Application of Software Agents Technology to Create Knowledge

Juan Pablo Soto, Aurora Vizcaíno, Javier Portillo and Mario Piattini

Alarcos Research Group, University of Castilla-La Mancha
Paseo de la Universidad, 4-13071, Ciudad Real, Spain
jpsoto@proyectos.inf-cr.uclm.es
{aurora.vizcaino, mario.piattini}@uclm.es
javier.portillo@alu.uclm.es

Abstract. Agent technology has shown promise in the area of information management and is therefore a good candidate for knowledge management solutions. Intelligent agents have thus played diverse roles in knowledge management systems, as assistants, recommenders, user trackers, etc. Intelligent agents have also been used in frameworks or architectures whose development was intended to make the implementation of knowledge management systems easier. It is on this point that we are going to focus in this paper. Many of these architectures and frameworks were developed from a technical point of view, forgetting the knowledge tasks that they should support. Bearing this in mind, we designed a knowledge life cycle which can be taken into account when developing knowledge management systems. Moreover, in this paper we describe a generic multi-agent architecture (based on the knowledge cycle) to help developers to implement knowledge management systems.

1 Introduction

In the last decades, Knowledge Management (KM) has captured the attention of companies as being one of the most promising ways to reach success in this information era [9]. A shorter life-cycle of products, globalization, and strategic alliances between companies demand a deeper and more systematic organizational knowledge management [1]. One way to assess an organization's performance is to determine how well it manages its critical knowledge.

KM can be defined as a discipline that enables an organization to take advantage of its intellectual capital, in order to reuse it and learn from previous experience [16]. Skyrme [19] suggests that KM is the purposeful and systematic management of vital knowledge, along with its associated processes of creating, gathering, organizing, diffusing, using, and exploiting that knowledge. Due to the importance of knowledge management, tools which support some of the tasks related to KM have been developed. Different techniques have been used to implement these tools. One of them, which is proving to be quite useful, is that of intelligent agents [22]. Software agent technology can monitor and coordinate events or meetings and disseminate information [2]. Furthermore, agents are proactive; this means they act automatically when it

Advances in Artificial Intelligence.
Research in Computing Science 26, 2006, pp. 127-136

Received 06/07/06
Accepted 03/10/06
Final version 09/10/06

is necessary. The autonomous behaviour of the agents is critical to the goal of this research; reducing the amount of work that employees have to perform when using a KM system. Another important issue is that agents can learn from their own experience. Because of these advantages, different agent-based architectures have been proposed to support activities related to KM [6]. Some architectures have even been designed to help in the development of KM systems. However, most of them focus on a particular domain and can only be used under specific circumstances. What is more, they do not take into account the cycles of knowledge when proposing to use knowledge management in the system itself. For these reasons, in this paper we propose a generic multi-agent architecture for creating knowledge in organizations that use knowledge management techniques. The agents used in this architecture have been designed to support the different phases of the knowledge life cycle and foster the creation, storage and dissemination of knowledge.

This paper is structured as follows. In section two we describe different knowledge models as they have been proposed in the relevant literature. In addition, we put forward our own knowledge model. In section three, previous works based on multi-agent architectures are outlined. Section four presents our proposal. Finally, conclusions are explained in section five.

2 Knowledge Processes

One of the most widely discussed approaches is the SECI process [12], where the interaction between tacit and explicit knowledge emerges as a spiral (knowledge spiral) that includes four layers of knowledge conversion: *socialization* involves the sharing and exchanging of tacit knowledge between individuals, to create common mental models and abilities, most frequently through the medium of shared experience; *externalization* is the process of articulating tacit knowledge into comprehensive forms that can be understood by others; *combination* involves the conversion of explicit knowledge into a more complex set of explicit knowledge, and *internalization* is the process of adding explicit knowledge to tacit new knowledge by experimenting in various ways, through real life experience or simulations.

