International Journal of Computer Science and Applications

Vol. 6, No. 2, pp 1-18, 2009

AGENT-BASED KNOWLEDGE COMMUNITIES

PIERRE MARET
Université de Lyon, F-69003, Lyon, France
UMR CNRS 5516, Laboratoire Hubert Curien
pierre.maret@univ-st-etienne.fr

JACQUES CALMET
University of Karlsruhe (TH), Computer Science, Am Fasanengarten 5
Karlsruhe, D-76131, Germany
calmet@ira.uka.de

Virtual communities are becoming increasingly popular, particularly on the Internet, as a means for like-minded individuals to pursue common goals. It is a way to access and to share knowledge and information among participants of such communities without physical or hardware constraints. The concept of a community of interest can be supported in a virtual community in order to bring the appropriate parties together and to share their knowledge with each other. The objective of our work is to investigate a community abstraction approach to agent-based knowledge management.

We propose a general model that extends the abstraction of an agent, such that it becomes an actor within a knowledge community. In our bottom-up model, agents themselves, software or human, are the members of the virtual knowledge sharing communities. Agents can choose to join, leave, create and destroy communities, and can be member of many communities simultaneously. A prototype design illustrates this approach.

Keywords: Virtual communities; multi-agent systems; knowledge management; abstraction.

1. Introduction

The expected goal of any organization (true or virtual) is ultimately to make decisions or at least to support decision making on a wide range of issues. Nowadays, the trend is to rely on learning methodology in most facets of what is often called computational intelligence. However, this is only possible when knowledge is sufficient and well enough mastered. This leads to investigate the knowledge facet of our project. A sentence cited in [Thomas *et al.* (2007)] says, where ICT means Information and Communication Technologies, "Whether involved in virtual teams focused on global, distributed projects or not, most knowledge workers currently spend most of their time working virtually through the ICTs, but do not take advantage of the benefits available". It means that knowledge workers are not taking full advantage

of their craft. This is not surprising in some sense since it is well-known that few, if any, commercial "knowledge product" do exist although our society claims that its growth will be based upon knowledge. A trivial remark is that knowledge has always been a marker of humanity. According to some philosophers including Ferry, ontology has its origins in Greek philosophy and specifically in Stoicism. The first definition seems to be due to Zenon von Kition (or Zeno of Citium depending on the language used) 334-262 BC. This definition was however not written down before Ciceron. Very simply stated, "theoria" was both an ontology (a doctrine defining the basic structure of a human being) and a knowledge theory. Today an ontology is defined as structured knowledge thus almost in agreement with this original definition. It has been mostly forgotten that algorithms have summarized the sort of mathematically defined knowledge that was required to program computers before the advent of artificial intelligence (AI). Then, knowledge and its representations was one of the main ingredients of AI. Another trivial fact is that multiagent systems have been defined to manage problems in distributed AI. We set our approach in the framework of multiagent systems.

Nowadays, the concept of knowledge is too often limited to the concept of ontology although knowledge comes in very different species: structured or not, certain or not, complete or not for instance. Knowledge engineering is too often inefficient for routine applications. Indeed, as previously stated, the amount of available knowledge is blowing up and generates such large knowledge bases that they are becoming difficult to master. Mediators provide a very sound theoretical background to query different classes of knowledge. One of the authors did design the KOMET mediator [Calmet *et al.* (1997)]. KOMET does permit to infer knowledge with a sound semantical meaning but is much too large for most applications. For instance, such bases and systems are, with the presently available technologies, too large to be ported on intelligent portable or mobile devices. Also, it is a top-down approach where the goal is to master all existing knowledge bases.

This paper deals with virtual knowledge communities (VKCs). We started with the decision to base our approach on Virtual Communities rather than on classical knowledge tools because this allows a bottom-up approach. VKCs are a way to let the users build their own knowledge sources, to design them as different threads in a given methodology. The specific goal of this paper is to present a software package that implements and manages VKCs. We want to point out that VKCs are a possible way to define building stones upon which to approach virtual organizations in a general meaning. Some of our previous publications cited in this work demonstrate that it is possible to define virtual enterprises through their main asset: their corporate knowledge. Then, we did show that VKCs may be used to define the corporate knowledge of a virtual or a true enterprise.

We want to master knowledge that is distributed, dynamic (flowing through a Grid for instance) and subjective (assessed by "users" or "clients"). The emphasis is shifted towards knowledge identification (instead of well-known sources) and knowl-

edge exchange and sharing. Also, VKCs are an answer to facilitate the evolution of knowledge engineering from mediator systems to pervasive or ambient or human centered computing and to the grid technology. They allow portability to intelligent mobile devices. This enables for instance personalized evacuation modeling for high risk situations such as fires in buildings, heavy smoke in tunnels or delayed intervention time for rescue crews but also for more usual hazardous cases such as the support of isolated elderly people.

Finally, the concept is arising from a proposed agent-oriented abstraction [Calmet *et al.* (2004)] thus providing a sound theoretical background. An important feature is to link the management of knowledge to trust and risk management. Indeed, to base a society on knowledge as Europe claims, requires to enforce trust and security. A well-known example is that the security of a grid is defined by its least secure element. This implies to maintain information and knowledge on all elements of its architecture. Also, it was shown that either in decision making under uncertain knowledge [Yang (2007)] or in defining a secure gateway for Web services [Huang (2007)] a key feature is to share knowledge efficiently. This is made relatively easy when defining knowledge as VKCs. However, we do not cover these applications here. In fact, the software we describe in this paper is applied in [Yang (2007)] and [Huang (2007)].

