resumen

*El presente trabajo fue desarrollado con el objetivo de plantear un modelo de seguridad adaptable a la plataforma para el desarrollo de sistemas multiagentes (SMA) BESA. Además, con el fin de implementarlo y brindar al diseñador herramientas para efectuar controles de seguridad en sus aplicaciones. Para conseguir dicho objetivo, se analizaron ocho principios de seguridad ya existentes para integrarlos con la teoría básica de los SMA, de tal forma que el modelo propuesto se pudiera desarrollar para la plataforma BESA.*

*Siguiendo la línea de los principios de seguridad, BESA.Security incluye: controles para autenticación basada en certificados digitales, autorización fundamentada en restricciones de interacción, confidencialidad apoyada en la infraestructura de clave pública, auditabilidad basada en registros de actividad, no repudio con base en firmas digitales, integridad basada en el ordenamiento de mensajes y disponibilidad y confiabilidad apoyadas en replicación de agentes. Los aportes más importantes del trabajo, incluyen una definición formal de los principios de seguridad para los SMA como base teórica en este ámbito y una plataforma para el desarrollo de aplicaciones basadas en SMA seguras*

# BESA, una plataforma para el desarrollo de Sistemas Multiagentes Seguros

**Guillermo Fonseca • María Camila González**

introducción

**Los sistemas** multiagentes (SMA) representan una manera diferente de resolver problemas de concurrencia y paralelismo, gracias a que permiten hacerlo de una forma más natural a la comúnmente usada. Este enfoque es especialmente útil para aplicaciones complejas que son requeridas por las organi-

zaciones actualmente. Sin embargo, se debe aclarar que aun no existen herramientas robustas para facilitar la creación de SMA.

Por otra parte, BESA es una plataforma para el desarrollo de sistemas multiagentes que fue concebida en el ámbito académico, por ello, no poseía opciones para aplicar controles de seguridad a los SMA creados. Considerando la creciente necesidad de la industria, y la carencia de herramientas para satisfacerla, se formuló el proyecto de implementación de un modelo de seguridad para BESA, con el fin de convertirla en una herramienta suficientemente robusta para la creación de aplicaciones comerciales basadas en SMA. Refiérase a la Tabla 1 para ver el pobre cubrimiento de las herramientas actuales respecto de ocho aspectos de seguridad.

| | Principio de Seguridad | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Plataforma** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| Aglets Software Development Kit | | X | | | | | | |
| Ajanta | X | | X | | | | | |
| Tryllian's Agent Development Kit | X | | X | | | | | |
| FIPA-OS | | | X | | | | | |
| Grasshopper | X | | X | | | X | | |
| JADE | X | X | X | | | X | | |
| JACK Intelligent Agent | | | | | | | | |
| Zeus | X | | X | | | | | |
| BESA | X | X | X | X | X | X | X | X |

Tabla 1. Herramientas para SMA y Seguridad

Los números corresponden a lo siguientes: 1) Autenticación. 2) Autorización. 3) Confidencialidad. 4) Auditabilidad. 5) No Repudio. 6) Integridad. 7) Disponibilidad. 8) Confiabilidad.

A lo largo de este artículo se presentan los resultados del trabajo examinando los conceptos básicos, el modelo de seguridad propuesto, la implementación y los trabajos futuros.

## Conceptos

*A. Sistema Multiagente SMA [2] [3] [4] [10]*

En los SMA, un agente es una entidad virtual con un grado de autonomía definido que es capaz de actuar dentro de un ambiente, que puede comunicarse directamente con otros agentes, que posee recursos propios, que percibe un ambiente y que tiene la habilidad de ofrecer servicios. El comportamiento de un agente tiende a la satisfacción de objetivos, teniendo en cuenta los recursos y habilidades disponibles dependiendo de su percepción y la comunicación con otros agentes.

Teniendo en cuenta lo anterior, un SMA se puede definir como un conjunto de agentes ubicados dentro de un ambiente, es decir, dentro de un espacio en el que éstos se encuentran acompañados de un conjunto de objetos que pueden asociarse a una posición en el ambiente.

Un SMA también contiene un conjunto de relaciones que unen objetos y agentes entre sí, y un conjunto de operaciones que hacen posible que los agentes perciban, produzcan, consuman, transformen y manipulen objetos del ambiente. Existe también un conjunto de operadores para la aplicación de las operaciones de modificación y para la reacción del ambiente frente a la misma.

