Sistemas multi-agente en entornos p2p

Rubén Mondéjar[1], Jordi Pujol[1], Pedro García[1] and Carles Pairot[1]

[1] Department of Computer Science and Mathematics, Universitat Rovira i Virgili
Avinguda dels Països Catalans 26, 43007 Tarragona, Spain
{ruben.mondejar, jordi.pujol, pedro.garcia, carles.pairot}@urv.cat

Resumen. En este artículo analizaremos como los sistemas multi-agente han sido portados a entornos p2p (peer-to-peer) y qué beneficios se han obtenido. Para ello, mostramos los fundamentos del p2p, de los sistemas multi-agente, así como la línea de investigación en el contexto que nos ocupa. Finalmente, presentamos una discusión sobre las razones por las cuales resulta adecuada la simbiosis de los sistemas multi-agente y el p2p.

1. Introducción

Los desarrollos sobre redes de gran escala de telecomunicaciones y sistemas distribuidos han visto incrementado su interés con los sistemas multi-agente [1]. La propia definición de agentes software ya nos indica que los agentes tienen características parecidas a la de los sistemas p2p, como son su funcionamiento de forma distribuida y autónoma. La extensión de los sistemas multi-agente puede ser facilitada si la tecnología de los sistemas de agentes pudiese interoperar con servicios p2p estándares.

Las especificaciones FIPA [2] definen un marco de trabajo multi-agente con servicios para la localización de agentes y un servicio de mensajes que soporta ontologías. De todas formas, aún no se ha propuesto un sistema multi-agente *wide-area*. En este sentido, AgentCities [3] utiliza un anillo central para interconectar diversas infraestructuras de agentes.

La utilización de los mecanismos p2p para dar soporte a los sistemas multi-agente era algo que debía ocurrir y que podemos ver reflejado en los primeros trabajos que se realizaron [4][5]. Estas primeras aproximaciones estaban motivadas por el uso de agentes que se beneficiaban de los de los mecanismos p2p.

En los últimos años el paradigma del p2p ha evolucionado de forma que los trabajos que hacen uso de él, también han ido adaptándose a esta evolución. En este artículo veremos la evolución de estos paradigmas en conjunto hasta el momento y discutiremos las ventajas y desventajas de estas asociaciones.

2. Antecedentes

Antes de entrar a definir que son los sistemas multi-agente, hemos de centrarnos en los elementos que lo forman: los **agentes**. Podemos definir un agente con la explicación ofrecida por el doctor Wooldridge en [6]: "Un agente inteligente es un proceso computacional capaz de realizar tareas de forma autónoma y que se comunica con otros agentes para resolver problemas mediante cooperación, coordinación y negociación. Habitan en un entorno complejo y dinámico con el cual interaccionan en tiempo real para conseguir un conjunto de objetivos". Las propiedades indispensables de un agente son:

- *Autonomía*: es la capacidad de operar sin la intervención directa de los humanos, y de tener algún tipo de control sobre las propias acciones y el estado interno.
- *Sociabilidad/Cooperación*: los agentes han de ser capaces de interactuar con otros agentes a través de algún tipo de lenguaje de comunicación.
- *Reactividad*: los agentes perciben su entorno y responden en un tiempo razonable a los cambios detectados.
- *Pro-actividad o iniciativa*: deben ser capaces de mostrar que pueden tomar la iniciativa en ciertos momentos.

Otras propiedades destacables serían:

- *Movilidad*: posibilidad de moverse a otros entornos a través de una red electrónica.
- *Continuidad temporal*: los agentes están continuamente ejecutando procesos.
- *Veracidad*: un agente no comunicará información falsa premeditadamente.
- *Benevolencia*: es la propiedad que indica que un agente no tendrá objetivos conflictivos, y que cada agente intentará hacer lo que se le pide.
- *Racionalidad*: el agente ha de actuar para conseguir su objetivo.
- *Aprendizaje*: mejoran su comportamiento con el tiempo.
- *Inteligencia*: usan técnicas de IA para resolver los problemas y conseguir sus objetivos.

