Knowledge Management in Learning Communities

Renata S. S. Guizzardi[1], Gerd Wagner[2], Lora Aroyo[3]

[1]Computer Science Department – University of Twente (UT)
P.O. Box 217 – 7500 AE Enschede – The Netherlands
souza@cs.utwente.nl
[2]Institute of Informatics, Brandenburg University of Technology at Cottbus – Germany
Email: G.Wagner@tu-cottbus.de
[3]Department of Computer Science – Eindhoven University of Technology (TU/e)
P.O. Box 513 – 5600 MB Eindhoven – The Netherlands
L.M.Aroyo@tue.nl

Abstract. Collaborative learning motivates active participation of individuals in their learning process, which often results in the attaining of creative and critical thinking skills. In this way, students and teachers are viewed as both providers and consumers of knowledge gathered in environments where everybody teaches and learns, by interacting with each other. Peer-to-peer networking reflects and supports this non-hierarchical relationship between teachers and students in a collaborative learning community. In this paper we present Help&Learn, an agent-based peer-to-peer helpdesk system to support extra-class interactions among students and teachers. Help&Learn expands the student's possibility of solving problems, getting involved in a cooperative learning experience that transcends the limits of classrooms. To model Help&Learn, we have used Agent-Object-Relationship Modeling Language (AORML), an UML extension for agent-oriented modeling. The aim of this research is two-fold. On the one hand, we aim at exploring Help&Learn's potential to support collaborative learning, discussing its knowledge management strategy. On the other hand, we aim at showing the expressive power and the modeling strengths of AORML.

Keywords: collaborative learning communities, knowledge management, peer-to-peer, agent-oriented architecture.

1 Introduction

Collaborative learning reflects the 'constructivist' approach, where the learner does not simply reproduce reality but actively creates it, usually in a collaborative dialogue with other actors [9]. Cognitive psychology theories, and especially the work of Piaget and Vygotsky [6], claim that collaboration is essential for the development of logical thinking and, ultimately, learning. In this way, instead of being based on information assimilation and memorization, collaborative methods are based on students of different performance levels working in groups, sharing a common goal.

Contrasting with traditional education practices, which view teachers as producers and students as consumers of knowledge, in collaborative learning, both teachers and students are seen, at the same time, as producers and consumers, gathered in an environment where everyone has something to teach and something to learn.

Consequently, instead of playing the role of detaining and transmitting knowledge, the teacher assumes other functions, such as those of motivator, guide and collaborator. Meanwhile, the students become more active and responsible for their own learning.

Papert's Constructionist theory [3] emphasizes the importance of sharing knowledge by the means of concrete artifacts. He claims that learning effectively occurs when the learner is "engaged in the construction of something external or at least shareable... a sand castle, a machine, a computer program, a book." [3]. According to Papert, building something meaningful and sharable leads to a cyclic process of externalizing knowledge that is in the mind of the learner and internalizing new structures, as a result of the social interaction around this external artifact. This externalization and internalization cycle seems to coincide with Nonaka & Takeushi's Knowledge Management (KM) theory [13]. According to them, there are two types of knowledge: explicit and tacit. The former refers to codifiable components, which can be disembodied and transmitted, while the latter refers to knowledge that is "confined in people's mind", being difficult to articulate and disseminate. Through social interaction and collaboration, tacit knowledge is turned into explicit, and individual knowledge is turned into organizational. Organizational knowledge creation is a result of a continuous and dynamic process of conversion between these two knowledge types.

Collaborative learning communities can be seen as organizations that share both explicit and tacit knowledge. A common problem of these settings is the fact that the community's resources are distributed among its members, thus it is not easy to find out who has the right piece of information, knowledge or advice. Targeting this problem, we propose Help&Learn (H&L), a peer-to-peer system aimed at supporting the organization and sharing of distributed knowledge in collaborative learning communities. Peer-to-peer technology allows sharing of resources via direct exchange among individual systems in a digital network, naturally supporting KM by closely adopting the conventions of face-to-face communication [16]. In such networks, there are no central servers controlling the interactions among peers. This horizontal relationship between peers allows the creation of rich knowledge sharing environments, in which people look for each other based on common interests, social affinity and personal characteristics. This configuration can be quite interesting to support collaborative learning communities, reflecting the non-hierarchic relationship between teachers, students and other members.

