A Multi-agent System for Knowledge Management based on the Implicit Culture Framework

Enrico Blanzieri Paolo Giorgini Fausto Giunchiglia
Claudio Zanoni

Department of Information and Communication Technology
University of Trento - Italy
via Sommarive 14, 38050 Povo Trento
{enrico.blanzieri,paolo.giorgini,claudio.zanoni}@dit.unitn.it

Abstract:
We present an implementation of a multi-agent system whose goal is to solve the problem of tacit knowledge transfer by means of sharing of experiences. In particular, we consider experiences of use of pieces of information. Each agent incorporates a systems for implicit culture support (SICS) whose goal is to realize the acceptance of the information suggested. The SICS permits a transparent, namely implicit, sharing of the information about the use, e.g. requesting and accepting, of pieces of information.

1 Introduction

In Knowledge Management, knowledge is categorized as being either codified (explicit) or tacit (implicit). Knowledge is said being explicit when it is possible to describe and share it among people through documents and/or information bases. Knowledge is said being implicit when it is embodied in the capabilities and abilities of the members of a group of people. Experience can be seen as a way of access and share this kind of knowledge. In [NT95], knowledge creation processes have been characterized in terms of tacit and explicit knowledge transformation processes, in which, instead of considering new knowledge as something that is added to the previous, they conceive it as something that transforms it. Supporting by means of IT systems the transfer of tacit knowledge, namely experience, among people in organizations represents a challenge whose difficulties are mainly in the need of explicitly representing tacit knowledge.

In [BG00] we have introduced the notion of Implicit Culture that can be informally defined (see [BGMR01a] for a formal definition) as the relation existing between a set and a group of agents such that the elements of the set behave according to the culture of the group. Systems for Implicit Culture Support (SICS in the following) have the goal of establishing an Implicit Culture phenomenon that is defined as a pair composed by the set and the group, in Implicit Culture relation. Supporting Implicit Culture is effective in solving the problem of improving the performances of agents acting in an environment where moreskilled agents are active, by means of an implicit transfer of knowledge between the group and the set of agents. In particular, Implicit Culture can be applied successfully in the context of knowledge management. The idea is to build systems able to capture implicit knowledge, but instead of sharing it among people, change the environment in order to make new people behave in accordance with this knowledge. As a first step in this direction we have shown how information retrieval problem can be posed in the implicit culture framework [BGMR01b]. In this framework supporting an Implicit Culture phenomenon leads to a solution of the problem of transfer tacit knowledge without the need to explicitly representing the knowledge itself.

Some assumptions underlie the concepts of Implicit Culture, Implicit Culture Phenomenon and SICS. We assume that the agents perform situated actions. Agents perceive and act in an environment composed of objects and other agents. In this perspective, agents are objects that are able to perceive, act and, as a consequence of perception, know. Before executing an action, an agent faces a scene formed by a part of an environment composed of objects and agents. Hence, an agent executes an action in a given situation, namely the agent and the scene at a given time. After a situated action has been executed, the agent faces a new scene. At a given time the new scene depends on the environment and on the situated executed actions. Another assumption is that the expected situated actions of the agents can be described by a cultural constraint theory. The action that an agent executes depends on its private states and, in general, it is not deterministically predictable with the information available externally. Rather, we assume that it can be characterized in terms of probability and expectations. Given a group of agents we suppose that there exists a theory about their expected situated actions. Such a theory can capture knowledge and skills of the agents about the environment and so it can be considered a cultural constraint of the group. Agents and objects, i.e. the environment, are specified for each application.

The goal of a SICS is to establish an implicit culture phenomenon. The general architecture we proposed in [BG00](Figure 1) allows to establish an implicit culture phenomenon by following two basic steps: defining a cultural constraint theory $\mathbf{\Sigma}$ for a group G; and proposing to a group G' a set of scenes such that the expected situated actions of the set of agents G' satisfies $\mathbf{\Sigma}$. Both steps are realized by using the information about the situated executed actions of G and G'. Implementation of a SICS was presented and shown to be effective in [BGMR01a] and [BGMR01b] where an unique SICS for the whole group of agents showed to improve agent coordination and reproduced collaborative filtering functionalities respectively.

