UNIVERSIDAD CENTROCCIDENTAL "LISANDRO ALVARADO"

APLICACIONES DE LOS SISTEMAS MULTIAGENTE

Prof. Niriaska Perozo G

"La manera tan impresionante en que los enjambres de hormigas, abejas, grillos, langostas, insectos en general y los grupos de animales, peces, pájaros, elefantes y lobos, entre otros, actúan formando con su comportamiento individual una especie de cerebro colectivo, una inteligencia de grupo donde emergen propiedades como la coordinación, la cooperación, la integración y la autorganización resultan interesante de estudiar y reflexionar, no sólo por el conjunto de aplicaciones que tienen en diversas áreas tales como: inteligencia artificial, economía, sociología e ingeniería, entre otras sino también por el aprendizaje que puede dejar a los seres humanos a nivel de su comportamiento individual y colectivo". **Niriaska Perozo**

APLICACIONES DE LOS SISTEMAS MULTIAGENTE

Prof. Niriaska Perozo G.

Trabajo Presentado Como Requisito Para Optar a la Categoria de Agregado en el Escalafon del Personal Docente y de Investigación

CONTENIDO:

1. BASAMENTO LEGAL
2. INTRODUCCIÓN
3. SISTEMAS MULTIAGENTE
4. INTELIGENCIA COLECTIVA
5. LA INTELIGENCIA COLECTIVA EN EL DESARROLLO DE APLICACIONES MULTIAGENTE
6. CONCLUSIONES
7. REFERENCIAS

MAYO 2008

1. BASAMENTO LEGAL

El presente trabajo de ascenso se sustenta en el Reglamento de Clasificación y Ascenso de los miembros del Personal Docente y de Investigación de la Universidad Centroccidental "Lisandro Alvarado", aprobado el 9 de mayo del 2007 en sesión extraordinaria del Consejo Universitario N° 1.772, publicado en Gaceta Universitaria 101.

Así, en el Capitulo 1, se encuentra el artículo N° 10, donde se describe los requisitos para ascender al escalafón de Agregado, específicamente en el ítem "e" se trata lo concerniente a la presentación y aprobación de un trabajo de ascenso, de conformidad a lo establecido en este Reglamento.

Además, en el Capitulo 3, de los Trabajos de Ascenso, específicamente en el artículo 35, se detallan las características que debe poseer dicho trabajo; y en el artículo 40, se mencionan las modalidades en que puede ser presentado el mismo, donde se señala entre otros, en el literal "a": *los trabajos de investigación* y en el literal "d": *las publicaciones científicas*.

Con respecto a éste trabajo, presentado para su aprobación como requisito de ascenso, tenemos que aplica tanto al literal "a", según lo establecido en el artículo 43, como al literal "d", según lo establecido en el artículo 46, del reglamento en referencia. En este sentido, confío al jurado la aplicación del Reglamento en cuestión.

2. INTRODUCCIÓN

Como miembro de la Unidad de Investigación en Inteligencia Artificial del Decanato de Ciencias y Tecnología de la Universidad Centroccidental "Lisandro Alvarado", mi trabajo de investigación y los artículos publicados presentan aportes al área de Inteligencia Artificial, específicamente, a la subárea relacionada a los sistemas multiagente e inteligencia colectiva, así tenemos que en el anexo 1, se presenta el diseño de una arquitectura multiagente para un sistema operativo Web que soporta y maneja un conjunto de servicios en un contexto heterogéneo, dinámico y adaptativo, bajo el enfoque de reconfiguración de aplicaciones; el Sistema Operativo Web (SOW) está conformado por cuatro subsistemas, representados por agentes que llevan a cabo una serie de funciones coordinadas, para permitir el uso eficiente de los recursos sobre Internet. Además, en el anexo 2, se define un método de verificación para el diseño de un sistema multiagente basado en MASINA [2]. Este método de verificación consiste en evaluar ciertos criterios en el sistema como coherencia, autonomía y robustez, entre otros para

verificar la arquitectura del sistema multiagente desde diferentes niveles de abstracción (nivel macro), y a través del cruce de los modelos (nivel micro).

Por otra parte, en el marco del desarrollo del Proyecto: "*Diseño de un Modelo Autorganizativo para un Sistema Manejador de Comunidades en un Sistema Operativo Web, utilizando Inteligencia Artificial*", código del C.D.C.H.T. – UCLA: 005-RCT-2005, ejecutado en el período 2005-2008, se logró obtener un prototipo funcional de un modelo autorganizativo para un sistema manejador de comunidades en un sistema operativo web utilizando inteligencia colectiva y dos publicaciones: "*Diseño de un Sistema Manejador de Comunidades para un Sistema Operativo Web*" (anexo 3) y el "*Modelo de un Sistema Manejador De Comunidades Autorganizativo para un Sistema Operativo Web Multiagente*" (anexo 4) enviadas a la Revista Publicaciones en Ciencias y Tecnología de nuestro Decanato. En estos artículos se describe por un lado, el diseño multiagente de un sistema manejador de comunidades para un sistema operativo Web, como un sistema de agrupamiento de recursos y servicios dinámico y autorganizado y por otro lado, el modelo funcional de este sistema con el análisis, calibración y evaluación de los parámetros claves en la simulación.

Debido a que los trabajos realizados son aplicaciones basadas o desarrolladas siguiendo el paradigma de agentes, y con la intención de facilitar la comprensión de los artículos anexados, en las secciones 3, 4 y 5 es presentado un marco teórico con los conceptos básicos relacionados al área de los sistemas multiagente e inteligencia colectiva respectivamente.

3. SISTEMAS MULTIAGENTE

En esta sección se presenta una visión general sobre los aspectos teóricos más relevantes en el área de la inteligencia artificial distribuida (IAD).

Los sistemas distribuidos han ganado importancia en los últimos años, debido por una parte al incremento de los recursos disponibles en las redes cada vez más rápidas y extensas, y por otra parte a la gran capacidad de cálculo que tienen los nodos de dichas redes. Esta situación ha favorecido la distribución de las aplicaciones hasta ahora consideradas monolíticas, y con esta distribución han aparecido nuevos servicios y productos. La Inteligencia Artificial (IA) no ha sido ajena a esta evolución, dando una nueva dimensión a las soluciones dadas a problemas existentes [5].

Dicha distribución, según [5], ha dado lugar a la IAD, como un subcampo de la IA que se encarga del estudio y el modelado de las acciones y el conocimiento de

sistemas colaborativos. A su vez, la IAD se divide en dos áreas de estudio; la *Resolución de Problemas Distribuidos* (RPD) y los *Sistemas Multiagente* (SMA). La RPD considera que un problema puede ser dividido en varios módulos, o nodos, que cooperan y comparten el conocimiento de que disponen, quedando toda la interacción entre los nodos prefijada en tiempo de desarrollo como parte integrante del sistema. Por otra parte, un SMA se puede definir como una red de solucionadores de problemas (agentes) con un nivel muy bajo de acoplamiento, que trabajan conjuntamente, lo que posibilita que se enfrenten a problemas más complejos que los abordables de forma individual.

A continuación se muestra la IAD desde ambos puntos de vistas. Además, se señala algunas diferencias entre ambos enfoques.

a) Sistemas Multiagente (SMA)

La teoría de agentes puede ser vista como una evolución de la inteligencia artificial en la búsqueda de aportar autonomía a los sistemas computacionales. De hecho, aun cuando la definición de agente ha sido motivo de un amplio debate entre la comunidad de investigación de la Inteligencia Artificial Distribuida, existe el acuerdo de que la autonomía es la característica principal que describe a un agente, entendiendo como autonomía la capacidad del agente de actuar sin la intervención de un usuario o de otro sistema. Una definición de agente ampliamente aceptada es citada por [6]: "*Un agente es un sistema computacional que está situado en un ambiente, y que es capaz de tomar acciones autónomas en ese ambiente con el fin de cumplir sus objetivos de diseño*". Así, los agentes inteligentes, además de atributos y métodos (propiedades del paradigma de orientación por objetos), poseen creencias, deseos e intenciones que los vinculan con su entorno y les proveen estados mentales de los cuales depende su comportamiento. Cada agente posee una serie de propiedades, entre las que se cuentan autonomía, movilidad, racionalidad, reactividad, sociabilidad y proactividad. Además, pueden estar dotados de mecanismos de razonamiento que les permiten abordar situaciones de manera inteligente y evolucionar por medio de la experiencia. En ese caso los agentes se denominan "*agentes inteligentes*".

Los SMA son sistemas que describen a los agentes en un entorno social, en el que dichos agentes cooperan para lograr tanto sus metas individuales como las metas colectivas de la comunidad multiagente [6]. Un SMA posee un lenguaje de agente que permite que los agentes se relacionen o interactúen.

Las características más representativas de los SMA, según [5], son:

- Cada agente tiene información incompleta, o no es capaz de resolver los problemas completos, es decir, tiene conocimiento limitado del problema.

- No existe un sistema de control global.
- Los datos están descentralizados.
- El procesamiento de la información es asíncrono.

Con respecto a la coordinación en los SMA, tenemos que las acciones en un entorno multiagente necesitan ser coordinadas por cuatro razones principales [1]:

I. Los agentes necesitan información y resultados que otros agentes pueden suministrar.
II. Los recursos son limitados. La coordinación llega a ser realmente importante cuando los recursos disminuyen, ya que los recursos tienen que ser compartidos de tal forma que permita optimizar las acciones a ser ejecutadas.
III. Para tratar de optimizar costos. La coordinación de acciones hace que se evite la realización de acciones redundantes y sin sentido.
IV. Para permitir que ciertos agentes tengan objetivos diferentes pero interdependientes, lo que permitirá realizar trabajos en conjunto.

Básicamente, la coordinación de acciones en un SMA puede ocurrir en *situaciones de cooperación* (objetivos compatibles, deben asistirse unos a otros para alcanzar sus metas) *y de competición* (objetivos incompatibles, las metas de uno contradice las de otro) [1].

b) Resolución de Problemas Distribuidos

Según [3], la RPD estudia los sistemas inteligentes distribuidos con una funcionalidad global. Los agentes cumplen unas características mínimas de: *Benevolencia* (los agentes cooperan con los demás siempre que les sea posible; no pueden "*mentir*", ni esconder información); *Objetivos Compartidos* (todos los agentes valoran el resultado de la actividad del grupo con la misma escala y desean contribuir para maximizar su calidad); *Diseño Central* (todos los agentes se diseñan para que se integren en un sistema inteligente, capaz de resolver un problema; el diseñador debe asegurar que los agentes utilicen el mismo lenguaje, que cada agente desempeñe un papel que influya en la consecución del objetivo global, entre otros).