From a software engineering perspective, the analysis and design of the distributed processes involved in knowledge management become increasingly sophisticated and require an agent-oriented approach, such as the Agent-Object-Relationship Modeling Language (AORML) [20], an extension of UML to model agent-oriented information systems. The strengths of AORML with respect to KM systems are: 1) it considers the organizations and actors of a domain as agents in the modeling process. In this way, it allows to model business processes on the basis of the interactions between (human and artificial) agents working on behalf of their organizations. Related work is mentioned in [5]. Although norms and contracts are not directly supported by AORML, it provides deontic modeling constructs such as commitments and claims with respect to external agents, and obligations and rights with respect to internal agents. 2) the fact that 'mentalistic' concepts of agents, such as beliefs and commitments, are explicitly considered in the system model, supports the software engineer to reason about and to model the behavior of agents, both internally and in interaction with other agents of the system; 3) it captures the behavior of agents with the help of rules. Besides these strengths, since AORML is an extension of UML,

preserving its principles and concepts, it is an accessible language, and it is likely to face less resistance for industrial acceptance and use.

The aim of this research is two-fold. On the one hand, we aim at exploring H&L's potential to support collaborative learning, discussing its KM strategy. On the other hand, we show that AORML is an appropriate language to model such a system, as well as other KM environments. In section 2, KM is described in connection with the educational context. Section 3 presents a description of H&L, introducing the main problems and activities in focus. Section 4 introduces AORML and its modeling constructs, which are then applied in section 5, presenting part of the Help&Learn system's modeling. Section 6 acknowledges some work related to Help&Learn and to AORML. Finally, some conclusions and future work directions are presented in sections 7.

2 Knowledge Management in Education

Collaborative Learning mediated by network-based environments have been the focus of many recent research initiatives and experiments [8,15], especially within the CSCL and the E-learning communities. The need for a different kind of learning approach has been also noted within the KM literature, like in [6]:

"The traditional education paradigm is inappropriate for studying the types of open-ended and multidisciplinary problems that are most pressing to our society. These problems, which typically involve a combination of social and technological issues, require a new paradigm of education and learning skills, including self-directed learning, active collaboration, and consideration of multiple perspectives."

Maçada and Tijiboy (1998) [8] consider three essential elements for collaborative learning to succeed in network-based environments: a) cooperative posture, which involves: non-hierarchical relationship between the participants, collaboration, constant negotiation, open-mindedness, etc.; b) collaborative technological infrastructure; and c) a non-hierarchical method, i.e. it is very important that all the participants get involved in the constant organization and re-organization of the environment dynamics (meaning the establishment of goals, norms, roles, priorities of tasks, etc.).

Especially focused in b), this work is based on the assumption that KM can be generally beneficial for learning [15]. KM can, for instance, motivate learners to be more active and to collaborate. While feeding a KM system, the users need to create artifacts, externalizing their knowledge, in order to make it available for other users (user-based approach for knowledge creation, similar to the one adopted in [6]). This process of externalization is an important step for learning. Supporting this idea, Constructionist learning theories emphasize the importance for the learner to produce something concrete, which he can share with his peers [3]. In other words, externalizing knowledge by means of a sharable artifact will help the learner to perform synthesis and learn, and at the same time it may motivate him for peer collaboration.

The knowledge resources exchanged in a learning environment cannot be much differentiated from those exchanged for other purposes. In this context: i) there is a share of physical resources, such as: books, articles, and other educational artifacts; ii) with the growing use of information technology and the Internet in these settings, there are plenty of electronic documents, references, and web links; and, finally, iii) there is also tacit knowledge [6], i.e. knowledge that is contained in people's minds

and that is usually informally exchanged among them by different means, for instance, in person, through messages, or via Internet communication tools integrated in virtual learning environments [15]. All these forms of knowledge need to be properly integrated and managed in order to bring about positive changes in the teaching/learning process.