2. Virtualization

Nowadays, virtualization is becoming a buzzword. Our world is becoming virtual. This trend started many years ago. It was immediately noticed in the entertainment industry and the public at large since, for instance, virtual reality is nowadays a basic ingredient to design many movies. Also, in ICTs, computer science and business communities the word "virtual" assess many new meaningful approaches.

Virtual enterprises and networks are becoming the right building stones of the e-economy. A Web search with either of the following key-words ECOLEAD, MANTICORE or FEDERICA[a], which refer to European projects, will return relevant information on this new paradigm. These concepts are spanning a large spectrum from virtual organizations and enterprises to virtual campuses or virtual representation of the environment of patients with psychological disorder. For several years the treatment of psychological disorders is achieved whenever possible by virtualizing the patient and thus projecting him into a virtual world. Some tools based on using neural networks to this end are described in [Fleischer (2005)]. There are even claims that we have a virtual vision of the world. The argument is that the image forwarded to the brain by the retina is solely virtual. It is however challenging to find meaningful references for this claim.

[a]See : http://ecolead.vtt.fi/, http://www.i2cat.cat/i2cat/ImgsPortal/FitxerContingut1389.pdf, http://www.fp7-federica.eu/

3. Virtual organization and knowledge

3.1. *Virtual enterprise, virtual team, virtual community*

A virtual enterprise is usually defined as the temporary or permanent alliance of organizations for the accomplishment of a task by way of information and communication technology. At a smaller scale, a virtual team is a group of people that rely primarily or exclusively on electronic forms of communication to work together in accomplishing goals [Palmer *et al.* (1997)]. While Kishore [Rajiv *et al.* (2002)] would prefer the expression Virtual environment instead of virtual enterprise or team, Seiber and Griese [Sieber *et al.* (1997)] mention that virtuality is based on both a real added-value chain without any newly created physical infrastructure, and on the exclusive use of information technology (IT) to support the business processes.

A Virtual Community can be defined as a group of people sharing common interests and making use of electronic forms of communication for exchanges. Thus, a virtual community is not necessarily related to a task, rather to a topic and to some knowledge. Since it can also be related to a task, one may say that it is more general than a virtual enterprise and a virtual team. The knowledge detained by people belonging to an organization, and eventually exchanged within the virtual and true communities represent the knowledge detained by the organization. It can be called corporate knowledge [Maret *et al.* (2005)].

3.2. *Distributed Knowledge Management*

The term Distributed Knowledge Management (DKM) has been used in [Bonifacio *et al.* (2002)] and [Bonifacio *et al.* (2003)], emphasizing the concept of Knowledge Nodes. Knowledge Nodes stress the values of autonomy of actors within a company and of coordination in-between actors. Authors argue that many traditional knowledge management systems, which are based on the idea that "all perspectival aspects of knowledge should be eliminated in favor of an objective and general representation of knowledge", are deserted by their users due to the "false epistemological and organizational assumptions made during the design of the systems".

Distributed Knowledge Management is based upon two fundamental principles; the *principle of autonomy*, which states that each organizational unit should be granted a high degree of autonomy, and in particular semantic autonomy, to manage its local knowledge and conceptualizations, and the *principle of coordination*, which states that each unit must be enabled to exchange knowledge with other units, without imposing the adoption of a single, common interpretative schema, but by using a method of mapping other unit's context onto its context from its own perspective.

A Knowledge Node (KN) is then presented as a means for obeying these principles and is defined as a reified organizational unit, which does exhibit some degree of semantic autonomy. Each KN has its own explicit representation of its reified

perspective of a given community called a context. This can be either a category system or an ontology or a collection of guidelines or a business process. Each KN also has a software agent associated to it, which knows the KN context. This agent acts as a go-between between its KN and other KN's by supporting users of a KN to compose outgoing queries, and answering incoming queries from other KN's.

3.3. *Toward a bottom-up approach of corporate knowledge*

We strongly support the principle of autonomy of actors within an organization. Moreover, the knowledge of an organization is the knowledge of its actors. Actors are the individuals, but also any artifact that holds knowledge and (automated) decision ability (for instance a database and related software programs). Thus, we feel that these principles are so strong that the idea of using Knowledge Nodes should go right the way down from the level of organizational unit to single units actors. Doing so, the corporate knowledge is no longer only a top-down concept, where structural and conceptual entities are imposed to the lower levels. Corporate knowledge is rather considered in a bottom-up approach, which is much more compliant with real world and may be the only way to implement effective knowledge management that will work well in practice.

We propose to carry out the modeling of the actors within an organization in using the Multi-Agent System paradigm. MAS have been invented to tackle problems within distributed computing. This is convenient for modeling autonomous and communicating actors, and it has been also proposed in [Elst *et al.* (2003)] or in [Dignum (2003)]. In addition to these research works, we will propose the so-called community abstraction, which is a way to aggregate actors into virtual communities in order to develop local, on the fly knowledge exchanges. The community abstraction will thus become one of the two cornerstones of knowledge management when the cornerstones refer to top-down and bottom up views of knowledge management.

4. Virtual Knowledge Communities

4.1. *Agent paradigm and agent-oriented abstraction*

[Ferber (1997)] proposes the following definition of an agent: "an abstract or physical autonomous entity which performs a given task using information gleaned from its environment to act in a suitable manner so as to complete the task successfully. The agent should be able to adapt itself based on changes occurring in its environment, so that a change in circumstances will still yield the intended result."