Llamaríamos sistema multi-agente (MAS) a aquél en el que un conjunto de agentes cooperan, coordinan y se comunican para conseguir un objetivo común. Las principales ventajas de la utilización de un sistema multi-agente son:

- *Modularidad***:** se reduce la complejidad de la programación al trabajar con unidades más pequeñas, que permiten una programación más estructurada.
- *Eficiencia***:** la programación distribuida permite repartir las tareas entre los agentes, consiguiendo paralelismo (agentes trabajando en diferentes máquinas).
- *Fiabilidad***:** el hecho de que un elemento del sistema deje de funcionar no tiene que significar que el resto también lo

hagan; además, se puede conseguir más seguridad replicando servicios críticos y así conseguir redundancia.

- *Flexibilidad*: se pueden añadir y eliminar agentes dinámicamente.

La administración de agentes establece el modelo lógico para la creación, registro, comunicación, movilidad y destrucción de agentes. En este proyecto vamos a seguir los estándares establecidos por la FIPA. En la figura 1 se puede ver el modelo de administración de agentes para este entorno.

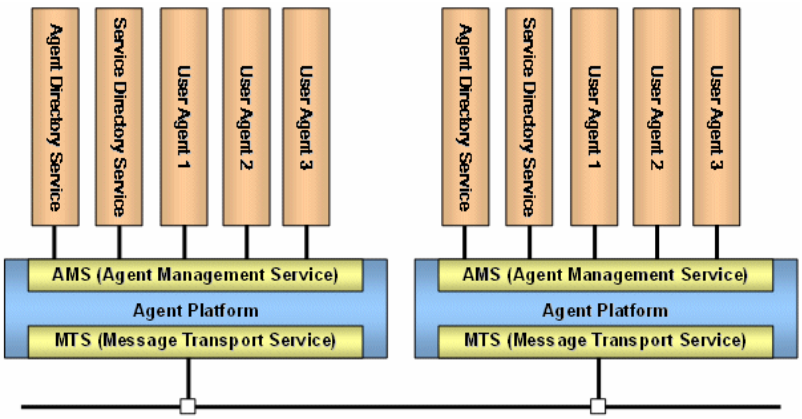

Figura 1. Arquitectura de un sistema multi-agente

Los componentes que forman parte de la figura 1 son:

- *Agentes* (User Agents)**:** unidad básica. Se podría definir como un programa que contiene y ofrece una serie de servicios.
- *Service Directory Service (SDS)***:** agente que proporciona un servicio de *páginas amarillas* dentro del sistema, es decir, conoce los diferentes servicios que ofrecen el resto de agentes de la plataforma. Los agentes se han de registrar en el SDS para ofrecer sus servicios.
- *Agent Directory Service (ADS)*: agente que proporciona el servicio de *páginas blancas* dentro del sistema. Por lo tanto, conoco todos los agentes actualmente activos en la plataforma.
- *Agent Management System (AMS)*: es el agente que controla el acceso y uso de la plataforma. Almacena las direcciones de los agentes, ofreciendo un servicio de páginas blancas.
- *Message Transport System (MTS)*: facilita la comunicación entre los agentes de diferentes plataformas.
- *Agent Platform (AP)*: proporciona la infraestructura básica para crear y ejecutar agentes.
- *Software*: cualquier programa accesible desde un agente.

Los agentes individualmente proporcionan una determinada funcionalidad, pero lo que les hace tan adecuados para ciertos sistemas es su capacidad de cooperar para resolver problemas. Para poder hacerlo, los agentes se han de comunicar entre sí, utilizando un lenguaje común: ACL (Agent Communication Language). Para garantizar la homogeneidad y compatibilidad entre los diversos agentes, la FIPA determina qué forma ha de tener un mensaje y cómo y cuando deben utilizarse; para ello, esta organización elabora las FIPA Specifications.

