Virtual Knowledge Communities for Corporate Knowledge Issues

Pierre Maret[1], Mark Hammond[2,3], and Jacques Calmet[3]

[1] INSA de Lyon, LIRIS (CNRS FRE 2672),
20, avenue Albert Einstein. F-69621 Villeurbanne, France
pierre.maret@insa-lyon.fr

[2] Department of Computing, Imperial College London,
180 Queen's Gate, SW7 2AZ, London, England
mrh00@doc.ic.ac.uk

[3] University of Karlsruhe, IAKS,
Am Fasanengarten 5, D-76131 Karlsruhe, Germany
calmet@ira.uka.de

Abstract. Corporate knowledge consists both of information that is available throughout a company and of information technology frameworks and paradigms. Considering an enterprise as a distributed computational paradigm, multi-agent systems can be proposed to address knowledge management issues within a company. We consider in this paper a new approach for corporate knowledge based on the agent oriented abstraction paradigm. This paradigm provides a high level of abstraction. We investigate here the concept of virtual knowledge communities, which is a convenient concept for addressing dynamical distributed knowledge management. It allows improved simulation and support for knowledge management processes, and therefore to innovate with new methods in this field. Our approach is well-suited for instance to filter the amount of knowledge that is transmitted throughout a company.

1 Introduction

Knowledge management (KM) is a critical issue within knowledge-intensive organizations [1]. Corporate knowledge consists of both information that is available throughout a company and of information technology frameworks and paradigms. Most approaches to knowledge management remain mainly founded on centralization and objectivity. They are generally based on the database paradigm. Examples of such systems are numerous. However centralization and objectivity appear incompatible with the very nature of knowledge. Bonifacio [2] criticizes most current knowledge management systems where "all perspectival aspects of knowledge should be eliminated in favor of an objective and general representation of knowledge". In [2] the authors propose a peer-to-peer architecture which emphasizes distributed knowledge management and knowledge nodes. Kornfeld [3] claimed years ago that diversity and concurrency of (scientific) communities are essential to their progress. We argue that approaches to knowledge

M.-P. Gleizes, A. Omicini, and F. Zambonelli (Eds.): ESAW 2004, LNAI 3451, pp. 33–44, 2005.

management must maintain compliance with the very nature of knowledge that is subjective, distributed and contextual.

Multiagent systems (MAS) have been introduced as a methodology to address distributed computing problems in artificial intelligence. They have evolved as a management methodology and a software engineering design principles leading to object-oriented-like systems. The main software agents developed in the KM area implement functionalities [4] such as extraction of knowledge from document bases [5, 6, 7], user's profile identification [5], and knowledge targeted diffusion [8, 9]. When adequately considering most of these tools, we observe that they solely cover single issues of corporate knowledge and that they do not propose a broad and generic view on corporate knowledge. Indeed, the level of abstraction remains mostly insufficient and the broad scope of available knowledge is not considered appropriately.

Considering an enterprise as a distributed computational paradigm, multi-agent systems can be proposed to address knowledge management issues within a company. Processes within the company tend to make agents produce and exchange knowledge with each other. This constitutes a key issue addressing the domain of agent societies. Examples of such systems are numerous (for instance [10], [11]) Within multi-agent societies, a balanced articulation must be found between organizational control and autonomous social behavior of agents. The works of Lesser concerning organizational design confront the organizational control and emergent organization [12]. Bradshaw proposes a framework to specify, manage and enforce agent behavior using DAML-based policies [13]. Exploring security issues inside open organizations, Omicini [14] calls for a systemic vision of MAS, explicitly accounting for social issues (social intelligence) as opposed to focusing on individual agent's intelligence. Calmet [15] proposed the liberal approach for agent communities, based on the work of Weber in sociology. Weber considers a society as the result of the actions of its actors. This liberal point of view has been taken to describe the Agent Oriented Abstraction [16] where agents are seen as made of two components: the knowledge component and a decision making system. The concept of knowledge management into a society of agents becomes fully meaningful in this context [17].

