DESARROLLO DE SISTEMAS MULTI-AGENTE METODOLOGÍA GAIA IMPLEMENTACIÓN DE BOOKTRADING

Autores:

Juan de Dios García Bravo
(jdd316490@hotmail.com),
Richard Linares Aliaga
(richard.linares45@gmail.com),
José Jaime Paredes Wong
(josejaime@gmail.com)

Co-autores:

Marcos Rivas Peña
(mrivasp@unmsm.edu.pe)
Rolando Maguiña Pérez
(rolando_maguina@yahoo.com)
Armando Fermín Pérez
(aferminperez@yahoo.com)

Universidad Nacional Mayor de San Marcos
Facultad de Ingeniería de Sistemas e Informática
Lima – Perú

Resumen

Actualmente esta en constante investigación el desarrollo de metodologías que modelen sistemas multi-agente, las cuales presentan nuevos términos abstractos para modelar esta nueva técnica de desarrollo de Sistemas de Información, además agregan sus propios modelos para representar dichos sistemas. Éste proyecto consiste en la aplicación de la Metodología GAIA para modelar Sistemas Multi-agente presentando sus modelos y los conceptos que la metodología utiliza. Para lo cual se utilizará un ejemplo proporcionado por la Plataforma Jade (Java Agents DEvelopment Platform) el cual es el BookTrading (Venta de libros), y se demostrará el funcionamiento de dicho ejemplo, que presenta las características de un sistema multiagente.

Palabras Claves: Agentes, Sistemas Multi-agente, GAIA, Jade.

Abstract

Nowadays is in constantly investigation the development of methodologies for modeling Multi-agent Systems, which have new abstract concepts to model this new technique of developing Information Systems, furthermore adding their own models to represent such systems. This project consists in the application of The GAIA Methodology for modeling Multi-agent Systems representing their models and the concepts used by GAIA. We use an example of Jade Platform (Java Agents DEvelopment Platform) which is the BookTrading Example, and we show the work of such example, representing the features of Multi-agent System.

Key words: Agents, Multi-agent Systems, GAIA, Jade.

1. INTRODUCCIÓN

La construcción de un sistema flexible que tenga un comportamiento complejo y sofisticado al combinar componentes es una de las ventajas que obtenemos aplicando un enfoque orientado a agentes. Cuando estos componentes llamados agentes interactúan, con un grado de "inteligencia" y con capacidad de relacionarse "socialmente" (interactuar con otros agentes) da como resultado un sistema Multi-agente (SMA) [1].

Dentro de las ventajas de los SMA tenemos: eficiencia (debido al paralelismo que se logra al disponer de múltiples agentes), robustez y confiabilidad (gracias a la redundancia), escalabilidad (ya que puede crecer junto con la demanda) y reusabilidad [2].

La aplicación de la Ingeniería del Software al paradigma de agentes, conocida como Ingeniería del Software Orientada a Agente (AOSE – Agent Oriented Software Engineering) [4] ha generado en los últimos años numerosos trabajos relacionados. Hasta el momento, los trabajos existentes se han centrado en intentar buscar métodos de desarrollo para poder modelar sistemas reales complejos y con características claramente distribuida.

a. Agentes

Un agente básicamente se le puede definir como un componente informático que esta situado en algún ambiente, y que es capaz de hacer acciones autónomas en su entorno para cumplir con sus objetivos específicos, además mediante sus acciones es capaz de percibir y modificar su ambiente.

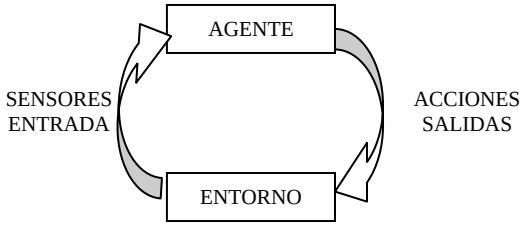

Fig. 1

Un agente posee las siguientes características:

- Autonomía: Un agente tiene su propio hilo de ejecución interna, como se mencionó, orientado a conseguir una tarea específica.
- Entorno: Todos los agentes realizan sus acciones sobre un entorno. Un entorno puede ser computacional (Website) o físico (una empresa), un agente puede percibir y modificar porciones de su entorno.
- Proactividad: Un agente debe estar orientado a conseguir sus objetivos en un ambiente dinámico e impredecible.