*B. Seguridad*

**Una teoría de seguridad informática se enfoca en siete principios de seguridad. Con el fin de cubrir los aspectos de los SMA, se realizó un estudio de ésta teoría y se decidió adicionar un octavo principio de seguridad que se relaciona con la naturaleza distribuida de los SMA.** Éste es el principio de confiabilidad que se encuentra explicado más adelante.

## Principios de Seguridad [5]

1) *Autenticación:* es la verificación de si un actor en un sistema es realmente quien dice ser.

2) *Autorización:* es el acto de una autoridad que permite o no a un actor realizar una acción.

3) *Confidencialidad:* es asegurar que sólo los actores autorizados pueden acceder a información que no es pública.

4) *Auditabilidad:* es poder hacer seguimiento a las actividades realizadas en un sistema.

5) *No Repudio:* es asegurar que un actor realizó una acción de manera que no lo pueda negar.

6) *Disponibilidad:* es mantener los servicios más utilizados siempre listos para responder las peticiones de los usuarios en todo momento.

7) *Integridad:* es la característica de un sistema que indica si se encuentra en un estado coherente.

8) *Confiabilidad:* consiste en la capacidad de un sistema de recuperarse ante un fallo en un proceso.

## Modelos de Seguridad [5] [6]

**Por otra parte, existen los modelos de seguridad. Los sistemas en general implementan alguno de los modelos para priorizar sus requerimientos. Estos modelos se encuentran a continuación debido a que son la base del modelo propuesto.**

*Modelo Bell-LaPadula:* considera como el aspecto más importante a proteger: la confidencialidad. Para esto utiliza niveles de clasificación de seguridad para sujetos e información utilizando reglas.

*Modelo Biba:* considera la integridad como lo más importante. Para esto se basa en niveles de integridad definidos para cada usuario y aplica reglas que restringen la lectura o escritura.

*Modelo Clark Wilson:* se enfoca en la integridad, como el de Biba y controla que los usuarios autenticados no modifiquen indebidamente la información y la

consistencia interna y externa del sistema. Para esto plantea la división de tareas garantizando que los usuarios no realicen operaciones indebidas.

*Modelo de Flujo de Información:* se fundamenta en la información y el análisis de su movimiento para determinar si el sistema es seguro o no a partir de reglas específicas. Se concentra en los principios de confidencialidad, integridad y autorización.

*Modelo de No Interferencia:* Este modelo se basa en no interferir con el trabajo de otra persona, se pueden realizar conjeturas acerca del sistema con el solo hecho de ver los movimientos de un usuario. Por esto, se enfoca en asegurar que las acciones realizadas en un nivel de seguridad no afecten otros niveles de seguridad haciendo énfasis en la confidencialidad, la integridad y la autorización.

*Modelo de Brewer y Nash:* Este modelo se conoce como la pared China debido a que se basa en crear una pared lógica entre un usuario y la información que no debe acceder. Dos usuarios pueden estar en el mismo nivel de seguridad pero eso no quiere decir que deban leer la misma información. Se concentra en la confidencialidad y la autorización.

*Modelo de Graham Denning:* Este modelo se basa en los modelos de Bell_LaPadula y Biba y hace énfasis en la autorización. Ofrece una solución basada en una matriz de objetos y derechos de acceso. Por medio de la matriz se puede relacionar a los sujetos con los objetos y las operaciones que éstos pueden realizar.

Modelo de Harrison-Ruzzo-Ullman: Este modelo se basa en los modelos de Bell_LaPadula y Biba, pero añade la especificación de la manera como pueden modificarse los derechos de acceso de un usuario en el tiempo. Se enfoca en la autorización.

*C. Arquitectura ASMA [2] [7] [8]*

La Arquitectura para Sistemas Multiagentes ASMA surgió de una iniciativa del Grupo de Investigación SIDRe del Departamento de Ingeniería de Sistemas de la Pontificia Universidad Javeriana, para trabajar en el desarrollo de herramientas computacionales y metodológicas relacionadas con los SMA. Esta arquitectura está definida en tres niveles: nivel de agente, nivel social y nivel de sistema en los cuales existen componentes que se interrelacionan.

El nivel de agente especifica los componentes internos de cada agente como muestra la Figura 1. Los componentes son: un canal, un conjunto de comportamientos y un estado. El canal se encarga de recibir los eventos dirigidos al agente por un único punto de entrada. Los comportamientos son procesos paralelos que trabajan juntos para lograr las metas del agente. El estado es una memoria compartida accedida por los comportamientos del agente usando sincronización de exclusión mutua.

