrastructure: An introduction". citeseer.ist.psu.edu/465178.html. Queen Mary, University of London. Fecha de consulta: Mayo ,2005.

Robles, S., J. Mir, J. Ametller, y J. Borrell 2002. "Implementation of secure architectures for mobile agents in marism-a". En: "Mobile Agents for Telecommunication Applications (MATA)", volume 2521 of Lecture Notes in Computer Science. Springer Verlag. 182-191 p.

Roth, V. y M. Jalali-Sohi 2001. "Concepts and architecture of a security-centric mobile agent server". En: "Proceedings of the 5th International Conference on Autonomous Distributed Systems (ISADS 2001)", March, Dallas, Texas, USA. IEEE Computer Society. 435 pp.

Saha, D. y A. Mukherjee 2003. "Pervasive computing: A paradigm for the 21st century". Computer, IEEE Computer Society, 36(3):25–31 p.

Satyanarayan, M. 2001. "Pervasive computing:vision and challenges". Personal Communications, IEEE, 8(4):10–17 p.

Schoeman, M. y E. Cloete 2003. "Architectural components for the efficient design of mobile agent systems". En: "SAICSIT '03: Proceedings of the 2003 annual research conference of the South African institute of computer scientists and information technologists on Enablement through technology", 17-19 September, Republic of South Africa. South African Institute for Computer Scientists and Information Technologists. 48–58 p.

Shehory, O. y K. Sycara 2000. "The retsina communicator". En: "AGENTS '00: Proceedings of the fourth international conference on Autonomous agents", 21-25 September, Barcelona, Spain. ACM Press. 199–200 p.

Sun-Developer-Network 2005a. "Security and the java platform". http://java.sun.com/security/. SUN. Fecha de consulta: Junio, 2005.

Sun-Developer-Network 2005b. "Seguridad en la plataforma java jdk 1.2". http://www.programacion.com/java/tutorial/security1dot2/. SUN. Fecha de consulta: Junio, 2005.

Thompson, M. R., A. Essiari, y S. Mudumbai 2003. “Certificate-based authorization policy in a pki environment”. ACM Trans. Inf. Syst. Secur., 6(4):566–588 p.

Weiser, M. 1991. “The computer for the 21st century”. Scientific American, 265(3):94–104 p.

Anexo 1

A continuación se presentan las conversaciones y correos intercambiados durante las pruebas de comunicación, con el objetivo de dejar prueba testimonial del proceso de pruebas y facilitar un trabajo futuro en este sentido.

Correo 1

Hola Marcel Isabel,

Nuestro equipo aun continua haciendo investigación en el tema de seguridad en agentes móviles. Nuestro marco de implementación para prototipos es Jade, plataforma para la que hemos desarrollado algunos componentes (como el http mtp). Actualmente Joan Ametller, de nuestro equipo, está escribiendo su tesis doctoral sobre el tema de la seguridad y la movilidad en agentes. Si le envias un mail quizás pueda ayudarte: Joan.Ametller@uab.es

Saludos cordiales,

Sergi Robles (Sergi.Robles@uab.es) Tel +34 93 581 2395 FAX +34 93 581 3033 Department of Information and Communications Engineering Universitat Autonoma de Barcelona-08193 Bellaterra,Spain

Correo 2

Hola,

nuestra principal linea de investigación es la proteccion de agentes móviles que siguen itinerarios. Los mecanismos de protección a los que nos hemos dedicado se basan en proteger los datos i el codigo de los agentes en entornos donde puede haber plataformas hostiles. Lamentablemente ninguno de los mecanismos propuestos ha sido implementado dentro de la plataforma JADE, que proporciona su propio modelo de seguridad.

Te mando algunas referencias de trabajos nuestros por si te interesa leer los temas que abordamos. Si tienes qualquier duda puedes preguntarmela.

[1] J. Ametller, S. Robles and J.A. Ortega-Ruiz. Self-Protected Mobile Agents. In 3rd International Conference on Autonomous Agents and Multi Agents Systems. ACM Press, 2004.

[2] J. Mir and J. Borrell, "Protecting Mobile Agent Itineraries", In Mobile Agents for Telecommunication Applications (MATA). Springer Verlag, vol. 2881 of Lecture Notes in Computer Science, 275-285, October 2003.

[3] S. Robles, J. Mir and J. Borrell. "MARISM-A: An Architecture for Mobile Agents with Recursive Itinerary and Secure Migration", In 2nd. IW on Security of Mobile Multiagent Systems. Bologna, July 2002.

Un saludo! joan

Joan Ametller Esquerra Departament d'Enginyeria de la Informació i de les Comunicacions Universitat Autònoma de Barcelona - 08193 Bellaterra, Spain Despatx QC/2003 Tel: +34935813577 E-Mail: Joan.Ametller@uab.es

Conversación con Joan Amettler.