An agent can also be described as an active object with the ability to perceive, to reason and to act. Moreover, we assume that agents possess their own knowledge and can communicate with each other. This concept of agents can be extended to the concept of intelligent agents which have the following additional properties [ASAP (2008)]. *Autonomy*: An agent does not depend on the actions of an external entity (human or other). It can act in an autonomous way and has the control

of its behavior, its state and its actions. *Mobility*: An agent can move and decide to join or not a system or an exchange platform. *Pro-activity*: An agent is able to make decisions and to take initiatives. It works towards personal goals, has its own behavior depending on its individual objectives. It can achieve complex actions which can be divided into sub-tasks organized according to its behavior. *Sociability*: Agents can interact and communicate with others. *Reactivity*: An agent has it's own view of the world and it can react to the changes in the environment. *Temporal continuity*: Agents are continuously running processes.

Also [Ferber (1997)] gives the following definition for a multi-agent system: "a system composed of a population of autonomous agents which cooperate with each other to reach common objectives, while simultaneously each agent pursues individual objectives."

We can thus see a Multi-Agent System as a system in which autonomous agents can communicate, exchange their individual knowledge and cooperate in order to solve complex problems and to achieve collective or individual goals [Huhn *et al.* (1999)]. This is the natural and most general available model for an agent-based knowledge sharing system. The Agent-Oriented Abstraction (AOA) has been proposed to abstract multi-agent systems as a society of agents [Calmet *et al.* (2004)]. The latter are composed of annotated knowledge and a decision mechanism. AOA is compliant with a societal approach of agents. Indeed, it is based on Weber's classical theory in Sociology [Weber (1922)]. Very briefly, it can be said that Weber states that a society is the result of the actions of individuals. Our approach to corporate knowledge matches exactly this view point: Corporate knowledge results from the coupling of the available knowledge with a decision mechanism governing its actors' behavior. This definition includes individual knowledge and informal exchanges as well as databases and file management systems and software such as for instance personal assistants and communication applications. Thus, corporate knowledge does not only refer to the knowledge contained in the system, but also to the means to exploit and to manage it. Therefore, in the next section, we introduce the basic concept of Virtual Knowledge Communities as a means for agents to share knowledge about a topic.

4.2. *The community abstraction*

Virtual communities are becoming increasingly popular, particularly on the Internet, as a means for like-minded individuals to meet other individuals they can learn to trust and to share and gain access quickly and efficiently to the information they are mostly interested in. The concept of a community of practice or a community of interest can be supported in a virtual community in order to bring the appropriate parties together and to share their knowledge. The advantage of this is that the members of a community centered on one specific topic or practice will only be presented with knowledge from domains they are, or at least are relatively likely to be, interested in. This knowledge needs not be something they have specifically

asked/searched for.

Many virtual communities applications already exist on the Internet. Some are using *agents* in various forms as part of the back office system. We propose an approach that extends the abstraction of an agent, such that it acts within the system, searching for or delivering knowledge within other agents and through communities. With such a model, agents can choose to join, leave, create and destroy a community, they can ask for information and send information to the community, and they can be member of several communities simultaneously. We call Virtual Knowledge Community (VKC) the virtual place where agents can meet, communicate and interact among themselves. Basically, a VKC is centered on a topic, corresponding to a domain of interest for which the interested agents have joined this community. This notion allows an increased availability of data and knowledge within the various communities.

The main basic concepts of VKCs are the Community of Communities, Agents and Community.

Community of communities is a yellow page system. Agents can hold and manage a Community of communities, or just refer to one or several Community/ies of communities. This allows agents to check existing communities and to join them according to their centers of interest. Also, within a community of communities other specific communities can be dynamically created and terminated by the agents according to their goals.

Agents are autonomous entities that have their own knowledge and can act and communicate with each other. Agent's knowledge is stored in the personal repository of the agent, which contains the relations between the concepts, the properties associated to these concepts, and the different instances of concepts and properties. The concept of knowledge cluster is introduced to represent a piece of knowledge from the agent's repository. Agents share and exchange knowledge clusters.

A *community* consists of a domain of interest (a knowledge cluster), a leader (an agent), a policy and an unspecified number of member agents. The leader has created this community to achieve a goal (corresponding to the domain of interest). Each community is associated to a single policy which defines the community and which is up to the community leader. For instance, depending on the policy, a message buffer stores for a given duration or under given rules exchanged messages within this community.

5. Virtual Knowledge Communities Design and Implementation

In this section, we first describe the design of the main concepts upon which the system is built. Then, we describe the implementation of agent's knowledge management and the life cycle of communities.

5.1. *Design of the VKC components*

In this section, we present the core concepts which are implemented in the prototype as Java classes.

CommunityOfCommunities. This class is implemented by agents through which other agents have access to a list of existing communities. An agent owner of a class instance manages a list of communities, and each time an action is made that implies a modification of this list, a message is sent to update it.

AgentViewOfCommunityOfCommunities. Every communityAgent has instances of this class which allows it to access the CommunityOfCommunities via their owners. This class is in fact an interface thanks to which the agents of the system can exchange messages with the CommunityOfCommunities.

MemberViewOfCommunity. Each agent has exactly one instance of this class for each community it is member of, and uses it as an interface to operate within each one of these communities.

CommunityPack. This class gathers all information describing a community. Each instance of the class MemberViewOfCommunity has a CommunityPack. This one is created by the leader of the community at the moment of its creation and lasts until its termination. When an agent joins the community, it obtains a reference to the CommunityPack for this community.

LeadersViewOfCommunity. This class has exactly the same role as the MemberViewOfCommunity class but seen from the point of view of the community's leader. Since the latter is also a member of the given community, it preserves a MemberViewOfCommunity nevertheless.

CommunityBuffer. Each community has a CommunityBuffer to (eventually) store the messages sent by the agents. A policy describes the management related to this buffer (access right, duration of messages,...). In the default case, only the leader can write and read in this buffer via his LeadersViewOfCommunity that has reference to the buffer. Thus, when an agent wants to write a message in the buffer, or when it wants to read one from it, it just asks the leader to do that.