2.2 El modelo p2p

Podemos definirlo como una arquitectura para sistemas distribuidos que típicamente dispone de muchos nodos que resultan no confiables y heterogéneos. Es tolerante a fallos y auto-organizable, operando en entornos dinámicos con frecuentes entradas y salidas de nodos. Su objetivo primordial es beneficiarse de los recursos distribuidos compartidos (como el porcentaje de CPU, ancho de banda o espacio de almacenamiento) entre los diferentes nodos que conforman la red.

Uno de los problemas principales de este tipo de arquitecturas es el tema de encontrar contenidos en la red. En los primeros esquemas, léase Napster, se utilizaba el mecanismo del **índice central**, donde todos los usuarios se registraban en un servidor central que servía para encontrar los contenidos. Las búsquedas se hacían en el servidor central, y las transferencias de datos entre los clientes afectados. Este esquema tenía el problemas de escalabilidad: el servidor central se convertía en un cuello de botella al verse saturado de peticiones de múltiples usuarios (entradas y salidas de usuarios, búsquedas, etc).

Para solucionar este problema, la siguiente generación de sistemas p2p apareció con la red **Gnutella**. El esquema utilizado era el de **inundación** (*flooding*), y consistía en una red de nodos no estructurados, conectados anárquicamente entre sí, los cuales no dependen de ningún servidor centralizado. No obstante, el problema de la localización de los recursos es un problema no determinista. Para encontrar un determinado recurso, un nodo envía un mensaje de búsqueda a cada uno de los nodos a los que está conectado. Estos, a su vez, realizan la misma operación, de forma que la búsqueda se expande por una cantidad de nodos exponencial desde el nodo origen. Este proceso se repite tantas veces como especifica el parámetro *TTL*, que es, sin duda, el número máximo de saltos, y tiene por objetivo el de no saturar la red con dichas búsquedas. El problema es que si un recurso se encuentra en la red pero está a un número de saltos demasiado grande respecto al origen de la búsqueda, se supondrá que no existe, cuando la realidad es otra. En conclusión, para esta generación de redes p2p, la localización de los recursos no está garantizada.

JxTA [7] es una plataforma open-source p2p creada por Sun Microsystems en 2001. Esta plataforma define una serie de protocolos basados en XML que permiten que cualquier dispositivo conectado a la red intercambie mensajes y colabora con el resto de la red. JxTA es un marco de trabajo p2p actualmente disponible y que fue diseñado para soportar un alto rango de dispositivos (PCs, teléfonos móviles, PDA’s, etc) para comunicarse de forma descentralizada.

Los componentes de JxTA están organizados en tres niveles: Núcleo, Servicios y Aplicaciones. El nivel del Núcleo equivale al núcleo del Sistema Operativo y

proporciona los servicios básicos de seguridad, control de pipes entre peers, la gestión de grupos y la monitorización de los peers. El nivel de Servicios realiza las funciones de las librerías del Sistema Operativo, proporcionando por ejemplo servicios de indexación, búsquedas y compartición de ficheros. Por último el nivel Aplicación es equivalente al nivel de usuario en un Sistema Operativo.

Los protocolos de JxTA actuales son: Protocolo de localización de peer, Protocolo de resolución de peer, Protocolo de publicación de peer, protocolo de enlace de pipe, Protocolo de enrutado al punto final y Protocolo de coordinación.

Los peers JxTA crean una red virtual superpuesta que permite a un peer interaccionar con el resto de peers directamente. Además, cada recurso está identificado por un identificador único, con lo cual, cada peer puede cambiar su dirección de localización manteniendo mientras mantiene constante su número de identificación.

En cuanto a la arquitectura de la red, se puede decir que una red JxTA es una red overlay heterogénea, con conexiones no fiables y enrutado no determinista. Los peers pueden aparecer y desaparecer dinámicamente. Se distinguen cuatro tipos de peers: peers mínimos, peers simples, peers de rendevouz y relay peers. Estos dos últimos surgen para dar solución a los problemas de los firewalls.