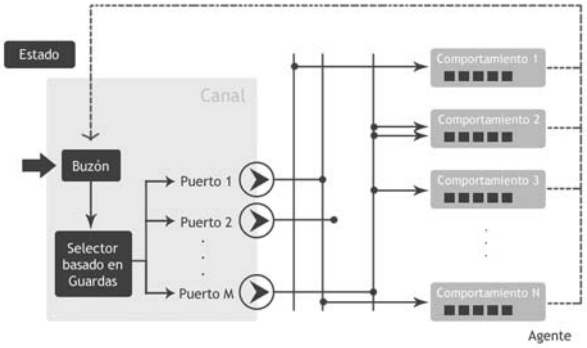


Figura 1. Nivel de agente de ASMA

El nivel social presentado en la Figura 2 surge del objetivo de permitir la creación y manipulación de sociedades jerárquicas de diferentes niveles tales como el nivel micro-social, el nivel de grupos y el nivel de sociedades globales.

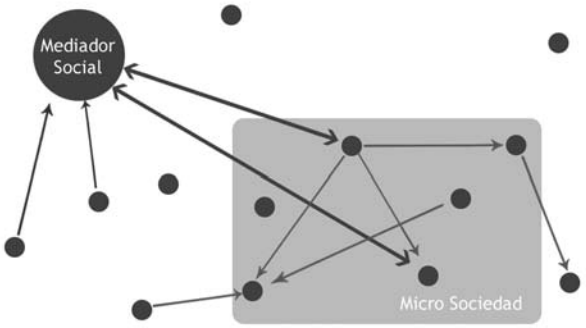


Figura 2. Nivel social de ASMA

Un contenedor BESA es un espacio de ejecución en el cual habitan los agentes. El contenedor es el responsable de manejar su ciclo de vida y de facilitar la comunicación por medio de un administrador local. El nivel de sistema es un sistema distribuido, como muestra la Figura 3, compuesto por mínimo un contenedor

BESA. Estos contenedores se pueden ejecutar en diferentes máquinas físicas o virtuales

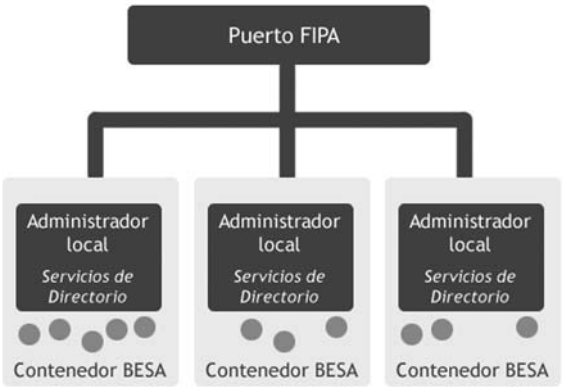


Figura 3. Nivel de Sistema de ASMA.

*D. Plataforma para el Desarrollo de SMA BESA*

**BESA es un herramienta que facilita la creación de aplicaciones orientadas a SMA ya que brinda funcionalidades implementadas y probadas listas para ser aplicadas por el programador.**

El término BESA surgió de la unión de sus tres características principales en inglés: Behavior-Oriented (Orientado a Comportamientos del agente), Event-driven (con una técnica de control de eventos que implementa un mecanismo de selección) y Social-based (basado en mecanismos sociales) Agent *Framework* (plataforma de agentes).

## Arquitectura ASMA con seguridad

La arquitectura ASMA mencionada dentro de los conceptos fue ajustada con el fin de cumplir con los requerimientos de seguridad identificados. Por lo anterior, surgió un nuevo modelo de ASMA que contiene los mismos tres niveles descritos. Para cada uno se identificaron los puntos críticos de seguridad y finalmente se planteó un componente lógico de seguridad (Figura 4) que se debe comunicar con cada nivel de la arquitectura.