In this paper, we propose a multi-agent architecture for knowledge management where each agent incorporates a SICS. The multi-agent architecture permits the basic operations of the SICS to be performed in a less invasive way. In fact, the agents contribute to propagate the information about the actions of the user to other agents. The system also adopts a distributed point of view of knowledge management opposed to a centralized one as pointed out by [BBT02]. The SICS incorporated in the agents can be seen as a generalization of a memory-based collaborative filtering that makes intensive use of similarity-based retrieval.

The paper is organized as follows: the next section presents the multi-agent architecture and Section 3 draws conclusions and future directions.

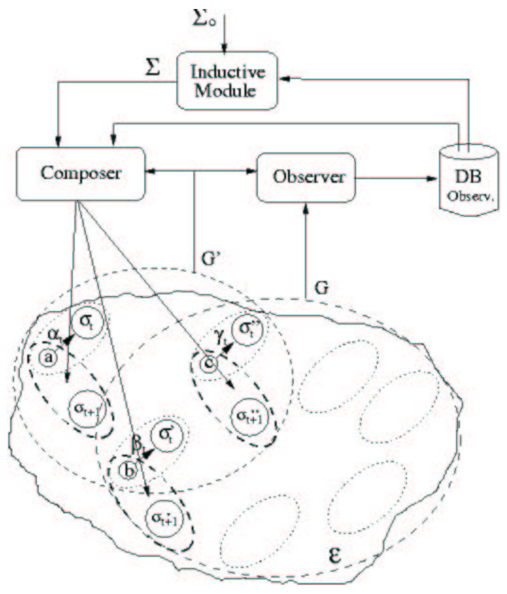

Figure 1: The basic architecture for Systems for Implicit Culture Support consists of the following three basic components: *observer* that stores in a data base (DB) the situated executed actions of the agents of G and G' in order to make them available for the other components;*inductive module* that, using the situated executed actions of G in DB and the domain theory $\mathbf{\Sigma}_0$, induces a cultural constraint theory $\mathbf{\Sigma}$; *composer* that, using the cultural constraint theory $\mathbf{\Sigma}$ and the executed situated action of G and G', manipulates the scenes faced by the agents of G' in such way that their expected situated actions are in fact cultural actions with respect to G. As a result, the agents of G' execute (on average) cultural actions w.r.t. G, and thus the SICS produces an Implicit Culture phenomenon. In the figure, the agents a, b, and c performs the actions α, β and γ while they face the scenes σ_t, σ'_t and σ''_t respectively. The composer produces a set of scenes σ_{t+1}, σ'_{t+1} and σ''_{t+1}

2 A Multi-agent System based on Implicit Culture

In this section we present the multi-agent system based on the Implicit Culture we have developed for Knowledge Management applications. The system has been built using JADE (Java Agent Development Framework) [BPR00], a software development framework for developing multi-agent systems conforming to the FIPA standards [FIP]. Basically, the system is a collection of personal agents that interact one another in order to satisfy the requests of their users. Each agent uses locally the SICS to suggest both its user and the other agents. Applying the SICS locally, each personal agent is able to provide suggestions from its perspective, namely on the base of the information it has collected observing the behavior of its user and those of the agents with which it has interacted with. In our system we have extended the FIPA protocols in order to allow the agents to exchange feedback about how the users use the information suggested by their personal agents.

A user asks her personal agent about a keyword and the agent starts to search for documents, links, and references to other users, related to the keyword. The personal agent tries to suggest the user using the observations done in the past on the user's behavior and on the behavior of the users whose personal agents it interacted with. Alternatively, the personal agent can submit the request to other agents which will treat the request as it were done by their users. In this case, however, the suggestions can include also other agents to contact. The selection of the agents to send the request is done applying locally the SICS again.