El proceso de razonamiento en un grupo de agentes de un sistema RPD se puede subdividir en cuatro diferentes etapas: *Descomposición de tareas* (una tarea se descompone en tareas menos complejas o más pequeñas); *Asignación de tareas y recursos entre agentes* (se determina qué agentes tendrán que resolver que tareas y los recursos de los que disponen); *Resolución de subproblemas* (cada

agente resuelve las tareas que le han sido asignadas) e *Integración de soluciones* (para componer y conseguir una solución a la tarea inicial).

El dilema básico de la RPD consiste en realizar tal proceso de forma globalmente coherente, ya que las decisiones globales se toman localmente pues ningún componente dispone de una visión completa del proceso de resolución del problema.

Los diferentes mecanismos de coordinación en la RPD se pueden clasificar en tres grandes grupos [4]:

I. **Estructuras de organización:** En el contexto de la RPD las estructuras de organización se conciben como patrones estáticos de relaciones de información y de control entre los agentes. Por ejemplo, se definen áreas de interés para cada agente, en las que se especifican las tareas de las que el agente es responsable, y se definen estructuras de autoridad para resolver los conflictos en caso que las áreas de interés se solapen.

II. **Planificación multiagente:** Antes de comenzar el proceso de resolución del problema los agentes generan un plan en el que se especifican todas sus futuras interacciones. La *planificación* se realiza de forma *centralizada*, todos los agentes mandan sus intenciones de actuación a un agente coordinador. Este coordinador detecta las relaciones conflictivas y sinérgicas entre las acciones potenciales, y las coordina para generar el plan del grupo. En el caso de *planificación distribuida*, ningún agente dispone de conocimiento sobre todas las actividades del grupo, lo cual dificulta considerablemente el proceso de planificación.

III. **Intercambio de información a metanivel:** Este mecanismo facilita la coordinación dinámica, es decir, articula el proceso de resolución de problemas con el de planificación. Los agentes mantienen los modelos sobre los demás intercambiando información en un metanivel referente a su percepción del estado de resolución del problema. De este modo predicen localmente las acciones de los demás y eligen las acciones locales compatibles.

Las arquitecturas clásicas de la RPD son los *actores*, la *red de contratos* y la *arquitectura de pizarra*. La RPD asume un enfoque constructivo, donde los agentes resuelven subproblemas y se coordinan para alcanzar la funcionalidad global (metodología bottom-up) [3].

c) RPD vs SMA

Según [7] hay algunos aspectos que pueden ser contrastados para establecer claramente cuando es posible usar un enfoque u otro.

- En los sistemas RPD se parte de un problema bien definido, "*dividido*" *a priori*, para cuya solución un controlador ejerce una distribución de las tareas estáticas, presuntamente óptima, asignándolas a nodos que crea para la ocasión según criterios de eficacia general, mientras que en los SMA los agentes con creencias, objetivos e intenciones propias construyen conjuntamente el plan para solucionar el problema.
- En la RPD se mantiene una visión monolítica de la representación de planes, mientras que en los SMA lo que interesa es la coordinación de planes. Las técnicas de planificación asociadas a los SMA son, planificación centralizada de planes (distribuidos y centralizados) y planificación distribuida de planes (distribuidos y centralizados), mientras que en la RPD se asocian generalmente con Planificación Centralizada. Vemos como los SMA tienen una diversidad de formas de planificación que forman parte de su riqueza, además que sólo en los SMA puede existir un conocimiento amplio de su entorno.
- En la RPD se enfoca el problema de la coordinación de arriba a abajo ("*top-down*"), ya que los agentes son diseñados para cumplir los requisitos especificados en la definición del problema; en los SMA se hace de abajo a arriba ("*bottom-up*") ya que, en principio, los agentes se diseñan en primer lugar, y la estrategia de solución para un problema determinado se especifica más tarde. Así, los sistemas RPD, se especializan en la tarea ("*task oriented*"), mientras que los SMA adoptan un punto de vista centrado en el agente ("*agent oriented*").

4. INTELIGENCIA COLECTIVA

Según [8], la inteligencia de enjambre ("*Swarm Intelligence*") o también llamada inteligencia colectiva la podemos definir como un área de investigación que se encarga del diseño de algoritmos y de sistemas de resolución de problemas, inspirados en el comportamiento de las sociedades de insectos. Los sistemas realizados bajo el enfoque de la inteligencia colectiva se caracterizan por exhibir características que hacen a las sociedades de insectos exitosas en su entorno, tales como: flexibilidad, robustez, control descentralizado y autorganización.

Cuando hablamos de inteligencia colectiva en las sociedades de insectos debemos tratar dos aspectos claves [8]: la autorganización y la interacción indirecta.

a) ***Autorganización.*** Según [9], la autorganización *es un proceso en el que el patrón (arreglo organizado particular de objetos en espacio o tiempo) en el nivel global de un sistema emerge solamente de las interacciones numerosas entre los componentes de nivel inferior del sistema*. Por otra parte, las reglas que especifican las interacciones entre los componentes del sistema se ejecutan usando solamente información local, sin referencia al patrón global. Por ejemplo, las estructuras emergentes en el caso de la búsqueda de comida (forrajeo) en las hormigas incluyen redes organizadas espacial y temporalmente de rastros de feromona (sustancia química excretada por algunos animales que influye en el comportamiento de los de su misma especie). La esencia de la autorganización es que un sistema adquiere una estructura espacial, temporal o funcional sin interferencia específica del exterior. La organización puede desarrollarse o en tiempo o espacio, puede mantenerse en una forma estable o puede mostrar fenómenos transitorios. La característica principal de estos sistemas autorganizativos es su capacidad de lograr tareas colectivas complejas con comportamientos individuales simples, sin un control central o estructura jerárquica. Esta capacidad para resolver problemas complejos se debe al comportamiento emergente (un comportamiento nuevo y estable) que es producido por el sistema y, que es más complejo que el comportamiento de componentes individuales del sistema.

   La autorganización requiere de cuatro insumos [8]:

   I. **Retroalimentación Positiva.** Está constituida por reglas de comportamiento simple que promueven la creación de estructuras. Por ejemplo, el reclutamiento y el reforzamiento realizado en las hormigas a través del rastro dejado con la feromona o por la danza en el caso de las abejas.

   II. **Retroalimentación Negativa.** Permite equilibrar con la retroalimentación positiva y ayuda a estabilizar el patrón colectivo: puede tomar la forma de saturación, agotamiento o competición. En el ejemplo de forrajeo, la retroalimentación negativa proviene del número limitado de hormigas disponibles, satisfacción, agotamiento de la fuente de alimento, multitud en la fuente de alimento o competición entre las fuentes de alimento.

   III. **Amplificación de las Fluctuaciones (Aleatoriedad).** No sólo las estructuras emergen a pesar de la aleatoriedad sino que la aleatoriedad es frecuentemente crucial, ya que permite el descubrimiento de nuevas soluciones y fluctuaciones que pueden actuar como semillas para que nuevas estructuras se desarrollen y se fortalezcan. Por ejemplo, una hormiga que se pierda siguiendo el rastro de sus compañeras. Aunque éste fenómeno puede parecer

ineficiente, esto favorece la exploración de nuevas áreas, poder encontrar fuentes de alimento no explotadas y el reclutamiento de compañeras para esas nuevas fuentes de alimento.

IV. **Múltiples Interacciones.** Un simple individuo puede generar una estructura autorganizada tal como un rastro estable provisto de un tiempo de vida feromonal suficiente. Sin embargo, la autorganización generalmente requiere de una densidad mínima de individuos tolerantes mutuamente. Además, los individuos deben ser capaces de hacer uso de los resultados de sus propias actividades también como de las actividades de los otros. Por ejemplo, redes de rastro pueden autorganizarse y ser usadas colectivamente si los individuos usan la feromona de los otros.

b) ***Interacción Indirecta* ("*Stigmergy*").** En el caso de "Stigmergy", es un *mecanismo de retroalimentación usado para reflejar las interacciones indirectas entre los agentes y el entorno. Permite alcanzar formas emergentes de comportamiento coordinado a nivel de la sociedad* [10] además, fue introducido por el estudio de las sociedades biológicas de insectos y sus conceptos de interacción e intercambio de información. "Stigmergy" describe una forma de interacción e intercambio de información asíncrona entre agentes mediado por un ambiente activo es decir, un ambiente que es modificado y a su vez, capaz de modificar el comportamiento de los agentes involucrados. Las investigaciones de algunas sociedades biológicas de insectos demuestran que se coordinan produciendo un campo disipante o de acción en su ambiente. Por ejemplo, la interacción de las hormigas se basa en la existencia de feromona que es depositada en el ambiente por las hormigas durante sus actividades. Tal deposición de feromona modifica el ambiente para otras hormigas y determina sus actividades de cierta manera. Un cambio en el ambiente es detectado por las hormigas y puede accionar cambios en su comportamiento. En el mundo biológico, "stigmergy" permite que los animales bastante simples alcancen un comportamiento inesperado altamente ordenado a nivel de la sociedad.

5. LA INTELIGENCIA COLECTIVA EN EL DESARROLLO DE APLICACIONES MULTIAGENTE

Un SMA necesita operar en ambientes heterogéneos y dinámicos, capaces de manejar frecuentemente los cambios requeridos, para ello deben ser flexible, robustos y adaptables a las circunstancias. Actualmente muchas aplicaciones que requieren cierto grado de autorganización y un comportamiento emergente, son diseñadas bajo el enfoque de un sistema multiagente que utiliza mecanismos de comunicación y coordinación, por ejemplo basados en el comportamiento de

algunas sociedades de insectos para aprovechar por un lado, las ventajas inherentes a los SMA como: *autonomía, robustez, flexibilidad, reactividad, localidad, entre otras* [1, 6] y por otro lado, el comportamiento colectivo, autorganizado y emergente de algunas sociedades de insectos.