As there is no consensus in defining the stages that form a KM life cycle, Davenport and Prusak identify three tasks of knowledge management: (generation, codification/coordination and transfer) [4]. Wiig, in [24] observes five KM processes: (knowledge creation, knowledge storing, knowledge use, knowledge leverage, knowledge sharing). The similarities found in the stages of the models described previously helped us to define a process that is able to integrate the different proposals. For example, all the proposals consider stages which create, acquire, transfer, distribute, and disseminate knowledge. Another important aspect to take into account is that the capture or storing, along with the access or retrieval of knowledge, also form an important part of a KM life cycle model.

We have therefore chosen those stages that we believe should be supported by an architecture for developing KM systems (see Figure 1) and which are critical for this kind of systems. In section four we will explain how our multi-agent architecture follows this knowledge life cycle model.

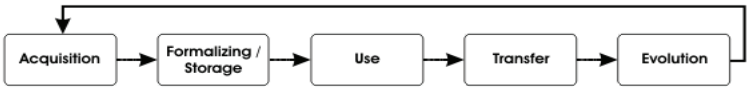

Fig. 1. Knowledge Management Life Cycle Model proposed

3 Related works

Traditional KM systems have received certain criticism, since they are often implanted in the company itself. Employees are consequently overloaded with extra work, as they have to introduce information into the KM systems and worry about updating this information. One proposal to avoid this extra burden was to add software agents to perform this task in place of the employees. Later, intelligent agent technology was also applied to other different activities, bringing several benefits to the KM process [15, 18].

The benefits of applying agent technology to knowledge management include distributed system architecture, easy interaction, resource management, reactivity to changes, interoperation between heterogeneous systems, and intelligent decision making. The set of KM tasks or applications in which an agent can assist is virtually unlimited, for example: to provide support to cooperative activities [23], to help people find information or experts which/who can assist them in their daily work [3] and to exploit an organizational memory [6].

Besides these works, we found others which focused on document classification [13], mailing list management [11], or data mining [7].

These and other existing systems were often developed without considering how knowledge flows and what stages may foster these flows. Because of this, they often support only one knowledge task, without taking into account that KM implies giving support to different process and activities. On the other hand, KM systems often focus on the technology, without taking into account fundamental problems that this kind of systems is likely to support.

4 A Multi-agent architecture to develop KM systems

As one of our goals is to develop a multi-agent architecture with KM itself, we shall start this section by describing each stage of our knowledge life cycle, explaining the type of agent that supports that stage.

4.1 Knowledge Model

- *Knowledge acquisition* is the stage responsible for making the organization knowledge visible. This stage considers the activities necessary to create organ-

izational knowledge. Furthermore, the acquisition stage determines the organization skills for importing knowledge from external sources. The definition of the knowledge to be acquired can be assisted by classifying types of knowledge and knowledge sources [5]. To support this stage we propose to include in our architecture an agent called *Captor Agent*. The Captor Agent is responsible for collecting the information (data, models, experience, etc) from the different knowledge sources. It executes a proactive monitoring process, to identify the information and experiences generated during the interaction between the user and groupware tools (email, consulted web pages, chats, etc.). In order to accomplish this, the Captor Agent uses a knowledge ontology which defines the knowledge to be taken into account in a domain. Another useful ontology is the source ontology, which defines where each type of knowledge might be found [20]. Both of these are based on Rodriguez's ontologies for representing knowledge topics and knowledge sources [15].

The Captor Agent communicates with another agent (the *Constructor Agent*) which is in charge of creating knowledge. For example, when the Captor Agent acquires information that should be converted into knowledge it sends this information to the Constructor Agent.

One advantage of this architecture is that the Captor agent can work in any domain, since by changing these ontologies the Captor knows what key knowledge should be found and where it might be.

