Analyzing Requirements of Knowledge Management Systems with the Support of Agent Organizations

Renata S. S. Guizzardi[1], **Anna Perini**[2]

[1]Computer Science Department – University of Twente (UT)
P.O. Box 217 – 7500 AE Enschede – The Netherlands
souza@cs.utwente.nl

[2]ITC-irst
via Sommarive 18, I-38050, Trento-Povo, Italy
perini@itc.it

Abstract. Knowledge Management (KM) is considered by many organizations a key aspect in sustaining competitive advantage. Designing appropriate KM processes and enabling technology face considerable risks, as they must be shaped to respond to specific needs of the organizational environment. Thus, many systems are abandoned or fall into disuse because of inadequate understanding of the organizational context. This motivates current research, which tends to propose agent organizations as a useful paradigm for KM systems engineering. Following these approaches, organizations are analyzed as collective systems, composed of several agents, each of them autonomously producing and managing their own local data according to their own logic, needs, and interpretative schema, i.e. their goals and beliefs. These agents interact and coordinate for goal achievement defining a coherent local knowledge system. This paper presents a novel methodology for analyzing the requirements of a KM system based on an iterative workflow where a pivotal role is played by agent-oriented modeling. Within this approach, the needs for KM systems are traced back to the organization stakeholders' goals. A case study is used to illustrate the methodology. The relationship of this work with current studies in agent organizations and organizational knowledge management is also discussed. Differently from other works, this methodology aims at offering a practical guideline to the analyst, pointing out the appropriate abstractions to be used in the different phases of the analysis.

Keywords: agent organizations modeling, knowledge management, agent-oriented methodologies

1. Introduction

Agents have frequently been proposed as appropriate entities to enable the analysis and design of complex systems, made up of several components that often behave in a distributed fashion and interact with each other in order to achieve a common objective (i.e. the system's overall functionality) [15,26,27]. The social and cognitive characteristics of agents are their main strength, turning them into promising constructs to emulate human interaction and rational behavior. Based on the analysis of the current social structures embedded in the organization may lead to more appropriate system proposals to enable such structure to evolve in terms of efficiency and performance.

This paradigm seems even more natural when carried out to the Knowledge Management (KM) domain, defining social behavior and processes underlying the organizational settings [6,12,14,18]. Knowledge Management (KM) can be broadly defined as the tools, techniques and processes that contribute to the most effective and

efficient management of organizational intellectual assets [16]. Advances in this area are mainly motivated by the assumption that, in order to remain competitive in the information society's market, organizations should focus on *knowledge assets*, generally maintained by the members of an organization.

Currently, organizational tasks and processes are often distributed in different branches, and follow dynamic kinds of control structure, such as those of market or collaborative network societies [6]. Such changes require present organizational structures and processes to be well understood and often redesigned. So, designing KM solutions presents both challenges of process re-engineering and of information system design, as they must be shaped to respond to the specific needs of the organizational environment. In fact, many systems are abandoned or fall into disuse because of inadequate understanding of the organizational context [7,19]. Hence, analysis and design activities claim for adequate modeling constructs, such as those proposed in the agent organization paradigm.

Current methodologies for designing agent organizations usually focus on the roles structures within the organization, modeling agents' responsibilities and norms. Moreover, they model the organization's dynamic aspects in terms of interaction patterns [6,9]. Although the definition of roles and their interaction leads to a clear view of the structure and general behavior of the organization, they do not capture the reasons behind organizational requirements. In fact, there is a gap between understanding organizational needs and defining the roles that the agents in an agent organization should play. With the main target of filling in this gap, this paper presents a novel methodology for analyzing the requirements of a KM system, adopting an agent-oriented approach.

In this work, we claim that more focus should be given to the *initial phases of system development*, aiming at grasping the *requirements of the system to be*, both in terms of the individual perspective of the organizational members and the overall objectives of the organization. This is especially important in the KM context, which focuses on the effective use of human intellectual capital, since much of human knowledge is tacit and intangible [16]. Moreover, issues such as community and community's practices [7,11,25] go much beyond those typically considered in the conception of traditional systems, and opens up many more ways to leverage information technologies to augment human and organizational capabilities and performances.

The methodology proposes the analysis of the *goals* of the system's stakeholders and their inter-dependencies as the initial steps towards understanding the requirements for a KM system. The main strengths of this approach can be summarized as anticipating the concerns of all actors involved in a given scenario, focusing on the stakeholders' aims while abstracting from unimportant issues, until the domain is well understood and the analyst is ready to propose a solution (either by changing current processes or by applying technology). In addition to that, the adopted notation is visually rich and accessible, besides being supported by existing modeling tools [17,23].

The paper is organized as follows: section 2 discusses how agent organizations, starting from the analysis of agent's cognitive mental structures, could support modeling of organizational KM settings; section 3 presents this work's proposed approach for systems requirements analysis and the applied notation; section 4 presents a fictitious KM scenario used here to illustrate our novel methodology; section 5 focuses on a case study for our methodology, using the scenario of section 4;

section 6 discusses related work; and, finally, section 7 presents conclusion and future directions of this work.

2. Agent Organizations as Metaphors in KM Modeling

Rational agents in Artificial Intelligence (AI) have been defined as cognitive beings having characteristics such as goals, beliefs, desires, commitments and claims, being influenced by studies from different research communities, including economics, philosophy, and cognitive science. Agents are embedded in an environment from which they perceive certain events (perceptors), and on which they act causing changes (effectors) [26]. The behavior of perceiving the environment and acting as a result of such perception defines agent *reactiveness*. But besides reacting, agents are able to adopt goal-driven behavior, deciding to act on its own (*proactively*), motivated by their given beliefs about the world and their desires with respect to how they would like the world to be. Moreover, agents may "live" in a community of agents, interacting with them in several ways, meanwhile pursuing its goals and/or reacting to events (which here include communicative events triggered by incoming messages from other agents).

Recently, research in this area has moved its focus from the individual characteristics of an agent, to the elements that occur as a result of agents' interactions. This has given life to a new research area known as *Agent Organizations*. Work in this area has focused, for example, on: a) the complexities of self-organizing communities [20]; b) on how the different roles played by internal and external agents may affect human organizations, and how this understanding might help organizations adapt to changes [5]; and c) on modeling organizational business processes [6,9,14].

Concurrent to the evolution in organizational models, more suitable agent-based abstractions have been developed, allowing the understanding of the organizational social, economic and technological dimensions. Advances in agent societies are frequently focused on coordination frameworks that enable agents' interaction, in such a way that they will autonomously but cooperatively achieve their goals. Some authors classify agent organizations as having more structure than agent societies, having in common the fact that the agents in the system work towards a common overall purpose. In this sense, the main differences between organizations and societies may be given by the emphasis on the decision processes that underlie organizations, making more explicit the division of labor among agents (usually through *roles*) [6,9]. However, organizations and societies could also be considered as synonyms, as work on both fields should be targeted at empowering agents with social structures, providing them with more complex abstractions to model and support organizations.

The features highlighted above make agents adequate constructs in representing humans in domain models and organizational abstractions. We can profit from the organizational view, defined by the notions of *purpose*, *structure*, *rules* and *norms* [6] when modeling systems to be adjusted to organizational processes and practices. The idea of applying agents as human abstractions is that agents may aid the analyst to abstract from some of the problems related to human complexity, and focus on the important issues that interfere with specific goals, beliefs and commitments of the domain agents in each modeling phase. This allows the analyst to clearly understand the current situation, and this is an essential factor for the proposal of the appropriate solution. Moreover, such kinds of model make communication with the stakeholders much more effective, since the analyst uses concepts that are more familiar to the

common user (e.g. goal, task and belief) than technology-oriented terminology (like tables, SQL query, middleware and threads).

