AGENTS THAT HELP TO DETECT TRUSTWORTHY KNOWLEDGE SOURCES IN KNOWLEDGE MANAGEMENT SYSTEMS

Juan Pablo Soto, Aurora Vizcaíno, Javier Portillo-Rodríguez and Mario Piattini
Alarcos Research Group, Information Systems and Technologies Department, Indra-UCLM Research and Development Institute, University of Castilla – La Mancha, Ciudad Real, Spain

Keywords: Knowledge Management Systems, Multi-agent architecture, Software Agents.

Abstract: Knowledge Management is a critical factor for companies worried about increasing their competitive advantage. Because of this companies are acquiring knowledge management tools that help them manage and reuse their knowledge. One of the mechanisms most commonly used with this goal is that of Knowledge Management Systems (KMS). However, sometimes KMS are not very used by the employees, who consider that the knowledge stored is not very valuable. In order to avoid it, in this paper we propose a three-level multi-agent architecture based on the concept of communities of practice with the idea of providing the most trustworthy knowledge to each person according to the reputation of the knowledge source. Moreover a prototype that demostrates the feasibility of our ideas is described.

1 INTRODUCTION

Knowledge Management (KM) is an emerging discipline considered a key part of the strategy to use expertise to create a sustainable competitive advantage in today's business environment. Having a healthy corporate culture is imperative for success in KM. Zand (1997) claims that bureaucratic cultures suffer from a lack of trust and a failure to reward and promote cooperation and collaboration. Without a trusting and properly motivated workforce, knowledge is rarely shared or applied, organizational cooperation and alignment are nonexistent.

Certain systems have been designed to assist organizations to manage their knowledge. These are called Knowledge Management Systems (KMS). KMS, described in (Alavi & Leidner, 2001), as an IT-based system developed to support/enhance the processes of knowledge creation, storage/retrieval, transfer and application. An advantage of KMS is that staff may also be informed about the location of information. Sometimes the organization itself is not aware of the location of the pockets of knowledge or expertise (Nebus, 2001). Moreover, a KMS is able to provide process improvements: it is better at serving the clients, and provides better measurement and accountability along with an automatic knowledge management.

However, developing KMS is not a simple task since knowledge per se is intensively domain dependant whereas KMS are often context specific applications. KMS have received certain criticism as they are often installing in the company thus overloading employees with extra work, since employees have to introduce information into the KMS and worry about updating this information. Moreover, the employees often do not have time to introduce or search for knowledge or they do not want to give away their own knowledge and or to reuse someone else's knowledge (Lawton, 2001). As is claimed in (Desouza et al, 2006) "employees resist being labeled as experts" and "they do not want their expertise in a particular topic to stunt their intellectual growth". Because of this resistance towards sharing knowledge, companies are using incentives to encourage employees to contribute to the knowledge growth of their companies (Huysman & Wit, 2000). Some of these incentives are organizational reward and allocate people to projects not only to work but also to learn and to share experiences. These strategies are sometimes useful. However, they are not are a '*silver bullet*' since an employee may introduce information that is not very useful with the only objective of trying to simulate that s/he is collaborating with the system in order to

Pablo Soto J., Vizcaíno A., Portillo-Rodríguez J. and Piattini M. (2007).
AGENTS THAT HELP TO DETECT TRUSTWORTHY KNOWLEDGE SOURCES IN KNOWLEDGE MANAGEMENT SYSTEMS.
In *Proceedings of the Second International Conference on Software and Data Technologies - PL/DPS/KE/WsMUSE*, pages 219-226
DOI: 10.5220/0001327602190226

generate points and benefits to get incentives or rewards. Generally, when this happens, the information stored is not very valuable and it will probably never be used. Based on this idea we have studied how the people obtain and increase their knowledge in their daily work. One of the most important developments concerning the nature of tacit, collective knowledge in the contemporary workplace has been the deployment of the concept 'communities of practice (CoPs)', by which we mean groups of people with a common interest where each member contributes knowledge about a common domain (Wenger, 1998). CoPs is necessarily bound to a technology, a set of techniques or an organization, that is to a common referent from which all members evaluate the authority or skill and reputation of their peers and the organization. A key factor for CoPs is provides an environment of confidence where their members can to share the information and best practices.