En la siguiente generación de redes p2p se pretende dar solución al problema comentado, de forma que la localización de los recursos sea ya **determinista**. De esta forma, si un determinado recurso se encuentra en la red, seguro que lo encontraremos. Para tal propósito, ahora las redes se construyen de forma **estructurada**, típicamente basados en **formas geométricas**, como hipercubos, anillo o árbol. La idea es designar nodos particulares para que almacenen un contenido particular. Cuando un nodo quiere buscar un contenido, se dirige al nodo que se supone que lo tiene.

Los retos clave de estos sistemas son:

- Evitar los cuellos de botella que se pueden producir en determinados nodos y por tanto, se distribuyen las responsabilidades **de igual forma** entre los nodos existentes.
- Adaptarse a las continuas entradas y salidas (y también caídas) de nodos. Para ello, hay que dar responsabilidades a los nodos que entran y redistribuir las responsabilidades de los nodos que salen de la red.

La idea, por tanto, es parecida a la utilización de las clásicas *tablas de hash*, pero en este caso, los *buckets* de hash son los nodos físicos de la red. Este tipo de servicio p2p se denomina comúnmente como **Tabla de Hash Distribuida (DHT)**.

Las DHTs proporcionan mecanismos para añadir, borrar o localizar claves de hash. Se construyen por encima de redes overlay p2p eficientes, resistentes a fallos, y auto-organizativas. Encima de estas redes se pueden construir, además, otros servicios interesantes, como enrutamiento y localización de objetos descentralizada, servicios de *multicast* y *anycast* escalables, y las ya mencionadas DHTs. En particular, la abstracción DHT almacena pares clave-valor. El valor siempre se guarda en el nodo vivo de la red overlay al cual le pertenece el *hash* de la clave. Su estructura interna es la de un grafo (anillo, árbol, …), lo cual satisface la condición que el número de saltos necesarios para encontrar un determinado valor en la DHT es típicamente $O(\log n)$, donde n es el número total de nodos en el sistema.

Ejemplos existentes de este tipo de sistemas incluyen Chord [8], Pastry [9] o Kademlia [10]. Comentar que las redes overlay p2p descentralizadas reciben también

el nombre de redes o sustratos **KBR** (*Key Based Routing*), ya que el encaminamiento de los mensajes se realiza de acuerdo con las claves o identificadores de los nodos.

3. Estado del arte

3.1 Aproximaciones sobre desestructurado

A continuación vamos a detallar varias alternativas en p2p desestructurado donde la gran mayoría de trabajos realizados utilizan el proyecto JxTA.

Podemos encontrar propuestas como OPAL [11][12], un sistema multi-agente que sigue las especificaciones de la FIPA Abstract Architecture (AA) y utiliza JxTA para el despliegue y descubrimiento dinámico de agentes sobre p2p. Estos trabajos muestran como suplir las carencias en las comunicaciones FIPA de los agentes, por las posibilidades de la comunicación p2p. Las limitaciones solventadas del servicio de transporte especificado en la FIPA AA consisten en que tan sólo permite la comunicación con mensajes ACL entre dos puntos. De este modo, se posibilitan otros mecanismos en la comunicación de los agentes, como por ejemplo el descubrimiento dinámico y el envío de mensajes uno a muchos.

Por último destacamos UbiMAS [13] un sistema de agentes móviles para entornos ubicuos –véase Figura 2. Este sistema está basado nuevamente en JxTA, y lo utiliza como infraestructura de comunicaciones, justificando esta elección describiendo un sistema ubicuo como un entorno muy dinámico donde es necesario un sistema distribuido descentralizado. Nuevamente nos encontramos con que la red p2p es utilizada como sistema de comunicaciones, en este caso, para envío unidireccional, bidireccional y propagación de mensajes uno a muchos. Una de sus aportaciones especialmente interesante es la utilización de la red p2p para la migración de agentes.

3.2 Aproximaciones sobre p2p estructurado

En este apartado veremos diferentes trabajos que utilizan diferentes protocolos p2p estructurados para construir sistemas distribuidos de agentes móviles.