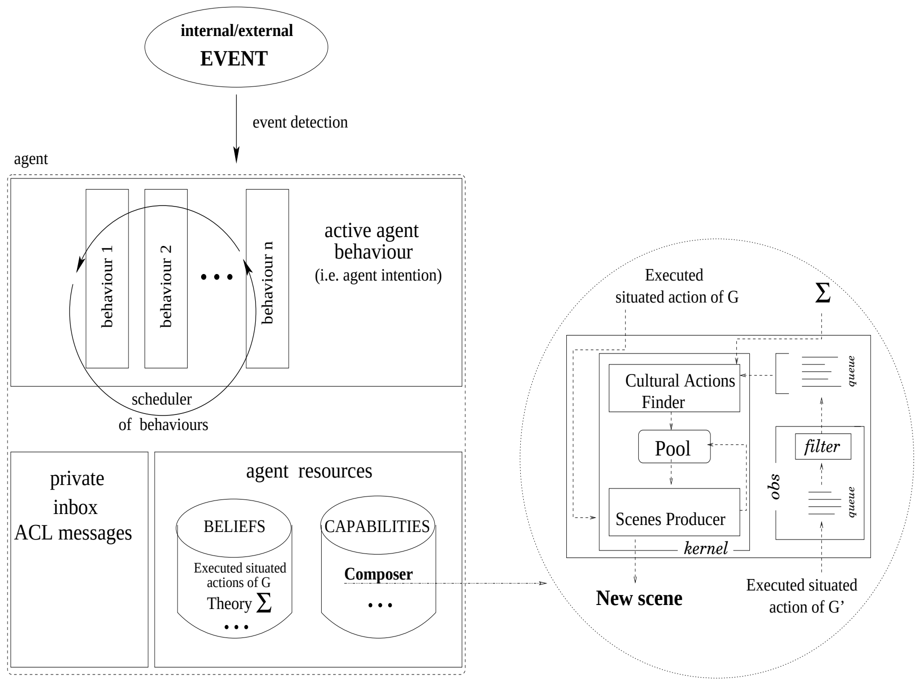

Figure 2: Internal architecture of a JADE agent implementing a SICS

Figure 2 presents the general architecture of each single personal agent implemented with JADE. The architecture of a JADE agent consists of four main components: *Behaviors*, *Scheduler*, *Inbox* , and *Resources*. In our implementation we have:

- *Behaviors*, an agent is able to carry out several concurrent tasks in response to different internal and external events. All tasks are implemented as behavior objects; we have a specific behavior for the SICS. A request from the user or from another agent activates the SICS behavior.
- *Scheduler*, that determines which behavior is the current focus of the agent and consequently it selects an action to perform.
- *Inbox*, a queue of incoming messages (ACL). It contains the messages coming from the user as well as those from other agents.

- *Resources*, consisting of beliefs and capabilities. The agent's beliefs are the information available to the agent and the capabilities are particular functionality used in the behaviors. In our implementation the three main components of the SICS (observer, composer and inductive module) are three different capabilities and the observations and the cultural constraint theory are stored as beliefs. Additionally, each personal agent has beliefs about a local schema useful to organize the information available. This schema is not mandatory.

The capability (the composer) and the beliefs (situated executed actions and cultural constraint theory) related to the SICS and reported in Figure 2 are presented in details in [BGGZ]. Here we concentrate on the other beliefs and behaviors. Each personal agent has among its beliefs a local schema in order to organize information available to its user. Basically, the schema is a tree where the nodes are labeled with strings that the user uses to describe her own areas of interest and the leaves are links. A link can be a reference to a document stored locally in the user system or it can be an Internet address or a reference to a person (e.g., a phone number, an email address or just the name of the person). The schema is a conceptual representation of how the user organizes locally its information and it does not say anything about how this representation matches with those of the other users. The schema is represented in XML.

Figure 3 shows the algorithm used by a personal agent when it receives a request of information from its user or from some other agent. The global variable `result` contains both links and names of agents of the platform. If the message is a query the SICS behavior is activated and it modifies `result`; if no agents appear in `result` the DF agent is added to it in order to propagate the query in any case; if the sender of the query is the user the links contained in `result` are sent back and a query is sent to all the agents contained in `result`. If the message is a reply from an agent the complete `result` (links and agents) is sent whereas an incomplete `result` (links only) is sent in case the reply comes from the user.