Among the various approaches to corporate knowledge, the one we adopt considers that corporate knowledge consists of *i)* the overall knowledge detained by agents and *ii)* the ability of agents to cooperate with each other for achieving their goals. Corporate knowledge encompasses then any piece of information available in an enterprise from the technology required to design and produce goods to management decision policy through human relations and internal or external communication. The decision mechanism encompasses (but is not limited to) the behavior of agents regarding domains of interests and knowledge exchanges.

In this paper we investigate the concept of virtual knowledge communities, which is a convenient concept for addressing dynamical distributed knowledge management. It is well-suited to filter the amount of knowledge that is transmitted throughout a company. The concept of community (of interest or of practice) is central in the knowledge management area. Examples are [18] and [19]. It seems that this concept has hardly been addressed in the framework of agent societies. We notice however that, like individuals and computer systems (and even internet nodes), agents are autonomous and heterogeneous. Moreover, relative to traditional approaches, agent-based modeling introduces openness and dynamicity, which is highly compatible with knowledge processes. Agent societies therefore constitute the right level of abstraction for modeling and engineering corporate knowledge systems which are complex articulated systems. The Agent Oriented Abstraction provides a high level of abstraction. E-business and enterprise-wide applications could therefore significantly gain from our approach for corporate knowledge. It allows improved simulation and support for knowledge management processes, and therefore allows innovation with new methods in this field.

This paper is organized as follow: section 2 introduces briefly key concepts for the management of knowledge of agents: ontology, knowledge clusters and instances. Then in section 3 we describe the concept of virtual knowledge community. Section 4 concerns the implementation of a Jade-based prototype system. Section 5 consists of a discussion and of some concluding remarks. We illustrate some of the concepts we do use with a purposely almost trivial example. A more significant example taken from corporate knowledge would not fit in the format of the paper.

2 Agents, Ontology and Knowledge

Agents are active objects with the ability to perceive, to reason and to act. In addition, it is assumed that agents have explicitly represented knowledge and communication ability [20]. For our purposes, we discuss hereafter three key notions used in our approach: ontology, knowledge cluster and knowledge instances.

The knowledge of an agent is represented in the vocabulary of an *ontology*. Agents are related to an ontology to talk and reason about things and facts. We consider a high-level ontology for frame-based description of knowledge. Knowledge is described in terms of Predicates, Concepts, Actions. This is compliant with the ACL-FIPA specifications. Attributes related to these terms are such as name, slots, arguments. We call an instance of the ontology a *knowledge cluster*. A knowledge cluster represents some structured knowledge. Basic operators on knowledge clusters can be defined, such as addition, filtering, search, is-sub-part-of, comparison. Knowledge clusters can be defined recursively. A knowledge cluster may be related to the overall knowledge of an agent, a specific task or to a given topic. A simple example of a knowledge cluster related to the domain of software maintenance is given hereafter (with a simplified syntax).

- Concept
  - name: Software
  - slots: Name-Software, Version

- Concept
  - name: Incident
  - slot: Description-Incident
- Concept
  - name: Location
  - slot: Description-Location
- Predicate
  - name: IncidentDecription
  - arguments: Software, Incident, Location

We call *knowledge instances* instances of objects defined into the knowledge clusters. An example of knowledge instances of an agent is given hereafter.

- Software : Jade("3.1")
- IncidentDescription : (Jade("3.1") , "System refuses to...", "While ...")

We assume that agent's knowledge consists of knowledge clusters and instances. Under this assumption, an agent's knowledge varies from agent to agent, which is fully compliant with individuals' knowledge. Moreover, while processing tasks, agents use, produce and acquire knowledge. Thus, knowledge can not be uniquely considered at design time (inherent knowledge). So, we assume that agent's knowledge evolves during the agent's life, thanks to individual activity and exchanges within the agent society. These assumptions are trivial regarding knowledge instances, but they are not trivial regarding knowledge clusters.