Applying agents as a metaphor on system development is not new and has been observed in [15,27]. However, especially in KM domains, agent organizations seem to be an interesting approach as agents may represent not only artificial beings, but also the *human users* and the *organizations* involved in a given scenario [6,12,13,14,18]. This allows, for example, the requirement engineer to understand, before modeling a KM system itself, how knowledge flows within the organization. As a result, besides inserting new technology, the business processes applied in the organization may be changed in order to enhance these knowledge flows. Moreover, if a technological solution is needed, agents enable legacy systems to be considered in the analysis, allowing the new solution to be based on approaches of integration of old and new components. This may lead to more satisfaction to end users, who are already familiar with the interface and methods applied in the systems in use. These aspects are compliant with the Distributed Knowledge Management approach [1] which prescribes that more attention should be given the *knowledge holders* and the natural processes they already use to *share knowledge* within organizations, which leads to a *bottom-up strategy* when proposing a KM solution.

3. The Proposed Approach for KM Systems Requirements Analysis

The agent paradigm offers appropriate constructs for modeling human organizations, as argued in section 2. However, having the right abstraction is not enough for guaranteeing the development of adequate solutions for the organization. For that, a consistent system engineering methodology is needed.

In our view, important requirements for an adequate methodology are the following:

- Offering the right set of concepts and constructs for the targeted system engineering phase;
- Providing a visual language besides textual descriptions, thus facilitating the communication between stakeholders and analysts, and among analysts and system designers;
- Being relatively accessible and not requiring too much overhead in the sense of extra work from the part of the analyst in understanding and using the given language and method.

With respect to the first issue above, in our particular case, we should focus on the right choice of agent cognitive characteristics to be applied in the different phases of the development cycle. Concepts such as agent's beliefs, goals, and plans are vastly discussed in literature and different models have been proposed [27]. However, it is hard to know how to go from theory to practice. In our approach, based on the *Tropos* agent-oriented methodology [2], we start by addressing agents' goals, as the appropriate type of concept in requirements analysis. Relevant KM literature justifies this choice. For instance, Nonaka & Takeuchi [16] mention *intention* as the first driving force for the adoption of KM practices within organizations. Nevertheless, these authors mainly focus on the organization's top management intention, defined as "an organization's aspiration to its goals". Here, in contrast, we consider the goals of all stakeholders involved, trying to understand the relations and possible discrepancies between their goals. Finally, our approach also complies with the last two requirements presented above, providing CASE tool supported [17] diagrams that

explicitate peculiarities of the stakeholders, serving as interesting communication artifacts between analysts and stakeholders.

3.1. Workflow and Methods Underlying the Analysis Process

The proposed methodology rests on the analysis process depicted in Fig. 1 (a). It starts with collecting information about the domain in different ways, for instance, by interviews with the target personnel of the organization, and by active observation of the employee's activities, e.g. through ethnographical analysis, Fig. 1 (b). The domain analysis itself is then carried out, applying an agent-oriented visual modeling approach. This analysis allows us to point out the rationale behind stakeholders' needs of process reorganization and of KM enabling technology. Finally, we are able to elicit the requirements for a KM solution and to trace them back to the fulfillment of the social and individual goals previously analyzed.

A review activity aimed at verifying the outcomes of this analysis and of choosing among possible alternative solutions is usually needed before starting the architectural design step. This last step is guided by the analysis of the elicited requirements and aims at synthesizing a candidate architecture. Notice that this chain of activities may be performed several times, in an iterative analysis process.

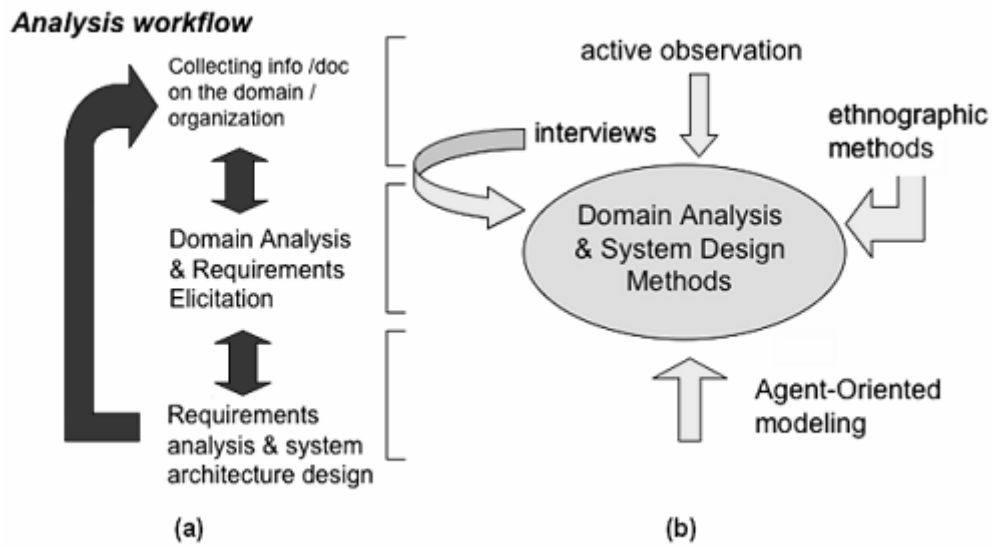

Figure 1. The analysis process: a) main steps in an iterative workflow; b) analysis methods that can be used in the workflow activities

3.2. Agent-Oriented Visual modeling in *Tropos*

The *Tropos* methodology is an agent-oriented software development methodology for engineering distributed systems [2]. The methodology adopts a model-driven approach, i.e. it guides the software engineer in building a conceptual model, which is incrementally refined and extended, from an early requirements model, namely a representation of the organizational setting where the system-to-be will be introduced, to system design artifacts. Indeed, a distinctive feature of the methodology with respect to current AO methodologies is that of filling the gap between requirements analysis and system architecture design, by adopting an uniform notation and an uniform analysis technique to model business goals, system requirements and system architecture.

Tropos uses a conceptual modeling language derived from the i* framework [28], which provides a graphical notation and a set of techniques for goal analysis. This notation has been extended in order to allow for informal and formal specifications. Basic constructs of the conceptual modeling language are those of actor, goal, plan, softgoal, and resource:

- an actor can represent a stakeholder in a given domain, a role or a set of roles played by an agent in a given organizational setting;
- a goal represents the strategic interests of actors. Two basic types of goals are considered, namely hard and soft goals, the latter having no clear-cut definition and criteria as to whether they are satisfied. This difference is captured in [4], which suggests to say that (hard) goals can be *satisfied*, while softgoals can be *satisficed.* Softgoals are useful to represent how a state of affairs should be reached, that is they can represent goal/plan qualities and non functional requirements.
- a plan (or task) specifies a particular way of doing something, i.e. a particular course of action that can represent a means for satisfying a goal or for satisficing a softgoal;
- a resource is a physical or informational entity used in a given task or to achieve a certain goal.