La autonomía propiedad de los agentes, pueden hacer que en un sistema, un agente pueda desempañarse sin necesidad de la intervención humana como es el caso de un Agente que vende libros (BookSellerAgent). Su autonomía esta expresada en que el mismo puede leer, modificar los libros que posea en su catalogo dando el servicio de ofrecer dichos libros a cualquier agente comprador (BookBuyerAgent), ambos negociarán en el entorno sin necesidad de intervención humana, el agente comprador podrá decidir con autonomía que libro comprar eligiendo el de menor precio. La proactividad se muestra en que el agente comprador de libros, busca el mejor libro (el que tenga más bajo precio) y ese elegirá para comprarlo. El entorno estará definido por el espacio donde se desarrollan estos dos agentes y por todos los catálogos de libros que están ofertados por los agentes vendedores.

b. Sistemas Multi-agente - MAS

MAS puede ser concebido en términos de una sociedad organizada de individuos (agentes) en el cual cada agente juega específicos roles e interactúa con otros agentes mediante protocolos determinados por los roles que envuelven a los agentes.

Desde el punto de vista de sociedad, los agentes pueden: Cooperar, Coordinar o Negociar entre ellos, para conseguir sus objetivos comunes, u objetivos compartidos en un sistema multiagente. También existe el caso en que los agentes no cooperen, es decir ellos mismos resuelvan determinadas tareas, en este caso serían identificados como especialistas en determinadas tareas.

En la figura 2 podemos observar como se comporta un sistema multiagente, una organización de agentes, donde cada agente se relaciona dentro de la organización con otros agentes, además se relacionan con otras organizaciones de agentes, para cumplir con los objetivos del sistema. Cada agente percibe una porción del entorno, según los permisos que tenga y los sensores que se hayan definido para él, y lo más importante según el rol que tenga asignado.

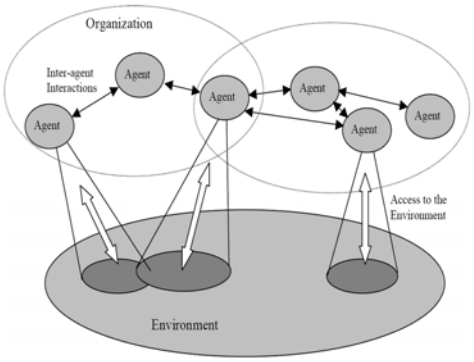

Fig. 2

Para el ejemplo proporcionado por Jade, el BookTrading, nos damos cuenta que la organización esta conformada por dos tipos de agentes el agente comprador y el agente vendedor, en este caso pueden existir muchos agentes vendedores y uno o más compradores.

La figura 2 también nos permite identificar los siguientes conceptos:

- El entorno en el cual MAS está inmerso.
- Los roles que son jugados por diferentes agentes en la organización.
- Interacción entre los roles.

Al observar la figura anterior vemos conceptos abstractos dentro de la organización como "El entorno", "Roles e interacciones", "Reglas de interacción", "Estructura Organizacional". Estos conceptos deben ser resueltos e identificados al

definir el sistema, pues nos dan diagramas preliminares para nuestro modelamiento.

2. LA METODOLOGÍA GAIA

La metodología GAIA [4] se centra en la idea de que la construcción de sistemas basados en agente es un proceso de diseño organizacional. Cabe resaltar el hecho de que la metodología que proponen es muy genérica, no indicando posibles mecanismos para poder llegar a un sistema directamente ejecutable.