3 Virtual Knowledge Communities

We have defined corporate knowledge as the overall knowledge detained by agents within a system and their ability to cooperate with each other in order to achieve their goals. We introduce now the concept of a virtual knowledge community as a means for agents to share knowledge about a topic. The description hereafter aims at equipping agents with a layer through which they have the ability to act as members of knowledge communities. Agents are in charge of tasks within the society and they are provided with knowledge and decision mechanisms (agent oriented abstraction). Membership in a knowledge community does not replace the intrinsic goal of an agent for which it was introduced into the system. The concept of virtual knowledge community aims to increase the efficiency with which information is made available throughout an organization. This leads firstly to a more efficient achievement of the goals assigned to the agents, and secondly, provides a learning or data-mining mechanism. This mechanism can be proactive or reactive depending on the circumstances. Business activities such as e-business or virtual enterprises are usually dynamic processes. Thus, agents ought to be able to create, join, feed, mediate and use knowledge communities dynamically. Also, agents ought to increase their knowledge.

3.1 Modeling Virtual Knowledge Communities

A virtual knowledge community is composed of a topic, members and a space for exchanging messages. We previously described the knowledge of agents in terms of knowledge cluster and instances. Thus, the topic of a knowledge community is described similarly. Agents participating in a community are supposed to send and access messages related to the community topic. We assume that the locations of knowledge exchanges are message buffers where agents' contributions are posted and accessed. The content of messages is composed of a knowledge cluster and a set of knowledge instances.

3.2 Knowledge Community Processes

Agents' actions related to knowledge communities are the following: initiate and terminate a community, join a community and exchange knowledge. Community initiation can be done by any agent which becomes then the community leader. Initiation consists of creating a topic and a message buffer, and of advertising about the community. Advertising is not done through a specific shared feature (for instance the Federation Directory Service in the approch of [10]), which would introduce some centralization. Initiation is done thanks to a broadcast message that each agent of the system is able to send through the system it belongs to. Advertising consists then in broadcasting a community initiation cluster and a buffer reference dedicated to messages' exchanges related to this community. The community initiation cluster is necessary a sub-part of the initiator's cluster. It can also contain instances. An example is given hereafter, continuing the simple previous example:

- Concept
  - name: Software
  - slots: Name-Software, Version

Community termination consists of erasing the community message buffer created during initiation. An agent considers joining communities in receiving and in evaluating the initiation message and more specifically the posted knowledge clusters. An agent may be willing to join a community when the intersection between a community cluster and its own cluster is not empty. It sends then a so-called *join message* to the leader. A negotiation process is initiated with the community leader, which evaluates the candidate and the conflicts that may arise [21]. The role of the leader can be compared to the negotiator introduced in the approach described in [11].

Exchanging knowledge within a community consists of posting and accessing request and inform messages. They are sent to the community buffer and contain a knowledge cluster or instances. An example of an inform message related to the previous community cluster is given hereafter. In this example, the sender agent proposes specializations of the concept Software.

- Concept
  - superconcept: Software
  - name: Proprietary-software
  - slots: Price
- Concept
  - superconcept: Software
  - name : Open-source-software

3.3 Social Behavior as Agent's Knowledge

In an open and moving environment, it can not be assumed that agents use the same terms for the same real world objects. Also, it is not possible to believe that agents will succeed in exchanging knowledge without a *minimum semantic effort*. However this effort and this ability must remain controlled by the agent itself. Agent's autonomy is then preserved. We assume that this ability is itself knowledge. It can take any format in the framework of the knowledge cluster: reference to any accessible "normalized" clusters, implementation of an own matching table or function... Semantic interpretation of the content of any knowledge clusters is then left to the capabilities and features of each agent. It must be outlined that this approach is a design decision for our model. One could select a more generic approach, using for instance KOMET [22] where a mediator system extracts the relevant knowledge from agents in a semantically sound way. The following example illustrates the basic expression of two knowledge matching abilities. The predicate "convert-into" enables the agent to match "SW" with "Software". The action "Convert" expresses the ability for the agent to initiate conversion functions for clusters.