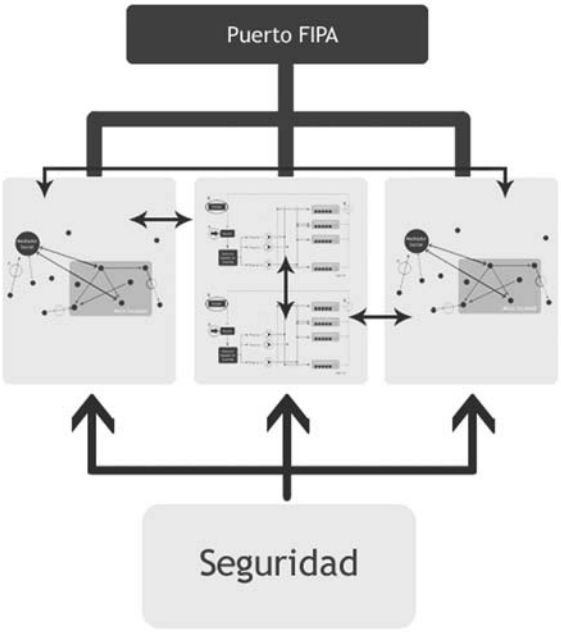


Figura 4. Nuevo modelo ASMA

## Modelo de seguridad en BESA

Dada la necesidad de considerar todos los aspectos de seguridad en relación con los SMA, se utilizó el enfoque de principios de seguridad orientado a resolver los aspectos de seguridad que se pueden presentar en los SMA. Luego de examinar los ocho modelos de seguridad se encontró necesario proponer un nuevo modelo basado en técnicas tomadas de los ocho anteriores. Sin embargo, los modelos no cubren todos los principios, por lo cual se plantearon unas estrategias adicionales. La Tabla2 muestra el cubrimiento de los modelos en relación con los principios.

Los principios de seguridad corresponden a los siguientes: 1) Confidencialidad; 2) Integridad; 3) Disponibilidad; 4) Autenticación; 5) Autorización; 6) Auditabilidad; 7) No Repudio.

A continuación se presenta el modelo propuesto desde la perspectiva de cada principio de seguridad. Para cada uno se realizó un análisis de vulnerabilidades en BESA 2.1 y luego se evaluaron diferentes opciones para solucionarlas.

| | Principio de Seguridad | | | | | | |
|---|---|---|---|---|---|---|---|
| **Modelo** | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Bell-LaPadula | X | | | X | X | | |
| Biba | | X | | X | X | | |
| Flujo de Información | X | X | | X | X | | |
| Clark Willson | | X | | X | X | | |
| No Interferencia | X | X | | X | X | | |
| Bewer y Nash | X | | | X | X | | |
| Graham Denning | X | X | | X | X | | |
| Harrison-Ruzzo-Ullman | X | X | | X | X | | |

Tabla 2. Nivel de sistema de ASMA

*D. Autenticación*

La autenticación es considerada implícitamente en todos los modelos de seguridad debido a que sin ésta no se pueden garantizar otros principios. Si se autoriza a un actor, pero éste puede ser suplantado, no sirvió el control, al igual que si se garantiza el no repudio de las acciones de un actor.

Ahora bien, en BESA 2.1 no existía un mecanismo de autenticación. Se asignaba una clave al agente que se utilizaba para hacer controles de autorización, como se mostrará más adelante. Debido a esto, podía haber suplantación de contenedores y, por tanto, de agentes en el sistema. Este era un problema de seguridad grande debido a que un contenedor con agentes no válidos podía efectuar operaciones riesgosas.

Para solucionar esto se implementó un mecanismo de autenticación basado en la infraestructura de clave pública asignando un par de llaves privada y pública a cada contenedor, o sociedad. Al enviar un mensaje con autenticación se utiliza la firma digital de éste de tal forma que cualquier actor que posee la llave pública del contenedor emisor puede corroborar que éste fue quien verdaderamente lo envió. En el esquema mencionado todavía había un problema de seguridad debido a que si se generaba un par de llaves y su certificado correspondiente, cualquier contenedor externo podía autenticarse en el sistema distribuyendo su certificado

falso. La solución a este problema es tener una autoridad certificadora, confiable que firma todos los certificados del sistema, asegurando que éstos son válidos.

Para que este mecanismo funcione, todas las máquinas del sistema deben tener los certificados digitales de los contenedores válidos y su propia clave privada. Con el fin de facilitarle la labor al programador o administrador del sistema, existe una aplicación que genera y reparte los certificados digitales cifrados a cada máquina requerida. Esta aplicación se debe ejecutar una vez para armar la infraestructura de certificados y luego se puede iniciar el SMA utilizando autenticación, y como se presenta más adelante, confidencialidad también.