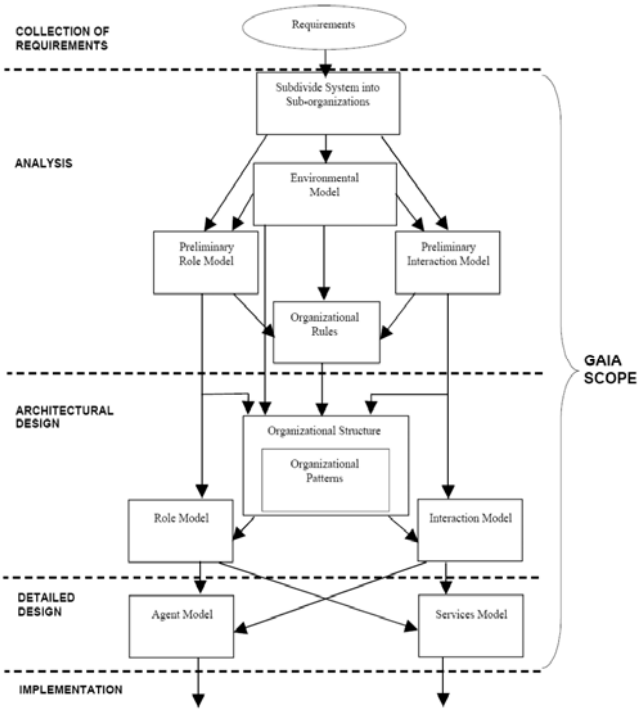

Fig. 3

GAIA tiene como alcance la fase de análisis y diseño si lo comparamos con el ciclo de vida del RUP, deja las otras fases a libertad del desarrollador de cómo capturar los requerimientos y la implementación. Como vemos en la figura 3 tiene varios modelos empezando por el modelo de la organización y sus sub-organizaciones, en la fase de análisis. Esta figura nos resume el trabajo de la Metodología GAIA.

2.1 FASE DE ANÁLISIS

Organizar la recolección de especificaciones y requerimientos para el sistema dentro de un modelo de entorno, modelos preliminares de roles, de interacción, y un grupo de reglas de organización, por cada sub-organización que componen todo el sistema. En resumen consiste en identificar las abstracciones de la organización principalmente.

La fase de análisis identifica específicamente lo siguiente:

- Los objetivos de la organización que constituye el sistema completo y su comportamiento global
- El modelo de entorno
- Modelo preliminar de Roles
- Modelo preliminar de Interacción
- Las reglas de la organización debe respetar orientadas a su comportamiento global.

a. La organización

La primera fase de GAIA en el análisis es concerniente con determinar, si la organización múltiple tiene que coexistir en el sistema y convertirse en una interacción autónoma, MAS. Es decir decidir si la organización puede ser modelada utilizando un sistema multi agente.

b. El modelo de entorno.

Se sugiere tratar el entorno en términos de recursos computacionales abstractos (variables). El modelo de entorno puede ser visto como todos los recursos con los que cuenta la organización, cuales pueden ser leídos y cambiados, y por que roles, en una definición general, identificando que roles deben estar en el sistema.

Para el ejemplo que se presenta posteriormente el entorno es definido: Por todos los libros y sus respectivos precios, en los catálogos de todos los agentes vendedores.

c. El modelo Preliminar de roles.

Los roles son la parte más importante del sistema, son los que definen el comportamiento especifico de la organización, cada rol es jugado dentro de la organización. Dichos roles tienes un conjunto de permisos y responsabilidades. Los roles tienen permisos de actuar sobre los recursos del sistema, y como los modifican. Además principalmente según como se defina el rol debe tener un conjunto de responsabilidades, que es como responde el rol a determinada acción. Las responsabilidades son divididas en 2 tipos: Liveness and Safety, estas responsabilidades aseguran que el rol siga en ejecución y no modifique los recursos haciendo perder la integridad del sistema.