- *Knowledge formalizing/storing* is the stage that groups all the activities that focus on organizing, structuring, representing and codifying the knowledge, with the purpose of facilitating its use [4]. To help carry out these tasks we propose a *Constructor Agent*. This agent is in charge of giving an appropriate electronic format to the experiences obtained so that they can be stored in a knowledgebase to aid retrieval. Storing knowledge helps to reduce dependency on key employees, because at least some of their expert knowledge has been retained or made explicit. In addition, when knowledge is stored, it is made available to all employees, providing them with a reference as to how processes must be performed, and how they have been performed in the past. Moreover, the Constructor Agent compares the new information with old knowledge that has been stored previously and decides whether to delete it and add new knowledge or to combine both of them. In this way, the combination process of the SECI model is carried out, producing new knowledge as the end-result of the merging of explicit knowledge with explicit knowledge.
- *Knowledge use* is one of the main stages, since knowledge is useful when it is used and/or reused. The main enemy of knowledge reuse is ignorance. Employers often complain because employees do not consult knowledge sources and do not take advantage of the knowledge capital that the company has [21]. KM systems should offer the possibility of searching for information; they can even give recommendations or suggestions, with the goal of helping users to perform their tasks by reusing lessons that have already been learnt, and previous experiences. In our framework the agent in charge of this activity is the *Searcher Agent*, which searches in the knowledgebase for information that is needed. The result of the search will be sent to the *Interface Agent.* This will be explained in the next section. This agent could be implemented with different retrieval tech-

niques. Since this architecture is proposed at a high level, these aspects will not be dealt with in this paper. However, we would like to emphasize that the *Searcher Agent* fosters the internalization process of the SECI model, since the employees have the opportunity of acquiring new knowledge, by using the information that this agent suggests.

- *Knowledge transfer* is the stage in charge of transfering tacit and explicit knowledge. Tacit knowledge can be transfered if it has been previously stored in shared means, for example: repositories, organizational memories, databases, etc. The transfer stage can be carried out by using mechanisms to inform people about the new knowledge that has been added. For this stage we propose a *Disseminator Agent*, which must detect the group of people, or communities, who generate and use similar information: for example, in the software domain, the people who maintain the same product or those who use the same programming language. So this agent fosters the idea of a community of practice in which each person shares knowledge and learns, thanks to the knowledge of the other community members. Disseminated information may be of different types; it may be information linked to the company's philosophy, or specific information about a given process. Finally, the Disseminator agent needs to know exactly what kind of work each member of the organization is in charge of, together with the knowledge flows linked to their jobs. In order to do this, the Disseminator Agent contacts a new type of agent called the *Personal Agent,* which gives him information about the profiles of the user. Comparing this stage with the SECI model we can say the *Disseminator Agent* fosters the socialization process, since it puts people who demand similar knowledge in touch with each other. Once in contact, they can share their experience, thus increasing their tacit knowledge.
- *Knowledge Evolution*. This stage is responsible for monitoring the knowledge that evolves on a daily basis. To carry out this activity we propose a *Maintenance Agent*. The main purpose of this agent is to keep the knowledge stored in the knowledgebase updated. Information that is not often used is therefore considered by the Maintenance Agent as information that could possibly be eliminated.

4.2 Multi-agent Architecture

Once the knowledge model which the architecture is based on and the agents which support the different stages are defined, we can then define how the agents are organized into different agencies. Figure 2 shows that the architecture is formed of two agencies, made up of several agents.

The *Knowledge Agency* consists of the Constructor Agent, the Captor Agent, the Searcher Agent, the Disseminator Agent and the Maintenance Agent, previously described in Section 4.1. Therefore, the agency is in charge of giving support to the KM process.

The *User Agency* consists of the *Interface Agent* and the *Personal Agent*. The *Interface Agent* acts as an effective bridge between the user and the rest of the agents. Thus, if any agent wants to give a message to the user, the agent needs to send it to the Interface agent, which is the only one allowed to "talk" to the user. The Interface Agent also communicates with the *Personal Agent,* which obtains user profiles and

information that are relevant to users' knowledge and which help to determine the expertise level and knowledge that each person has, or that a given person may need.

Another component used in this architecture is the *Shared Ontology*, this ontology is shared by all agents and provides a conceptualization of the knowledge domain. The Shared Ontology is used for consistent communication of the agencies.