Exemplifying the common difficulties of this context, we mention the fact that all these resources are distributed among people and that it is not easy to find out who has the right piece of information, knowledge or advice. The nature of these problems suggest that KM systems (KMSs) can be highly recommendable for learning settings. In addition to that, software agents' specific characteristics turn them into promising candidates in providing a KMS solution [5]. These agents can be used both as a metaphor to model the domain in which the system will be deployed, and as software components to develop the actual KMS.

Targeting the above highlighted problems, we propose H&L to support the organization and sharing of distributed knowledge. The specifics about the KM strategies applied in H&L are presented in the next session.

3 Help&Learn: A Peer-to-Peer Architecture for Knowledge Management in Learning Settings

Peer-to-peer introduces a set of concepts that takes a human centered view of knowledge as residing not just in people's minds but also in the interaction between people and between people and documents [19].

H&L expands the student's possibility of solving their doubts, getting involved in a cooperative learning experience that transcends the limits of classrooms. By collaborating with other peers, the students learn with the doubts of others, besides developing cognitive abilities, such as to state clearly their doubts and thoughts; to interpret questions; to mediate discussions; and to solve problems. In this open context, other interested parties may join the learning community, such as business employees and online organizations. They bring different perspectives to the discussions, making the cooperation richer. Figure 1 shows the peer-to-peer architecture of the proposed scenario.

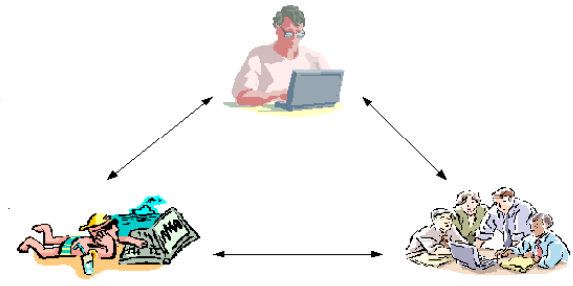

Fig. 1. A teacher, a student, and employees of a company, interacting to ask and answer questions in the proposed peer-to-peer environment

We use the metaphor of a helpdesk, where somebody asks for help (the helpee) and somebody provides the needed help (the helper). Each peer in the network is seen as a source of knowledge. The agents of the system are responsible for managing the

exchanges between these sources. This includes: a) handling a peer request for help and delivering help in a personalized way; b) finding the best peer to answer to a help request; and c) searching through previously asked questions/answers [15].

In H&L, knowledge is created and integrated in use-time, including users participation in these processes, and not in design time with the help of a knowledge engineer. This model has many advantages, such as: avoiding that knowledge artifacts become obsolete, for being dependant on the knowledge engineering; and motivating the users of the system to engage in collaboration and learning, while creating and sharing the artifact [6]. The knowledge in H&L is exchanged by the system peers in the form of HelpItems. These HelpItems can be what-is or how-to-do explanations, bibliographic or Web references, electronic documents, or even hardcopies, depending on the peers setting (e.g. inside a school or a company, hardcopies can be exchanged in addition to electronic copies).

The users are not required to perform knowledge formalization. The exchanged questions and answers are expressed and stored in natural language. Besides mediating this exchange of help, the system agents are responsible for searching through previously asked questions and answers to provide the users with suitable help.

The quality of knowledge artifacts is an important issue in KMSs [6]. In H&L, this is measured by the peers themselves. The help provided is annotated by the helpee, and this information is shared among the agents of the system, to be considered in future helper indication.

As in a typical peer-to-peer application [19], a key issue here is finding the best peer to satisfy a certain help request. A helper is selected if she can fulfill a help request, by providing the helpee with appropriate HelpItems. Besides expertise, the time and availability of the peer are also considered for the best helper indication. As an example, a teacher may know the answer to a student's question but she may have less time than an advanced student to spend on it.

A common problem in KM settings is motivating the users of the system to use it in its full potential [6,8]. The peer motivation to participate in discussions and answer to helpees' questions can be given by a sense of belonging to a learning community, or by the desire of having a good social status [8]. However, this motivation can also be caused by external factors, like teacher's reinforcements or an external grading system.

4 Agent-Object-Relationship Modeling

The Agent-Object-Relationship (AOR) modeling approach [20] is based on an ontological distinction between active and passive entities, that is, between *agents* and *objects*. This helps to capture the semantics of complex processes, such as the one that involves teachers and students, owners and employees of a company, and other actors involved in a KM environment. The agent metaphor subsumes both artificial and natural agents. This way, the users of the information system are included and also considered as agents in AOR modeling.