- Predicate
  - name: Convert-into
  - arguments: Software, SW
- Action
  - name: Convert
  - arguments: conversion-function, cluster-in, cluster-out

Regulation of the knowledge community is also a societal issue. Each agent may define policies specifying their behavior regarding knowledge communities: what are the circumstances in which they would choose create, delete, join a community, send and access knowledge. Of course, this policy can be different for every agent and a single agent may implement several policies. For instance, it can range from joining a community if a given cluster or a given instance appears into the community cluster, to joining a community when the intersection with the community cluster reaches a given percentage, etc. The initiator of a community can also perform regulation regarding the use of its community. This may prevent any inappropriate functioning due to (purposely or not) unfair agents. The implemented policy can be very liberal or it can filter or moderate contributions (considering for instance the message sender or the message content). Again, we assume that agent's potential behaviors are part of its knowledge.

The following example sketches behaviors of agents when joining a community. Instances of these behaviors follow. In the example, the agent "knows" three behaviors for deciding to join a community: when the concept "Software" is present into the community cluster, when the instance "Jade(3.1)" is present into the community cluster, or when the community cluster is covered by more than p% of the agent's cluster. One, both or none of the behaviors can be activated.

- Action
  - name: Join-on-cluster
  - arguments: Cluster
- Action
  - name: Join-on-knowledge-instance
  - arguments: Instance
- Action
  - name: Join-on-cluster-intersection-rate
  - arguments: value

- Knowledge instances
  - Join-on-cluster (concept(Software))
  - Join-on-knowledge-instance (Jade("3.1"))
  - Join-on-cluster-intersection-rate (p)

Exchanges in-between agents suppose the existence of communication mechanisms. Again, in an open and generic approach, it can not be considered that agents always use a same and unique communication mechanism. For instance, knowledge exchange within a community can be carried out in a negotiation or in a cooperation mode. Thus, we consider that communication mechanisms belongs to agent's knowledge. Each agent can then possess several communication mechanisms and it selects the appropriate one when creating or joining a community. We are aware that it is easy to argue that many of the implicit societal problems involved in this model could be better discussed. However, we set our approach in a well-defined model of agent society, as previously quoted, and any discussion is more relevant for [15] than for this paper.

4 Implementation of a Prototype System

4.1 Aims of the System

The aim of the prototype is to design and create a working system, in which agents with heterogeneous personal ontologies can create, destroy, leave and join communities in order to share knowledge clusters and instances with each other. Emphasis is on the sharing of knowledge, the creation of mappings between personal concepts and normalized concepts and on establish the mechanisms with which agents can dynamically move between communities and update their personal knowledge.

To summarize what is provided within the prototype we list the inputs to the system, the decisions a decision making system makes on the agents behalf and the results we observe. The prototype allows the creation of multiple agents on different machines, each with a given personal ontology, given personal knowledge instances, a set of given normalized ontologies and a set of mappings for each normalized ontology as inputs. Each one of the sets of mappings details any mappings an agent already has from its personal ontology predicates and concepts to the normalized ones. The agents are allowed to join and leave the system at any given time, in any given order. The agents are not hard-coded with any knowledge about existing communities to join (except for the global 'community of communities') or other agents to communicate with.

In light of the agent-oriented abstraction approach previously cited, agents are provided with a decision making system linked to their knowledge component. The decision making system is involved for instance when an agent creates a new community. Here it specifies which one of the normalized ontologies the agent should use to create this community. The decision making system is also involved whenever the agent encounters a new concept for which it does not already possess a mapping. Here the decision making consists in determining which concept, if any, to map to. Finally, the decision making system is involved when adding goals to an agent (for goal driven agents) or when choosing which part of its knowledge it should share (for altruistic agents). For all of these types of decisions, we implemented a human-based approach. An automated approach could be designed with no conceptual changes.

The results of the prototype system take the form of a user readable interactive output, showing the actions the agent is undertaking while it performs them. Thus, we can analyse the behavior of the agents. This is quite useful since the actions the agents make depend on other agents in the system as well as on themselves. This way, the actions decided by an agent may be non predictable in advance. We observe that agents can update their personal ontology, personal knowledge instances and personal mappings when appropriate.