In order to provide to companies the conditions to develop trustworthy knowledge management systems we propose a multi-agent systems that simulates the member's behaviours of CoPs to detect trustworthy knowledge sources. Thus in Section 2, we explain why agents are a suitable technology with which to manage knowledge. Then, in Section 3 we describe our proposal. After that, in Section 4 we illustrate how the multi-agent architecture has been used to implement a prototype which detects and suggests trustworthy knowledge sources for members in CoPs. Finally, in Section 5 the evaluation and future work are presented.

2 AGENTS IN KNOWLEDGE MANAGEMENT

2.1 Why Intelligent Agents?

Due to the fundamentally social nature of knowledge management applications different techniques have been used to implement KMS. One of them, which is proving to be quite useful is the agent paradigm (van-Elst et al, 2003). Different definitions of intelligent agents can be found in literature. For instance, in (Mohammadian, 2004) agents are defined as computer programs that assist users with their tasks. One way of distinguishing agents from other types of software applications and to characterize them is to describe their main properties (Wooldridge & Jennings, 1995):

- Autonomy: agents operate without the direct intervention of humans or others, and have some kind of control over their actions and internal states.
- Social ability: agents interact with other agents (and possibly humans) via some kind of agent communication language.
- Reactivity: agents perceive their environment and respond in a timely fashion.
- Pro-activeness: in the sense that the agents can take the initiative and achieve their own goals.

In addition, intelligent agent's specific characteristics turn them into promising candidates in providing a KMS solution (Mercer & Greenwood, 2001). Moreover, software agent technology can monitor and coordinate events, meetings and disseminate information (Balasubramanian et al, 2001), building and maintaining organizational memories (Abecker et al, 2003). Another important issue is that agents can learn from their own experience. Most agents today employ some type of artificial intelligence technique to assist the users with their computer-related tasks, such as reading e-mails, maintaining a calendar, and filtering information. Agents can exhibit flexible behaviour, providing knowledge both "*reactively*", on user request, and "*pro-actively*", anticipating the user's knowledge needs. They can also serve as personal assistants, maintaining the user's profile and preferences. The advantages that agent technology has shown in the area of information management have encouraged us to consider agents as a suitable technique by which to develop an architecture with the goal of helping to develop trustworthy KMS.

Therefore, we have chosen the agent paradigm because it constitutes a natural metaphor for systems with purposeful interacting agents, and this abstraction is close to the human way of thinking about their own activities (Wooldridge & Ciancarini, 2001). This foundation has led to an increasing interest in social aspects such as motivation, leadership, culture or trust (Fuentes et al, 2004). Our research is related to this last concept of "trust" since artificial agents can be made more robust, resilient and effective by providing them with trust reasoning capabilities.

2.2 Previous Work in the Field

This research can be compared with other proposals that use agents and trust in knowledge exchange. For instance, in (Abdul-Rahman & Hailes, 2000), the authors propose a model that allows agents to decide which agents' opinions they trust more and propose a protocol based on recommendations. This model is based on a reputation or word-of-mouth mechanism. The main problem with this approach is that every agent must keep rather complex data structures that represent a kind of global knowledge about the whole network. In (Schulz et al, 2003), the authors propose a framework for exchanging knowledge in a mobile environment. They use delegate agents to be spread out into the network of a mobile community and use trust information to serve as the virtual presence of a mobile user. Another interesting work is (Wang & Vassileva, 2003) where the authors describe a trust and reputation mechanism that allows peers to discover partners who meet their individual requirements through individual experience and by sharing experiences with other peers with similar preferences. This work is focused on *peer-to-peer* environments.