5.2. *Knowledge management of the agents*

Knowledge of the agents is stored in the Web Ontology Language (named OWL). It rests primarily on two existing and linked APIs from JENA [Jena (2008)] and Protégé-OWL [Protege (2008)]. The JENA framework, an open-source Semantic Web framework for Java that provides APIs to build semantic Web applications managing the RDF and OWL languages. It also provides methods to write and read in RDF/XML, along with a SPARQL (Query Language for RDF, [SPARQL (2008)]) query engine. The Protégé-OWL plugin is described as an extension of Protégé that supports OWL. Protégé is a free, open source ontology editor and knowledge-base framework. The Protégé-OWL API provides methods to load and save OWL and RDF ontologies, and to edit these ontologies in a powerful way. This plugin is tightly linked to the previously mentioned JENA framework. Using these

two frameworks makes it possible to work with repositories filled with ontologies written thanks to the Protégé software, and also to store them while conforming to the same standard.

Our prototype implements also a class called KnowledgeCluster to represent the personal knowledge of the agents. This class provides all needed methods to carry out the most basic operations necessary for an agent to manage its ontological knowledge contained into the repository. The main and most useful methods of this class are writeToFile, extractSubCluster, addSubCluster and findClass.

The class KnowledgeCluster provides mechanisms to load and save ontologies from files. These two functions are provided by the JENA API and the Protégé-OWL plugin together. Technically, each instance of KnowledgeCluster has an *onto uri*[b] attribute which corresponds to the path of the OWL file containing the personal knowledge of the agent. Each time an agent needs to access its knowledge, this file is loaded in a *edu.stanford.smi.protegex.owl.model.OWLModel* (provided by the Protégé- OWL API). Then, this OWL Model provides all the methods to get the existing concepts, properties and individuals of the knowledge, and also to add some. To save the knowledge of an agent, the method writeToFile(String uri) of the implemented KnowledgeCluster class is needed. This method saves the OWLModel in the OWL File specified by the uri parameter.

Thanks to the KnowledgeCluster class, an agent can extract part (i.e. sub-cluster) of its repository. This function is provided by the method extractSub-Cluster(String head). This method returns a sub KnowledgeCluster related to the concept head given as input argument. The way in which a sub cluster is extracted is straightforward: the method determines all the dependencies between the head concept and all the other concepts. This means that it extracts all the concepts, properties and individuals related to this head concept. For each newly extracted concept, it undertakes the same task recursively, until the graph has been totally investigated and no more dependency does exist. A more sophisticated algorithm should be implemented in the future, in taking advantage of semantic links.

The addSubcluster method adds a sub-cluster to the existing ontology within the repository. The latter merges the new KnowledgeCluster to the existing Knowl-edgeCluster which calls this method. In order to do so, it just adds all the concepts, properties and individuals of the sub cluster that do not exist yet, and also all the inheritance links.

The findClass method returns an OWLNamedClass contained in the Knowl-edgeCluster whose name is given through the input parameter name. Its main purpose is to check if the repository contains this given class, so that an agent can assess if its knowledge overlaps a knowledge cluster within a community. For instance, if it processes a knowledge request for a concept "Car", it just calls findClass(Car), and if the result is not empty, then it can extract a sub cluster with head "Car"

[b]URI : Uniform Resource Identifier, the generic term for all types of names and addresses that refer to objects on the World Wide Web.

and send it to the interested agent.

5.3. *System implementation*

Implementation of VKCs have been done under the Jade system, a Java based software development framework that conforms to FIPA standards for intelligent agents [JADE (2008)]. The fact that JADE conforms to all the appropriate FIPA standards and enables portable and easily maintainable agent development using the Java language, makes it a perfect framework in which to develop a multi agent-based system for knowledge management using virtual knowledge communities. The implementation intends to address the knowledge sharing activities of the agents, so all the other behaviors of the agents are not considered in the implementation.

The UML diagram of the VKC component is depicted in figure 1. It illustrates that most classes of the VKC package implement the class Jade.util.leap.Serializable so that these objects can be transmitted via SOAP messages for Web Services, or via the ACL messages exchanged within the communities. It can also be noticed, that CommunityOfCommunitiesAgent and CommunityAgent both implement the class Jade.core.Agent. Each of these two types of agent have a graphic interface to display their messages to the user of the system, which is respectively the CofCGui and the AgentGui. Furthermore, we can see in this figure all the components which belong to an agent and which have been outlined in the beginning of this section: MemberViewOfCommunity and LeadersViewOfCommunity to access all the communities it belongs to or is a leader, and AgentViewOfCommunityof-Communities which provides access methods to the CommunityOfCommunities. We should finally note that a CommunityBuffer can contain an unspecified number of Jade.lang.acl.ACLMessages sent by the agents of the community. In the next subsections, the agent communication is depicted, and then the life cycle of a community and of the agents running within it is described.

5.4. *Agent communications*

Agents in JADE communicate with each other using messages. Non-leader agents send all messages to community leaders, since leaders are in charge of allowing agents to join their community, and control access to the community's message buffer.

All agents can send the following types of messages:

JOIN_REQUEST message if they wish to join a community. In our implementation a leader will always accept such a request, but this type of message could be used also to implement security mechanisms for controlling access to communities.

LEAVE_REQUEST message if they wish to leave a community. This will be accepted always by the leader.

INFORM message to share something with the community. The content slot contains the DomainCluster that expressed the knowledge the agents wish to share with the community.