Las arquitecturas de un agente estigmérgico son reactivas y manejadas a través del ambiente. Los principios de este enfoque son:

- Los agentes son simples, reactivos, e inconscientes de otros agentes o de las actividades complejas que emergen de la sociedad de agentes.
- El ambiente es un mecanismo importante para dirigir actividades de estos agentes y para acumular información sobre las actividades en curso de la sociedad entera de agentes.
- La coordinación de actividades a través de la comunicación directa es remplazada por interacciones indirectas que permiten tener agentes más simples y reducir los requerimientos de comunicación entre agentes.

Los beneficios de la inteligencia colectiva basada en el comportamiento de las sociedades de insectos son ilustrados por una variedad de aplicaciones descritas en [8, 9]. Estas aplicaciones incluyen el enrutamiento del trafico de la red en sistemas de telecomunicaciones, problemas de transporte multi-robot, análisis y clasificación de datos y búsquedas de soluciones aproximadas a problemas complejos computacionalmente, entre otras.

6. CONCLUSIONES

En este trabajo de ascenso se presentan 4 publicaciones donde se describen detalladamente los aportes de cada uno, destacando los siguientes: el diseño multiagente de dos subsistemas de un SOW, un método para verificar el diseño de un SMA basado en MASINA y el prototipo funcional de un modelo autorganizativo para un sistema manejador de comunidades en un SOW utilizando inteligencia colectiva. Además, se presenta un marco teórico acerca de los sistemas multiagente e inteligencia colectiva para facilitar la comprensión del trabajo presentado.

7. REFERENCIAS

[1] Ferber, J. *"Multi-Agent Systems, An introduction to Distributed Artificial Intelligence"*. Addison-Wesley, 1.999.

[2] Aguilar, J., Cerrada, M., Hidrobo, F. "*MASINA: A Methodology to Specify Multiagent Systems*". Lecture Notes in Computer Science, Springer Berlin / Heidelberg, ISBN: 978-3-540-72829-0, Volume 4496/2007, pp. 92-101, 2007.

[3] Ossowski, S., García, A., (1998). "*Inteligencia Artificial Distribuida y Sistemas Multiagentes*". Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial 6: 1-12.

[4] Nwana, H. S. and Lee, L. C. and Jennings, N. R. (1996). "*Coordination in Software Agent Systems"*. The British Telecom Technical Journal, 14 (4). pp. 79-88.

[5] Universidad Pontificia Comillas de Madrid (2003). "*Sistemas Multiagente*". Disponible en: [http://www.iit.upco.es/~gustavop/sma.htm].

[6] Weiss, G. "*Multiagent Systems. Modern Approach to Distributed Artificial Intelligence*". MIT Press, Cambridge, MA (USA), 1.999.

[7] Alonso, E. "*Inteligencia Artificial Distribuida: Cómo Entenderla y Usarla*". Disponible en: [http://perso.wanadoo.es/jjreina/divulgacion/100tifica/articulo/iad/iad.htm]

[8] Bonabeau E., Dorigo M., Theraulaz G. "*Swarm Intelligence: from Natural to Artificial Systems*". Oxford University Press, 1999.

[9] Camazine, S., Deneubourg, J., Franks, N., Sneyd, J., Theraulaz, G. and Bonabeau, E. "*Self-Organisation in Biological Systems"*. Princeton University Press, Princeton, NJ, USA, 2001.

[10] Hadeli, P. Valckenaers, C. Bala Z., H. Van Brussel, B. Saint G., T. Holvoet, E. Steegmans. 2003. "*Self-Organising in Multi-Agent Coordination and Control Using Stigmergy*". K.U. Leuven Clestijnenlaan.

ANEXO 1

"Architecture of a Web Operating System based on Multiagent Systems"
Lecture Notes in Artificial Intelligence, Springer-Verlag, Vol. 3681, pp. 700-706, 2005

http://apps.isiknowledge.com/ISIP_GeneralSearch.do

ANEXO 2

"Definition of a Verification Method for the MASINA Methodology"

International Journal of Information Technology Vol 12, No.3, 2006

http://www.icis.ntu.edu.sg/scs-ijit/1203/1203_12.pdf

ANEXO 3

"Design of a Community Manager System for a Multiagent Web Operating System"

Enviadas para revisión a la Revista:
Publicaciones en Ciencias y Tecnología.
Decanato de Ciencias y Tecnología,
Universidad Centroccidental "Lisandro Alvarado"

ANEXO 4

"Modelo de un Sistema Manejador de Comunidades Autorganizativo para un Sistema Operativo Web Multiagente"

Enviadas para revisión a la Revista:
Publicaciones en Ciencias y Tecnología.
Decanato de Ciencias y Tecnología,
Universidad Centroccidental "Lisandro Alvarado"

ANEXO 5

"Constancia de Trabajo"

ANEXO 6

"Constancia de Proyecto Inscrito en el CDCHT"

ANEXO 7

"Constancia de Línea de Investigación"

MODELO DE UN SISTEMA MANEJADOR DE COMUNIDADES AUTORGANIZATIVO PARA UN SISTEMA OPERATIVO WEB MULTIAGENTE

Niriaska Perozo [1], **Docarly Romero**[2]
1 Unidad de Investigación en Inteligencia Artificial, Decanato de Ciencias y Tecnología. Universidad Centroccidental "Lisandro Alvarado", Barquisimeto 3001-Venezuela, nperozo@ucla.edu.ve
2 Postgrado en Ciencias de la Computación. Decanato de Ciencias y Tecnología. Universidad Centroccidental "Lisandro Alvarado", Barquisimeto 3001-Venezuela, docarlyrp@cantv.net

Resumen. Este articulo describe la simulación y evaluación funcional de un Sistema Manejador de Comunidades (SMC) para un Sistema Operativo WEB Multiagente (SOWM), siguiendo un diseño de referencia y los conceptos asociados al área de Inteligencia de Enjambre ("Swarm Intelligence") evaluando así, la autorganización y emergencia en la gestión de comunidades que requiere este tipo de sistemas. Para ello, se ha empleado un algoritmo de agrupamiento basado en el comportamiento emergente de las hormigas, para recoger y depositar cadáveres y formar pilas de ellos. Siguiendo un esquema de gestión de comunidades dinámica, autorganizativa y emergente, se realizó la búsqueda de servicios y recursos a nivel Intra e Inter en el SMC de acuerdo con los requerimientos y objetivos del SOWM y sus subsistemas.

Palabras Claves: Sistemas Multiagente, Comunidades Virtuales, Autorganización, Inteligencia Colectiva, Sistema Operativo Web.

1. Introducción

El Sistema Operativo Web Multiagente (SOWM) surge como una solución para integrar los numerosos recursos disponibles sobre la INTERNET, suministrando así una plataforma computacional que permita a los usuarios compartir recursos y resolver problemas de heterogeneidad y adaptabilidad dinámica. Para satisfacer estas necesidades se propone en [1] una arquitectura conformada por cuatro (4) entidades o subsistemas que se interrelacionan entre sí y que tienen una identidad propia. Una de esas entidades o subsistemas del SOWM es el *Sistema Manejador de Comunidades (SMC)*. El SMC propuesto plantea una solución adaptativa que pretende mejorar la eficiencia y el desempeño del SOWM. El SMC establece un mecanismo de agrupamiento dinámico de nodos, es decir, crea comunidades virtualmente organizadas que exhiben afinidades funcionales que favorece la búsqueda de recursos y servicios en la Web, en otras palabras, se buscaría por comunidades y no, de nodo en nodo. El SMC es autónomo en cuanto a la toma de decisiones, actualización y agrupamiento de

los nodos involucrados en las comunidades que forman parte de él, por ello cada comunidad es capaz de autorganizarse y adaptarse a su entorno, emergiendo nuevas comunidades. Así, se simula el comportamiento de este subsistema realizando, verificando y evaluando el diseño propuesto en [14] desde el punto de vista de implementación.

2. Antecedentes

Por ser el modelo propuesto autorganizativo, es interesante mencionar algunas aplicaciones descentralizadas y autorganizativas que operan en base a sus interacciones locales. Estos sistemas son particularmente robustos porque se adaptan a los cambios del ambiente. En algunos casos los sistemas autorganizativos están complementados con un comportamiento emergente en el sentido que los componentes individuales del sistema ejecutan tareas simples que favorecen la emergencia de tareas complejas mediante sus interacciones. Las propiedades de autorganización y emergencia han sido aplicadas en sistemas de control de manufactura [9], en sistemas de redes sensoriales para el control de tráfico [7], ha inspirado e influenciado desarrollos algorítmicos en optimización combinatoria, enrutamiento en redes de comunicaciones, análisis de datos exploratorios, partición de grafos, entre otros [3, 6, 2, 13].

En el caso de los sistemas naturales autorganizados tenemos como ejemplo las sociedades de insectos tales como las hormigas, las termitas o las abejas, la comunicación entre ellos ocurre mediante un mecanismo llamado "Stigmergy" (comunicación indirecta vía el entorno, Grassé 1959) [8, 9]. El comportamiento social de los humanos también es autorganizativo, dado que el hombre comienza a realizar una tarea con información propia y mediante interacciones locales directas o indirectas producen sociedades complejas que manifiestan un comportamiento global emergente; además en el ramo de la biología esta propiedad se pone de manifiesto en el sistema inmune y en la regeneración de células del cerebro garantizando la emergencia y autorganización de los mismos [7, 11].