- **Liveness Responsability:** Son las que permiten al rol seguir en ejecución, esta identificada por expresiones regulares compuestas por los protocolos y actividades que debe ejecutar dicho rol con otros roles. Identificando como es su ejecución de dicho rol.
- **Safety Responsability:** Son restricciones para el rol, que son cosas que no debe hacer, como por ejemplo en un stock no debe ser menor a cero. Son las restricciones o reglas encontradas en los otros modelos.
- **Protocolos:** Son las interacciones entre los distintos roles, básicamente es una actividad que necesita una interacción entre un rol y otro. Los protocolos son detallados en el modelo de interacción.
- **Actividades:** Desde el punto de vista de la programación orientada a objetos (POO) puede ser un método o un servicio, es decir un conjunto de instrucción que cambien la información de algún recurso. En este modelo las actividades se identifican pues va subrayadas.

d. El modelo preliminar de Interacción.

Este modelo captura las dependencias y relaciones entre varios roles en la organización MAS, en términos de la definición de un protocolo por cada tipo de interacción de roles.

Se identifica la entrada y la salida de esa interacción, quien inicia la comunicación y con quien se esta comunicando, y una breve descripción de la utilidad de dicho protocolo.

2.2 FASE DE DISEÑO

2.2.1 DISEÑO ARQUITECTURAL

La importancia de la fase de análisis es principalmente comprender el funcionamiento del sistema como un sistema multiagente, la fase de diseño es donde las decisiones deben ser tomadas sobre las características del MAS.

a. Completar los modelos de roles e interacción

Describe en detalle las características de todos los roles y estados envueltos en MAS que son realmente producidos. La estructura de la organización identifica, el diseño conocido de roles que tiene que interactuar con otros roles y cuales protocolos tienen que ser ejecutados.

2.2.2 DISEÑO DETALLADO

En esta parte del diseño se hace una definición exacta de los agentes, quienes son definidos por los roles. Un rol puede ser mapeado directamente en un agente, como también se da el caso de un agente puede realizar varios roles, quedando claro que el primer caso es el más usado y más sencillo de implementar.

a. Definición del Modelo de Agente.

Identifica que clases de agentes son definidos para jugar roles específicos y como varios instancias de cada clase debe ser instanciada en el actual sistema. Identifica los Agentes y sus relaciones dando una multiplicidad. Como sabemos pueden existir varios roles en el sistema pero no necesariamente un rol va a ser ejecutado por un agente, sino según como se necesite modelar vamos a tener que un agente encapsula varios roles, entonces este agente sería un tipo de agente.

b. El modelo de servicio.

Identifica los servicios asociados con cada clase de agente o equivalentemente con cada rol que son jugados por las clases de agentes. Asi de este modo, el modelo de servicio, en el caso de asignación estática de roles a clases de agentes, y en el caso donde agentes pueden dinámicamente asumir roles. Comparando con la POO sería el diagrama de secuencia que diagrama los métodos en los objetos, lo mismo se cumple para este diagrama. Identifica los servicios (métodos) indicando sus entradas, salidas, pre y post condiciones.

c. El modelo de comunicación:

Identifica las relaciones entre los agentes, que agentes se comunican indicando principalmente donde se encontraran los cuellos de botella para el sistema. Permitiendo identificar los puntos débiles del mismo en lo que respecta a la comunicación. Este modelo se basa en el modelo de interacción. Para saber quien se comunica con quién y con que objetivo existe dicha comunicación, pudiendo reducir el modelo en caso no sea necesaria.

Con este modelo termina el Alcance de GAIA, la parte de Análisis y Diseño. De aquí en adelante el desarrollador es libre de elegir como implementar el sistema diseñado. Pudiendo incluso implementar el sistema con objetos, para lo cual sería inútil haber modelado con GAIA. Para implementar o modelar en bajo nivel, se puede utilizar AUML (UML para agentes), u otras herramientas libres.