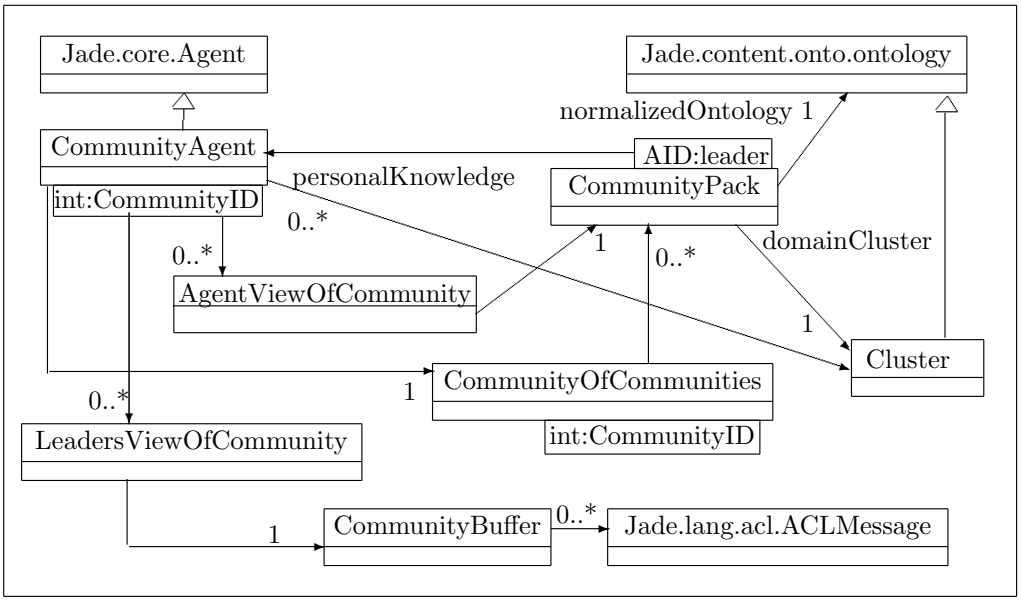

Fig. 1. UML Class diagram of the prototype.

REQUEST message if they want to find some specific knowledge in the community. The content slot will contain the DomainCluster specifying the type of knowledge they require.

READ message if they wish to read the next message on the buffer that they have permission to read, and have not yet read. The content slot specifies the time of the last message read, i.e. the agent wishes to read any messages that were added to the buffer after the specified time. A leader agent responds to this message type in forwarding an appropriate INFORM or REQUEST message to the agent, or in sending a REFUSE message, depending on the role of the agent.

ACCEPT message if they wish to acknowledge the receipt of a NEW_ROLE message.

All agents will expect to receive the following types of message from leaders:

ACCEPT message if the action that their message specified was accepted. If the original message sent was a join request, then the agent's role will be returned in the content slot. If the original message was an inform or request, then a score will be sent in return, to provide feedback on the quality of the agent's communication.

REFUSE message if the action that their message specified was refused, due to lack of permissions or otherwise.

NEW_ROLE message telling them whether their role within the community has changed. The content slot will contain the role id. This is sent by a leader when an agent fulfills specific criteria.

INFORM message informing the agent that some new knowledge is available. The content slot will contain a DomainCluster representing the new knowledge.

REQUEST message asking the agent for some new knowledge. The content slot will contain the DomainCluster specifying the type of knowledge they require.

5.5. *Life cycle of a community*

The life cycle of a community can be divided into several steps that are now described.

Community creation

An agent creates a community with a head concept corresponding to the piece of knowledge it wants to extend. This corresponds to a domain of interest of this new community. The agent becomes then the leader of this community and it advertises for it into a Community of communities. Note that it may be also the owner of a Community of communities.

Community life

The community life consist of exchange of message in-between the leader and the community members. Subparts of the community life if the community expansion and reduction, and the content exchanges.

The *community expansion* starts with the community creation posting in Community of communities. Agents that want to join a community send a join request to the leader. The latter processes these requests and sends a join ACCEPT message as reply to the agents to acknowledge their addition to the community.

During *content exchanges* agents which want to take advantage of the community to extend their personal knowledge, send knowledge requests for a given head concept to the leader which adds them to the list of messages in the community buffer.

Periodically, each community leader reads the next message which was sent to him. If it is a join or leave request, the new agent is added to or removed from the community member list. If it is a knowledge request, it checks if it can answer it. If yes, it extracts a corresponding sub-cluster from its knowledge repository and sends it to the interested agent. It puts also the message in the community buffer such that it can be read by the other agents. Lastly, if it is a knowledge cluster, it merges it with its personal repository[c]. Periodically, the members ask the leader for the next unread message in the community buffer. When it is a community's termination inform message, they know that the community was terminated by the leader thus, they must remove their MemberViewOfCommunity for this community.

[c]Note that it cannot be guaranteed that the meaning of the information (to be) merged with the agent's repository can be semantically understood by the agents. Solutions rely on references to normative shared ontologies, and in the case of user-controlled agents, on the owner's own decision.