Primero destacar que existen diferentes marcos de trabajo de agentes, como por ejemplo Extensible Mobile Agent Architecture (EMAA), Decentralised Information Ecosystem Technologies Agents Platform (DIET), Cognitive Agent Architecture (Cougaar), Java Agent Development Framework (Jade) y otros tantos. Estos marcos han sido los pilares de diferentes esfuerzos de investigación, entre los que destacamos los dos siguientes.

En [16] el CAN (Content Addressable Network) fue desarrollado con agentes móviles, utilizando EMAA como base, y además manteniendo unos costes de comunicación muy parecidos a los del propio CAN. En este caso, cada nodo CAN es implementado como un agente, llamado DHT Server, ofreciendo la misma funcionalidad de un nodo DHT. Además, sustituye los mensajes por agentes ligeros

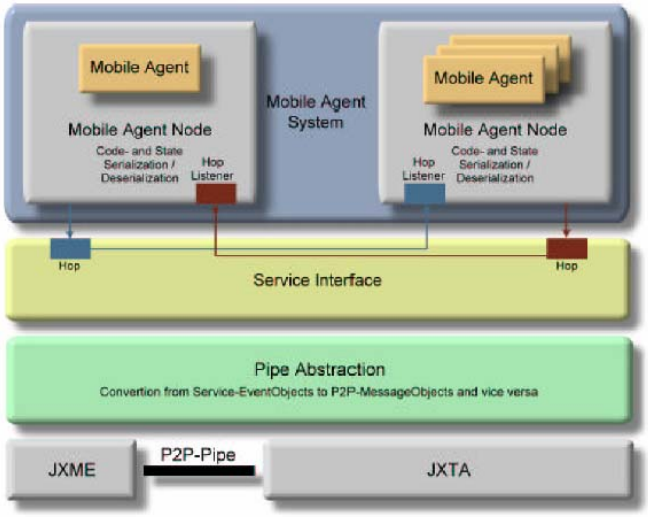

Figura 2. Arquitectura UbiMAS

móviles, para así responder a necesidades de enrutamiento de la misma red. Finalmente aporta un segundo desarrollo del sistema para entornos en que los agentes sean altamente móviles.

En [17] han desarrollado un SWAN (Small World Adaptative Networks) con DIET como infraestructura. DIET es un marco que ofrece un sistema de agentes altamente ligeros, y que se ejecutan (obtienen un hilo de ejecución) solamente cuando tienen carga de trabajo. De nuevo, cada nodo SWAN es implementado como un agente, llamado ServiceProvider. Aquí, por el contrario, consiguen una optimización del número de conexiones establecidas, debido a que las centralizan en un agente llamado SwanEngineManager, existente en cada nodo físico. Los autores, utilizan las capacidades de DIET para que SwanEngineManager se ponga en contacto con el ServiceProvider concreto y así cumplir los objetivos de los agentes. De este modo, obtienen una gran cantidad de nodos SWAN por nodo físico. Como en el caso anterior, los mensajes son nuevamente sustituidos por agentes.

3.3 Aplicaciones

Como aplicaciones que hacen uso de este tipo de sistemas encontramos ejemplos como Photo Agent [14] o Personal Data Backbone [15].

Photo Agent proporciona una forma de gestionar y compartir fotografías digitales, sin ningún tipo de manipulación de ficheros explícita ni comunicación de datos. Los agentes comparten fotografías anónimamente y de forma pro-activa, de forma que los usuarios pueden simplemente especificar qué imágenes quieren compartir, con quién quieren compartirlas y sin tener en cuenta como son distribuidas y buscadas. El prototipo Photo Agent utiliza JxTA para la comunicación p2p dando soporte a una compartición eficiente en un entorno distribuido.