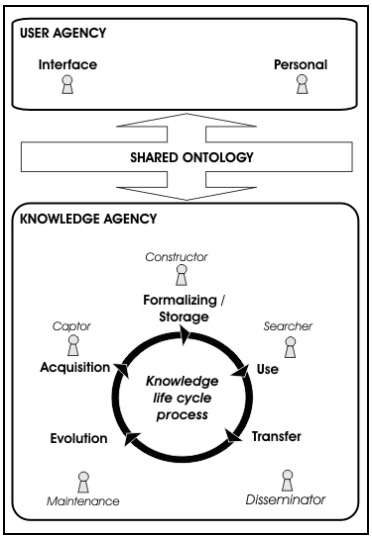

Fig. 2. Multi-agent Architecture for KM Systems

4.3 Agents Collaboration

As was mentioned before, the agents must collaborate with other agents. In order to show how they collaborate, we are going to describe a possible scenario that can take place in an organization.

Scenario

Let us imagine that a person is writing a mail and that the agents start to work, in order to check whether the mail contains information that should be stored in the data base (we are supposing that the employees know that the mails are being looked at in this way and that they agree to this). As Figure 3 shows, the Interface Agent captures each event that is triggered by the Employee. In this case the employee sends an email. Then the Interface Agent advises the Captor Agent that an event has been triggered. Afterwards, the Captor Agent determines the type of groupware tool used (email) to identify and obtain information topics about related tasks. In order to obtain information from the mail, a new agent can be added to the system (it would not form part of our architecture) but would be an agent that has been already developed to assist in this task. There are several implemented agents in existence for dealing with email [8]. Most of the current implementations are text classifiers [17] or keyword extractors [10]. The Captor Agent would study whether the information sent by the "email agent" should be transformed into knowledge. Finally, the Constructor

Agent receives the information which is structured in the form of, for instance, cases, for its later storage.

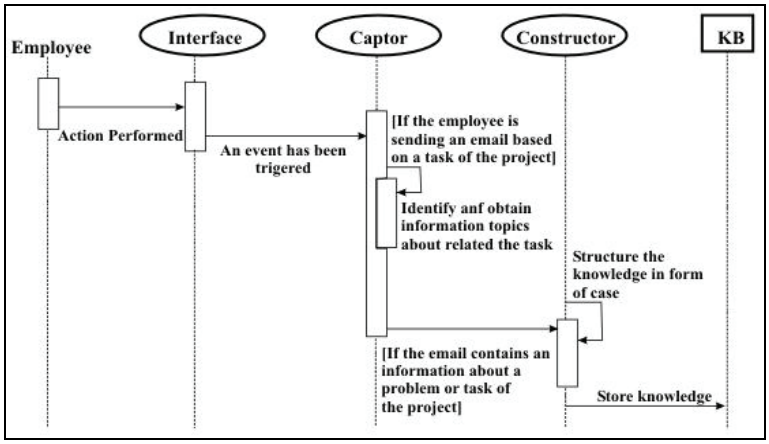

Fig. 3. Scenario of Agent Collaboration

4.4 Design of the Multi-agent Architecture

Methodology

This architecture has being designed by using INGENIAS [14] which provides meta-models to define MAS, and support tools to generate them. Using meta-models facilitates the development of the system enormously, since they are oriented to visual representations of concrete aspects of the system. Below, we are going to use the agent meta-model diagrams to describe the roles and tasks of each agent proposed in our architecture.

Figure 4 shows the goal, roles and tasks performed by the Captor Agent. The goal of this agent is to obtain information that should be stored. Its role is "filter" since it must decide what information should be transformed into knowledge; the purpose being to use it in future projects.

In the following lines, we describe each one of the task carried out by this agent.

- *CaptureInfo*: The agent must capture information.
- *IdentifyIS*: This task consists of identifying available information sources in the system.
- *SendToConstructor*: Once the suitability of storing the information has been analyzed, the Captor sends it to the Constructor Agent.