A dependency link between pairs of actors allows us to model the fact that one actor depends on another in order to achieve a goal, execute a plan, or acquire a resource. The former actor is called the *depender*, while the latter is called the *dependee*. The object (goal, plan resource) around which the dependency centers is called the *dependum*. If the dependee fails to deliver the dependum, the depender would be adversely affected in its ability to achieve its goals. In this sense, the depender becomes vulnerable due to its dependency links. This type of information can be graphically depicted in an *actor diagram*, a graph whose nodes represent actors (circles) and whose arcs represent dependencies (a couple of arrows linked by its dependum).

The concept of actor dependency, as defined in [28], has been inspired by Catelfranchi et al [3] work. In [3], the concept of agent dependency is analyzed with the aim of providing a first computational theory of dependence to be applied for communication control. This theory has been extended and refined in following works, for instance in [21] an abstract structure called dependence graph has been proposed to model OR- and AND- dependencies between a group of agents. Dependence graphs are use to identify the level of complexity, internal cohesiveness/ fragility of a multi-agent system. Even if it rests on the same basic intuitive meaning, actor dependency as proposed in [28] is a simpler concept (a ternary relationship) which is intended to model explicit actor dependencies that can be stated by roles and procedures adopted by the organization we are analyzing, or may result from an analysis of how a stakeholder may achieve its individual goals. Moreover, the purpose of using this concept is different, i.e. the aim is that of modeling the rationality behind the requirements of the organization of a new system or of a change in the currently used one.

The process of model building in *Tropos* has been specified in [2] in terms of a non deterministic concurrent algorithm, here we give a qualitative description. Model building begins with the definition of a number of actors, each with a list of associated main goals (or softgoals). Notice that at the beginning, the minimum set of actor goals

which relates to the analysis purpose is explicitly modeled. Throughout the refinement of the model, further goals may be needed to be included.

Each root goal is analyzed from the perspective of its respective actor and depicted in a sort of balloon, called the *goal diagram*. For instance, goal means-end analysis proceeds by refining a goal into sub-goals, plans, and resources that provide means for achieving the goal (the end). Contribution analysis allows the analyst to point out goals and softgoals that can contribute positively or negatively in reaching the goal being analyzed. Decomposition allows for a combination of AND and OR decompositions of a root goal into sub-goals, thereby refining a goal structure. The generated sub-goals are delegated to other actors, or remain a responsibility of the actor itself. Sometimes new actors (roles) need to be introduced, to whom some goals and/or tasks are delegated. For instance, in order to represent the role of technology at support of the organization's processes, new actors are introduced, refining a model of the organization's needs into a model of the requirements for an information system able to meet these needs. Softgoal analysis is typically used to drive the choice of one among different alternatives that may emerge during OR-goal decomposition [4]. Modeling is complete when all goals have been dealt with to the satisfaction of the actors who pursue them.

Among the advantages of adopting *Tropos* visual modeling for KMS requirements analysis is the possibility of pointing out the idiosyncrasies of a given environment, for instance: a) verifying inconsistencies between models elaborated on the basis of interviews with different actors in the organization; b) realizing that several actors perform the same exact task, thus suggesting that the process can be more efficient if that task is attributed to only one or two actors; c) understanding that too much or too little time and effort are dedicated to KM activities; and d) realizing the problems behind the non-adoption of the proposed KM methods and systems [19], for example, detachment of the system from the daily practices of organizational members, lack of trust and motivation to share knowledge, etc.

4. Knowledge Management in CoPs: a Fictitious Scenario

In order to demonstrate our proposed methodology, we use here a fictitious scenario. Although not a real case study, this scenario was carefully tailored to be realistic, taking into consideration the available literature [7,11,25]. Here follows the scenario description.

"Luca starts working in BHI Software Company. He is a programmer with 10 years of experience. As a newcomer at BHI, he needs to adjust to the organization's work practices, adapting to the work style of his working team and learning about the company's policies and management directives. Aiming at providing its workers with a rich environment for knowledge sharing, BHI Management fosters the development of Communities of Practice (CoPs) across the organization. These communities are self-organizing groups whose members share interests and goals, or perform similar tasks within the organization. They are not necessarily from the same working team or division, and their members are dispersed across the 10 branches of BHI. Through a special division named KM Division (KMD), BHI Management legitimates and supports the CoPs' activities, granting incentives for those that stand out (thus contributing to the organization as a whole), besides providing technological infrastructures and tools. The CoP needs special procedures to motivate Luca, as well as the other members, on sharing knowledge. Especially when business processes are tight, making one's knowledge available will get very little priority. Besides, it may

not be very clear for Luca what he will get in return for his willingness to contribute to the CoP members. External incentives provided by the CoP, as well as an adequate information system, may play an important role here."

5. Analysis of the Scenario

The main steps of the analysis carried out on the given scenario are discussed in this section, being illustrated with excerpts of visual models specified in the *Tropos* notation.

Section 5.1 starts presenting a very general model, giving us an overview of the actors of the scenario and of their goal dependencies. Hence, sections 5.2 and 5.3 presents a refinement of this first model, giving us details regarding respectively the newcomer perspective and the general internal structure of a CoP belonging to the organization of our scenario. In these diagrams, one can already understand which main goals of the newcomer are important to answer to his/her specific needs while adjusting to the organizational setting. Following this model, section 5.4 depicts the diagram containing our proposed solution for this case, which requires the development of a software system named KARe (Knowledgeable Agent for Recommendations) [22]. This is actually the last model belonging to the domain analysis and requirements elicitation phase. Next, section 5.5 presents a review of the requirements analysis phase, and introduces the phase of architectural design. Finally, section 5.6 is dedicated to KARe's global architecture, which is described and sketched on the basis of the analysis of the elicited requirements. The model of section 5.6 concludes our analysis, since KARe's architecture refinement and detailed design are no longer in the scope of this paper. These models have been prepared mostly in this presented sequence, but of course a few refinement cycles have been necessary, leading to changes in the different diagrams until the final version presented here.

5.1. The domain stakeholders and their strategic dependencies

According to the *Tropos* methodology, domain analysis starts by identifying the main stakeholders, represented as actors, with their goals. Figure 2 shows an excerpt of the initial model where the BHI company top management is modeled as the actor Organization, depicted as a circle. The organization has an initial softgoal relative to having the organization's team working well[1], which expresses how BHI intends to achieve more general objectives such as pursuing high quality of the products and of the production processes (pursuing high quality products / processes), as well as innovation (innovating) by considering human resources as a main asset.

The BHI's Knowledge Management Division and the communities within the organization play critical roles with respect to BHI strategic goals, according to the scenario, so they are also modeled as specific actors, namely the KM Division actor and the CoP actor. Luca plays the role of a newcomer in the organization (Newcomer actor), with his main goal of adjusting to the working practices of the organization (adjusting to the organization practices goal).

[1] The reason for modeling team working well as a softgoal refers to the fact that the organization is not monitoring and measuring explicitly the team work quality. In the process of analyzing goals from the point of view of the organization, positive and negative contributions of the team working well softgoal to the other goals of the organization will be modeled explicitly.