Las técnicas de agrupación o clustering también ofrecen un aporte importante para el modelo propuesto, en este sentido tenemos el trabajo: "*Towards Improving Clustering Ants: An Adaptive Ant Clustering Algorithm"*[18] que consiste en incorporar propiedades adaptativas al algoritmo estándar de agrupación "Standard Ant Clustering Algorithm" (SACA) generalizado por Lummer y Faieta en base a un esquema de visión progresiva y heurística de feromona, este algoritmo propuesto denominado "Adaptive Ant Clustering Algorithm (A^2CA)" se propuso evaluar en un conjunto de datos bioinformáticos las propiedades de ejecución y convergencia garantizando una estabilización luego de un número de pasos, quedando demostrado que con estas modificaciones, el A^2CA es robusto en cuanto a que se encuentra el número correcto de grupos ("clusters"), existen bajas variaciones de los resultados y se estabiliza después de un número fijo de iteraciones definidas automáticamente por el algoritmo. Otro trabajo en esta área es: *"Ant-Based Clustering and Topographic Mapping"* [10] donde el agrupamiento y el ordenamiento basado en colonias de hormigas y aplicado en el contexto de minería de datos ("data-mining"), este trabajo incorporó una mejora a la versión básica del algoritmo de

agrupación de hormigas ("Ant Clustering", AC) llamada "*Adaptive Time-dependent Transporter Ants*" (ATTA), incorporando adaptatividad y heterogeneidad, a las actividades de transporte dependiente del tiempo, y un método de agrupación que transforma el espacio, es por esto que ATTA fue objeto de rigurosas evaluaciones de funciones y colecciones de datos para verificar que el agrupamiento y ordenamiento basado en colonias de hormigas identifica automáticamente el número de grupos en una colección de datos y produce soluciones de buena calidad. Además, es importante señalar según [17] que el clustering o agrupamiento tiene múltiples aplicaciones dentro de la ciencia de la computación, como compresión de imágenes y voz digitalizadas; en la recuperación de información; en minería de datos; en la segmentación de imágenes médicas; en la clasificación de componentes de software y en el ambiente de la Web, entre otros.

3. Aspectos Generales

Un sistema multiagente (SMA) necesita operar en ambientes heterogéneos y dinámicos, capaces de manejar frecuentemente los cambios requeridos, para ello deben ser flexible, robustos y adaptables a las circunstancias [7]. Actualmente muchas aplicaciones que requieren cierto grado de autorganización y un comportamiento emergente, son diseñadas bajo el enfoque de un sistema multiagente que utiliza mecanismos de comunicación y coordinación.

En [3] se define la inteligencia de enjambre en término de dos conceptos: Autorganización y Stigmergy. Así tenemos según [4] que Autorganización, *es un proceso donde el patrón (arreglo organizado particular de objetos en espacio o tiempo) en el nivel global de un sistema emerge solamente de las interacciones numerosas entre los componentes de nivel inferior del sistema.* Por otra parte, las reglas que especifican las interacciones entre los componentes del sistema se ejecutan usando solamente información local, sin referencia al patrón global. La esencia de la autorganización es que un sistema adquiere una estructura espacial, temporal o funcional sin interferencia específica del exterior. La característica principal de estos sistemas autorganizativos es su capacidad de lograr tareas colectivas complejas con comportamientos individuales simples, sin un control central o estructura jerárquica. Este comportamiento usualmente emerge de todas las interacciones que se producen en el sistema [15, 16].

En el caso de "Stigmergy", es un *mecanismo de retroalimentación usado para reflejar las interacciones indirectas entre los agentes y el entorno. Permite alcanzar formas emergentes de comportamiento coordinado a nivel de la sociedad* [8]. Además, fue introducido por el estudio de las sociedades biológicas de insectos y sus conceptos de interacción e intercambio de información. "Stigmergy" describe una forma de interacción e intercambio de información asíncrona entre agentes mediado por un ambiente activo es decir, un ambiente que es modificado y a su vez, capaz de modificar el comportamiento de los agentes involucrados.

Los beneficios de la inteligencia de enjambre son ilustrados por una variedad de aplicaciones descritas en [3, 4, 12]. Estas aplicaciones incluyen el enrutamiento del trafico de la red en sistemas de telecomunicaciones, problemas de transporte multi-robot, análisis y clasificación de datos y búsquedas de soluciones

aproximadas a problemas complejos computacionalmente, entre otras. Por otro lado, el agrupamiento o clustering consiste en dividir un conjunto de objetos en un número de clusters o grupos [5]. La motivación de agrupar un conjunto de datos está en encontrar una estructura inherente de datos que exhiban un alto grado de similaridad para poder agruparse; estos grupos deben tener un *alto grado de cohesión* (similitud entre los elementos de los nodos del mismo grupo), y *bajo acoplamiento* (los grupos deben ser lo mas diferentes entre sí) con respecto a otros grupos [17].

4. Descripción de la Propuesta

De acuerdo al diseño del SMC propuesto en [14], se tienen dos (2) componentes: el *manejador de comunidades* (responsable de la gestión de comunidades) y el *coordinador de búsqueda* (encargado de la búsqueda de recursos y servicios a nivel de comunidades). A continuación se describe el algoritmo utilizado y las políticas establecidas para la implementación del SMC con respecto a los componentes mencionados.

4.1 Algoritmo Utilizado

Recientemente se han propuesto algunos algoritmos de agrupamiento basados en el comportamiento de algunas sociedades de insectos, tal es el caso del Algoritmo de Agrupamiento basado en el Comportamiento emergente de las hormigas al recoger y depositar cadáveres para formar pilas de ellos ("Ant Clustering (AC)"); este modelo estándar del algoritmo de agrupamiento AC fue generalizado por Lummer y Faieta para aplicarlo al análisis exploratorio de datos [3]. El Algoritmo de Agrupamiento utilizado es presentado en la tabla 1. En el algoritmo de agrupamiento basado en el comportamiento de las hormigas ("Ant Clustering") [3], la probabilidad de recoger un dato i viene dada por:

$$Pp(i) = (k_p \ / \ (k_p + f(i)\ ))^2 \qquad (1)$$

y la probabilidad de depositar por:

$$P_d(i) = \begin{cases} 2f(i) & ; \text{si } f(i) < k_d \\ 0 & ; \text{caso contrario} \end{cases} \qquad (2)$$

Donde k_p y k_d son constantes iguales a 0.1 y 0.15 respectivamente y f(i) es la medida de densidad de similitud del dato i en una localidad en particular r y se define así:

$$f(i) = \begin{cases} 1/S^2 \ \sum_{j \in Vecindad(r)} (1 - (d(i,j) / \alpha)) & ;\text{si } f > 0 \\ 0 & ; \text{si } f <= 0 \end{cases} \qquad (3)$$

Donde S^2 es el área de la región de percepción de un agente con centro en su localidad, α es un factor de escalamiento de la medida de similitud, y d(i,j) es la medida de similitud usada en este caso (distancia euclidiana) dada por:

$$d(i,j) = \sqrt{\sum_{h=1}^{D} (x_i - y_j)^2} \quad (4)$$

La distancia euclidiana en los elementos de vectores binarios, representa la presencia del atributo como 1 y la ausencia como 0.

Tabla 1. Algoritmo Utilizado para el Agrupamiento

	INICIO DEL ALGORITMO

Inicialización	**For Cada Elemento Oi Do**
	Colocar Oi aleatoriamente sobre el Grid
	End For
	For Todos los Agentes Do
	Colocar el agente en un lugar seleccionado aleatoriamente en el grid bidimensional
	End For

Ciclo Principal	**For t=1 to tmaxpasos Do**
	For Todos los agentes Do
	If (Agente_Desocupado) and (Lugar_Ocupado_o_i)
	Entonces Calcular_F(o_i) and Pp(o_i)
	Obtener aleatoriamente un valor R (Entre 0 y 1)
	If R<=Pp(o_i) then Recoger_o_i
	Else If (Agente_Cargando_Oi) and (Lugar_Vacio)
	Entonces Calcular_F(o_i) and Pd(o_i)
	Obtener aleatoriamente un valor R (Entre 0 y 1)
	If R<=Pd(Oi) then Depositar_o_i
	End if
	Moverse aleatoriamente en el grid a un lugar no ocupado por otro agente
	End For /*Agentes*/
	End For /*Tiempo*/
	Imprimir_Localización_Elementos

	FIN DEL ALGORITMO

4.2 Políticas Establecidas en el Manejador de Comunidades del SMC

- La incorporación de un nodo a una comunidad del SMC, a través del Manejador de Comunidades, se realiza siguiendo el diagrama presentado en la figura 1.
- La eliminación de un nodo del SMC, a través del Manejador de Comunidades se realiza siguiendo el diagrama presentado en la figura 2.
- La transferencia de un nodo es otra de las tareas a cargo del Manejador de Comunidades, la transferencia de un nodo es virtual entre comunidades y se apoya en las tareas de *eliminar nodo* (para quitar el registro del nodo de la comunidad origen) y de *localizar comunidad* (para incorporar el nodo a otra comunidad), siguiendo las políticas de agrupamiento de nodos establecidas.

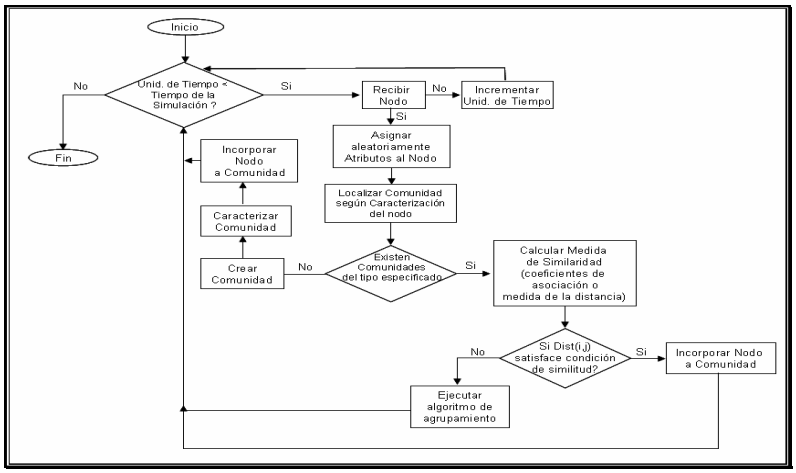

Figura 1. Diagrama de tareas involucradas en la incorporación de un nodo a través del Manejador de Comunidades

4.3 Políticas establecidas en el Coordinador de Búsqueda del SMC

El coordinador de búsquedas del SMC se encarga de recibir peticiones de búsqueda de servicios de un SMR o de otro SMC y procesarlas realizando búsquedas internas (Nivel Intra, dentro de las comunidades a las que pertenece el nodo) y/o externas (Nivel Inter, en las comunidades que no pertenece el nodo), dependiendo del éxito o no de la búsqueda y siguiendo el diagrama presentado en la figura 3.