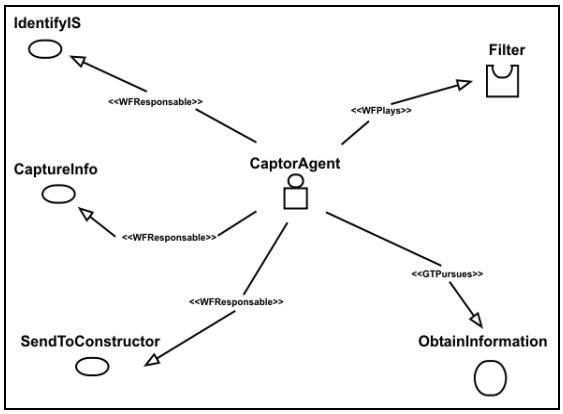

Fig. 4. Captor Agent diagram

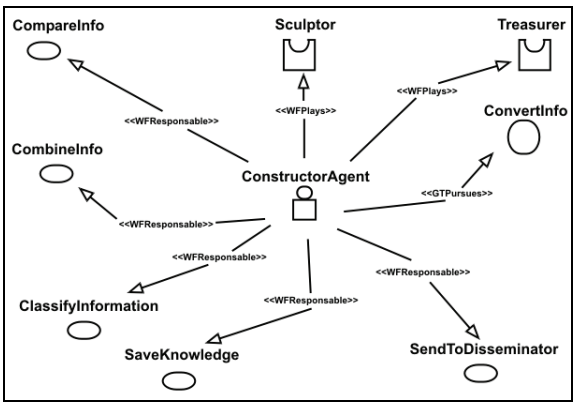

Fig. 5. Constructor Agent diagram

Figure 5 shows the role and tasks performed by the Constructor Agent whose roles are: sculptor and treasurer since it is in charge of giving an appropriate electronic format to the information (sculptor) and of storing it in the knowledgebase (treasurer). The tasks developed by Constructor Agent are:

- *CompareInfo*: The agent is in charge of comparing the new information with the previously stored knowledge.
- *CombineInfo*: The agent is also in charge of combining the new information with the previously stored knowledge. In this way, the combination process of the SECI model is carried out, producing new knowledge which comes about as a result of the merging of explicit knowledge with explicit knowledge.
- *ClassifyInformation*: Another task is to classify the information received by the Captor Agent (for instance: models, structures, files, diagrams, etc.).
- *SendToDisseminator*: This is a critical task which consists of sending knowledge to the Disseminator Agent.

- *SaveKnowledge*: One of the most important tasks is to store the new knowledge in the knowledgebase.

In this paper, only the Captor and Constructor Agent diagram are shown due to space constraints.

5 Conclusions

KM is a hot topic nowadays, as companies have realized that it offers them a competitive advantage. Because of this, various knowledge management systems have been developed, to support specific knowledge tasks. Moreover, different frameworks have been proposed, in the quest to make the development of certain knowledge management systems easier. Surprisingly, however, most of these frameworks have not taken into account how knowledge is created and how it flows through companies. In this paper we have described a knowledge life cycle to be taken into account when KM systems are being developed. Moreover, we have described a multi-agent architecture to support the different stages of that cycle. The main feature of the architecture is that it is so generic that it can be used to develop KM systems for almost any domain. Furthermore, it has been designed according to the INGENIAS methodology whose meta-models help future developers to understand how the different agents work.

Acknowledgement. This work is partially supported by ENIGMAS (PIB-05-058), and MECENAS (PBI06-0024) project, Junta de Comunidades de Castilla-La Mancha, Consejería de Educación y Ciencia, both in Spain.