Barber and Kim (2004) present a multi-agent belief revision algorithm based on belief networks. In their model the agent is able to evaluate incoming information, to generate a consistent knowledge base, and to avoid fraudulent information from unreliable or deceptive information sources or agents. This work has a similar goal to ours. However, the means of attaining it are different. In Barber and Kim's case they define reputation as a probability measure, since the information source is assigned a reputation value of between 0 and 1. Moreover, every time a source sends knowledge the source should indicate the certainty factor that the source has of that knowledge. In our case, the focus is very different since it is the receiver who evaluates the relevance of a piece of knowledge rather than the provider as in Barber and Kim's proposal.

3 A THREE-LEVEL MULTI-AGENT ARCHITECTURE

Before defining our architecture it is necessary to explain the conceptual model of an agent which, in our case, is based on two related concepts: trust and reputation. The former can be defined as confidence in the ability and intention of an information source to deliver correct information (Barber & Kim, 2004) and the latter as the amount of trust an agent has in an information source, created through interactions with information sources. There are other definitions of these concepts (Gambetta, 1988; Marsh, 1994). However, we have presented the most appropriate for our research since the level of confidence in a source is, in our case, based upon previous experience of this.

The reputation of an information source not only serves as a means of belief revision in a situation of uncertainty, but also serves as a social law that obliges us to remain trustworthy to other people. Therefore, people, in real life in general and in companies in particular, prefer to exchange knowledge with "trustworthy people" by which we mean people they trust. People with a consistently low reputation will eventually be isolated from the community since others will rarely accept their justifications or arguments and will limit their interaction with them. It is for this reason that the remainder of this paper deals mainly with reputation.

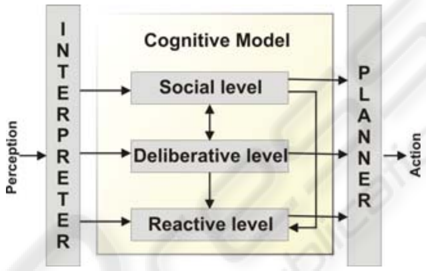

Figure 1: General architecture.

Taking these concepts into account we designed a multi-agent architecture which is composed of three levels (see Figure 1): reactive, deliberative and social. The reactive and deliberative levels are considered by other authors as typical levels that a multi-agent system must have (Ushida et al, 1998). The first level is frequently used in areas related to robotics where agents react to changes in the environment, without considering other processes generated in the same environment. In addition, the deliberative level uses a reasoning model in order to decide what action to perform.

On the other hand, the last level (social) is not frequently considered in an explicit way, despite the fact that these systems (multi-agent systems) are composed of several individuals, interactions between them and plans constructed by them. The social level is only considered in those systems that try to simulate social behaviour. Since we wish to emulate human feelings such as trust, reputation and even intuition we have added a social level that considers the social aspects of a community which takes into account the opinions and behaviour of each of the members of that community. Other previous works have also added a social level. For instance in (Imbert & de Antonio, 2005) the authors try to emulate human emotions such as fear, thirst, bravery, and also uses an architecture of three levels.

In the following paragraphs we will explain each of these levels in detail.

Reactive level: This is the agent's capacity to perceive changes in its environment and to respond to these changes at the precise moment at which they happen. It is in this level when an agent will execute the request of another agent without any type of reasoning. That is to say, the agent must act quickly in the face of critical situations.

Deliberative level: The agent may also have a behaviour which is oriented towards objectives, that is, it takes the initiative in order to plan its performance with the purpose of attaining its goals. In this level the agent would use the information that it receives from the environment, and from its beliefs and intuitions, to decide which is the best plan of action to follow in order to fulfill its objectives.