4.2 System Implementation

JADE (Java Agent Development Framework), a Java based software development framework that fully conforms to FIPA standards for intelligent agents [23], was chosen to implement the prototype system. The most important aspect of JADE that makes it useful for the prototype system is its support for message passing and user defined ontologies. Agents are created by simply extending the `jade.core.Agent` class. This makes the development of different kinds of agents very straightforward.

Classes. The main class of our system is the `CommunityAgent` class. It is an abstract class that all different types of agents must extend. We propose two extensions of the `CommunityAgent`: the `IndividualAgent` and the `SocialAgent`. These two different types of agent have different behaviors. An individualistic agent instigates or joins a community because it needs knowledge from a certain domain, and will leave a community when it has got what it wants from the community or if it is not getting enough from it (e.g. it has been in the commu-

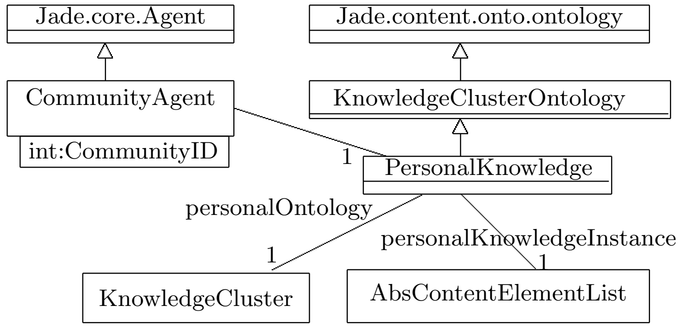

Fig. 1. Simplified class diagram (Unified Modeling Language formalism)

nity too long with no benefit). Its goals are chosen through its decision making mechanism, in the form of concepts that the agent wants to add to its knowledge. An individualistic leader will kill a community once it has completed the knowledge it requires. A *sociable agent* instigates a community purely because it has nothing else to do and for the good of all. The community will only die if everyone leaves the community. Sociable agents join every community that concern them (whose community cluster overlaps his own domain of knowledge) and never leave active communities where they may still be able to offer something. From a security point of view it is difficult to distinguish in this model between social agents and intruding passive agents. However, since JADE is not a secure platform as a forthcoming publication will show, this is not a crucial comment.

Each CommunityAgent has exactly one `PersonalKnowledge` object. The `PersonalKnowledge` class extends the `KnowledgeClusterOntology` class which represents an ontology that describes the classes of objects that make up knowledge clusters, namely concepts and predicates. One might say that the KnowledgeClusterOntology is a meta-ontology that describes knowledge clusters. The personal knowledge consists of both personal ontology (`KnowledgeCluster` class), and personal knowledge instances (`AbsContentElementList` class). The class diagram can be seen in Fig.1 where '1' means 'contains one attribute of class...'.

Agent Communications. Any communication in the system is done through message passing. Messages sent are of type `jade.lang.acl.ACLMessage`, `inform` or `request`. There are three types of *request* messages that can be sent by `CommunityAgent`'s: `joinCommunity`, `readFromBuffer` (request to read from the community buffer) and `KnowledgeCluster` (an agent wishes to increase a cluster with both ontological and instance knowledge). There are four types of *inform* messages: *inform* content message containing a `knowledgeCluster` or `knowledge instances` that are being shared by an agent, `updateRole` (from leaders to members following an update of an agent's role), `leavingCommunity` and `communityDead`.

System run. Let us take the example of two agents (`Mark` and `Jose`), instances of the `IndividualAgent` class. They have different personal names for all the concepts in their ontology (for instance concepts *SW* and *Software*), but they have both a reference to the same normalized ontology. `Jose` already has the mappings from his personal concepts to the ones in the normalized ontology contained in his mapping file (*Program* stands for *SW*). `Mark` does not. Moreover, `Jose` has ontological knowledge that `Mark` does not have (*OpenSourceSoftware* is a *SW*). `Jose` has instances of this knowledge also (*JADE* is instance of *OpenSourceSoftware*). While performing the test run, we see that through knowledge communities and agent's goals, knowledge sharing takes place. `Mark` can gain both ontological knowledge, as well as instances, even though `Jose`'s concepts are not included in the normalized ontology, and `Mark` and `Jose` have different names for all the terms they share in common. Agents can then dynamically collaborate, pool their knowledge and gain information, although they did not build it into their initial system.