3. CASO DE ESTUDIO: BOOKTRADING

Para presentar el uso de la Metodología GAIA en combinación con la plataforma JADE, usaremos el ejemplo que nos proporciona JADE que es el BookTrading. El cual consiste en dos agentes un agente comprador (BookBuyerAgent) y un agente vendedor (BookSellerAgent) los cuales representaremos con la metodología GAIA. La implementación se realizará con la plataforma JADE, para lo cual se necesita una interface de usuario para adjuntar los libros que se ofertarán por cada agente vendedor. Cada agente comprador al crearse se enviará como parámetro el libro a solicitar. El agente comprador solicitará a todos los agentes vendedores si tienen el libro que esta buscando, eligiendo el que tiene mejor precio. Los agentes vendedores tienen un catalogo de libros los cuales tienen un precio, los cuáles serán ofrecidos a los agentes compradores.

3.1 FASE DE ANÁLISIS

Empecemos definiendo los roles de los dos agentes del caso de estudio:

Primero llenamos en una plantilla el esquema de rol del BookBuyer (Comprador de libros), indicando una descripción de lo que hace y también indicando los protocolos, actividades, permisos y responsabilidades tipo liveness (ciclo de vida) y tipo safety. Lo mismo hacemos para el caso del BookSeller (Vendedor de Libros).

Un rol necesita un conjunto de *Permisos* que identifique aquellos recursos que puede acceder para poder cumplir con sus responsabilidades. Estos recursos son generalmente fuentes de información en las que puede leer, hacer modificaciones y enriquecer bajo ciertas circunstancias.

Esquema Rol: BookBuyer		
Descripción: Envía peticiones de libros que necesita comprar, y decide que libro comprar, seleccionando el que posea un mejor precio.		
Protocolos y Actividades: RequestPerformer, ReceiveProposal, ReceiveRefusals, SendPurchaseOrder, ReceivePurchaseOrder, SelectBestBook		
Permisos:		
Lectura	**supplied** priceBooks	// contacto con la información de los catálogos
Escritura	**change** bestPrice, bestBook	// Cambia los atributos de mejor precio de libro y selecciona el mejor libro
Responsabilidades:		
Liveness:	**BookBuyer =** **(**RequestPerformer.ReceiveProposal.SelectBestBook SendPurchaseOrder.ReceivePurchaseOrder)^{w}	
Safety:	**true**	

Recordar que el tipo de responsabilidad liveness (ciclo de vida) define la trayectoria potencial de ejecución través de las actividades e interacciones asociadas al rol.

Esquema Rol: BookSeller		
Descripción: Recibe petición de libro, procede a buscar en su catalogo de libros, envía el resultado de la búsqueda con el precio de dicho libro.		
Protocolos y Actividades: OfferRequestServer, PurchaseOderServer		
Permisos:		
Lectura		// contacto con la información de los catálogos
Escritura	**change** catalogue	// Cambia los atributos de mejor precio de libro y selecciona el mejor libro
Responsabilidades:		
Liveness:	**BookSeller = (OfferRequestServer.PurchaseOrderServer)^{w}**	
Safety:	**Price_of_books > 0**	

Luego seguimos con las plantillas de protocolo indicando el nombre, el iniciador (initiator), el contestador (partner), los datos de entrada y salida (input y output) y una descripción.

Protocolos

Protocol Name: RequestPerformer		
Initiator: BookBuyer	**Partner:** BookSeller	**Input:** Book for purchase
Description: Cuando se requiere buscar un libro en los diferentes vendedores de libros se llama al protocolo RequestPerformer		**Output:** Price of book

Protocol Name: ReceiveProposal		
Initiator: BookSeller	**Partner:** BookBuyer	**Input:** Book for purchase
Description: Según como haya sido la búsqueda del libro pedido el agente vendedor devuelve la propuesta que tiene de dicho libro (su precio y el nombre del libro).		**Output:** Book found

Protocol Name: ReceiveRefusals		
Initiator: BookSeller	**Partner:** BookBuyer	**Input:** Book for purchase
Description: Este protocolo se ejecuta cuando el libro buscado no ha sido encontrado.		**Output:** Null

Hay que recordar que los protocolos especifican patrones para la interacción entre los agentes, los cuales son formalmente definidos de manera que constituyen algo más que una simple secuencia de pasos.