References

1. Ale, M., Chiotti, O., Galli, M.R.: Agent-Supported Ontology-Based Knowledge Management Systems. Proceedings of ASIS (Simposium Argentino de Sistemas de Información) and JAIIO (33º Jornadas Argentinas de Informática e Investigación Operativa) (2004).
2. Balasubramanian, S., Brennan, R., Norrie, D.: An Architecture for Metamorphic Control of Holonic Manufacturing Systems. Computers in Industry, Vol.46 (2001) 13-31.
3. Crowder, R., Hughes, G., Hall, W.: An Agent Based Approach to Finding Expertise in the Engineering Design Environment. Proceedings of the 14th International Conference on Engineering Design, (2003).
4. Davenport, T.H., Prusak, L.: Working Knowledge: How Organizations Manage What They Know. Project Management Institute. Harvard Business School Press, Boston, Massachusetts (1997).
5. Dickinson, A.: WP3 - KM Framework. Enhancing Knowledge Management in Enterprises (ENKE) IST Project, IST-2000-29482, http://www.ist-enke.com (2000).
6. Gandon, F.: Multi-Agent System to Support Exploiting an XML-based Corporate Memory. Proceedings PAKM, Basel, Switzerland (2000).
7. Giannella, C., Bhargava, R., Kargupta, H.: Multi-agent Systems and Distributed Data Mining. Cooperative Information Agents VIII: 8th International Workshop, CIA 2004, Vol. 3191/2004. Springer-Verlag, Erfurt, Germany, (2004) 1-15.

8. Maes, P.: Agents that reduce work and information overload. Communications of the ACM, Vol.37 (1994) 31-40.
9. Malone, D.: Knowledge Management: A Model for Organizational Learning. International Journal of Accounting Information Systems, Vol.3 (2002) 111-123.
10. Mock, K.: An Experimental Framework for Email Categorization and Management. 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New Orleans, Louisiana, USA (2001).
11. Moreale, E., Watt, S.: An Agent-Based Approach to Mailing List Knowledge Management. Proceedings of the Agent-Mediated Knowledge Management (AMKM) (2003) 118-129.
12. Nonaka, I., Takeuchi, H.: The Knowledge Creation Company: How Japanese Companies Create the Dynamics of Innovation. Oxford University Press (1995) 304.
13. Novak, J., Wurst, M., Fleischmann, Strauss, W.: Discovering, Visualizing and Sharing Knowledge through Personalized Learning Knowledge Maps. Proceedings of the Agents-Mediated Knowledge Management (AMKM) (2003) 213-228.
14. Pavón, J., Gómez-Sanz, J.: Agent Oriented Software Engineering with INGENIAS. Multi-Agent Systems and Applications II, LNAI 2691. Springer-Verlag (2003) 394-403.
15. Rodríguez, O., Martinez-García, Ana., Vizcaíno, A., Favela, J., Piattini, M.: Identifying Knowledge Flows in Communities of Practice. In Coakes, Elayne and Steve Cleark: Encyclopedia of Communities of Practice in Information and Knowledge Management. IDEA Group (2005) 210-217.
16. Rus, I., Lindvall, M.: Knowledge Management in Software Engineering. IEEE Software, Vol. 19 (2002) 26-38.
17. Segal, R.B., Kephart, J.O.: MailCat: An Intelligent Assistant for Organizing E-Mail. Proceedings of the 3rd International Conference on Autonomous Agents, Seattle, WA, USA (1999).
18. Silverman, B.G., Bedewi, N., Morales, A.: Intelligent Agents in Software Reuse Repositories. CIKM Workshop on Intelligent Information Agents, Baltimore, USA (1995).
19. Skyrme, D.: Knowledge Management: Making Sense of an Oxymoron. Management Insight, Vol.22 (2003).
20. Soto, J.P., Vizcaíno, A., Piattini, M., Rodríguez, O.: A Multi-Agent Architecture to Develop Knowledge Management Systems. 1er. Congreso Internacional de Ciencias Computacionales (Cicomp'06), Ensenada México (2006).
21. Szulanski, G.: Intra-Firm Transfer of Best Practices Project. American Productivity and Quality Centre, Houston Texas (1994) 2-19.
22. van Elst, L., Dignum, V., Abecker, A.: Agent-Mediated Knowledge Management. International Simposium AMKM 2003. Springer, Stanford, CA, USA (2003) 1-30.
23. Wang, A., Reidar C., Chunnian, L.: A Multi-Agent Architecture for Cooperative Software Engineering. Proceedings of the Eleventh International Conference on Software Engineering and Knowledge Engineering (SEKE'99), Germany (1999) 162-169.
24. Wiig, K.M.: Knowledge Management: Where Did it Come from and Where Will it Go? : Expert Systems with Applications, Vol. 13 (1997) 1-14