Social level: This level is very important as our agents are situated within communities and they exchange information with other agents. Thanks to this level they can cooperate with other agents. This level represents the actual situation of the community, and also considers the goals and interests of each community member in order to solve conflicts and problems which may arise between them. In addition, this level provides the support necessary to measure and stimulate the level of participation of the members of the community.

Two further important components of our architecture are the *Interpreter* and the *Planner* (see Figure 2). The former is used to perceive the changes that take place in the environment. The planner indicates how the actions should be executed.

In the following subsections we will describe each of the levels of which our architecture is composed in more detail.

3.1 Reactive Architecture

This architecture was designed to the reactive level of the agent. The architecture must respond at the precise moment in which an event has been perceived. For instance when an agent is consulted about its position within the organization. This architecture is formed of the following modules:

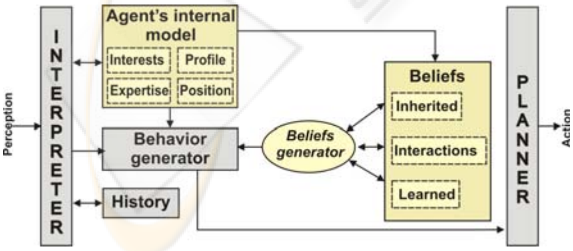

Figure 2: Reactive architecture.

Agent's internal model: As a software agent represents a person in a community this model stores the user's features. Therefore, this module stores the following parts:

- The *interests*. This part is included in the internal model in order to make the process of distributing knowledge as fast as possible. That is, the agents are able to search for knowledge automatically, checking whether there is stored knowledge which matches with its own interests. This behaviour fosters knowledge sharing and reduces the amount of work employees have to do because they receive knowledge without making searches.
- *Expertise*. This term can be briefly defined as the skill or knowledge of a person who knows a great deal about a specific thing. Since we are emulating communities of practice it is important to know the degree of expertise that each member of the community has in order to decide how trustworthy a piece of knowledge is, since people often trust in experts more than in novice employees.
- *Position*. Employees often consider information that comes from a boss as being more reliable than that which comes from another employee in the same (or a lower) position as him/her (Wasserman & Glaskiewics, 1994). Such different positions inevitably influence the way in which knowledge is acquired, diffused and eventually transformed in the local area. Because of this these factor will be calculated in our research by taking into account a weight that can strengthen this factor to a greater or to a lesser degree.
- *Profile*. This part is included in the internal model to describe the profile of the person on whose behalf the agent is acting. Therefore, a person's preferences are stored here.

Behaviour generator: This component is necessary for the development of this architecture since it has to select the agent's behaviour. This behaviour is defined on the basis of the agent's beliefs. Moreover, this component finds an immediate response to the perceptions received of the environment.

History: This component stores the interactions of the agents with the environment.

Belief generation: This component is one of the most important in the cognitive model because it is in charge of creating and storing the agent's knowledge. Moreover, it defines the agent's beliefs.

Beliefs: The beliefs module is composed of three kinds of beliefs: inherited beliefs, lessons learned and interactions. *Inherited beliefs* are the organization's beliefs that the agent receives. For instance: an organizational diagram of the enterprise, the philosophy of the company or community. *Lessons learned* are the lessons that the agent obtains while it interacts with the environment. The information about *interactions* can be used to establish parameters in order to know which the agent can trust (agents or knowledge sources). This module is based on the interests and goals of the agent, because each time a goal is realised, the lessons and experiences generated to attain this goal are introduced in the agent's beliefs as lessons learned.

3.2 Deliberative Architecture

This architecture was designed to the deliberative level of the agent (see Figure 3).

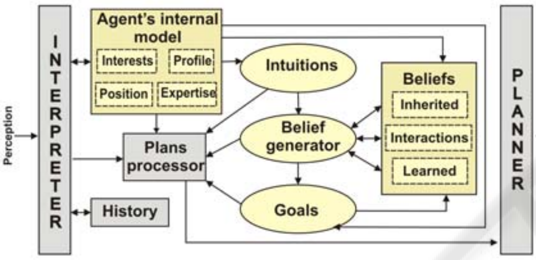

Figure 3: Deliberative architecture

Its components are:

Agent's internal model: this module is the same as that which is described in the reactive architecture. It is composed of the interests, profile, position and expertise of the agent.