Intuitively, some connections can already be identified between the knowledge artifacts in a KMS and objects, and between the KMS users and human agents. The KMS itself can also be composed of multiple software agents, which perform different tasks, accomplishing various goals, in order to mediate the processes of

knowledge creation, integration and sharing. These agents can be identified and modeled with the aid of AORML.

AOR distinguishes between agents and objects according to these two main points: 1) while the state of an object in OO programming has no generic structure, the state of an agent has a 'mentalistic' structure: it consists of mental components such as beliefs and commitments. 2) while messages in object-oriented programming are coded in an application-specific ad-hoc manner, a message in Agent-Oriented Programming is coded as a 'speech act' according to a standard agent communication language that is application-independent.

In AORML, an entity can be an agent, an event, an action, a claim, a commitment, or an ordinary object. Agents and objects form, respectively, the active and passive entities, while actions and events are the dynamic entities of the system model. Commitments and claims establish a special type of relationship between agents. These concepts are fundamental components of social interaction processes and can explicitly help to achieve coherent behavior when these processes are semi or fully automated.

Only agents can communicate, perceive, act, make commitments and satisfy claims. Ordinary objects are passive entities with no such capabilities. Besides human and artificial agents, AOR also models institutional agents. Institutional agents are usually composed of a number of human, artificial, or other institutional agents that act on its behalf. Organizations, such as companies, government institutions and universities are modeled as institutional agents, allowing us to model the rights and duties of their internal agents.

There are two basic types of AOR models: external and internal models. An external AOR model adopts the perspective of an external observer who is looking at the (prototypical) agents and their interactions in the problem domain under consideration. In an internal AOR model, we adopt the internal (first-person) view of a particular agent to be modeled.

This paper is focused on the exemplification of external AOR models, which provide the means for an analysis of the application domain. Typically, these models have a focus, that is an agent, or a group of agents, for which we would like to develop a state and behavior model. Figure 2 shows the elements of an external AOR model, in which the language notation can be seen.

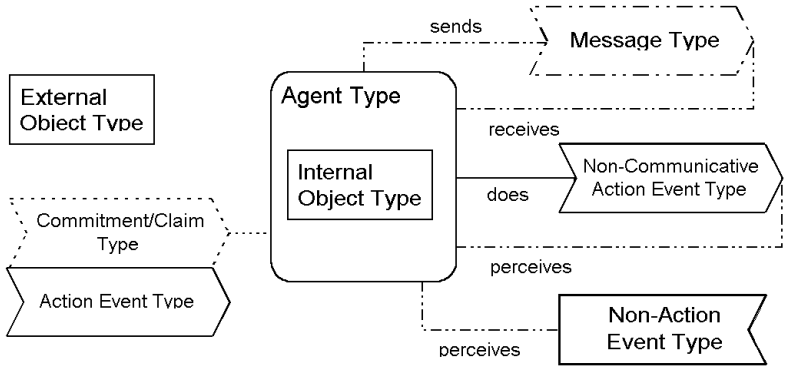

Fig. 2. The core elements of AOR external models

Object types belong to one or several agents (or agent types). They define containers for beliefs. If an object type belongs exclusively to one agent or agent type,

the corresponding rectangle is drawn inside this agent (type) rectangle. If an object type represents beliefs that are shared among two or more agents (or agent types), the object type rectangle is connected with the respective agent (type) rectangles by means of an UML aggregation connector.

As it can be seen in Figure 2, there is a distinction between a communicative action event (or a message) and a non-communicative action event. Also, AOR distinguishes between action events and non-action events. The figure also shows that a commitment/claim is usually followed by the action event that fulfills that commitment (or satisfies that claim).

An external model may comprise one or more of the following diagrams:

- *Agent Diagrams (ADs)*, depicting the agent types of the domain, certain relevant object types, and the relationship among them. An AD is similar to a UML class diagram, but it also contains the domain's artificial, human and institutional agents.
- *Interaction Frame Diagrams (IFDs)*, depicting the action event types and commitment/claim types that determine the possible interactions between two agent types (or instances).
- *Interaction Sequence Diagrams (ISDs)*, depicting prototypical instances of interaction processes.
- *Interaction Pattern Diagrams (IPDs)*, focusing on general interaction patterns expressed by means of a set of reaction rules defining an interaction process type. Reaction rules are the chosen component by AOR to show the agent's reactive behavior and it can be represented both graphically and textually.