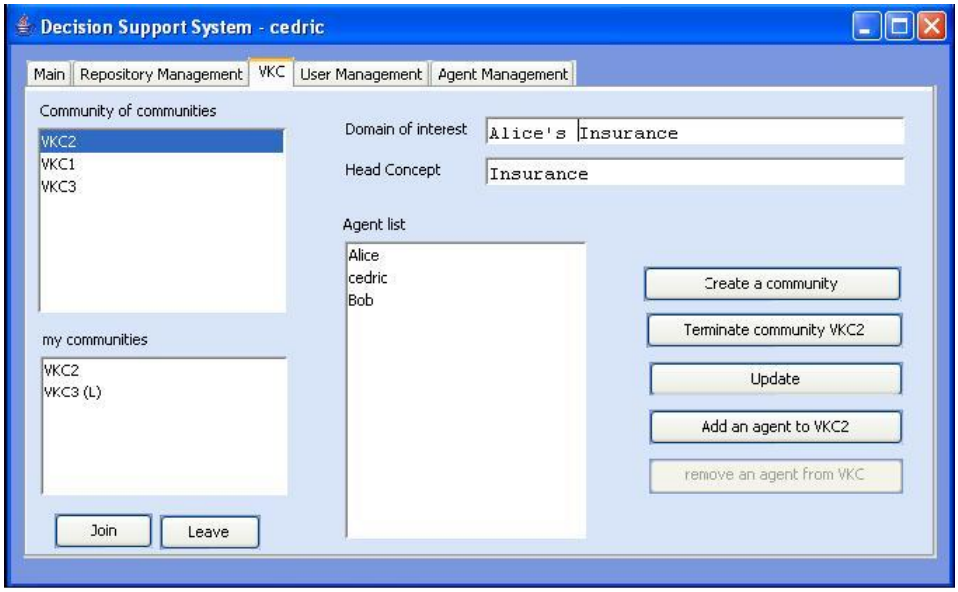

Fig. 2. Screenshot of the Community Management Tab

If it is a knowledge request, they check if they can answer to it, and if yes, they extract a corresponding sub cluster from their knowledge base and send it to the requiring agent. Members can also directly receive knowledge clusters in reply to their knowledge requests. Then, they merge it with their personal repositories.

Community extinction or reorientation

A leave request is sent to the leader from a member which decides to leave the community. The agent is then removed from the MemberViewOfCommunity list of this community.

A leader can decide to leave the leadership of a community or to terminate it. The transfer of the leadership is a message sent to members. A community's death inform message is sent by the leader willing to terminate the community. CommunityOfCommunitiesAgent remove the community from the CommunityOfCommunities's list, and the leader removes its MemberViewOfCommunity and LeadersViewOfCommunity related to this community.

Note that in the prototype, all the actions such as creation, removal, joining and leaving a community are orchestrated by the user who controls the corresponding agent through the VKC tab of the graphic user interface (Fig. 2). The agents plays then the role of a user assistant for distributing and gathering knowledge from the system.

6. Discussion

The concept of corporate knowledge can be tackled with at least two facets. A first one lies fully in business and refers to the amount and quality of know-how and information that is available throughout a company. The second one is set in the framework of information technology and is tightly linked to the various and diverse paradigms that are used to represent and manage knowledge.

In a system with no formal knowledge management system, all actors rely on themselves alone to identify and locate all knowledge requirements and useful sources for satisfying these requirements. The only interaction between actors is informal. This is clearly not an optimal methodology to maximize knowledge management potential, since it provides no formal means to re-use knowledge found by other actors, nor any means for improving accessibility to data.

In the "traditional" knowledge management approach knowledge queries are usually carried out through a centralized system. This methodology is great as it allows much knowledge-reuse and accessibility to data from all parts of an organization; however it is fundamentally flawed in that the knowledge has to be edited in some way in order to be shared in a standard manner. It may disregard the context of the stored information, and therefore not only encourages the sharing of imperfect knowledge, but also discourages the users of a system from using it at all, due to the fact they have to adopt very tight and strong new practices in order to make the system work. This system may be then too rigid and requires from the actors of the system to re-assess their knowledge rather than to let the system doing the work to accommodate the knowledge in its original context. VKCs enable to share better knowledge since the size of the communities are much smaller than with usual knowledge bases. Some previous works of us (dissertation of Yang) have shown that to enforce trust under uncertain knowledge, knowledge data must be shared and reused. This is rather easy to perform when using VKCs. In addition, too large knowledge bases make reputation trust less reliable while keeping the abstract communities small enable to master better the concept of trust. Also, this fact minimizes the risk of implicit consequences that are always present when practical reasoning is used in decision-making systems.

Adopting the IT point of view leads to think of an enterprise as a distributed computational paradigm. Multiagent systems have been invented to tackle problems within distributed computing. They have been then naturally introduced to address knowledge management issues. A general overview on the works on these topics can be found in [Elst *et al.* (2003)].

In our approach, each actor is seconded by an autonomous agent having full control over its own knowledge. The actors do not have to try to fit their own knowledge into another 'view' of the world, rather the mechanics of the system is in place to allow fluid interaction between agents with similar domains of interest, without imposing some kind of fixed view of the world upon them. This is where things are slightly different to the knowledge nodes approach presented in [Bonifacio

et al. (2002)], where the concept of knowledge nodes makes the fixed view of the world more 'local' but still imposes this on to the actors within one knowledge node. The dynamism of this system allows new agents to join a community of communities, and as long as they have one domain concept in common with other agents in the organization, it is possible for them to find these agents and instigate some kind of exchange of ideas in the context of a community of interest. In essence, the abstraction level of an agent and a community have been chosen to be as general as possible and to leave the power in the hands of the individuals, rather than in the system itself.

The community approach enables agents to exchange data in a formal manner with other agents on a peer-to-peer basis. As long as agents are in the same community, they have the potential to exchange knowledge, and since agents can join any community they wish to, they have the possibility to exchange knowledge with any agents in the entire organization in a formal manner, as long as they share a domain of interest.

Our proposal of considering virtual communities can be related to [Houari *et al.* (2007)], where Houari considers that distributed knowledge management, intelligent software agents and XML based knowledge representation are "three research challenges which are changing the face of knowledge management solutions". This article explores structural requirements of distributed knowledge management and how to empower it with software agents and XML based knowledge representation. While we are using a similar point of view, our proposal consists of the description of the abstraction called *virtual knowledge communities* which is a means for knowledge exchanges among agents.