Protocol Name: SendPurchaseOrder		
Initiator: BookBuyer	**Partner:** BookSeller	**Input:** Book for purchase
Description: Luego de que el comprador selecciona el libro a comprar, ejecuta el protocolo de SendPurchaseOrder, para que se ejecute la compra.		**Output:** Book to sell

Protocol Name: ReceivePurchaseOrder		
Initiator: BookSeller	**Partner:** BookBuyer	**Input:** Book to sell
Description: El vendedor recibe la orden de compra por parte del comprador.		**Output:** Book for selling

Protocol Name: PurchaseOderServer		
Initiator: BookSeller	**Partner:** BookBuyer	**Input:** Book for selling
Description: Una vez que se recibe la orden de compra, se debe ejecutar la venta de dicho libro con el protocolo PurchaseOrderServer.		**Output:** Book sold

3.2 FASE DE DISEÑO:

Ya en la fase de diseño se pasa a la fase donde eventualmente se identifica el modelo del agente, estos es identificar que clases de agente hay para especificar los roles específicos que jugarán, y cuántas instancias de cada clase tienen que ser creadas en el sistema.

Modelo de Agentes

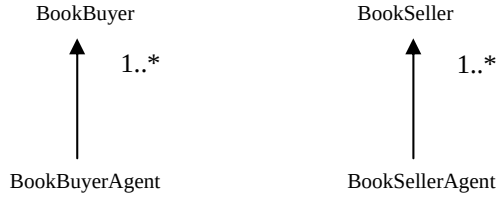

Fig. 4

Luego de definir el modelo de agentes pasamos a definir el modelo de servicio el cual identifica los servicios asociados a cada rol, y especifica las propiedades esenciales de estos servicios.

Modelo de Servicio

Por cada servicio es necesario conocer las entradas, salidas, pre-condiciones y post-condiciones, estas dos últimas representan restricciones en el servicio así cada rol identificado en la etapa de análisis esta asociado con al menos un servicio.
En la siguiente cuadro (Fig. 5) vemos los 3 servicios hemos definido el OfferRequestServer (Ofrecer Pedido), el PurchaseOrderServer (Comprar orden) y el RequestPerformer (Pedido Ejecutante).

Servicio	Entradas	Salidas	Pre-condiciones	Post-condiciones
OfferRequestServer	Nombre del Libro	Precio del Libro	Numero de libros en catalogo > 0	true
PurchaseOrderServer	Titulo del Libro	Libro vendido	Libro ha sido encontrado	Catalogo actualizado
RequestPerformer	Titulo del libro	Libro y precio del mejor libro ofertado	Se ingresó el nombre del libro a comprar	- Libro comprado - No se encontró ningún libro.

Fig. 5

Por ejemplo para el caso de OfferRequestServer se entiende que la entrada es el nombre del libro solicitado y la salida el precio de tal libro teniendo en cuenta como pre-condición el número de libros en el catálogo (sea mayor a cero) y como post-condición un valor de true (no hay post-condiciones).

Modelo de Comunicación

El modelo de comunicación solo grafica que agentes se comunican entre sí sin especificar qué o cuándo se envía. Para nuestro caso de estudio solo tenemos la relación entre el agente comprador y el agente vendedor.

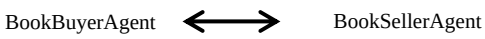

Fig. 6

Con este último modelo finaliza el alcance de la metodología.

IMPLEMENTACIÓN EN JADE:

Se utilizan 2 clases tipo Agente el BookBuyerAgent y el BookSellerAgent.