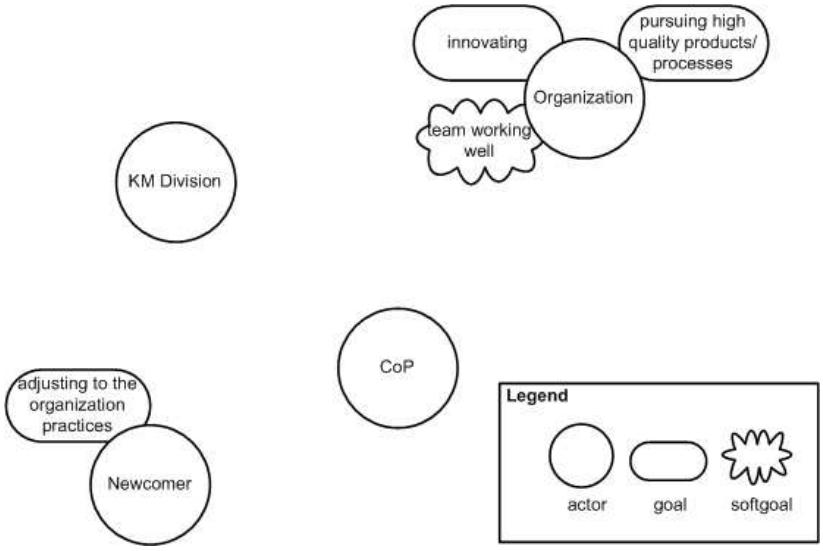

Figure 2. Initial domain model of the BHI scenario, according to the *Tropos* methodology.

Further modeling steps consist in analyzing each actor's goal from the point of view of the actor itself, aiming at identifying the strategic dependencies between actors, i.e. the dependencies which allow for goals achievement. Moving on with the analysis, Figure 3 shows basic goal dependencies between the scenario's actors.

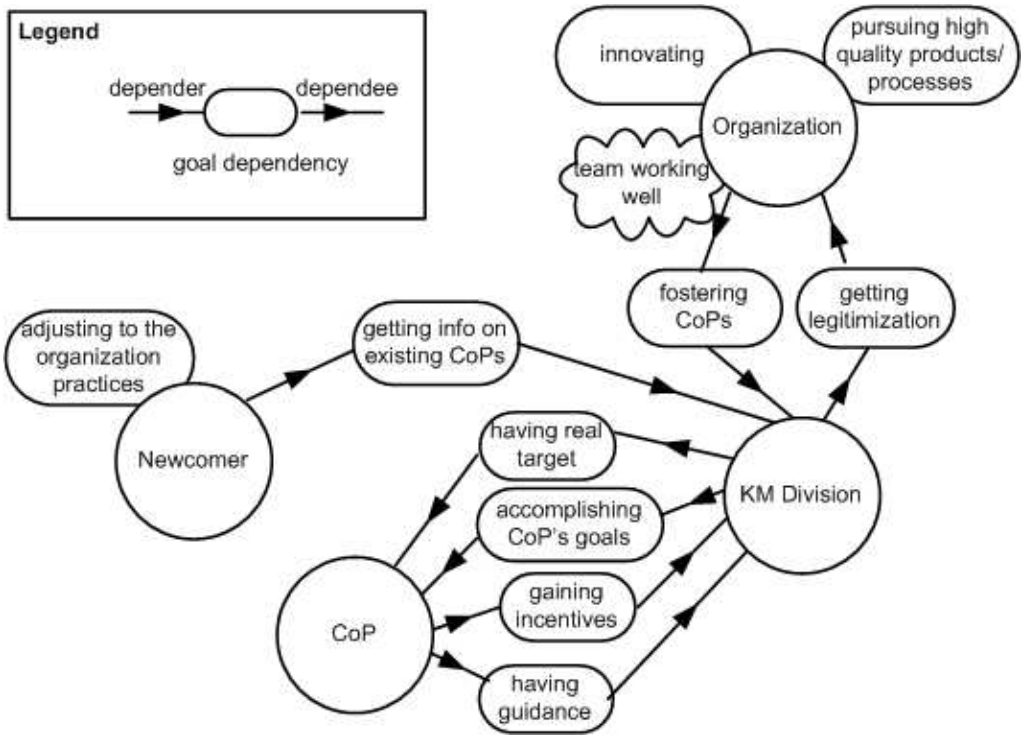

Figure 3. A first domain model showing the strategic dependencies between the actors involved in the scenario

The analysis of the Organization softgoal team working well points out a strategic goal of the organization, i.e. CoPs fostering, which is then delegated to the Knowledge Management Division (KM Division actor). On the other hand, the KM Division depends on the Organization to be legitimized for playing the specific role of motivating and supporting KM practices (legitimization getting goal). Note that a

dependency represents, at the same time, *delegation* and *commitment*: the depender delegates the goal to the dependee, while the dependee commits itself to the achievement of the dependum. This mutual dependency characterizes what intentional analysis names *"sustainable relationship"*, i.e. a relationship in which two actors depend on each other to achieve one or more of their own goals. Sustainable relationships indicate that there is some kind of balance between the two actors, thus helping them achieve personal goals. On the other hand, if there are dependencies only from one side, this indicates vulnerability by this dependee actor towards the depender, which should be corrected in order to guarantee that both actors are committed to each other. Analyzing the different strengths between each dependency can also indicate if a specific situation needs to be balanced [28]. The other dependencies depicted in Fig. 3 like the ones between the CoP and the KM Division may be easily derived from the scenario described in section 4.

5.2. Modeling an Individual Actor's Perspective: the Newcomer's Point of View.

The particular perspective of an actor can be analyzed using the three basic goal analysis techniques provided by *Tropos*: means-end analysis, contribution analysis and AND/OR decomposition. This allows the refinement of the domain model by identifying new actor dependencies. Figure 4 shows a goal diagram that models the Newcomer's perspective. Here, the internal goals of the Newcomer are analyzed and the dependencies towards the CoP, motivated by these goals, are identified.

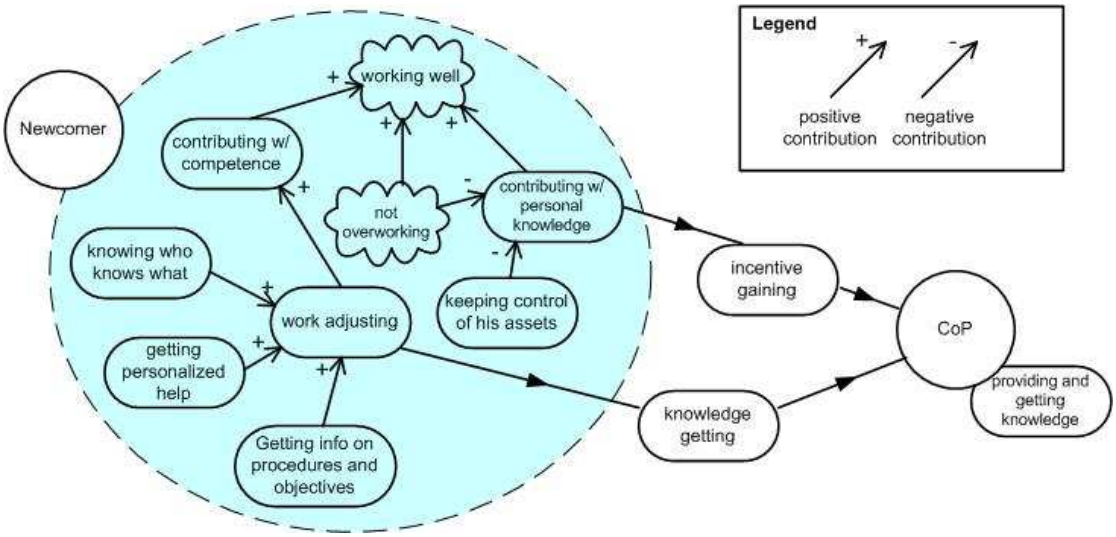

Figure 4. A Goal Diagram showing the Newcomer's perspective

The Newcomer's most general goal is the working well softgoal, i.e. he aims at doing his work efficiently, while also feeling good about himself and about the organization as a whole. In order to accomplish this, he aims at contributing with his competence and contributing with personal knowledge, gained in previous personal and professional experiences. Going deeper in the analysis of this last goal, we see that two other goals contribute negatively towards it (not overworking and keeping control of his assets goals). These are common problems already noted by the KM community [7,19]. Issues of trust (keeping control of his assets goal) and

motivation (not overworking goal) often lead to dissatisfaction towards the traditional centralized KM systems.