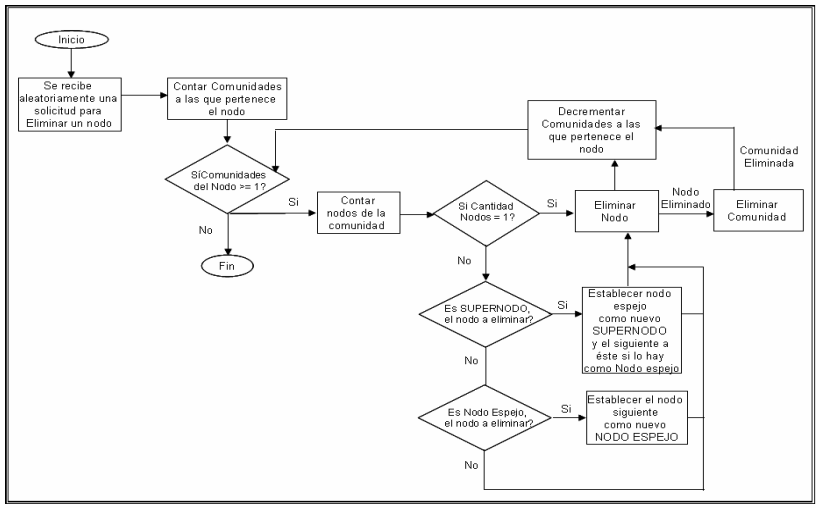

Figura 2. Diagrama de la tarea Eliminar Nodo a través del Manejador de Comunidades

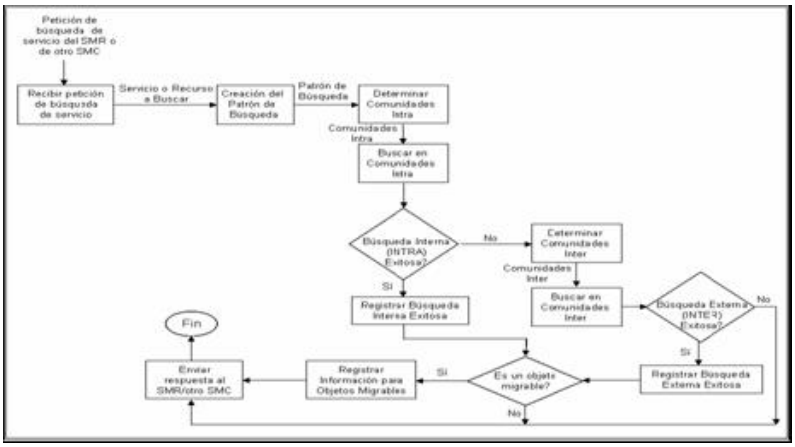

Figura 3. Diagrama de las Tareas de Localización del CB

5. Ejecución y Evaluación de la Propuesta

La implementación de esta propuesta se realizó usando el simulador Starlogo, que es una herramienta libre y con un ambiente de modelado programable para explorar el procesamiento de sistemas descentralizado. Starlogo provee un ambiente de ejecución para correr miles de agentes (o tortugas como son llamadas en el entorno) en paralelo y facilita un comando observador central para configurar e influenciar el sistema. En la figura 4 se muestra la interfaz gráfica elaborada para la simulación del SMC y sobre la que se utilizó una configuración inicial (ver tabla 2) que servirá de escenario inicial para realizar las evaluaciones del modelo propuesto.

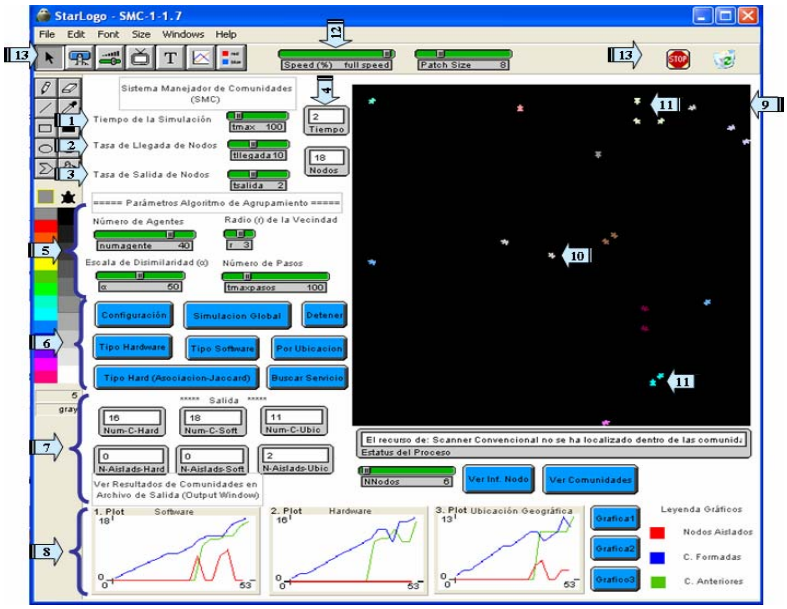

Figura 4. Interfaz Gráfica de la Simulación del SMC

Tabla 2. Configuración Inicial utilizada en la Simulación del SMC

Parámetro	Valor
Tiempo de la Simulación (tmax)	100 unidades de tiempo
Tasa de Llegada (tllegada)	10 nodos/unidad de tiempo
Tasa de Salida (tsalida)	2 nodos/unidad de tiempo
Número de Agentes (numagente)	40
Radio de la Vecindad de cada agente (r)	3
Escala de Disimilaridad (α)	50
Número de Pasos o Iteraciones (tmaxpasos)	100

Para efectos de la simulación [20], se crean 3 tipos de comunidades: *Software, Hardware y Ubicación Geográfica* para las que se ejecutó por separado la misma configuración inicial con la finalidad de obtener resultados mas detallados.

5.1 Resultados Obtenidos

5.1.1 Comunidades Tipo Hardware

En la tabla 3 se muestra el comportamiento de la gestión de comunidades de tipo hardware con los datos de la configuración inicial.

Tabla 3. Resultados Obtenidos con la Configuración Inicial en el Tipo de Comunidad: Hardware

Elemento	Resultados		
Unidades de Tiempo Transcurridas	40	70	100
Nodos Recibidos	399	699	999
Comunidades de Tipo Hardware Creadas	80	80	80

El número de comunidades de hardware que se crearon luego de la ejecución de la simulación es bastante elevado y el mismo, es producto de la cantidad de nodos aislados que quedaban luego de cada iteración del algoritmo de agrupamiento, ya que los mismos se convierten en comunidades con un sólo nodo, esto se debe a que el nodo es recogido por un agente para encontrarle ubicación y no logra depositarlo en ninguna comunidad existente, o el nodo, no es seleccionado durante la simulación antes de finalizar la ejecución de la misma.

5.1.2 Comunidades Tipo Software

Los resultados obtenidos de la gestión de comunidades de tipo software con los datos de la configuración inicial son mostrados en la tabla 4.

Tabla 4. Resultados Obtenidos con la Configuración Inicial en Comunidades de Tipo: Software

Elemento	Resultados		
Unidades de Tiempo Transcurridas	40	60	100
Nodos Recibidos	399	591	999
Comunidades de Tipo Software Creadas	80	85	85

El comportamiento de los resultados de las comunidades de tipo software es similar al obtenido con las comunidades de tipo hardware, puesto que ambos expresan la presencia o ausencia de atributos en vectores binarios.

5.1.3 Comunidades Tipo Ubicación Geográfica

La administración de comunidades de tipo ubicación geográfica es más dinámica en la ejecución de la simulación puesto que se nota el incremento y decremento de comunidades a medida que llegan los nodos, esto no se nota en los otros tipos de comunidades, ya que el cálculo de la medida de similitud basada en la distancia favorece la creación de este tipo de comunidad debido a que se calcula la distancia entre las coordenadas que dan ubicación a los nodos y no en base a los atributos de un vector binario. En la tabla 5 se muestra los resultados del comportamiento de las comunidades en diferentes unidades de tiempo.

Tabla 5. Resultados Obtenidos con la Configuración Inicial en Comunidades de Tipo: Ubicación Geográfica

Elemento	Resultados		
Unidades de Tiempo Transcurridas	40	70	100
Nodos Recibidos	399	699	999
Comunidades de Tipo Ubicación Geográfica Creadas	35	40	47

Una vez analizados los resultados obtenidos de la simulación bajo una configuración de parámetros establecida como inicial, se pudo observar que se obtuvieron resultados satisfactorios de acuerdo con los objetivos planteados, pero es necesario comparar estos resultados con los que resulten luego de evaluar algunos parámetros claves en el algoritmo de agrupamiento, y así poder detectar su sensibilidad ante diversos cambios intencionados. A continuación se muestra los resultados obtenidos al ajustar los parámetros: número de agentes (“numagente”), radio de la vecindad (“r”), número de iteraciones (“tmaxpasos”) y cambio de la medida de similitud.

5.2 Evaluación y Calibración de Parámetros

5.2.1 Ajuste del Parámetro "numagente" (Número de Agentes)

El número de agentes es uno de los principales parámetros que intervienen en el algoritmo de agrupamiento implementado, debido a que un agente es el encargado de recoger o depositar un determinado nodo en una posición especifica del gris, siempre y cuando satisfaga ciertas condiciones establecidas, ver resultados en la tabla 6.

Tabla 6. Comportamiento de Comunidades de Tipo Hardware al Ajustar el Parámetro "numagente"

"numagente"	**10**	**40**	**70**
Unidad de Tiempo	30	100	100
Nodos Recibidos	296	999	999
Nro. Comunidades de Tipo Hardware creadas	**+ 100**	**91**	**80**

5.2.2 Ajuste del Parámetro "r" (radio de la vecindad)

Para la evaluación de este parámetro se tomaron las comunidades por ubicación geográfica para notar su comportamiento cuando el radio de la vecindad de los nodos se incrementa o decrementa, ver resultados en la tabla 7.

Tabla 7. Comportamiento de Comunidades de Tipo Ubicación Geográfica al Ajustar el Parámetro "r" Radio de la Vecindad

"r"	**1**	**3**	**5**
Unidad de Tiempo	100	100	100
Nodos Recibidos	999	999	999
Nro. Comunidades de Tipo Ubicación Geográfica creadas	**42**	**47**	**35**

5.2.3 Cambiando la Medida de Similitud

Una de las evaluaciones clave es el cambio de la *medida de similitud*, que se emplea para asociar dinámicamente un nodo a una comunidad de acuerdo a grado de similaridad y en el cálculo de la función de densidad, para determinar la probabilidad de recoger o depositar un nodo en el algoritmo de agrupamiento. En el desarrollo de la simulación se tomó la distancia euclidiana como medida de similitud, pero los coeficientes de asociación son también utilizados como medida de similitud [19] en el caso de utilizar vectores binarios, están basados en la medición del número de atributos coincidentes de un vector en relación al otro, debido a que se resalta la ausencia o presencia de un atributo, ver resultados obtenidos en la tabla 8.