Vemos el diagrama (Fig. 7) de secuencia para su implementación:

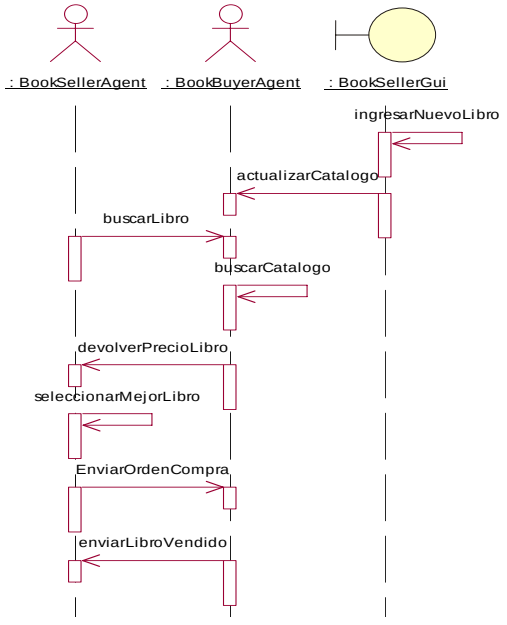

Fig. 7

4. TRABAJO A FUTURO

El desarrollo de sistemas multi-agente viene atrayendo cada vez más desarrolladores para modelar sus sistemas, y una de las metodologías que lo hace con una gran facilidad es GAIA. Para lo cual GAIA no cuenta con una herramienta de modelamiento propia, se plantea llevar a cabo una herramienta de modelamiento para sistemas multi-agente que soporte GAIA. Además GAIA no cubre todas las fases del ciclo software, pudiendo ser completadas con en dicho software.

5. CONCLUSIÓN

- Los sistemas multiagente se basan en la organización humana, mostrando su interacción entre Agentes.
- Los sistemas multiagente ofrecen flexibilidad e independencia de los componentes de software (Agentes).
- GAIA es una de las metodologías más flexibles, pues permite elegir la herramienta de modelamiento, pudiendo combinarse con otras herramientas para el modelado a bajo nivel.
- Las metodologías están en continúo desarrollo, pudiendo construir una herramienta para alguna metodología para sistemas multiagente.
- JADE es la plataforma con mayores avances en lo que respecta a la implementación de Sistemas Multiagentes, definiendo ontologías, basado totalmente sobre los estándares de FIPA.

6. REFERENCIA BIBLIOGRÁFICA:

[1] **Wooldridge M. "Intelligent Agents". En "Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence", The MIT Press. 1999. Pág. 27-78.**

[2] **Wooldridge M. "An introduction to MultiAgent Systems", England. Wiley, 2002. 368p.**

[3] **Green, Shaw, Leon Hurst, and Brenda Nangle, "Software agents: A review", Tech. Rep. no. TCD-CS-1997-06, Intelligent Agents Group (IAG), 1997.**

[4] **Wooldridge M., Jennings N., and kinny D. "The Gaia Methodology for Agent-Oriented Analysis and Design", 2000. P. 285-312.**

[5] **Zambonelli F, Jennings N., and Wooldridge M. "Developing Multiagent Systems: The Gaia Methodology", ACM, Vol. 12 Nº , July 2003, Pág. 317-370**

[6] **Moraitis P., Petraki E., and Spanoudakis N. "Engineering JADE agents with the Gaia methodology". En R. Kowalszyk, J. Miller, H. Tianfield, and R. Unland, editors, Agent Technologies, Infrastructures, Tools, and Applications for e-Services, volume 2592 of Lecture Notes in Computer Science, Pág. 77--91. Springer-Verlag, 2003.**

[7] **Caire Giovanni. "Jade Programming for Beginners". TILAB. 2003. 22p.**

[8] **Bellifemine F., Caire G., Trucco T. "Jade Programmer's Guide". TILAB. 2005. 52p.**

[9] **Bellifemine F., Caire G., Trucco T. "Jade Administrator's Guide". TILAB. 2005. 35p.**