Joan dice:

ok, pon el jade en modo grafico

marcel dice:

ya

Joan dice:

y enciende un dummy agent

marcel dice:

ya

Joan dice:

ok vete a receiver, boton derecho add

marcel dice:

yap

Joan dice:

en NAME

Joan dice:

ping@tao.uab.es:1099/JADE

marcel dice:

ya

Joan dice:

en addresses, boton derecho add

Joan dice:

http://tao.uab.es:7778/acc

Joan dice:

y le das a ok

marcel dice:

ya

Joan dice:

Content del mensaje

Joan dice:

ping Joan dice:

asegurate que el comunicative act es ACCEPT-PROPOSAL

marcel dice:

sip *Joan dice:*

envia

marcel dice:

ya

Joan dice:

no he recibido nada,... un momento

Joan dice:

dime tu ip

marcel dice:

158.97.83.224

Joan dice:

ei, algunu problema ?

marcel dice:

intenta con la 158.97.80.94

marcel dice:

158.97.80.194

marcel dice:

lo intente de nuevo y nada

Joan dice:

no, ahora soy yo que lo tengo apagado

Joan dice:

tienes xindows XP verdad ?

marcel dice:

sip

Joan dice:

tienes el puerto 7778 filtrado

Joan dice:

no admites conexiones entrantes por este puerto

Joan dice:

tienes el service pack 2 instalado?

marcel dice:

si

Joan dice:

mira, vete al panel de control

Joan dice:

hay una opcion que es firewall de windows

Joan dice:

lo ves

marcel dice:

ya

Joan dice:

siendo administrador

Joan dice:

tienes la opcion de abrir determinados puertos

Joan dice:

un momento

Joan dice:

ok

Joan dice:

tienes una pestaña que se llama excepciones

Joan dice:

y una opcion que se llama añadir puerto o algo asi

Joan dice:

alli te pde un nombre i un numero de puerto

Joan dice:

el nombre puedes poner lo que quieras, pero tienes que hacer esta operacion con el puerto 7778 i el puerto 1099

Joan dice:

creo que este era tu problema

marcel dice:

ya los agregue

Joan dice:

si quieres probamos entonces

marcel dice:

sip

Joan dice:

hazlo en todas las maquinas que esten involucradas en la comunicacion

Joan dice:

pues hablamos mañana si te parece y volvemos a hacer la prueba si quieres

marcel dice:

ok

Joan dice:

hasta mañana

Anexo 2

Durante el tiempo de experimentación en el uso del protocolo HTTP-S para sustentar comunicaciones seguras inter-platformas, también se probó con otro protocolo como IIOP.

La experimentación con el protocolo IIOP estuvo basada en el tutorial [1] que proporciona JADE para este propósito .

Descripción del escenario: se conectaron 2 equipos de cómputo formando una LAN, con las direcciones IP´s: 158.97.80.194 y 158.97.80.66 respectivamente.

Instrucción en una de los equipos de cómputo:

C:/jade/examples/src: java jade.Boot -gui -mtp jade.mtp.iiop.MessageTransportProtocol ping:examples.PingAgent.PingAgent

Cuando se ejecuta la instrucción anterior se genera el IOR, el cual, es utilizado para la conexión con el otro equipo de cómputo.

Instrucción en el otro equipo de cómputo que conforma la LAN.

C:/jade: java jade.Boot -gui -mtp jade.mtp.iiop.MessageTransportProtocol

Error que resulta cuando se realiza la prueba con el PingAgente.

((action

(agent-identifier:name da0@chap:1099/JADE

:addresses(sequence IOR: 000000000000001149444C3A464950412F4D54533A312E300

000000000000001000000000000006400010200000000E31

35382E39372E38302E31393400050900000019AFABCB00000

00002F9D906F6000000080000000000000000A0000000000

000100000001000000200000000000100010000000205010

001000100200001010900000000100010100))

(ACLMessage))

(MTS-error

(agent-identifier

[1] http://jade.tilab.com/doc/tutorials/JADEAdmin/JadePlatformTutorial.html

:name ping@Marcel:1099/JADE@chap:1099/JADE

:addresses

(sequence

IOR:000000000000001149444C3A464950412F4D54533A312E3

00000000000000001000000000000006E000102000000000D31

35382E39372E38302E3636000007A300000019AFABCB0000000

002F9D0665C000000800000000000000001400000000000002

000000010000002000000000000010001000000205010001000

100200001010900000001000101000000026000000020002))

(internal-error

"Agent not found:

getContainerID()

failed to find agent **ping@Marcel:1099/JADE@chap:1099/JADE**")))

Como se puede observar el error se encuentra en el nombre del agente de la plataforma remota, el cual, es concatenado con **@chap:1099/JADE**, la cual representa la dirección local.

Este error se hace presente tanto en el uso del protocolo IIOP como HTTP y HTTP-S. Lo cual indica que es un error de las clases internas de JADE, dónde se esta realizando las concatenaciones de los agentes remotos de una forma equivocada.