Tabla 8. Comportamiento de las Comunidades de Tipo Hardware al cambiar la medida de similitud

Medida de Similitud	Unidad de Tiempo	Cantidad Nodos	Comunidades de Hardware Formadas
Distancia Euclidiana	100	999	66
Coeficiente de Asociación de Jaccard	100	999	10

5.2.4 Ajuste del Parámetro "tmaxpasos" (numero de iteraciones del algoritmo de agrupamiento)

La evaluación de este parámetro se realizó en base a las comunidades por ubicación geográfica tomando cuatro valores (100, 200, 300 y 400), ver los resultados obtenidos en la tabla 9.

Tabla 9. Comportamiento de las Comunidades por Ubicación Geográfica al Ajustar el Parámetro "tmaxpasos"

"tmaxpasos"	100	200	300	400
Unidad de Tiempo	100	100	100	100
Nodos Recibidos	999	999	999	999
Comunidades x Ubicación Geográfica Creadas	**47**	**26**	**14**	**10**

La evaluación de las tareas del coordinador de búsquedas del SMC se realizó en base a la información de la comunidades de tipo hardware, software y por ubicación geográfica obtenidas luego de ejecutar la simulación global, para luego comenzar el proceso de recibir solicitudes de búsqueda de recursos desde el SMR u otro SMC, y verificar el comportamiento del CB en tres posibles situaciones: recurso no encontrado en las comunidades existentes; recurso encontrado a nivel intra y recurso encontrado a nivel inter.

6. Conclusiones

Al modelar el SMC para el SOWM se determinó después de un número de evaluaciones que el algoritmo de análisis exploratorio de datos propuesto por Lummer y Faieta [10], favorece la gestión de comunidades de manera dinámica, autoorganizada y emergente de acuerdo con el diseño propuesto. Además, se estableció los valores ideales (calibración) para los parámetros claves del algoritmo utilizado.

Con respecto a las medidas de similaridad en el agrupamiento de ítem u objetos, representan un aspecto clave, por tal razón se evaluaron dos (2) tipos de medidas; los coeficientes de asociación y la distancia euclidiana, determinando que los coeficientes de asociación específicamente el de Jaccard, reduce la creación de comunidades o agrupamiento de los nodos en comunidades, cuando se plasman los atributos en vectores binarios, tal es el caso de las comunidades

de Hardware y Software. Por otro lado, la medida de similaridad basada en la distancia euclidiana ofrece mejores resultados en las comunidades por ubicación geográfica, debido a que se mide la similaridad en base a la distancia real que existe entre nodos.

En las evaluaciones realizadas a los diversos parámetros que participan en el algoritmo se determinó, que a mayor número de agentes interactuando en el proceso de ejecución del algoritmo de agrupamiento utilizado menor es el número de comunidades o grupos que se forman; que a mayor número de iteraciones (en el intervalo [300, 400]) se logra una mejor convergencia y estabilización del sistema.

En el ajuste del parámetro radio de la vecindad, no varió significativamente el número de comunidades que se forman, pero si fue interesante los resultados obtenidos en relación al nivel de cohesión y acoplamiento de las comunidades, mientras menor sea el radio, mayor es el grado de similitud que existe entre los nodos de una comunidad y mayores las diferencias entre las distintas comunidades. Debido a esta situación se recomienda establecer el valor de este parámetro en r=1.

En cuanto a las tareas del coordinador de búsqueda del SMC, se realizó el proceso en tres (3) escenarios distintos de acuerdo a los requerimientos del SMC y de los otros subsistemas. Como trabajo futuro se podría ensayar con otros tipos de medidas de similaridad, distintas a las medidas implementadas para comparar los resultados en cuanto al numero de comunidades formada. Se podría implementar el algoritmo de agrupamiento con agentes heterogéneos que tenga la capacidad de memoria a corto plazo, de tal manera, que ellos puedan recordar cuales fueron los últimos nodos visitados para no volver a visitarlos en un tiempo determinado, ya que este comportamiento podría reducir el numero de comunidades formadas y acelerar el tiempo de convergencia del algoritmo. Finalmente, se podría variar dinámica e incrementalmente el parámetro radio de vecindad en el intervalo [1, 5], para acelerar la disolución de pequeñas comunidades preliminares.

7. Referencias

[1] Aguilar, J., Perozo, N., Ferrer, E., Vizcarrondo, J. 2005. “Architecture of a Web Operating System based on Multiagent Systems”. Publicado en Lecture Notes in Artificial Intelligence, Springer-Verlag, Vol. 3681, pp. 700-706.

[2] Aguilar, J. 2004. “La Inteligencia Colectiva en la Resolución de Problemas de Optimización Combinatoria Dinámicos”. Proceeding of the XIV Simposio Internacional de Métodos Matemáticos Aplicados a las Ciencias", pp. 21-22, San José, Costa Rica, Febrero 2004.

[3] Bonabeau E., Dorigo M., Theraulaz G. “Swarm Intelligence: from Natural to Artificial Systems”. Oxford University Press, 1999.

[4] Camazine, S., Deneubourg, J., Franks, N., Sneyd, J., Theraulaz, G. and Bonabeau, E. “Self-Organisation in Biological Systems”. Princeton University Press, Princeton, NJ, USA, 2001.

[5] Cui, X., Gao, J., Potok, T. 2006. “A flocking based algorithm for document clustering analysis”. Journal of Systems Architecture. USA.

[6] Di Caro, G., Dorigo, M. 1998. “AntNet: Distributed Stigmergic Control for Communication Networks”. Journal of Artificial Intelligence Research 9.

[7] Di Marzo, G., Pierre, M., Karageorgos, A. 2006. “Self-Organisation and Emergence in MAS: An Overview”. Informática 30 (2006): 45-54.

[8] Grassé P.P. 1959. “La reconstruction du nid et les coordinations inter-individuelles chez Belicositermes natalensis et Cubitermes sp. La théorie de la Stigmergie : Essai d'interprétation du comportement des termites constructeurs”. Insectes Sociaux, 6, p.p. 41-80. 3.

[9] Hadeli, P. et al. 2003. “Self-Organising in Multi-Agent Coordination and Control Using Stigmergy”. K.U. Leuven Clestijnenlaan.

[10] Handl, J., Knowles, J., Dorigo, M. 2006. “Ant-Based Clustering and Topographic Mapping”. Artificial Life Volume 12. pp 35-61. Massachusetts.

[11] Holland, O., Chris, M. 1999. “Stigmergy, Self-Organization, and Sorting in Collective Robotic”. Instituto de tecnología de Massachussets (Artificial life r: 173-202).

[12] Izquierdo Torres, Eduardo. 2004. “Collective Intelligence in Multi-Agent Robotics: Stigmergy, Self-Organization and Evolution”. University of Sussex.

[13] Montes de Oca M. et al. 2005. “Efectos de la comunicación directa entre agentes en los algoritmos de agrupación de clases basados en el comportamiento de insectos sociales”. Revista Iberoamericana de Inteligencia Artificial Nro 25.

[14] Perozo N., Aguilar J. “Design Of A Community Manager System For A Multiagent Web Operating System”. Enviado para evaluación a la revista Publicaciones en Ciencia y Tecnología, 2008.

[15] Rennard, J.P. 2003. “Social-Insect and self-organization”, URL: http://www.rennard.org/alife. (Consulta Septiembre 27, 2006).

[16] Rodriguez, M.A.. 2005. “A Self-Organizing Collective-Intelligence approach to the Peer-Reviewed Publication Process”. University of California, Santa Cruz (USA).

[17] Vicente, E., Tupia, M., Rivera, L.. 2006. “GraspKM en la Recuperación de la Estructura de Software”.

[18] Vizine, A., De Castro, L., Hrusehka, R., Gudwin, R.. 2004. “Towards Improving Clustering Ant: A Adaptive Ant Clustering Algorithm”. Informática 29 (2005) pp 143-154.

[19] Wiggerts, T. A. 1997. “Using clustering algorithms in legacy systems remodularization”. In WCRE’97 Proceedings of the Fourth Working Conference On Reverse Engineering. Washinton, USA. Pag 33.

[20] Romero D. “Modelo de un Sistema Manejador de Comunidades Autorganizativo para un Sistema Operativo Web Multiagente”. Tesis de Maestría, Universidad Centroccidental “Lisandro Alvarado”, 2008.

Design Of A Community Manager System For A Multiagent Web Operating System

Niriaska Perozo [1], Jose Aguilar[2]
1 Unidad de Investigación en Inteligencia Artificial, Decanato de Ciencias y Tecnología. Universidad Centroccidental "Lisandro Alvarado", Barquisimeto 3001-Venezuela, nperozo@ucla.edu.ve
2 CEMISID, Facultad de Ingeniería. Universidad de los Andes, Mérida 5101- Venezuela, aguilar@ula.ve

Abstract. This paper describes the design of a Community Manager System for a Multiagent Web Operating System. The system allows virtual gathering of nodes in communities, according to specific criteria, with the objective of improving the efficiency and performance of the Multiagent Web Operating System. The design is made and verified according to MASINA, a methodology for agent specification.

Keywords: Multiagent Systems, Web Operating Systems, Self-Organization, Virtual Communities.

1 Introduction

Given the wide variety of unimaginable services everyday in the web, it's difficult to design an operating system to support those services individually. The Multiagent Web Operating System (MWOS) has been developed to cover these needs, with a main objective of providing a platform that allows users to benefit from the computational potential offered by the web, through resource sharing and problems solving of heterogeneity and adaptability. Thus, in [1] is proposed an architecture for a MWOS, defining it as an intelligent versioned operating system, which can be dynamically self-configured in order to allow easy and clear access to the resources of Internet. This MWOS is made up of four sub-systems amongst which the Community Manager System (CMS) is found. The MWOS demands great interaction between different nodes, since each one offers different services and applications. In addition, since the number of existing nodes can grow at an incredibly way, there is a need to gather them in communities dynamically created in a specific environment and context. Thus, the groups of nodes that continually interact with each other due to the relationships between them and their environment will form communities. This

way, the architecture proposed in [1] for a MWOS defines a CMS with the goal of establishing the necessary policies for the dynamic and emerging creation, modification, and elimination of the existing communities in the MWOS. In this project we will create a detailed design of the CMS using the agent's theory, for which we will use the MASINA methodology [2, 3].