Let us now analyze the contributing with competence goal a bit further. In order to fully and most effectively contribute with his acquired competence, the Newcomer needs to adjust to his work environment (work adjusting goal). In order to do so, the Newcomer needs new knowledge about his work and about the organization as a whole, modeled by the goals getting info on procedures and objectives and knowledge getting, depending on the CoP for this. Three other goals contribute to the Newcomer's adjusting to work, namely: knowing who knows what, getting personalized help, and getting info on procedures and objectives.

5.2. Adding New Actors: Detailing the CoP Structure

New actors can be added during the analysis in order to model specific roles associated to a specific actor, or to specify new actors needed for goal delegation. In Figure 5, inside the dotted rectangle, we analyze the structure of the CoP actor, which can provide a solution to the goal dependencies involving the newcomer, with respect to knowledge providing/getting.

Here, we show two members of the CoP (Member i and Member i+1) and the two different roles, concerning KM, that they can assume: knowledge seeker and knowledge provider.

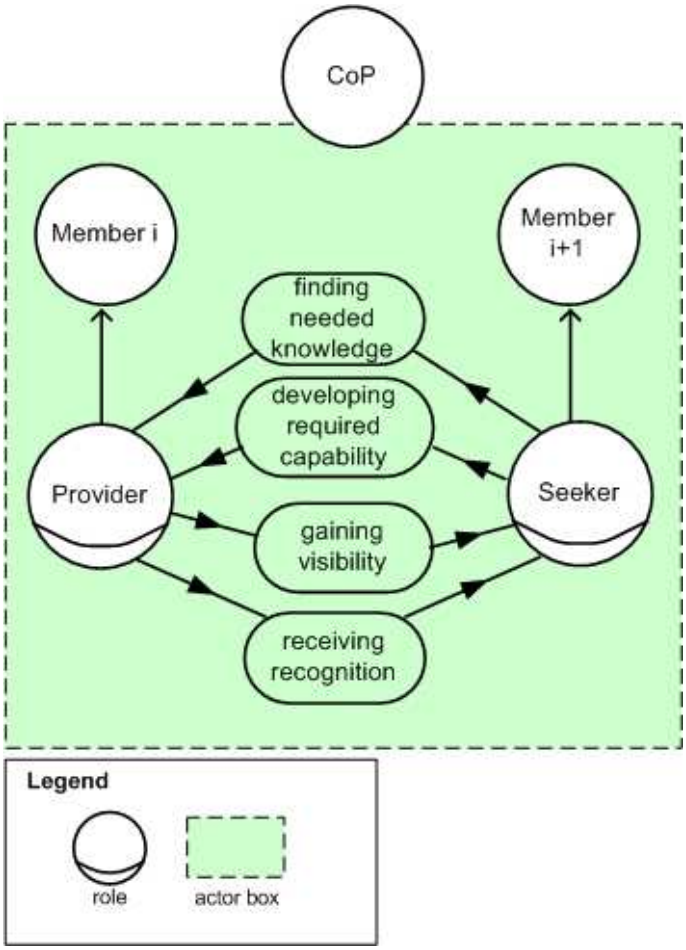

Figure 5. Actor Diagram showing the dependency between the Organization and the CoP actors

The two roles played by CoP members are illustrated in Fig. 5 by the Seeker and Provider roles. In this example, Member i assumes the role of Provider, while Member i+1 plays the role of Seeker. The diagram shows the mutual dependency between these actors. The Seeker depends on the Provider to find the knowledge he needs for a specific task (finding needed knowledge goal dependency), and to develop new capabilities (developing required capability goal dependency). On the other hand, the Provider depends on the Seeker to gain visibility (gaining visibility goal dependency) and to help the Seeker to develop new capabilities, so receiving recognition (receiving recognition goal dependency).

Note that CoP members may play both roles in different situations, generating a community of peers. Another interesting point is that as soon as a newcomer participates in a CoP, he/she will play the CoP member roles.

Considering the objectives a CoP is designed for, as shown in the actor diagram depicted in Fig. 5, this will assure the satisfaction of the previously identified goal dependencies between the Newcomer and the CoP. But could the Newcomer rest on the CoP to get personalized help when having a problem to solve (getting personalized help goal), and to find out who has a specific piece of knowledge (knowing who knows what goal), goals that are crucial for a newcomer (see Fig. 4)? Should CoP goals and structure be modified?

5.4. Identifying the Needs for a System Actor: the KARe System

In order to fulfill some of the goals of the Newcomer, the CoP structure has to be revised and needs to accommodate new actors (roles), having the ability to satisfy them. Figure 6 depicts an excerpt of the model showing that the CoP delegates some of the goals of the Newcomer to an actor named KARe (Knowledgeable Agent for Recommendations), namely: the goals of a) letting users keep control of their knowledge assets. The KARe system should allow each user to keep their assets in their own PCs, while making them available to other community members; b) allowing members to ask and answer questions through messages exchange. This feature is important because some of the Newcomer's questions may not be answered by reading artifacts. Sometimes, it could be necessary to communicate with CoP members for building the solution to a specific problem. KARe should mediate this interaction, by finding the best colleague to answer to a specific knowledge request; c) informing who knows what; and d) providing members with personalized help, by considering their personal characteristics when providing knowledge. Therefore, these four goals become KARe's main requirements

By analyzing the four goals from the point of view of the system actors, we can identify more detailed requirements, and analyze alternative solutions. For instance, we may consider satisfying KARe's goals by defining new artifacts to be produced along the organization's processes or we may look for what KM enabling technology can be considered to design a better solution.