Personal Data Backbone (PDB) es un servicio similar al servicio Passport de Microsoft, pero encima de una plataforma p2p llamada MyP2P. PDB permite la adquisición de datos personales flexible y segura en un entorno espontáneo formado por todos los usuarios. El principal objetivo de esta aplicación es devolver el control de los datos personales a su dueño. Gracias al uso de tecnología p2p, los usuarios pueden colaborar entre ellos y establecer servicios sin recurrir a ninguna entidad central o corporación, eliminando conceptos como privacidad, seguridad o monopolio. PDB está implementado por agentes móviles. La movilidad permite a éstos replicarse y migrar a través de la red p2p para evitar fallos de red y realizar balanceo de carga.

4. Discusión

Denotamos una similitud entre los entornos p2p y los sistemas multi-agente. Ambos se definen como auto-organizables, los cuales operan en entornos potencialmente dinámicos con frecuentes entradas y salidas, bien de nodos o bien de agentes, respectivamente. Uno de sus objetivos principales es beneficiarse de los recursos distribuidos compartidos (como por ejemplo, el porcentaje de CPU, ancho de banda o espacio de almacenamiento) entre las diferentes máquinas que conforman la red.

La tecnología p2p aporta mejor rendimiento, escalabilidad, disponibilidad y fiabilidad, comparado con los sistemas tradicionales centralizados. Por otro lado, los sistemas multi-agente están basados generalmente en sistemas distribuidos tradicionales y por tanto migrar hacia entornos p2p es una buena alternativa para compensar las dificultades de los sistemas dinámicos, como los sistemas distribuidos de gran escala, ubicuos o móviles.

Conjuntamente, ambas partes obtienen beneficios de tal simbiosis. Por un lado, los sistemas multi-agente se benefician de los mecanismos p2p de conectividad, tolerancia a fallos y balanceo de carga para mejorar la calidad de sus servicios. Y viceversa, según argumentan los distintos autores [16][17], el p2p se beneficia por la cómoda y sencilla implementación de los nodos mediante agentes, gracias a la alta modularidad ofrecida en los sistemas multi-agente. Además, estos agentes habitualmente son mucho más ligeros, por lo que este tipo de desarrollo es apto para ser desplegado en dispositivos móviles.

En este sentido, y adentrándonos en el paradigma de los agentes, se cambia la perspectiva sobre el modelo de red overlay, haciendo que cada agente personifique a un nodo de la misma. Al mismo tiempo, otros agentes, esta vez mucho más ligeros, pueden representar a los mismos mensajes de la red overlay y así encapsular la lógica de enrutamiento.

A todo esto, las aplicaciones multi-agente tradicionales se impregnan de la filosofía p2p, conllevando la compartición y libre acceso a los recursos, pudiendo activar políticas para evitar el free-riding como en [18]. A su vez, estas aplicaciones se benefician de la gestión de la información de forma inteligente, de acuerdo con el propio objetivo, de una forma transparente para el usuario final y eficientemente a nivel de arquitectura.

Soluciones tradicionales en sistemas distribuidos, y concretamente en sistemas multi-agente, son totalmente válidas para entornos reducidos o redes de área local. En cambio, cuando este tipo de soluciones resultan insuficientes, ya sea por temas de escalabilidad o de dinamismo, necesitamos soluciones como la que nos aportan la tecnologías p2p.

4. Conclusiones

En este artículo hemos podido comprobar que tanto los sistemas multi-agente y los entornos p2p ofrecen por separado propiedades atractivas. Así, los sistemas multi-agente ofrecen movilidad de agentes, servicio de directorio –tanto de páginas amarillas, como de páginas blancas- y una alta modularidad para el desarrollo de los propios agentes, como características más destacables. Por otro lado, los entornos p2p, y sobretodo los estructurados, ofrecen una conectividad, resistencia a fallos, balanceo de carga y mecanismos de localización de información completamente determinista.