Plans processor: This module is the most important of this architecture as it is in charge of evaluating the beliefs and goals to determine which plans have to be included in the Planner to be executed.

Belief generator: This component, as in the previous architecture, is in charge of creating, storing and retaining the agent's knowledge. In addition, it is also in charge of establishing the agent's beliefs. The belief creation process is a continuous process that is initiated at the moment at which the agent is created and which continues during its entire effective life.

Intuitions: Intuitions are beliefs that have not been verified but which it thinks may be true. According to (Mui et al, 2002) intuition has not yet been modelled by agent systems. In this work we have tried to adapt this concept because we consider that in real communities people are influenced by their intuitions when they have to make a decision or believe in something. This concept is emulated by comparing the agents' profiles to obtain an initial value of intuition that can be used to form a belief about an agent.

History: This component stores the interactions of the agents with the environment.

3.3 Social Architecture

This architecture (see Figure 4) is quite similar to the deliberative architecture. The main differences are the social model and social behaviour processor, which are explained in the following paragraphs.

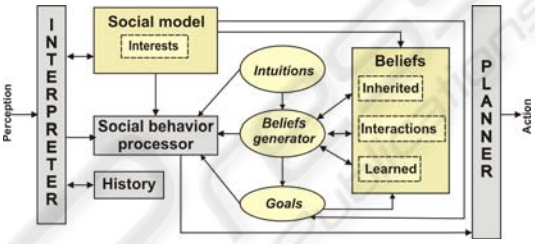

Figure 4: Social architecture.

Social model: This module represents the actual state of the community, the community's interests and the members' identifiers.

Social behaviour processor: This component processes the beliefs of the community's members. To do this, this module needs to manage the goals, intuitions and beliefs of the community in order to make a decision.

The social focus that this architecture provides permits us to give the agents the social behaviour necessary to emulate the work relationships in an organization. In addition, this layer permits the decentralization of decision making, that is, it provides methods by which to process or make decisions based on the opinions of the members of a community.

4 IMPLEMENTATION OF THE ARCHITECTURE

To evaluate the feasibility of the implementation of the architecture, we have developed a prototype into which people can introduce documents and where these documents can also be consulted by other people. The goal of this prototype is to allow software agents to help employees to discover the information that may be useful to them thus decreasing the overload of information that employees often have and strengthening the use of knowledge bases in enterprises. In addition, we try to avoid the situation of employees storing valueless information in the knowledge base.

A feature of this system is that when a person searches for knowledge in a community, and after having used the knowledge obtained, that person then has to evaluate the knowledge in order to indicate whether:

- The knowledge was useful.
- How it was related to the topic of the search (for instance a lot, not too much, not at all).

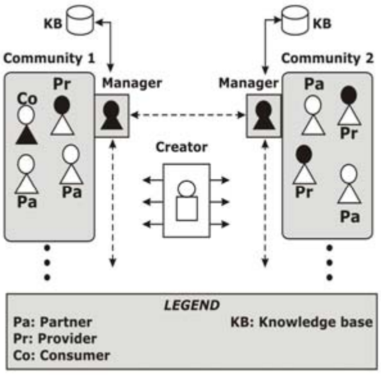

Figure 5: Agent’s distribution.