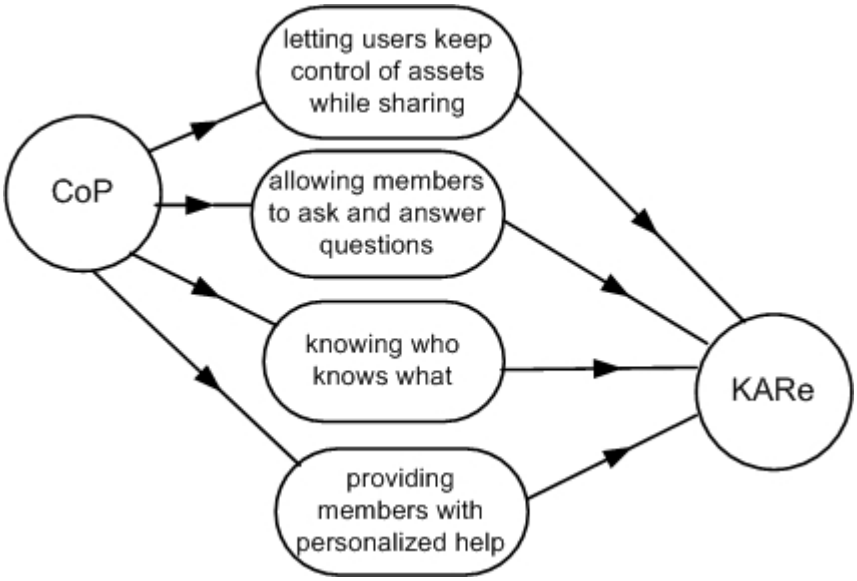

Figure 6. Actor Diagram showing the delegation of goals from the CoP to the KARe System

5.5 Reviewing the Domain Analysis & Requirements Elicitation

The review activity, at the end of this step, is aimed at verifying the achievement of the main objectives of domain analysis and requirements elicitation, or in other words the understanding of critical dependencies between domain actors for individual goal achievement and the identification of needs for new or alternative roles and dependencies to manage unsatisfied goals. Decisions on what to focus on in the following steps and on which alternative solutions to be refined in an architectural design, are taken at this point.

For instance, at that point of the analysis of the case study we choose to go ahead analyzing a KM solution resting on the adoption of a KM System based on peer-to-peer technology. This choice complies with the Newcomer's wish to keep control of his knowledge assets, besides resting on considerations about the effectiveness of this technology in favoring knowledge sharing, promoting individuals proactiveness. Moreover, the analysis of our scenario[2] pointed out the relevance of managing *tacit knowledge*, i.e. the knowledge that is confined in people's mind, and to transform it from *tacit to explicit and back to tacit*, completing the knowledge creating cycle as proposed in [16].

Taking a flexible approach on supporting this kind of knowledge conversion, KARe relies on the real potential of human communication to support knowledge creation and sharing. This emphasis is motivated by the assumption that such a process and, especially the content of the messages exchanged by community members, may eventually result in the disambiguation of tacit knowledge.

The next phase of our analysis is the system's architectural design. In this phase, new agents will emerge as sub-agents of the KARe system. These new agents are the choice of the system designer to fulfill the requirements captured in the requirements

[2] The models showed in the previous sections are too limited to justify our current choices, for instance, we did not show in details all the domain actors' points of view, nor did we show the refinement of the relationship between KM Division and CoP, sketched in Fig 4.

analysis phase. The designer usually bases her/his choice of architecture on previous experience or on available architectural patterns, previously used for similar purposes.

Besides supporting us on requirements analysis, *Tropos* is also applied for the architectural design. The main advantages this approach gives us are: a) allowing us to analyze which of the system's general goals are adopted by each of the internal agents; and b) supporting us on capturing the goal dependencies between the system's internal agents. For detailing the design of the system, it is necessary to rely on a different language, which can support information modeling, besides capturing agents' behavior and interaction, such as AORML [24]. The use of *Tropos* for architectural design provides a smooth transition from the problem level analysis to the system level analysis. Using it instead of a different notation makes it easier to trace back the functionalities of the system to the goals of the domain actors.

5.6. Analyzing the system requirements: KARe Preliminary Architectural Model.

The analysis of the requirements of the KARe system leads to the identification of a possible structure of the system actor in terms of roles (sub-actors), i.e. the system global architecture is identified through delegation of main system goals to internal sub-actors. For instance, the roles of Peer Assistant and User Model Engine may be designed in order to take care of goals respectively related to representing and searching knowledge on behalf of the CoP members, and providing personalization and configurability, while a Broker role may be proposed to achieve goals related to matchmaking peers with similar interests as adequate knowledge sources for specific requests. The emerging structure is that of an agent organization (or more generally of a peer-to-peer system), whose high level architecture may be modeled in terms of actor dependencies, according to *Tropos*, as in the example depicted in Figure 7. Note that, in this model, we start using technology-oriented terminology, such as *platform*, *directory service*, and *peer-to-peer infrastructure*.

KARe is depicted on the top of Fig. 7, delegating the four goals previously adopted by the system on behalf of the CoP (refer to fig. 6) to three main actors: the Peer Assistant (PA), the Broker and the User Model Engine (UM Engine). The PA is responsible for three of the main goals of the system: providing the basic platform, making recommendations, and providing question & answer services. These goals materialize, by the use of technological solutions, three important goals delegated by the KARe system and are further refined, providing us with more details regarding the proposed architecture. Granting the users with a basic platform means that the PA should empower the users with suitable means for storing and managing their knowledge assets (offering directory service goal), besides providing the basic infrastructure to allow the CoP Members (here represented by the Peer actor) to search for and deliver knowledge (providing peer-to-peer infrastructure goal). The choice for a recommendation scheme (making recommendations goal) is the strategy adopted by the CoP to motivate its members on collaborating and sharing knowledge. Such recommendations may come both proactively or in response to a specific knowledge request. The general goal of making recommendations may be refined according to two kinds of recommendations: referrals to experts in particular subjects or topics (recommending experts goal) or suggestions of knowledge assets according to different contents of interest (recommending artifacts goal). Artifacts are classified as *documents* or *messages*. The former refers to resources used in daily organizational routines (e.g. spreadsheets, reports and summaries) while the latter are result of communication

between the users, through questions and answers about particular content or processes. By focusing on messages, besides the usual documents referred to in KM applications, we aim at supporting the emergence and sharing of *tacit knowledge*, usually triggered by social interaction among members of a community. Finally, the PA also supports the users on asking and answering questions, through a question interface such as a web form (providing question & answer interface goal) and searching for knowledge on user's request (searching for knowledge goal).