5 Discussion and Conclusion

Considering an enterprise as a distributed computational paradigm, we proposed in this paper a generalization of corporate knowledge based on the agent paradigm. We used the Agent Oriented Abstraction paradigm, which has been proposed to describe the concept of agents in a fully generic way. It provides a high level of abstraction and considers that agents consist of knowledge and decision mechanisms. This abstraction mechanism leads to practical applications for corporate knowledge.

We described the concept of virtual knowledge community to model instances of corporate knowledge. This concept can be useful in real applications as well as in theoretical researches. The approach extends the field of knowledge management to societies of agents. Virtual knowledge communities constitute a nice framework for addressing and testing various aspects of corporate knowledge, especially knowledge modeling, autonomy of actors and exchange processes. Our work now provides the possibility to simulate and support knowledge management processes more appropriately and therefore to innovate with new methods in this field. Virtual knowledge communities also constitute a non trivial domain for applying and testing agents' key properties such as autonomy, heterogeneity, openness and dynamicity.

The virtual knowledge community concept enables agents to diffuse and to extend their knowledge within a society of agents. Knowledge is not limited to a specific "domain of interest", rather it is considered that knowledge owned by agents also comprises knowledge about communication mechanisms, heterogeneity resolution, and societal behavior. Knowledge exchanges are carried out within a set of agents concerned about a common topic. Exchanging knowledge instances is common and necessitates that the agents possess roughly the same knowledge structure. Thanks to the generalized approach of agent's knowledge, including knowledge matching ability in particular, agents have the ability to share knowledge structures while preserving autonomy. Future works will consider improving the structure and the modeling of agent's knowledge. The concept of knowledge annotations has been introduced in the agent oriented abstraction paradigm to structure the knowledge. However, support for such a paradigm can not be found in traditional agent platforms.

A societal issue is to define well adapted agent policies enabling the emergence of pertinent and fruitful knowledge exchanges. Society's organization may be impacted by the emergent organizations arising from knowledge societies. This is a well known issue in sociology and in human resource management. In this paper, we did not mention the security issue, which is also a societal issue. Since the system is open, it requires security policies to ensure only trustworthy agents can access the communities, and to prevent malicious attacks from untrustworthy agents. Work on this issue is in progress [24].

Practical applications of our approach are numerous. Agents can consist of intelligent knowledge assistants, as described in [25], [26], [23]. The interest of our approach compared to [23] is that we do not consider a unique description of the domain of interest of agents. The approach is also well suited to filter the amount of knowledge that is transmitted throughout a company. Indeed, knowledge broadcast is a high-relevance issue within companies. Another application concerns mobile systems. In this area, agents typically meet others ones that were not designed in coordination. Through knowledge communities, an agent could communicate with others to gain knowledge that is of common interest. In a traditional system, an analyst may need to redesign at least part of the existing system to accommodate this new source of knowledge.

References

1. Fisher, G., Ostwald, J.: Knowledge management: problems, promises, realities and challenges. Intelligent Systems **16** (2001) 60–72
2. Bonifacio, M., Bouquet, P., Mameli, G., Nori, M.: Peer-mediated distributed knowledge management. In: American Association for Artificial Intelligence Spring Symposium Technical Report SS-03-01, 1–8, ISBN 1-57735-078-9. (2003)
3. Kornfeld, W.A., Hewitt, C.: The scientific community metaphor. IEEE Transaction on systems, man and cybernetics **11** (1981) 24–33
4. van Elst, L., Dignum, V., Abecker, A.: Towards agent-mediated knowledge management. In: Agent-Mediated Knowledge Management. Volume 2926 of LNAI., Springer-Verlag (2003)
5. CoMMA: Available electronically at http://www.si.fr.atosorigin.com/sophia/comma/htm/homepage.htm (2004)
6. J.-R. Chen, S.R.W., Wragg, S.D.: A distributed multi-agent system for collaborative information management and sharing. In: Proc. of the 9th ACM International Conference on Information and Knowledge Management. (2000) 382–388
7. FRODO: Home page (2004) Available electronically at http://www.dfki.uni-kl.de/frodo/.
8. Campiello: Home page (2004) Available electronically at http://klee.cootech.disco.unimib.it/~campiello/.
9. Benjamin, V.R.: Skills management in knowledge-intensive organizations. In: Proc. of the EKAW 2002 International Conference. Volume 2473 of LNCS., Springer Verlag (2002) 80–95