One type of agent in our prototype (see Figure 5) is the User Agent which is in charge of representing each person that may consult or introduce knowledge in a knowledge base. The *User Agent* can assume three types of behavior or roles similar to the tasks that a person may carry out in a knowledge base. Therefore, the User Agent plays one role or another depending upon whether the person that it represents carries out one of the following actions:

- The person contributes new knowledge to the communities in which s/he is registered. In this case the User Agent plays the role of **Provider**.
- The person uses knowledge previously stored in the community. Then, the User Agent will be considered as a **Consumer**.
- The person helps other users to achieve their goals, for instance by giving an evaluation of certain knowledge. In this case the role is that of a **Partner**. So, Figure 5 shows that in Community 1 there are two User Agents playing the role of Partner (Pa), one User Agent playing the role of Consumer (Co) and another being a Provider (Pr).

The second type of agent within a community is called the *Manager Agent* (represented in black in Figure 5) which is in charge of managing and controlling its community. In order to approach this type of agent the following tasks are carried out:

- Registering an agent in its community. It thus controls how many agents there are and how long the stay of each agent in that community is.
- Registering the frequency of contribution of each agent. This value is updated every time an agent makes a contribution to the community.
- Registering the number of times that an agent gives feedback about other agents’ knowledge. For instance, when an agent “A” uses information from another agent “B”, the agent A should evaluate this information. Monitoring how often an agent gives feedback about other agents’ information helps to detect whether agents contribute to the creation of knowledge flows in the community since it is as important that an agent contributes new information as it is that another agent contributes by evaluating the relevance or importance of this information.
- Registering the interactions between agents. Every time an agent evaluates the contributions of another agent the Manager agent will register this interaction. But this interaction is only in one direction, which means, if the agent A consults information from agent B and evaluates it, the Manager records that A knows B but that does not means that B knows A because B does not obtain any information about A.

Moreover, when a user wants to join to a community in which no member knows anything about him/her, the reputation value assigned to the user in the new community is calculated on the basis of the reputation assigned from others communities where the user is or was a member. For instance, an User Agent called j, will ask each community manager where he/she was previously a member to consult each agent which knows him/her with the goal of calculating the average value of his/her reputation (R_{Aj}). This is calculated as:

$$R_{Aj} = \left(\sum_{j=1}^{n} R_{sj}\right)/n \qquad (1)$$

where *n* is the number agents who know j and $\mathbf{R}_{sj}$ is the value of reputation of j in the eyes of s. In the case of being known in several communities the average of the values $\mathbf{R}_{Aj}$ will be calculated. Then, the User Agent j presents this reputation value (similar to when a person presents his/her curriculum vitae when s/he wishes to join a company) to the Manager Agent of the community to which it is “applying”. This reputation value permits to assign a reputation value taking into account the previous experiences and relations with others agents, generating a flow and exchange of information between the agents. This mechanism is similar to the “word-of-mouth” propagation of information for a human (Abdul-Rahman & Hailes, 2000).

In addition, $\mathbf{R}_{sj}$ value is computed as follows:

$$R_{sj} = w_e * E_j + w_p * P_j + w_i * I_j + \left(\sum_{j=1}^{n} QC_j\right)/n \qquad (2)$$

where E_j is the value of expertise which is calculated according to the degree of experience that the person upon whose behalf the agent acts has in a domain.

P_j is the value assigned to a person's position. This position is defined in the agent's internal model of the reactive architecture described in Section 3.1.

I_j is the value assigned to intuition which is calculated by comparing each user's profile. Intuition is an important component both in the deliberative and in the social architecture because it helps agents to create their beliefs and behavior according to their own features.

In addition, previous experience should also be calculated. We suppose that when an agent A consults information from another agent B, the agent A should evaluate how useful this information was. This value is called QC_j (Quality of *j*'s Contribution). To attain the average value of an agent's contribution, we calculate the sum of all the values assigned to these contributions and we divide it between their total. In the expression n represents the total number of evaluated contributions.

Finally, w_e, w_p and w_i are weights with which the Reputation value can be adjusted to the needs of the organizations or communities. These weights represent different values depending on the category of each employee. For instance, if an enterprise considers that all its employees have the same category, then w_p=0. The same could occur when the organization does not take its employee's intuitions or expertise into account.