Security issues are addressed into numerous papers such as [Boella *et al.* (2006)] and [Portillo-Rodrguez *et al.* (2007)]. We are clearly not considering this key issue in this paper. In [Boella *et al.* (2006)] normative multiagent systems for secure knowledge management based on local access-control policies are studied. The authors argue that their approach respects the autonomy of the knowledge providers into the virtual community composed of multiple knowledge providers. In [Portillo-Rodrguez *et al.* (2007)], authors propose a three-level multi-agent architecture for considering reputation and trust within communities of practice where knowledge is exchanged. Trust in the frame of Virtual Communities is also considered in [Abdul-Rahman *et al.* (2000)]. However, these approaches to trust and security only scratch the surface of what is required for more serious investigations. The latter require first to assess whether agent platform (for instance JADE) enables to implement secure system. A theoretically sound approach to forbid the intrusion of agents as well as the capture of exchanged information is presented in [Endsuleit (2007)]. It is inspired by methodologies available for cryptographic protocols. Although the complexity analysis shows that this approach is feasible to be implemented, this would require still extensive efforts.

The words virtual, organization, business, enterprises, networks or knowledge

have been mixed in a great variety of projects and proposals. An overview of state-of-the-art is quickly meaningless, even when restricting it to professionally oriented projects. Two recent major publications in this area are the proceedings of the PRO VE 2007 conference [Camarinha-Matos *et al.* (2007)] and the white paper made available by the ESoCE project [Santoro *et al.* (2006)]. This project deals with the concept of professional virtual communities. It is fully different from our concept of virtual knowledge community although the latest is strongly connected to corporate knowledge. It is enough to read [Santoro *et al.* (2006)] to be convinced of the difference. Related to ESOCE, the document "A Roadmap towards the Collaborative Enterprise - CE Vision 2010" dated 2004 gives a pretty good outline of this approach. Our claim is that our VKCs are general enough to encompass most, if not all of, the projects covered at PRO VE 2007 as well as ESoCE. In particular, the roadmap for the latter mentions legal issues (see p.14-15) that have a more restricted span than those we may address within our approach.

Notice that our work is not directly related to Community-Based Learning (see for instance [Lee *et al.* (2003)]), where the learning issues are predominant. However, we can imagine that concepts from this domain (virtual class room, learner profile, adaptive and collaborative learning, etc.) could be adopted to extend our actual model for virtual community, where agents would represent the learners and would search for knowledge thought communities.

7. Conclusions and future works

We consider that the bottom-up agent-based approach to knowledge management using virtual communities is the appropriate abstraction to use for knowledge sharing in large, heterogeneous, knowledge environments such as corporations and the Internet. It can be used to model knowledge management scenarios, and the same approach can be used to create useful peer-to-peer (via 'community') software systems. Thus, it can cope with dynamic changes in the knowledge available from constantly changing different sources.

Since the system is, in its most general form, an open one, it requires security policies to ensure that only trustworthy agents can access the communities, to prevent malicious attacks from untrustworthy agents. Such policies have not been explicitly built in to the design of the system in this paper, but should be implemented at lower level (individuals, leaders of the communities) in the future. Also, the question of heterogeneity of knowledge sources has not been adequately addressed in this paper. The more the knowledge models will be loose (multiple interpretations, incomplete or uncertain knowledge, etc.), the more the algorithms will have to be powerful to cope with heterogeneity. Investigations are under way to combine different ontologies and thus better master this issue.

Finally, an open question is to identify those optimal parameters that may tune finely the behavior of agents which act in models for a community of virtual knowledge communities. To find them would help to build better models on which to

build future knowledge management systems. This is an almost experimental task that ought to be done using prototypical methods of economy (and also sociology): first suggest the relevant parameters and then perform a statistical evaluation of their relevance. An introductory paper [Thomas *et al.* (2007)] on making knowledge work in virtual teams can provide some paths for such a task.

References

A. Abdul-Rahman and S. Hailes (2000) *Supporting Trust in Virtual Communities* HICSS '00: Proceedings of the 33rd Hawaii International Conference on System Sciences. Vol. 6. 2007.

ASAP Research Group. Automated Scheduling, Optimisation and Planning. Jubilee Campus, Wollaton Road. `http://www.asap.cs.nott.ac.uk`. (2008)

G. Boella and L. van der Torre (2006) *Security policies for sharing knowledge in virtual communities* IEEE Transactions on Systems, Man and Cybernetics Vol.36, N.3, 2006, pp439–450

M. Bonifacio, P. Bouquet, R. Cuel (2002). *Knowledge Nodes: the Building Blocks of a Distributed Approach to Knowledge Management.* Journal of Universal Computer Science, 8(6), 652–661, Springer Pub. & Co.

M. Bonifacio, P. Bouquet, G. Mameli and M. Nori (2003) *Peer-Mediated Distributed Knowledge Management.* American Association for Artificial Intelligence Spring Symposium 2003, Technical Report SS-03-01, 1–8, ISBN 1-57735-078-9.

J. Calmet, S. Jekutsch, P. Kullmann and J. Schue (1997) *KOMET: A System for the Integration of Heterogeneous Sources.*. In Foundations of Intelligent Systems: 10th International Symposium, ISMIS '97, Z.W. Ras, A. Skowron (eds.); Springer, LNAI Vol. 1325, 1997.

J. Calmet, P. Maret and R. Endsuleit (2004) *Agent-oriented abstraction.* RACSAM (Revista Real Academia de Ciencias, Serie A de Matemticas) Special issue on Symbolic Computation in Logic and Artificial Intelligence. Vol. 98 (1), pages 77-83, Dec 2004.