**Con los controles implementados se evita la suplantación de contenedores y sociedades de manera transparente para el programador debido a que éste sólo debe escoger la opción de autenticar y utilizar la aplicación de distribución de certificados.**

*E. Autorización*

Como se mencionó anteriormente, en BESA, a los agentes se les asignaba una clave que se utilizaba para realizar dos controles de autorización: Destruir el agente y mover el agente a otro contenedor. Al examinar las vulnerabilidades de BESA se identificaron problemas de seguridad como la carencia de autorización en la recepción de eventos, en la adición de contenedores y en el movimiento de agentes. Por lo anterior, se podían enviar eventos inválidos, adicionar contenedores sin control y mover agentes sin tener precaución.

Luego de estudiar las soluciones que proponen los modelos de Graham Dening y Harrison-Ruzzo-Ullman en cuanto a autorización se definió utilizar la matriz de permisos que plantean estos modelos. Para realizar el control de autorización en BESA se definieron tres tipos de permisos según la interacción que puede existir entre contenedores del sistema: 1) Envío-Recepción de Eventos 2) Adición de contenedores al sistema 3) Movimiento de agentes. Adicionalmente se implementó una matriz de permisos para cada contenedor en la cual se incluyen y remueven permisos de los tres tipos.

**La ventaja de este mecanismo es que los permisos se pueden modificar en tiempo de ejecución permitiendo el dinamismo característico de un SMA. En un trabajo futuro se puede ampliar este esquema agregando más tipos de permisos.**

*F. Confidencialidad*

La confidencialidad es un principio de seguridad importante en las empresas que tienen información sensible. En BESA no existían controles de confidencialidad. En el paso de mensajes por la red se podía capturar todo tipo de información con sólo poner a escuchar una tarjeta de red en modo promiscuo. Se podía de obtener la clave del agente cuando éste se movía de un contenedor a otro.

Utilizando la infraestructura de llaves y certificados de autenticación, el problema se resolvió cifrando los mensajes entre contenedores con la llave pública del destinatario y para el caso de mensajes multicast existe un certificado del sistema que todos conocen junto con la llave privada correspondiente.

*G. Auditabilidad*

En BESA era necesario implementar un mecanismo que permitiera hacer seguimiento a las acciones que se llevaban a cabo en un contenedor. Para eso se implementó el principio un registro acciones como la creación de contenedores, envío de eventos, entre otras, facilitando el seguimiento detallado de lo que sucede en un contenedor. Esto se hizo utilizando archivos XML.

La auditabilidad en BESA es configurable, debido a que se puede indicar cuántos archivos se requiere crear y el tamaño de ellos además de seleccionar los tipo de mensajes a registrar. Este mecanismo de auditabilidad no es el más seguro considerando que una base de datos proporcionaría mayor control. Sin embargo, los archivos XML brindan un balance entre rapidez y velocidad de almacenamiento.

*H. No Repudio*

Este principio está orientado al no repudio de acciones generadas en un contenedor y registradas en el principio de auditabilidad. Esto se implementó con la

firma digital del mensaje almacenado en auditabilidad. El proceso es el siguiente: Cuando se genera un mensaje para registrar, se calcula el código hash y se cifra utilizando la llave privada del contenedor. Posteriormente se almacena el hash calculado y firmado en el log como una acción que lleva un número que relaciona el mensaje original con el hash firmado. De esta manera se garantiza que el contenedor fue el único que pudo registrar esa acción. Sin embargo, cabe resaltar que el espacio en disco se puede duplicar si cada acción tiene su mensaje de no repudio. Para tener control sobre esto se puede configurar el tipo de mensajes que requieren no repudio. Con lo anterior se permite hacer un seguimiento de acciones en caso de que se presente un incidente. Sin embargo, implica espacio en disco y carga de procesamiento.

*I. Integridad*

La integridad en BESA ya estaba cubierta en cierta parte por el protocolo de Ethernet, ya que éste cuenta con el algoritmo CRC 32 que garantiza la integridad de la mayoría de mensajes unicast. Por otra parte, los mensajes multicast se enviaban a todos los contenedores sin ningún control de recepción. Esto representaba un problema debido a que el sistema podía quedar inconsistente porque los contenedores podían poseer información diferente en caso de la pérdida de un mensaje multicast. Para solucionar este problema, se implementó un mecanismo de números de secuencia en los mensajes multicast que se almacenan en una lista realizando control de las secuencias de todos los contenedores. De esta manera si un mensaje multicast se pierde, los números de secuencia no serán consecutivos para ese contenedor y se expulsará del sistema. Como trabajo futuro se puede implementar un mecanismo de reenvío de mensajes perdidos.