The agents interact one another using the FIPA-Iterated-Contract-Net Protocol, that starts with a call for proposal to perform a given action. In particular, we use the call for proposal for checking the availability of an agent to perform a search action. Differently, the user interacts with its personal agent using the FIPA-Query Protocol. Additionally, we have introduced a third protocol for the propagation of the user feedback about the suggestions provided to him. In particular, the protocol guarantees that the user informs the personal agent about the acceptance of the refusing of a suggestion, and that the personal agent informs about this the other agents it asked. In practice, the sending of an inform whose content is "accept" is triggered by an action of the user, e.g. following a link, maintaining it implicit.

An example of interaction. Let us consider the case in which a *user* searches information about *"train timetable"* and asks his *personal agent*. Let us suppose that the SICS suggests an Internet address (*www.fs-on-line.it*) and another agent, *agent-1*. The personal agents inform the user about the address *www.fs-on-line.it* and send a request to *agent-1*. Supposing that *agent-1* replies with another internet address *www.trenitalia.it* and another agent, *agent-2*, then the *personal agent* will send a request to *agent-2*. When *agent-2* replies with

```
global result
for all message in INBOX do
  if (message.type == 'query') then
    result := nil
    SICS-behavior(query.sender,query.content, result.links,result.agents)
    if (result.agents == nil) then
      add(DF,result.agents)
    end if
    if (query.sender == user) then
      inform(self,user,result.links)
      for all result.agent do
        request(self,result.agent,query.content)
      end for
    end if
  else if (message.type == 'reply') then
      if (reply.sender == user) then
        inform(self,user,result.links)
      else inform(self,message.sender,result)
      end if
    end if
  end if
end for
```

Figure 3: The algorithm used by the personal agent for processing the messages

th email address *info@trenitalia.it*, the *personal agent* informs the user with the results it has collected (namely, *"www.fs-on-line.it"* +*"www.trenitalia.it"*+*"info@trenitalia.it"*). Finally, if the *user* executes an action considered of acceptance for example of *"info@trenitalia.com"* an inform with that content is sent. The *personal-agent* informs *agent-2* because it has suggested such an address, and *agent-1* because it has suggested *agent-2*. Figure presents the sequence of messages exchanged by the agents.

The example shows how the variant of the FIPA communication protocol permits to the agents to propagate the feedback of the user. In this way each personal agents has access locally to information about the use of the information done by the requester. The availability of the information permits to the agent to observe a wider number of actions permitting the transfer of knowledge between the users. In fact, if the personal agent would limit its observations only to the actions performed by its user the effect achieved by the user would be a simple personalization. With the communication protocol we adopted each SICS can observe also actions done by the users of the personal agents he has been put in contact to. It is worth to note that this is transparent to the user. As a summary, the personal agent acts on behalf of the user in a complex way. It uses the observations of the behavior of its user to provide a better service to the user herself (personalization) and to the other users (collaboration). Moreover and with the same goal, it integrates locally the observations of the user with the observations of the other users and contribute to propagate the observations of its own user in order to give feedback to the other agents. In other terms the user delegates to the personal agent the capacity of sharing information about

1. *request(user,personal-agent,“train timetable”)*
2. *inform(personal-agent,user,“www.fs-on-line.it”)*
3. *request(personal-agent,agent-1,“train timetable”)*
4. *inform(agent-1,personal-agent,“www.trenitalia.it”+“agent-2”)*
5. *request(personal-agent,agent-2,“train timetable”)*
6. *inform(agent-2,personal-agent,“info@trenitalia.it”)*
7. *inform(personal-agent,user,“www.trenitalia.it”+“info@trenitalia.it”)*
8. *inform(user,personal-agent,“accept(info@trenitalia.it)”)*
9. *inform(personal-agent,agent-1“accept(info@trenitalia.it)”)*
10. *inform(personal-agent,agent-2,“accept(info@trenitalia.it)”)*

Figure 4: The interaction example

use of information.

3 Conclusions and future work

We have presented a multi-agent system that exploits the architecture of the Systems for Implicit Culture Support in order to solve the problem of transfer of tacit knowledge in a knowledge management context. We have argued that tacit knowledge transfer requires the sharing of experiences and that the main difficulty relies in the need of explicitly representing the tacit knowledge. Our approach aims to by-pass the problem of explicit representation.