2 Theoretical Aspects

2.1 Background

The concept of community has been used and expanded to different areas, but it is currently having an important utilization in the Internet, which is one of the fastest and most penetrating phenomena in the society. The key consists in creating and maintaining virtual communities thought as efficient mechanisms of resource management over the Internet. Different works have been proposed at level of communities, for example the notions of *Groups* (users grouped according to common interests), *Chat Rooms* (online dialogue channels), *Intranets* (company employees using Internet technology), *Extranets* (people who do not belong to a given company, using that organization's intranet), among others have been used.

Particularly, the notion of *virtual communities* emerges from Internet. Among the most relevant works in this domain we find: "*Adaptive Web-Based Database Communities*" (creating data base communities specialized by area of interest) [6], "*Adaptive Content Management in Structured P2P Communities*" (using an adaptive, distributed algorithm to duplicate and replace contents in a community) [7], "*Building Adaptive E-Catalog Communities Based on User Interaction*" (it creates communities of electronic catalogs that continually adapt and restructure themselves in a dynamic environment) [8], "*Ws-Catalog-Net: An Infraestructure for Creating Peering, and Querying e-Catalog Communities*" (it consists of the creation of communities for making flexible the search when the information is not available locally in the portals) [11], "*Toward Self-organizing service communities*" (it is a structure where the communities of catalogue of services are made, linked, checked and adapted for constant interactions) [12], "*Virtual Clusters for Grid Communities*" (it allows to the grid's authorized clients, to negotiate the creation of virtual groups ("clusters") on virtual machines which are formed for satisfying the client's requirements in a determined time) [13]. The communities presented in these studies have adaptive, intelligent and emerging properties.

But the MWOS not only considers the communities as a means to integrate resources through isolated entities (like federations in Jini [4]), but the evolving entities developed within a specific context and/or define a set of implicit or explicit relationships between its components (like WOS ™ [9, 10]) in a multiagent system.

2.2 Proposed MWOS

The architecture of the MWOS studied here is detailed in [1], let us take a look at the most relevant aspects. Our MWOS is a versioned and intelligent operating system that dynamically self-configures in order to allow an easy access to resources distributed over the Internet. The basic model of the proposed MWOS is composed of four subsystems: the **Resource Manager Subsystem (RMS)** offers the means to locate and assign resources in the MWOS, and is also responsible of activities such as the management of the inference engine, offering the configuration of a given service according to the user's requirements as a final result. The **Repository Manager Subsystem (LRMS y RRMS)** provides techniques to organize the information stored in the local and remote repositories and also coordinates the access to these. The **Community Manager Subsystem (CMS)** establishes mechanisms that allow the grouping of nodes into communities organized by attributes or properties that must be present in these. And the **Web Object Manager Subsystem (WOMS)** provides mechanisms to manage the migration and replication of Web objects over the Internet in order to reduce the number of communications and to improve the tolerance to failures in the system.

Due the dynamics present in the Web and the large number of nodes that can be connected to the MWOS, **communities** are created, which simply are a virtual gathering of the nodes that belong to the MWOS, exhibiting functional and behavioral similarities. The use of communities will optimize the search of any service, since instead of looking node by node; the search will be done community by community.

3 Description of the Proposed Community Manager System

3.1 Proposal

The nodes interact in the MWOS through mechanisms of request/response and negotiation. The nodes in the MWOS are defined according to their skills and behavior. These are the elements considered as node features, which as a whole (with its resources and elements among others) is associated to a given community, that is, it's grouped according to certain similarities with other nodes (see figure 1).

The CMS must have protocols for the communication between nodes in a given community and between the different communities. Additionally, it must have mechanisms to manage the elements that belong to a given community. In addition, there are protocols that manage the creation, modification, and elimination of communities, and in general, the incorporation, elimination or migration of nodes between the different existing communities.

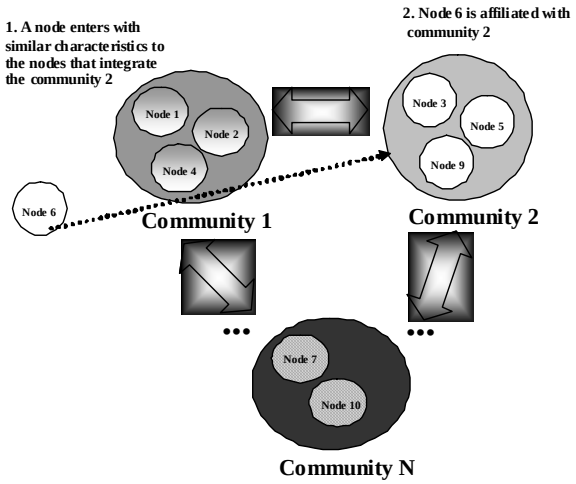

Fig. 1. Integration of a node to the set of communities of the MWOS

3.2 Architecture

In the CMS of the MWOS, the operations are coordinated by two components: the *community manager* and the *search manager*. A description of each one of these units (the architecture of the CMS is shown in figure 2) is seen below.

Community Manager: it is the main unit in the CMS, since it's in charge of managing the existing communities in the MWOS. It characterizes a new emerging community in the MWOS through its *community characterization module*, and with its *updating module* it is in charge of creating, eliminating and modifying a given community. Additionally, it allows the location of some communities with certain features or profiles through its *community search module*, finally, through its *node characterization module*; it is in charge of describing a node to be incorporated into a given community.

Search Manager: it is the unit in charge of receiving requests for search services from an RMS or from another CMS, and of processing them through its *service processing module*. It conducts the search for a given service, primarily through its *internal search module*, and if that is not successful, through its *external search module*.

For designing purposes we must take into account the following CMS features:

Each node in the MWOS holds information about the communities to which it belongs. Additionally, each node is capable of belonging to various communities, and each community belongs to a given type of community, understanding for a type of community a subset of attributes generated from the CMS generic template. For each type of community, ranges are established for each attribute, and different communities are derived for this type of community according to the values of their attributes. The search for services is carried out

first at the *Intra level* (in the communities to which the node belongs), and in case of failure, at the *Inter level* (communities external to the node).

Each community is capable of self-organizing and adapting to its environment. Communities emerge when a node is incorporated into the MWOS with features or a profile that does not fit into existing communities, or to lodge a node that needs to be part of a community different from the currently existing ones, because its features have changed and do not match with one of the current communities. A community disappears when it is empty that is, without any nodes.

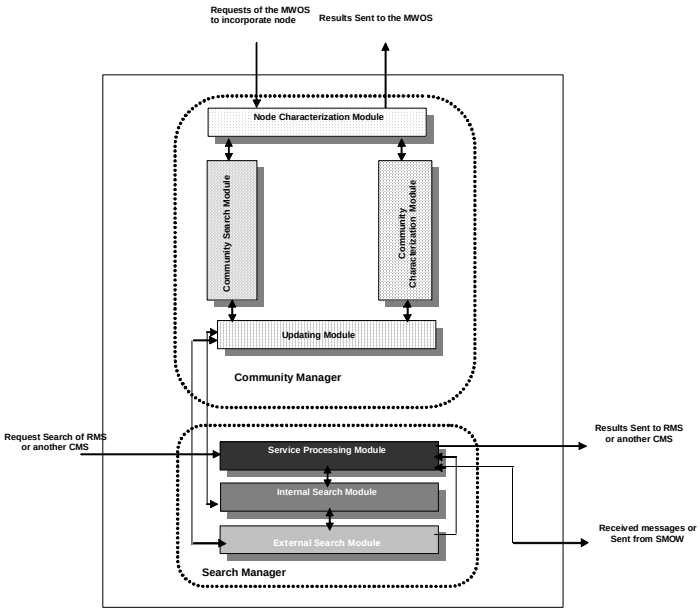

Fig. 2. Architecture of the CMS

4 Design

The CMS can be seen as a system made up of intelligent agents capable of cooperating to obtain the solutions to problems related to the management of communities. This way, the CMS is designed following an approach based on Multi-agent Systems (MASINA [2]) considering the following phases: conceptualization, analysis, design and implementation. The conceptualization phase consists of defining the actors and cases of use (see figure 3 y 4).

In the analysis phase the models for agents, tasks, communication, coordination and intelligence are developed [2]. For the purpose of this paper, we will first present the agents and tasks of the CMS and then an example of the model for agents and intelligence of the Search Coordinator Agent. Agent identification is carried out based on the actors defined in the conceptualization phase. Depending on the roles defined during this phase, we have the following functionalities: *community updating and service location at the community level*. We will keep all of these actors as agents of our system. We can then identify two agents, which will be called: **Community Administrator Agent (CAA)**

and Search Coordinator Agent (SCA). These two agents will be distributed in each node of the MWOS. Let's next see an example of the agent model of the Search Coordinator Agent [5].

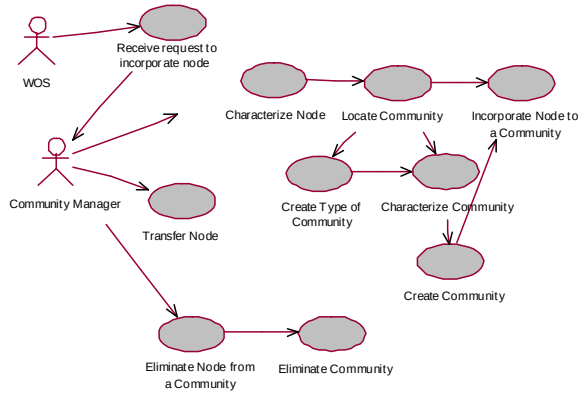

Fig. 3. Use Cases of the Actor Community Manager of CMS

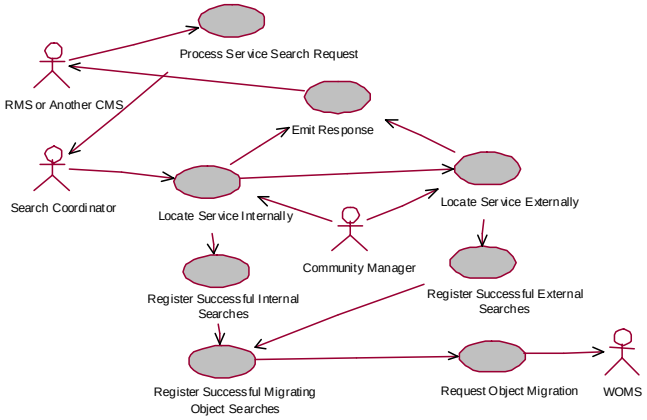

Fig. 4. Use Cases of the Actor Search Coordinator of CMS

Table 1. Search Coordinator Agent (SCA)

AGENT	**SEARCH COORDINATOR (SC)** **1.1**
NAME	Search Coordinator.
TYPE	Software Agent: Reactive Agent
ROLE	Management of service requirements.
POSITION	It's an agent with specific tasks related to locating a particular service.
DESCRIPTION	This is the agent in charge of providing mechanisms that help locate a required service, aiming to satisfy a received request.