*J. Disponibilidad y Confiablidad*

**La disponibilidad y la confiabilidad son principios indispensables en cualquier sistema distribuido ya que de esto dependen los servicios y la recuperación del sistema ante fallas de procesos críticos.**

Estos dos principios están muy relacionados en BESA debido a que fueron implementados por medio de la replicación. Este mecanismo permite realizar copias idénticas de agentes y distribuirlas a distintos contenedores BESA.

Para el caso de la disponibilidad, el usuario define los agentes que prestan los servicios más utilizados a ser replicados. El usuario tiene la responsabilidad de utilizar agentes copia para balancear las cargas.

**Para la confiabilidad, el usuario define los agentes críticos que requiere replicar y ante una falla de uno de estos, una de sus réplicas toma su lugar y continúa prestando los servicios. De esta manera se logra superar la falla del agente.**

La replicación está implementada de forma pasiva [9], es decir, hay un agente que actualiza todas sus réplicas cada determinado tiempo especificado por el programador. Las réplicas se encuentran en contenedores remotos para evitar su caída en caso de presentarse una falla en una máquina. Cabe aclarar que para sobrevivir a n fallas de un agente es necesario crear n+1 réplicas del mismo.

## Pruebas

Se realizaron pruebas unitarias por cada principio de seguridad. Luego se analizó la relación entre los principios para plantear y realizar las pruebas de integración. Finalmente se hicieron pruebas de sistema y con los resultados obtenidos se corrigieron los errores detectados.

## Conclusiones

Para los SMA es necesario tener en cuenta que la teoría general de seguridad debe ser ajustada para solucionar problemas que surgen por su naturaleza distribuida. El enfoque de principios de seguridad asegura un cubrimiento completo.

El análisis de los modelos de seguridad fue el punto de partida para la investigación de seguridad en los SMA y representó el foco del trabajo, de la medición de los resultados y del carácter innovador de la investigación e implementación.

Ahora, la plataforma BESA es suficientemente robusta para el desarrollo de aplicaciones comerciales basadas en SMA, especialmente para facilitar el manejo de la complejidad y el paralelismo de éstas. Sin embargo, cabe aclarar que la seguridad

siempre tiene un costo que en este caso es carga de procesamiento y espacio en disco.

BESA Security es un aporte novedoso en el ámbito de los SMA y puede ser extensible a los sistemas distribuidos en general. Adicionalmente se considera uno de los primeros pasos en el desarrollo de aplicaciones SMA seguras.

## Referencias


[1] Nguyen G, Dang T.T, Hluchy L, Laclavik M, Balogh Z, Budinska I. Agent Platform Evaluation and Comparison. Junio de 2002.

[2] BESA: Arquitectura para la construcción de Sistemas Multi-Agentes. Enrique González Guerrero, Jamir Antonio Avila Mojica, César Julio Bustacara Medina. Bogotá : s.n.

[3] Wooldridge, Michael. An Introduction to MultiAgent Systems. Baffins Lane : Chichester : John Wiley, 2002.

[4] Bordini, Rafael, y otros. Multi-Agent Programming (Languages, Platforms and Applications). New York : Springer, 2005.

[5] Valoración de la evidencia digital: Análisis y propuesta en el contexto de la administración de justicia en Colombia. Cano, Jeimy y Rueda, Andrea. Bogotá : s.n., 2006.

[6] El Modelo de Clark Wilson (CW). Cesar Beltrán, Sergio García, Mauricio Gómez, Andrea Herrera. Bogotá : s.n., 2003.

[7] E, González y C. Bustacara. Desarrollo de Aplicaciones basadas en sistemas multiagente. s.l. : Editorial Pontificia Universidad Javeriana, 2007. ISBN 978-958-683-871-4.

[8] González, Enrique y Bismark, Adith. Agentes Racionales. Bogotá : s.n.

[9] Coulouris, George F. Sistemas Distribuidos : Conceptos y Diseño. Madrid, España : Addison Wesley, 2001.

[10] Murch, Richard y Johnson, Tony. Intelligent *Software* Agents. Upper Saddle River, New Jersey : Prentice Hall PTR, 1999.



***Guillermo Fonseca.*** *Estudiante de décimo semestre de Ingeniería de Sistemas de la Pontificia Universidad Javeriana, con énfasis en seguridad informática.*

***Maria Camila González.*** *Estudiante de décimo semestre de Ingeniería de Sistemas de la Pontificia Universidad Javeriana, con énfasis en seguridad informática.*