The system incorporates a SICS in each agent. The SICS is used in order to provide information to the user and also to the other users by means of a communication protocol between the agents. The SICS observes the local actions of its own user and, thanks to a variant of the communication protocol w.r.t FIPA standards, also the actions of other users on the information suggested. The multi-agent architecture permits this exchange of information about the actions improving the range of actions that each local SICS can observe. The overall effect is an implicit transfer of information about the use of the items suggested. In other terms, the system supports a sharing of the experience of the use of some pieces of information.

As argued in [BBT02], an architecture for a knowledge management system should be designed with the distributed social form in which knowledge is created within organizations. Basic characteristics of agent-based systems, such as, autonomy, intentionality and sociability, can be used to design distributed knowledge management systems that allow us to overcome the limitation of centralized systems. A purely distributed approach to knowledge management is being consistently addressed in the EDAMOK project [BBT02]. The system-development part of the project adopts a peer-to-peer architecture with an explicit notion of *context*. Based on the published material it is possible to sketch some differences. Architecturally, our agent-based approach relies on different architecture and technology and insert a learning functionalities in order to discover and propagate information about the other entities (agents or peers) in the system. Theoretically, their approach tends to solve *a posteriori* the problem of matching between the local perspectives (contexts) whereas our system tends to support the formation of compatible local perspectives.

In our opinion the present proposal represents a viable way of supporting the transfer of tacit knowledge between individuals in an organization. Each personal agent contributes locally to a realization of an implicit culture phenomenon. It is important to note that the local perspective of each agent permits the existence of different practices, given the fact that not all the agents will converge to the same set of observations and consequently of suggestions. Further work requires an experimentation on the field of the system proposed. On the field, the notion of implicit culture can be of great help in order to boost acceptance of the transfer of tacit knowledge, namely experience. In fact, the user can be explicitly asked to participate at the knowledge transfer process without imposing any specific additional activity. On the other hand, accepting to have her own actions partially propagated in the multi-agent system can be facilitated by the idea of contributing to a culture and by the perspective of sharing the advantages.

References

[BBT02] Matteo Bonifacio, Paolo Bouquet, and Paolo Traverso. Enabling Distributed Knowledge Management. Managerial and Technological Implications. *Informatik/Informatique*, III(1), 2002. Special Issue on Knowledge Management, S. Lueg (ed.).

[BG00] E. Blanzieri and P. Giorgini. From Collaborative Filtering to Implicit Culture. In *Proceedings of the Workshop on Agents and Recommender Systems*, Barcellona, 2000.

[BGGZ] Enrico Blanzieri, Paolo Giorgini, Fausto Giunchiglia, and Claudio Zanoni. Personal Agents for Implicit Culture Support. DIT Technical Report DIT-02-084.

[BGMR01a] Enrico Blanzieri, Paolo Giorgini, Paolo Massa, and Sabrina Recla. Implicit Culture for Multi-agent Interaction Support. In Carlo Batini, Fausto Giunchiglia, Paolo Giorgini, and Massimo Mecella, editors, *Cooperative Information Systems, 9th International Conference - CoopIS 2001*, volume 2172 of *Lecture Notes in Computer Science (LNCS)*. Springer-Verlag, 2001.

[BGMR01b] Enrico Blanzieri, Paolo Giorgini, Paolo Massa, and Sabrina Recla. Information Access in Implicit Culture Framework. In *Proceedings of the Tenth ACM International Conference on Information and Knowledge Management (CIKM 2001)*, Atlanta, Georgia, November 2001.

[BPR00] F. Bellifemine, A. Poggi, and G. Rimassa. Developing multi-agent systems with JADE. In *Seventh International Workshop on Agent Theories, Architectures, and Languages (ATAL-2000)*, Boston, MA, 2000.

[FIP] FIPA. *Foundation for Intelligent Physical Agents*. http://www.fipa.org.

[NT95] I. Nonaka and H. Takeuchi. *The Knowledge Creating Company*. Oxford University Press, New York, 1995.