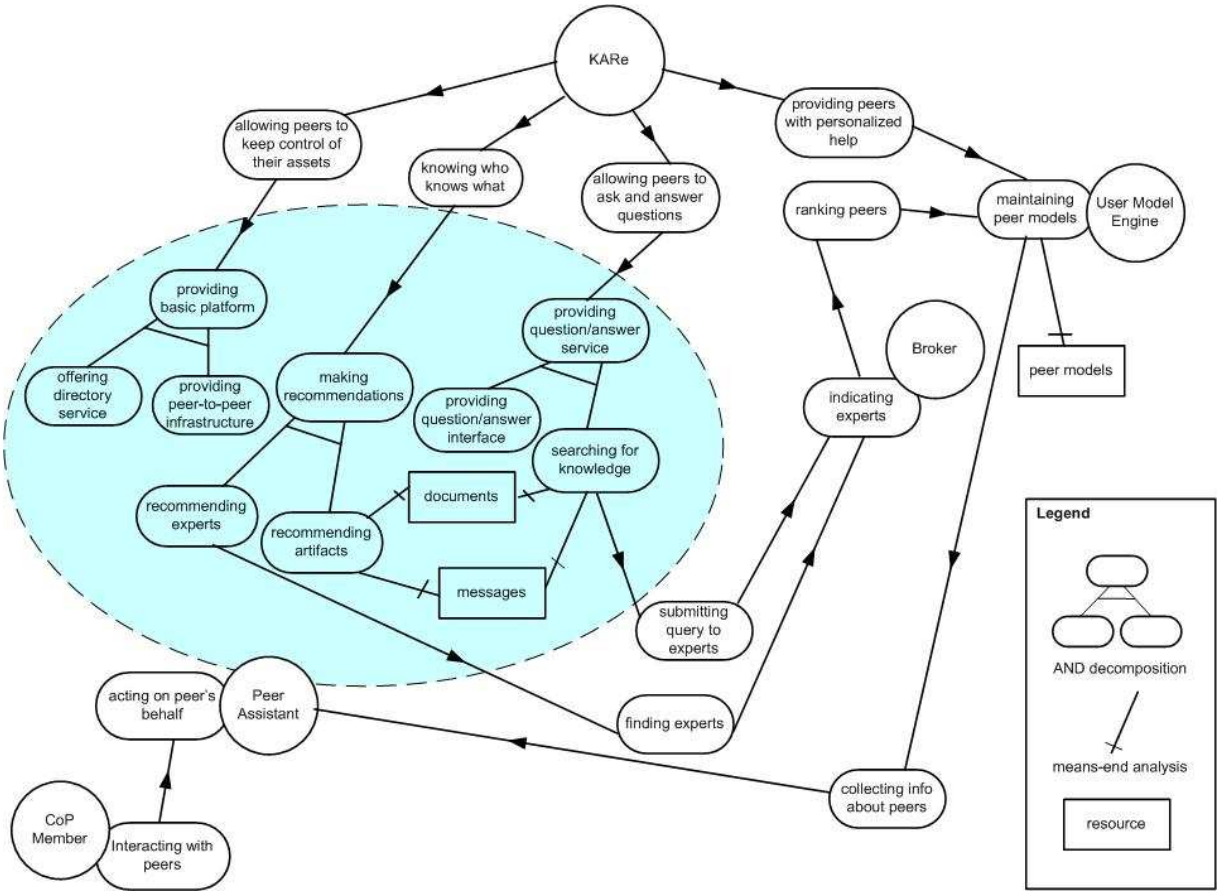

Figure 7. Actor diagram showing the high level architecture of the KARe system.

While the PA provides the most basic services of KARe, the Broker and the UM Engine focus on the activities of connectivity between peers and personalization. More specifically, the UM Engine maintains the user models describing each user's expertise and interests, besides other personal information. Having access to these user models, the Broker can propose 'experts' to solve particular knowledge requests. These two agents cooperate with the PA on fulfilling the CoP's goal of 'knowing who knows what', delegated to KARe (Fig. 6). Analyzing the goals of Fig. 7, we can understand how this cooperation occurs. The PA relies on the Broker to have information about experts that may answer a particular request, in the case of a question being sent by a Peer or when searching for knowledgeable peers to recommend (finding experts dependency from PA's recommending experts goal, and submitting query to experts from PA's searching for knowledge goal).The Broker has the goal of indicating, among all peers in the network, who would be the most knowledgeable to answer about specific topics (indicating experts goal). This goal also serves the UM Engine actor when elaborating the peer model that represents a particular user (maintaining peer models goal). This model provides

information about the peers' interaction and how they rated one another, regarding the knowledge provided, resulting in a sort of rank where the most knowledgeable peers are rated higher in relation to others (ranking peers dependency). The peer model is created and maintained with the aid of the PA, who provides authorized personal information about the peer it represents (collecting info about peers goal).

All user interactions with KARe are handled by the PA, each community member being represented by one PA (acting on peer's behalf goal) which can be noted by the only dependency from the CoP Member towards the PA (interacting with peers goal). By getting involved in social interaction, the CoP Member is able to exchange knowledge about work content and practices with other community members, getting involved in a rich knowledge exchange that may foster community evolvement and Newcomer's adjusting to the organizational context.

The KARe system, earlier described in [22], is inspired on developments of an earlier system proposal [14] (we refer to this publication for details on the agent's information and interaction design), and is developed on top of an existing application [1], which presents a semantically enriched technique for content management, and performs the basic peer-to-peer operations needed for this work's purpose. KARe extends this application by providing some degree of proactiveness in recommending items to users, and on building over the existing methods for evolving the available *querying* and *retrieval* techniques.

6. Related Work

As already mentioned, a few approaches to the application of the agent organization paradigm for the development of KM systems have been recently proposed [6,9,18,24]. Here we shall focus on those that are mostly influencing our methodology.

Perini et al. [18] supports the use of agents to model organizational processes, proposing a methodology for analyzing KM requirements based on intentional analysis, claiming that, in order to develop effective KM solutions, it is necessary to analyze the intentional dimension of the organizational setting, i.e. the interests, intents, and strategic relationships among the actors of the organization. Their methodology is based on the use of the i* framework [28], which models the organization as a set of actors, goals, softgoals, dependencies, tasks and resources. In a sense, our work builds on this initiative since it applies the same modeling framework and with similar purposes. In our approach Agent-Oriented modeling is proposed as one of the techniques at support of a more complex analysis process (see Fig. 1). Moreover, having adopted the *Tropos* methodology allows using a more clear semantics of i* elements (in [2] a metamodel for the modeling language used in *Tropos* is given) and will enable a smooth transition to our design approach.

Dignum proposes OperA [6], recognizing that, like multi-agent systems, KM environments can be seen as distributed systems where different actors, each pursuing its own goals, need to interact in order to achieve common targets and realize organizational objectives. In order to model the different roles, goals and interactions within an organization, this methodology proposes: a) an *organizational model* describing the domain's structure and global characteristics, considering that the society's goals determine agent roles and interaction norms; b) a *social model*, describing the commitments that regulate roles enactment by individual agents; and c) an *interaction model*, in which the society's agents interactions is described by the means of interaction contracts. The similarity between our approach and OperA is

mainly given by the use of actors, roles and goals. However, the modeling constructs applied are completely diverse, for instance, while OperA is based on scripts and logic formulas, our work proposes the use of a visual language and is much less formal.

Wagner [24] has proposed AORML, an UML-based modeling language in which an agent can be *institutional*, *human* or *artificial*. Institutional agents are usually composed of a number of human, artificial, or other institutional agents that act on their behalf. This distinction allows the analyst to make a domain model and then, gradually introduce artificial agents to support KM. To represent rights and responsibilities, AORML introduces, respectively the deontic modeling constructs of *claims* and *commitments*. Besides agents, ordinary objects are used to model the passive entities of the domain. In this way, the knowledge artifacts exchanged can be modeled as objects, providing the system analyst and designer to represent and reason about them. We refer to [14] for a more extensive discussion and exemplification on the use of AORML for KM, in the context of collaborative learning. Although we acknowledge the possibility of using AORML in domain modeling, we feel that this language lacks the concepts and constructs to support requirements analysis. In general, modeling with AOR starts with information modeling (like in UML class diagrams), jumping over the requirements analysis step. The solution proposed by Wagner for these initial phases is the use of UML Use Cases, however, we claim that our approach is more appropriate for focusing on goals, supported by Nonaka & Takeuchi's emphasis on intention (i.e. goals) [16] as the basis of any KM project.

7. Conclusion and Future Work

The paper presented a novel methodology for analyzing the requirements of a Knowledge Management System, and illustrated it with a case-study. The analysis process rests on an iterative workflow in which agent-oriented modeling plays a crucial role in understanding the domain's (organization) stakeholders needs for KM systems, basically, by tracing system requirements back to the stakeholders goals.

The agent-oriented modeling approach that has been exploited is based on the *Tropos* software development methodology [2]. It offers both a visual modeling language and some analysis techniques. The main contributions of this approach is supporting domain modeling in terms of the organizational structure and the emergent goals of each human or organizational agent involved in a given context. Other benefits are reached by the adoption of a visually rich and accessible notation, supported by several modeling tools [17,23].