10. Cenk-Erdur, R., Dikenelli, O., Seylan, I., Gurcan, O.: Semantically federating multi-agent organizations. In: Engineering Societies in the Agents World (ESAW'04), 20-22 October 2004. (2004)
11. Sibertin-Blanc, C., Hameurlain, N.: Participation components for holding roles in mas protocols. In: Engineering Societies in the Agents World (ESAW'04) 20-22 October 2004, Universit Paul Sabatier, Toulouse, France. (2004)
12. Sims, M., Goldman, C., Lesser, V.: Self-organization through bottom-up coalition formation. In: Proceedings of the second international joint conference on Autonomous agents and multiagent systems, Melbourne, Australia. ACM Press. (2003) 867–874
13. Bradshaw, J.: Representation and reasoning for DAML-based policy and domain services in kaos and nomads. In: Proceedings of the second international joint conference on Autonomous agents and multiagent systems, ACM Press (2003) 835–842
14. Omicini, A., Ricci, A.: Integrating organisation within a mas coordination infrastructure. In: Engineering Societies in the Agents World (ESAW'03), 29-31 October 2003, Imperial College, London, UK. (2003) 165–172
15. Calmet, J., Daemi, A., Endsuleit, R., Mie, T.: A liberal approach to openness in societies of agents. In: Engineering Societies in the Agents World (ESAW'03), London, UK, 29-31 October 2003. Volume 3071 of LNAI., Springer-Verlag (2003) 81–92
16. Calmet, J., Maret, P., Endsuleit, R.: Agent oriented abstraction. Royal Academy of Sciences Journal. Special Issue on Symbolic Computation in Logic and Artificial Intelligence **98** (2004)
17. Maret, P., Calmet, J.: Corporate knowledge in cyberworlds. IEICE Transaction on Information and Systems. Special Issue on Cyberworlds (2005)
18. Bonifacio, M., Bouquet, P., Cuel, R.: Knowledge nodes: the building blocks of a distributed approach to knowledge management. Journal of Universal Computer Science **8** (2002) 652–661
19. Gordon, M., Fan, W., Rafaeli, S., Wu, H., Farag, N.: The architecture of commknowledge. combining link structure and user actions to support an online community. Int. J. Electronic Business **1** (2003) 69–82
20. Weiss: Multi-agent Systems. MIT Press (1999)
21. Munroe, S., Luck, M.: Motivation-based selection of negotiation partners. In: Engineering Societies in the Agents World (ESAW'04) 20-22 October 2004, Universit Paul Sabatier, Toulouse, France. (2004)
22. Jacques Calmet, Sebastian Jekutsch, P.K., Schü, J.: KOMET – a system for the integration of heterogeneous information sources. In: International Symposium on Methodologies for Intelligent Systems. Volume 1325 of LNAI., Springer-Verlag (1997) 318–327
23. FIPA: Personal assistant specification (2004) Available electronically at http://www.fipa.org/specs/fipa00083/XC00083B.html.
24. Endsuleit, R., Mie, T.: Secure multi-agents computations. In: Proceedings of International Conference on Security and Management (CSREA) , Las Vegas. (2003)
25. Ford, K.M., Glymour, C., Hayes, P.: Cognitive prostheses. AI Magazine **18** (1997)
26. Hoffman, R.R., Ford, K.M., Hayes, P.J., Bradshaw, J.M.: The Borg hypothesis. Intelligent Systems **18** (2003) 73–75