In this way, an agent can obtain a value related to the reputation of another agent and decide to what degree it is going to consider the importance of the information obtained from this agent. The formulas (1) and (2) are processed in the social and deliberative architecture respectively.

5 EVALUATION AND FUTURE WORK

Once the prototype has finished we will evaluate it. To do this, different approaches can be followed, from a multi-agent point of view or from a social one. First of all we have focused on the former and we are testing the most suitable number of agents advisable for a community. Therefore, several simulations have been performed. As result of them we found that:

- The maximum number of agents supported by the Community Manager Agent when it receives User Agents' evaluations is approximately 800. When we tried to work with 1000 agents for instance, the messages were not managed conveniently. However, we could see that the Manager Agent could support a high number of petitions, at least, using simpler behavior.
- On the other hand, if we have around 10 User Agents launched, they need about 20 o more interactions to know all agents of the community. If a User Agent has between 10 and 20 interactions with other members it is likely that it interacts with 90% of members of its community, which means that the agent is going to know almost all the members of the community. Therefore, after several trials we detected that the most suitable number of agents for one community was around 10 agents and they needed a average of 20 interactions to know (to have a contact with) all the members of the community, which is quite convenient in order to obtain its own value of reputation about other agent.

All these results are being used to detect whether the exchange of messages between the agents is suitable, and to see if the information that we propose to be taken into account to obtain a trustworthy value of the reputation of each agent is enough, or if more parameters should be considered. Once this validation is finished we need to carry out further research to answer one important question, which is how the usage of this prototype affects the performance of a community. This is the social approach that we mentioned at the beginning of this section. As claimed in (Geib et al, 2004) to measure the performance of communities is a challenge since communities only have an indirect impact on business results. In order to do this we are going to take some ideas of the performance measurement framework for communities propose by (McDermott, 2002) where the performance of communities is measured in terms of output and values such as: personal knowledge, strength of relationships (this could be one of the most important values for our research) and access to information. This research will be critical to find how our proposal affects communities of practice.

ACKNOWLEDGEMENTS

This work is partially supported by the ENIGMAS (PIB-05-058), and MECENAS (PBI06-0024) project. It is also supported by the ESFINGE project (TIN2006-15175-C05-05) Ministerio de Educación y Ciencia (Dirección General de Investigación)/ Fondos Europeos de Desarrollo Regional (FEDER) in Spain and CONACYT (México) under grant of the scholarship 206147 provided for the first author.

REFERENCES

Abdul-Rahman, A., Hailes, S., (2000),*Supporting Trust in Virtual Communities.* 33rd Hawaii International Conference on Systems Sciences (HICSS'00).

Abecker, A., Bernardi, A., van-Elst, L., (2003), *Agent Technology for Distributed Organizational Memories: The Frodo Project*. In *Proceedings of 5th International Conference on Enterprise Information Systems*. Angers, France.

Alavi, M., Leidner, D. E., (2001), *Knowledge Management and Knowledge Management Systems: Conceptual Foundations and Research Issues.* MIS Quarterly, Vol. 25, No. 1, pp. 107-136.

Balasubramanian, S., Brennan, R., Norrie, D., (2001), *An Architecture for Metamorphic Control of Holonic Manufacturing Systems.* Computers in Industry, Vol. 46(1), pp. 13-31.

Barber, K., Kim, J., (2004), *Belief Revision Process Based on Trust: Simulation Experiments*. In *4th Workshop on Deception, Fraud and Trust in Agent Societies*, Montreal Canada.

Fuentes, R., Gómez-Sanz, J., Pavón, J., (2004), *A Social Framework for Multi-agent Systems Validation and Verification*, in *Wang, S. et al (Eds.) ER Workshop 2004, Springer Verlag, LNCS 3289*, pp. 458-469.

Gambetta, D., (1988), *Can We Trust Trust?* In D. Gambetta, editor, Trust: Making and Breaking Cooperative Relations, pp. 213-237.