These diagrams will be exemplified in the following section. For further reference, we refer to [20] and to the AOR website: http://aor.rezearch.info/.

5 Help&Learn Modeling

AORML can be used throughout the whole development cycle of a system. In this paper, we will focus on the analysis phase, in which we applied AOR external models. Figure 3 depicts the agent diagram, which includes all human, artificial and institutional agents (distinguished by UML stereotypes) involved in the helpdesk, and their relationships. Note that this diagram is very similar to the UML class diagram, showing the system's classes and relationships between them. For clarity purposes, the attributes of agents and objects are omitted in this diagram. However, they can be expressed following the traditional UML syntax.

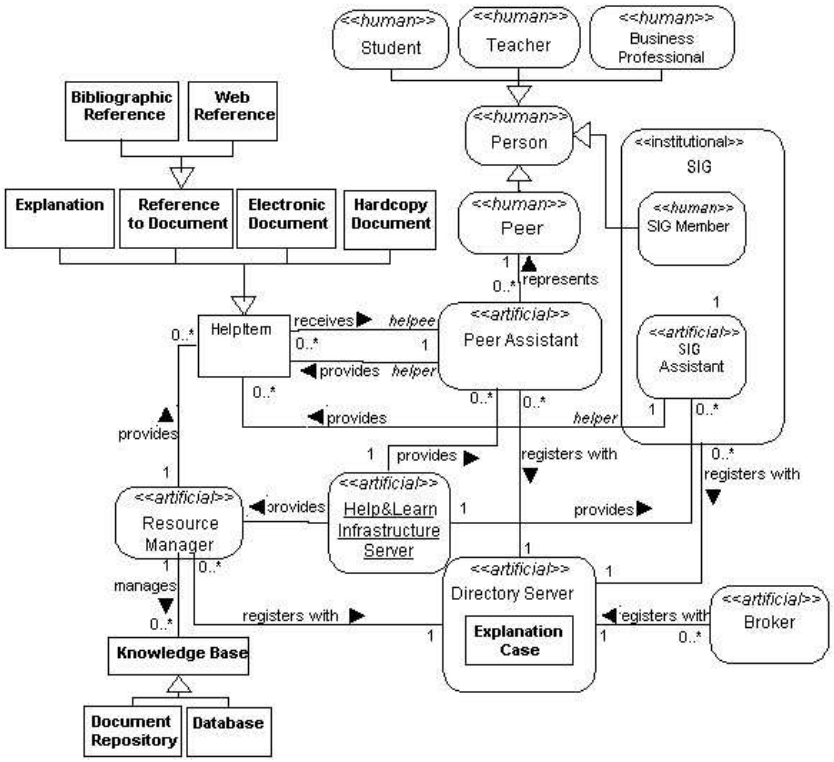

Fig. 3. Helpdesk System Agent Diagram

As the diagram in Fig. 3 shows, H&L brings together students, teachers and general business professionals as peers of a learning community. Below, we give a brief description of each artificial agent of the system.

Help&Learn Infrastructure Server (IS). This agent addresses the management of the H&L system itself. It provides the other artificial agents of the system, as well as periodic updates.

Peer Assistant (PA). In order to start participating on discussions in the system, a Person downloads the Peer Assistant (PA) from the H&L IS. This way, this Person becomes one of the system Peers, being able to act both as a helpee and as a helper for other Peers.