Table 2. SC Agent Objective

	OBJECTIVE – SC AGENT **1.2**
NAME	Locating a given service.

TYPE	Persistent Objective.
INPUT PARAMETERS	Service information (ID, Name, Version).
OUTPUT PARAMETERS	Service location.
ACTIVATION CONDITION	To receive a search request from the RMS or another CMS.
FINALIZATION CONDITION	Service finding, exception in the search.
SUCCESS CONDITION	The location of the required service is obtained.
FAILURE CONDITION	Exit condition, the allowed waiting time expires.
REPRESENTATION LANGUAGE	Natural language.
ONTOLOGY	CMS Service ontology
DESCRIPTION	The SC agent has as its objective to provide search mechanisms that allow for the location of a required service.

Table 3. SC Agent Service

	SERVICE – SC AGENT	**1.3**
NAME	Service finding	
TYPE	Free, concurrent	
INPUT PARAMETERS	Service information (ID, Name, Version).	
OUTPUT PARAMETERS	Service address found. Exception parameters such as: service not found, estimated waiting time expired.	
REPRESENTATION LANGUAGE	Natural language.	
ONTOLOGY	CMS Service ontology	

Table 4. SC Agent Service

	SERVICE – SC AGENT	**1.4**
NAME	Statistics control	
TYPE	Free, concurrent	
INPUT PARAMETERS	Successful outcome in the search for a service.	
OUTPUT PARAMETERS	Entry with Node, Community and service location. It's necessary to quantify the use frequency of migrating objects used to satisfy a service.	
REPRESENTATION LANGUAGE	Natural language.	
ONTOLOGY	CMS Service ontology	

Table 5. SC Agent Service-Property

	SERVICE PROPERTY – SC AGENT	**1.5**
PROPERTIES	Service performance Service reliability	

Table 6. SC Agent Service-Property Performance

	SERVICE PROPERTY – SC AGENT	**1.5.1**
NAME	Performance.	
VALUE	Bad, Good, Excellent.	
DESCRIPTION	This property has to do with the response time that is achieved after carrying out all the necessary activities in order to satisfy a required service.	

Table 7. SC Agent Service-Property Reliability

	SERVICE PROPERTY – SC AGENT	**1.5.2**
NAME	Reliability.	
VALUE	Reliable, Non-reliable.	
DESCRIPTION	This property has to do with the SC agent's ability to give useful, timely information in the decision-making process.	

Table 8. SC Agent General Capacity

	GENERAL CAPACITY – SC AGENT **1.6**
ABILITIES	It has the capacity to intelligently coordinate the search of a particular service, both at the internal level (searching in the communities to which the node requesting the service belongs (Intra)) as well as at the external level (looking in the rest of the communities (Inter)) and for this purpose it has the ability to compare patterns. It's additionally in charge of initiating an object's migration process, following the established object migration policies.
REPRESENTATION LANGUAGE	Natural language.

Table 9. SC Agent Restriction

	RESTRICTION – SC AGENT **1.7**
RULES	The search coordinator agent must activate efficient search mechanisms in order to provide acceptable response times. It's the agent responsible of remote searching for a service in the MWOS, thus, its actions are not subject to any other agent in the MWOS.
PREFERENCES	The requested service is satisfied at the internal level as long as possible. (Intra search).
CLEARANCE	Only the AC, RMS, SMOW agents have access to the search coordinator agent.

Finally, the Intelligence Model lets us describe the aspects that give an agent its intelligence, such as its reasoning, learning and experience accumulating mechanisms. We next present the intelligence model for the search coordinator agent.

Table 10. SC Agent Reasoning Mechanism

	REASONING MECHANISM **2.1**
INFORMATION SOURCE	Results from service searches.
ACTIVATION SOURCE	Service search request.
INFERENCE TYPE	Inductive
REASONING STRATEGY	Diffuse expert system, neuronal nets for pattern recognition and data mining; search mechanisms based on evolving computation. It's necessary to generalize, based on experience, the conduction of the search itself in an easier way in the future (induction). To face unknown situations in order to get more experience.

Table 11. SC Agent Learning Mechanism

	LEARNING MECHANISM **2.2**
NAME	Diffuse Classifying System
TYPE	Adaptive
REPRESENTATION TECHNIQUE	Rules
LEARNING SOURCE	Historic information, stemming off the dynamic search results.
UPDATING MECHANISM	Rule modification. Learning is updated considering previous experiences.

Table 12. SC Agent Experience

	EXPERIENCE **2.3**
REPRESENTATION	Rules
TYPE	Case-based
DEGREE OF RELIABILITY	Moderate

5 Design Verification and Obtained Results

The CMS was verified following a verification method proposed in [3] for the MASINA methodology [2]. Verification was made at two levels: Macro and Micro.

At the **macro level** it was verified that the CMS design satisfies the objectives of the MWOS. Regarding the CMS, we can say that it is divided into two agents: *the Search Coordinator Agent and the Community Administrator Agent*, which help satisfy the objectives set for the CMS. Additionally, the CMS relationships described in figure 2 are kept by the CMS agents, and the tasks to be carried out by the CMS are distributed between the two agents.

At the **Micro Level**, the model cross proposed in [3] was carried out. The results achieved for example for Agent model cross with the others models were the following:

Agents Model vs. Tasks Model*:* it was verified that every task in the CMS be assigned to one of the CMS agents; all the ingredients required or provided by the tasks are provided or received by an agents defined in the MWOS; every CMS agent satisfied its objectives by carrying out the tasks defined in the tasks model and the capacities required by the tasks are provided by the agents that carry them out (defined in the agents model). **Agents Model vs. Communication Model*:*** It was verified for each act of speech in the designed communication model, the used ontologies, the involved agents, and who initiated them, coinciding with the defined agents and the ontologies known by each CMS agent. **Agents Model vs. Coordination Model*:*** It was validated that the agents involved, the objects to be satisfied, the required capacities, the provided services and the synchronizations required by the conversations in the CMS coordination model correspond to the agents, objectives, capacities and services defined in the CMS agent model. **Agents Model vs. Intelligence Model*:*** it was verified that the reasoning capacities expressed in the intelligent model for the Community Administrator and the Search Coordinator agents, correspond to the reasoning capacities defined for the Community Administrator and the Search Coordinator agents in the CMS agent model.

6 Conclusions

Through the proposed CMS the MWOS could optimize the management of the services requested at the community level. It helps manipulate and search through communities, and it also could contribute to offer acceptable response times. Additionally, considering that the grouping of nodes and the updating processes of existing communities are carried out in a dynamic and emerging way, the proposed CMS could allow the MWOS to achieve these objectives in a flexible way. Thus, this CMS proposal is perfectly adaptable to the necessities of a Multiagent Web operating system, since in the Web there are not anything predefined due to its dynamic and volatile nature.

Further work involves implementing a simulation prototype, for a specific problem, in order to instantiate each agent and component, validating and evaluating the design from an implementation point of view.

7 References

[1] Aguilar, J., Perozo, N., Ferrer, E., Vizcarrondo, J. "Architecture of an Operating System based on Multiagent Systems". *Lecture Notes in Artificial Intelligence*, Springer-Verlag, Vol. 3681, pp. 700-706, 2005.

[2] Aguilar, J., Rivas F., Hidrobo F., Cerrada M. "Análisis y Diseño de Sistemas Multiagente usando la Metodología MASINA". Technical Report - Universidad de los Andes, January 2004.

[3] Aguilar, J., Perozo, N., Vizcarrondo, J. "Definition of a Verification Method for the MASINA Methodology". *International Journal of Information Technology*, Springer-Verlag, Vol 12, No.3, 2006.

[4] Morgan S. "Jini to the Rescue". IEEE Spectrum, 37(4):44-49, 2.000.

[5] Perozo N. "Diseño de un Sistema Manejador de Repositorios Locales y un Sistema Manejador de Comunidades para un Sistema Operativo Web". Tesis de Maestría, Universidad de los Andes, 2.003.

[6] Bouguehaya A. "Adaptive Web - Based Database Communities", Department of Computer Science, Virginia Tech, USA, 2.002.

[7] Kangasharju J, Ross, K. "Adaptive Content Management in Structured P2P Communities", http://citeseer.nj.nec.com/571891.html, 2.002.

[8] Paik H., Benatallah B. "Building Adaptive E-Catalog Communities Based on User Interaction". IEEE Intelligent Systems, November-December p.p. 2-10, 2.002.

[9] Kropf P., Plaice J. "WOS Communities – Interactions And Relations Between Entities in Distributed Systems". 1.999.

[10] Kropf P., Plaice J. "Intentional Communities". Proceedings of Distributed Communities on the Web (DCW'2000) workshop, 2.000.

[11] Baina, K., Benatallah, B., Paik, H.Y., Toumani, F., Rey, C., Rutkowska, A., Harianto, B. "WS-CatalogNet: An Infraestructure for Creating, Peering, and Querying e-Catalog Communities". Proceedings of the 30th VLDB Conference, Toronto-Canada, 2004.

[12] Paik, H.Y., Benatallah, B., Toumani, F. "Toward self-organizing service communities". System, Man and Cybernetics, Part A, IEEE Transactions. Mayo, 2005.

[13] Foster, I., Freeman, T., Keahey, K. Scheftner, D., Sotomayor, B., Zhang, X. "Virtual Clusters for Grid Communities". Proceedings of the 3rd Europena Across Grids Conference, 2005.