We believe that providing such kind of informal visual methodology can be quite beneficial for the KM community, since business analysts and consultants will be able to use it without having to get acquainted with more formal approaches. As an alternative, computer science expert analysts can also combine the use of Tropos with the *Formal Tropos* language [10], intertwining formal and informal modeling to obtain more precision and consistency.

Concerning the fundaments of our approach, recent works in agent organizations [7] elaborate on relevant concepts for analyzing and supporting communities, including: a) *norms*, which describe and enforce common standards of behavior and therefore enable identification of members with the community; b) *identification*, that is, the awareness of each member of its connection to others and to the community as a whole; and c) *trust*, which will enable predictability of action and comfortable handling for the members. While concepts of obligations between the

actors of an organization and identification may be dealt with in our model, further work is needed to effectively represent concepts of trust and norms.

For future work, we shall point out two distinct directions: a) going ahead with the KM system analyzed in the paper, i.e. the KARe system's development and validation; and b) extending the fundaments of the methodology.

The detailed design of KARe will be based on the use of AORML [24], which allows us to move on with the description of agent's mental model, considering beliefs, commitments and the interaction between agents and objects, representing the passive entities used by the agents to accomplish their goals. Benefits as a result of the application of this language may be attributed to our proposals on how these abstract concepts can be materialized in practical elements of a system. This will require us to deal with interesting research issues such as defining a conversion method between elements of the Tropos notation to AORML. A preliminary version of this conversion method has been developed [13] and is currently being revised, for achieving semantic consistency regarding the metamodels of the two modeling approaches. This refinement will allow the integration of both methodologies in a CASE tool [17] (currently under-development), thus facilitating the tasks of analysts and designers on system development.

Bibliographic References

1. M. Bonifacio, P. Bouquet, G. Mameli, and M. Nori. Peer-Mediated Distributed Knowledge Management. In [8], pages 31-47.
2. P. Bresciani, P. Giorgini, and F. Giunchiglia, J. Mylopoulos, A. Perini. Tropos: An Agent-Oriented Software Development Methodology. In *International Journal of Autonomous Agents and Multi Agent Systems*, 8(3):203–236, May 2004.
3. C. Castelfranchi, A. Cesta, and M. Miceli. Dependence Relations in Multi-Agent Systems. In Y. Demazeau and E. Werner (Eds.) *Decentralized AI - 3*, North-Holland: Elsevier, 1992.
4. L. K. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos. Non-Functional Requirements in Software Engineering. Kluwer Publishing, 2000.
5. V. Dignum, L. Sonenberg, and F. Dignum. Dynamic Reorganization of Agent Societies. In *Proceedings of the Workshop on Coordination in Emergent Agent Societies at ECAI 2004*, Valencia, Spain, 2004.
6. V. Dignum A model for organizational interaction: based on agents, founded in logic. PhD Thesis. Utrecht University, Jan. 2004.
7. V. Dignum, and P. van Eeden. Seducing, Engaging and Supporting communities at Achmea. In *Proceedings of the 4th European Conference on Knowledge Management*, Oxford, UK, 2003.
8. L. van Elst, V. Dignum, and A. Abecker (Eds.) Agent-Mediated Knowledge Management, Heidelberg: Springer-Verlag, 2004.
9. J. Ferber, O. Gutknecht, and F. Michel. From Agents to Organizations: An Organizational View of Multi-agent Systems. In P. Giorgini, J. P. Müller, and J. Odell (Eds.) *AOSE 2003*, Heidelberg: Springer-Verlag, pages 214–230, 2004.
10. A. Fuxman, L. Liu, J. Mylopoulos, M. Pistore, M. Roveri, Paolo Traverso. Specifying and analyzing early requirements in Tropos. In *Requirements Engineering Journal*, 2004.
11. P. Gongla and C. R. Rizzuto. Evolving communities of practice: IBM Global Services experience. In *IBM Systems Journal*, 40(4), 2001.

12. R. S. S. Guizzardi, A. Perini, and V. Dignum. Providing Knowledge Management Support to Communities of Practice through Agent-oriented Analysis. In *Proceedings of the 4th International Conference on Knowledge Management*, Graz, Austria, pages 320-328, 2004.
13. R. S. S. Guizzardi, A. Perini, V. Dignum. Using Intentional Analysis to Model Knowledge Management Requirements in Communities of Practice. Technical Report TR-CTIT-03-53, Dec. 2003.
14. R. S. S. Guizzardi, L. Aroyo, G. Wagner Agent-oriented Knowledge Management in Learning Environments: A Peer-to-Peer Helpdesk Case Study. In [8], pages 57-72.
15. N. R. Jennings, K. P. Sycara, and M. Wooldridge. A Roadmap of Agent Research and Development. In *Journal of Autonomous Agents and Multi-Agent Systems*. 1(1): 7-36. 1998.
16. I. Nonaka and H. Takeuchi. The Knowledge Creating Company: How Japanese Companies Create the Dynamics of Innovation. New York: Oxford University Press, 1995.
17. A. Perini and A. Susi. Developing tools for Agent-Oriented visual modeling. In *Proceedings of the 2nd German Conference on Multiagent System Technologies*, Erfurt, Germany, 2004.
18. A. Perini, P. Bresciani, E. Yu, A. Molani. Intentional Analysis for Distributed Knowledge Management. In [8], pages 351-367.
19. D. Pumareja, T. Bondarouk, and K. Sikkel. Supporting Knowledge Sharing Isn't Easy - Lessons Learnt from a Case Study. In *Proceedings of the Information Resource Management Association International Conference*, Philadelphia, 2003.
20. G. M. Serugendo, A. Karageorgos, O. F. Rana, and F. Zambonelli. Engineering Self-Organising Systems: Nature-Inspired Approaches to Software Engineering. Heidelberg: Springer-Verlag, 2004.
21. J. S. Sichman and R. Conte. Multi-Agent Dependence by Dependence Graph In C. Castelfranchi and W.L. Johnson (Eds.) *Bringing People and Agents Together, Proceedings of AAMAS 2002,* ACM Press, Part I, pages 483-490, 2002.
22. A. Soller, R. S. S. Guizzardi, A. Molani, and A. Perini. SCALE: Supporting Community Awareness, Learning and Evolvement in an Organizational Learning Environment. In *Proceedings of the 6th International Conference of the Learning Sciences*, Santa Monica/CA, 2004.
23. Tropos Project Homepage: Requirements-Driven Development for Software Agents – Tools section [online] at: http://dit.unitn.it/~tropos/tools.html, Nov. 2004.
24. G. Wagner. The Agent-Object-Relationship Meta-Model: Towards a Unified View of State and Behavior. In *Information Systems*, 28(5), 2003.
25. E. Wenger. Communities of Practice: learning, meaning and identity. New York: Cambridge University Press, 1998.
26. M. J. Wooldridge. Intelligent Agents. In G.Weiss (Ed.) *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence.* Cambridge: MIT Press, pages 27-77, 1999.
27. M. J. Wooldridge and P. G. Ciancarini. Agent-Oriented Software Engineering: The state of the art. In P. G. Ciancarini and M. J. Wooldridge (Eds.) *AOSE 2000*, Heidelberg: Springer-Verlag, pages. 1–25, 2001.
28. E. Yu. Modeling Strategic Relationships for Process Reengineering. PhD thesis, University of Toronto, 1995.