L. Camarinha-Matos et al (Eds) (2007) *Establishing the Foundation of Collaborative Networks.* Proc. of PRO-VE 2007, In IFIP Springer series, vol. 243, 2007

J.R. Chen, S.R. Wolfe and S.D. Wragg (2000) *A Distributed Multi-Agent System for Collaborative Information Management and Sharing.* In Proc. 9th ACM International Conference on Information and Knowledge Management (2000), 382–388, ISBN: 1-58113-320-0.

R. Cuel, M. Bonifacio and P. Bouquet (2002) *Knowledge nodes: the building blocks of a distributed approach to knowledge management.* Journal of Universal Computer Science, 2002.

V. Dignum (2003) *Using Agent Societies to Support Knowledge Sharing.* In AAMAS-03 Workshop on Autonomy, Delegation and Control, Melbourne, Australia, July 14th 2003.

E.H. Durfee (1999) *Distributed Problem Solving and Planning*, Chapter 3, 121–164. Weiss, G. (1999). Multi-Agent Systems, MIT Press.

L. van Elst, V. Dignum and A. Abecker (eds.) (2003) *Agent-Mediated Knowledge Management.* International Symposium AMKM 2003, Stanford, CA, USA. LNCS/LNAI Vol. 2926, 2004

R. Endsuleit (2007) *Robust and Private Computations of Mobile Agent Alliances*, PhD Dissertation, University of Karlsruhe, June 2007.

J. Ferber (1997) *Les systmes multi-agents : un aperu gnral* Technique et Science Informatiques, 16,8, 979-1012.

G. Fisher and J. Ostwald (2001). *Knowledge management: problems, promises, realities and challenges.* IEEE Intelligent Systems, 16(1), 60–72.

F. Fleischer (2005) *Insights into Neural Network Models of Traumatic Memories and Dissociative Amnesia*. Master Thesis, University of Otago, New Zealand and Diplomarbeit (in German), University of Karlsruhe 2005

M. Gordon, W. Fan, S. Rafaeli, H. Wu and N. Farag (2003) *The architecture of comm-Knowledge: combining link structure and user actions to support an online community*, Int. J. Electronic Business, 1(1), 69–82.

N. Houari and B.H. Far (2007) *An Agent-based Approach to Support Performance Management for Dynamic and Collaborative Work* Proceedings of the 9th International Conference on Enterprise Information Systems (ICEIS 2007), pp. 178-184, 12-16, June 2007, Funchal, Madeira Portugal, 2007.

D. Huang (2007) *A Security Gateway for Web Services Protocols.*. Dissertation, University of Karlsruhe, 2007.

M. Huhn and L. Stephens. (1999) *Multiagent systems and societies of agents.* In *Multiagent Systems: A Modern Introduction to Distributed Artificial Intelligence*, chapter 2, pages 79–120. Cambridge, MA, USA, MIT Press, 1999.

JADE Home page. `http://sharon.cselt.it/projects/jade/`. (2008)

Jena's development team. Home page. `http://jena.sourceforge.net/index.html`. (2008)

Y. Lee and Q. Chong (2003) *Multi-agent systems support for Community-Based Learning* Interacting with Computers (Elsevier). Vol. 15, N.1, January 2003, pp. 33-55(23)

P. Maret and J. Calmet (2005) *Corporate Knowledge in Cyberworlds.* IEICE Journal. Information and Systems. Special Issue on Cyberworlds. Vol.E88-D, N.5. May 2005.

P. Maret, M. Hammond, and J. Calmet (2004). *Virtual Knowledge Communities for Corporate Knowledge Issues.* 5th International Workshop on Engineering Societies in the Agents World (ESAW, Toulouse, France), Springer LNCS 3451, pp.33–44.

L. Orbst, H. Liu, R. Wray (2003) *Ontologies for Corporate Web Applications*, AI Magazine, Fall 2003

J.W. Palmer and C. Speier (1997) *A Typology of Virtual Organizations : an empirical study.*. Proceedings of the Association for Information systems, J. Gupta (ed.), 1997 America conference, Indianapolis, 1997.

J. Portillo-Rodrguez et al. (2007) *Fostering Knowledge Exchange in Virtual Communities by Using Agents* Lecture Notes in Computer Science Volume 4715 pp32-39. 2007.

Protégé's development team. Home page. `http://protege.stanford.edu/` (2008)

K. Rajiv and E.R. McLean (2002) *The Next Generation Enterprise: A CIO Perspective on the Vision, its Impacts, and Implementation Challenges.*. Information Systems Frontier; Kluwer Academic Publishers; Apr 2002; pp.121-138.

R. Santoro and A. Bifulco (2006) *Concurrent innovation paradigm for integrated product/service development.* ESoCE White paper, May 2006. www.ESoCE.net

P. Sieber and J. Griese (eds.)(1997) *Organizational Virtualness.* Proceedings of the VoNet Workshop, Simowa Verlag Bern, 77-83, www.virtual-organization.net

SPARQL. `http://www.w3.org/TR/rdf-sparql-query/` (2008)

D.M. Thomas, R. P. Bostrom and M. Gouge (2007) *Making knowledge work in virtual teams.* Communication of the ACM, pp. 85-90, vol. 50 No. 11, Nov. 2007

M. Weber (1922) *Economy and society*, University of California Press, 1922/1986, ISBN : 1558607978.

Y. Yang (2007) *A Framework for Decision Support Systems Adapted to Uncertain Knowledge.* Dissertation, University of Karlsruhe, 2007.