Directory Server (DS) and Broker (B). Every time the PA goes online, it registers with the Directory Server (DS), becoming available to answer help requests. When doing this, the PA will provide the DS with a minimal Peer profile, indicating what topics can be answered by him. On the other hand, the Broker creates his own Peer profile by contacting the PAs and also by applying data mining techniques on the DS profiles, in order to make rankings and classifications. The Broker ranks the Peers based on expertise, availability and reliability and it classifies them based on interests. This way, when queried by the PAs, it can provide information on the most appropriate Peers to answer a certain help request. The DS also maintains a repository of previously provided explanations, along with their respective request (typically, a question). This way, the PA consults this agent every time a question is forwarded to it by a helpee, to check whether or not this question has been already answered. If so, the answer is immediately recovered to the helpee; otherwise, the PA consults the

Broker for a best helper indication. In this repository, Information Retrieval Techniques are used in order to group similar questions and aid the retrieval of the relevant ones, as well as to support the creation of an automatic FAQ, according to the proposals of a previous work [15].

SIG Assistant. Special Interest Groups (SIGs) are also allowed to participate in the system (this is indicated by the inclusion of the institutional agent SIG in the agent diagram of Fig. 3). These SIGs usually pre-exist the system, but can also be created by suggestion of the Broker. It is not necessary that all the members of a SIG are Peers, only one member is enough (note this, again in the agent diagram, which generalizes a SIG Member as a Person, instead of relating it with the Peer class). The Broker has a representation of the SIGs and can also suggest that a PA contacts one of the SIG Assistants in order to ask the SIG for help. The SIG Assistant broadcasts the message to all members of the SIG. Then, the answers are sent back to the PA. Today, there are many SIGs advertised in the Web, specialized in several different areas. By introducing them to the helpdesk system, we hope to broaden their interaction scope, at the same time that we give the opportunity for other Peers to have their help request answered by an expert on the topic.

Resource Manager (RM). The Resource Manager brings to the system existing knowledge bases, which can be databases, document repositories etc. This way, HelpItems that are not owned by any of the system Peers can also be considered and consulted by the PAs. These knowledge bases can be consulted through keyword or query search. A System Peer does not directly contact a RM. Instead, this is done through the PA. In the case of a query search, the Peer uses an interface, based on established query languages such as SQL, XML-Query, or RDF-Query, which will then be translated by the contacted RM to the query language of the specific knowledge base. These agents are typically downloaded by the owners of existing knowledge bases, who will create the translation specification.

5.1 Interactions in Help&Learn

The next step after defining the agents in the system is to model their interactions using Interaction Sequence Diagrams (ISDs) for concrete examples. In H&L, a Peer can request explanation, or for a document (reference, electronic copy or hardcopy). For reasons of lack of space, only the first one is exemplified in this paper.

A prototypical interaction sequence triggered when a Peer issues a request for an explanation is shown on Figures 4 and 5. Such sequence should generally be maintained in the same ISD, integrating the whole process. This is especially useful for automatic code generation. Here, we chose to divide the sequence in two phases in order to facilitate our exploration of the modeling language specifics. Moreover, this way the general understanding of the interaction sequence may be eased.

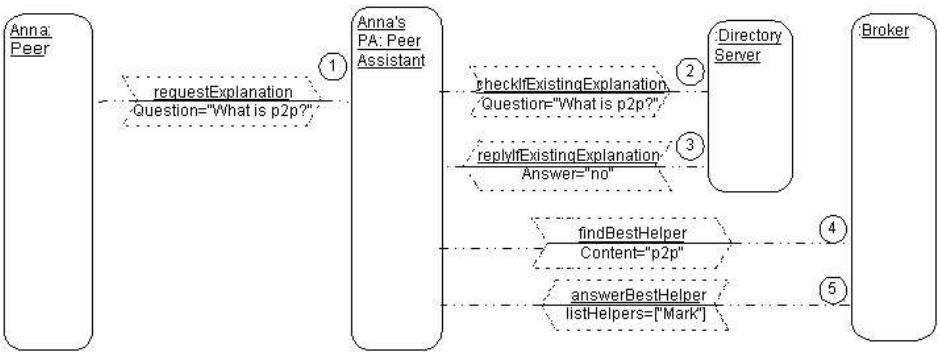

Fig. 4. Interaction Sequence Diagram, showing a help request being issued by Anna, a H&L Peer, and the best helper indication by the Broker

Figure 4 shows the Peer request and the best helper indication by the Broker. Here, Anna, a system Peer, issues a request for help to her PA, asking "what is p2p?". The PA attempts first to find out if this question has already been asked, by querying the DS maintaining the Explanation Case (see Fig.3). Since this question is asked for the first time, the PA cannot provide a direct answer and asks the Broker to find the best helper to answer this question. The Broker returns a ranked list of possible Peers for the PA to select. In our example, this list contains only one indication: Mark.