Geib, M., Braun, C., Kolbe, L., Brenner, W., (2004), *Measuring the Utilization of Collaboration Technology for Knowledge Development and Exchange in Virtual Communities.* Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04). IEEE Computer Society Press, Vol. 1.

Huysman, M., Wit, D., (2000), *Knowledge Sharing in Practice*. Kluwer Academic Publishers, Dordrecht.

Imbert, R., and de Antonio, A.,(2005), *When emotion does not mean loss of control.* Lecture Notes in Computer Science, T. Panayiotopoulos, J. Gratch, R. Aylett, D. Ballin, P. Olivier, and T. Rist, Eds. Springer-Verlag, London., pp. 152-165.

Desouza, K, Awazu, Y., Baloh, P., (2006), *Managing Knowledge in Global Software Development Efforts: Issues and Practices.* IEEE Software, Vol. 23, No. 5, pp. 30-37.

Lawton, G., (2001), *Knowledge Management: Ready for Prime Time?* Computer, Vol. 34, No. 2, pp. 12-14.

Marsh, S., (1994), *Formalising Trust as a Computational Concept*, PhD Thesis, University of Stirling.

McDermott, R., (2002), *Measuring the Impact of Communities.* Knowledge Management, 2002. Vol. 5, No. 2, pp. 26-29.

Mercer, S., Greenwood, S., (2001), *A Multi-Agent Architecture for Knowledge Sharing*. In *Proceedings of the Sixteenth European Meeting on Cybernetics and Systems Research*.

Mohammadian, M., (2004), *Computational Intelligence Techniques Driven Intelligent Agents for Web Data Mining and Information Retrieval*, in *Intelligent Agents for Data Mining and Information Retrieval. IDEA Group Publishing*. pp. 15-29.

Mui, L., Halberstadt, A., Mohtashemi, M., (2002), *Notions of Reputation in Multi-Agents Systems: A Review*, International Conference on Autonomous Agents and Multi-Agents Systems (AAMAS'02), pp. 280-287.

Nebus, J., (2001), *Framing the Knowledge Search Problem: Whom Do We Contact, and Why Do We Contact Them?* Academy of Management Best Papers Proceedings, pp. 1-7.

Schulz, S., Herrmann, K., Kalcklosch, R., Schowotzer, T., 2003, *Trust-Based Agent-Mediated Knowledge Exchange for Ubiquitous Peer Networks.* AMKM, LNAI 2926, p. 89-106.

Ushida, H., Hirayama, Y., Nakajima, H., (1998) *Emotion Model for Life like Agent and its Evaluation*. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence and Tenth Innovative Applications of Artificial Intelligence Conference (AAAI'98 / IAAI'98)*. Madison, Wisconsin, USA.

van-Elst, L., Dignum, V., Abecker, A., (2003) *Agent-Mediated Knowledge Management*. in *International Simposium AMKM'03*, Stanford, CA, USA: Springer.

Wang, Y., Vassileva, J., (2003), *Trust and Reputation Model in Peer-to-Peer Networks.* Proceedings of IEEE Conference on P2P Computing.

Wasserman, S. and Glaskiewics, J., (1994), *Advances in Social Networks Analysis.* Sage Publications.

Wenger, E., (1998), *Communities of Practice: Learning Meaning, and Identity*, Cambridge U.K.: Cambridge University Press.

Wooldridge, M. and Ciancarini, P., (2001), *Agent-Oriented Software Engineering: The State of the Art.* In Wooldridge M., Ciancarini, P. (Eds.), Agent Oriented Software Engineering. Springer Verlag, LNAI 1975.

Wooldridge, M., Jennings, N.R., (1995), *Intelligent Agents: Theory and Practice*. Knowledge Engineering Review, Vol. 10, No. 2, pp. 115-152.

Zand, D., (1997), *The Leadership Triad: Knowledge, Trust, and Power*. Oxford University Press.