La unión de ambos ámbitos permite obtener un producto que une lo mejor de cada uno. Estos beneficios difieren dependiendo del enfoque realizado:

- En un escenario, los agentes hacen uso y se apoyan sobre una infraestructura p2p, donde se benefician de un nuevo motor de comunicaciones, ofrecido por la propia arquitectura p2p subyacente, habilitando además una localización de la información determinista y eficiente. Así por ejemplo, veíamos como los agentes utilizaban dichas propiedades de JxTA en [11][12][13].
- En el otro tipo de escenario, los entornos p2p se pueden aprovechar de un desarrollo mediante agentes altamente modular y ligero, siempre bajo un sistema multi-agente y adoptando el mismo protocolo p2p como base de sus comunicaciones –véase [16][17]. Esto permite que este tipo de redes pueda llegar a otro tipo de dispositivos, como teléfonos móviles, PDAs, etc., además de que al consistir cada nodo en agentes pro-activos, pueden ofrecer una mayor calidad en sus servicios.

Referencias

1. Jennings, N.R. “Agent-oriented software engineering”. *In Procedding of the 12th Internacional Conference on Industrial and Engineering Application of AI*, 1999.
2. Especificaciones FIPA http://www.fipa.org
3. Agentcities Project, http://www.agentcities.org
4. Penserini, L. et al. “Modeling and Evaluating Cooperation Strategies in P2P Agent Systems”. *In Proceedings of 1st International Workshop on Agents and Peer-to-Peer Computing* (AP2PC 2002), July 2002, Bologna, Italy.
5. Montresor, A., Meling, H., Babaoglu, O. “Messor: Load-Balancing through a Swarm of Autonomous Agents”. *In Proceedings of the 1st Workshop on Agent and Peer-to-Peer Systems*, July 2002.
6. Wooldridge, M., “An introduction to multiagent systems”. John Wiley Ed., 2002.

7. Project JxTA: http://www.jxta.org/
8. I. Stoica, R. Morris et al. “Chord: A Scalable Peer-To-Peer Lookup Service for Internet Applications”. *Proc. ACM SIGCOMM*, Aug. 2001.
9. Rowstron, A., Druschel, P. “Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems”. *In Proc. 18th IFIP/ACM Internacional Conference on Distributed Systems Platforms*, pp. 329-350, 2001.
10. Maymounkov, P., Mazières, D. “Kademlia: A Peer-to-peer Information System Base on the XOR Metric”. *In Proc. 1st International Workshop on Peer-to-peer Systems* (IPTPS’02), MIT, March 2002.
11. Purvis, M. et al. “Multi-agent System Technology for P2P Applications on Small Portable Devices”. *Lecture Notes in Computer Science*, Volume 3601, 2005, pp 153-160.
12. Purvis, M. et al. “Multi-agent Interaction Technology for Peer-to-Peer Computing in Electronic Trading Environments”. *Lecture Notes in Artificial Intelligence (LNAI) 3157*, Heidelberg Germany (2004) 625-634, ISBN 3-540-22817-9.
13. Bagci, F., Petzold, J., Trumler, W., Ungerer, T. “Ubiquitous Mobile Agent System in a P2P-Network”. *Workshop at the Fifth Annual Conference on Ubiquitous Computing*, October 12-15, 2003, Seattle, USA
14. Hsu J.Y. et al. “Photo Agent: An Agent-Based P2P Sharing System”. *Proc. Third International Workshop AP2PC 2004*, July 19, 2004, New York, USA.
15. Cha, S.C., Joung, Y.J., Lue, Y.E. “A Passport-Like Service over an Agent-based Peer-to-Peer Network”. *Proc. 2nd International Workshop on Agents and Peer-to-Peer Computing (AP2PC), Lecture Notes in Computer Science Vol. 2872*, July 14, 2003, Melbourne, Australia.
16. Thomas, M., Regli, W. “Peer-to-Peer Data Lookup for Multi-agent Systems”. *In LNAI* 3601, pp. 201-212, 2005.
17. Bonsma, E., Hoile, C. “A distributed implementation of the SWAN peer-to-peer look-up system using mobile agents”. *1st International Workshop on Agents and Peer-to-Peer Computing (AP2PC 2002)*, AAMAS2002, July 2002, Bologna, Italy.
18. Yu, B., Singh, M.P. “Incentive mechanisms for agentbased peer-to-peer systems”. *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS2003, July 2003, Melbourne, Australia