Having the Broker's indication, the PA will then try to get the HelpItem that fulfills its user's request. This is depicted in Fig. 5. It starts with Anna's PA contacting Mark's PA with the request for help. Mark's PA replies with an acknowledge message, confirming it received the request. In this moment, a commitment is established from Mark's PA towards Anna's PA, fixing that the first will try to get help (from its peers) to answer to the latter's request.

This commitment is also represented in the ISD of Fig. 5. It is created by the acknowledge message (dashed arrow along with a "C", for "Create") and it has two arguments, a provideHelp and a noHelpAvailable message. These two messages compose an Or-Split (diamond containing an "x"), which represent the possible outcomes if the commitment is fulfilled. If any other possibility occurred, it would mean the commitment had been broken. Proceeding in the ISD, we will see this is not the case in this example. Mark's PA forwards the request to Mark, who provides the following answer: "p2p is a distributed technology…". This message is then forwarded to Anna's PA (note an arrow from this message to the commitment, indicating its fulfillment). At last, the help (i.e. the explanation) gets to its destination: Anna.

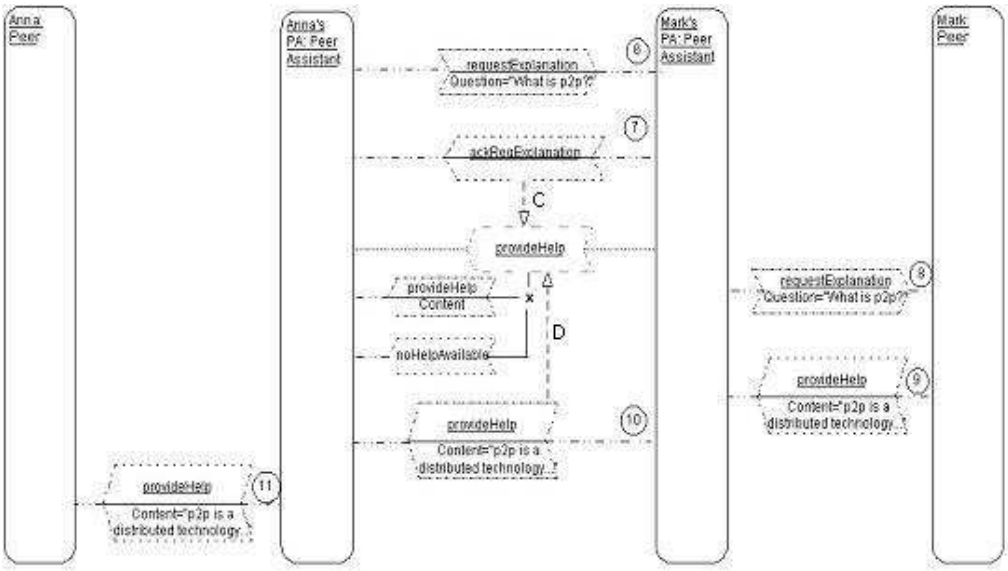

Fig. 5. Interaction Sequence Diagram, showing how a PA deals with the request for an explanation, on behalf of its user

The use of commitments supports situations in which the communication between two agents is asynchronous, as in this case. Mark's PA confirms it is going to provide the help. However, Anna's PA knows this can take some time, depending on Mark's availability and willingness to respond. Commitments are also good constructs to treat agent's autonomy. If it were useful, we could represent, for example, a commitment between Mark and its Peer, establishing that Mark commits to answer to the help request. At first sight, this does not seem very natural, since Mark is a human and, as such, has full autonomy over the system. In other cases, though, dealing with life-threatening situations and, of course, with artificial agents, this can be rather a good approach. Furthermore, commitments can be used as triggers for exception handling. For example, what should Anna's PA do in case Mark's PA does not meet its commitment? In H&L, this agent tries to find another Peer to answer to the help request.

Besides requesting an explanation, a Peer can also ask for documents, providing its PA with a list of keywords. Figure 6 depicts the interactions between Help&Learn's agents, when a request of this type is issued.

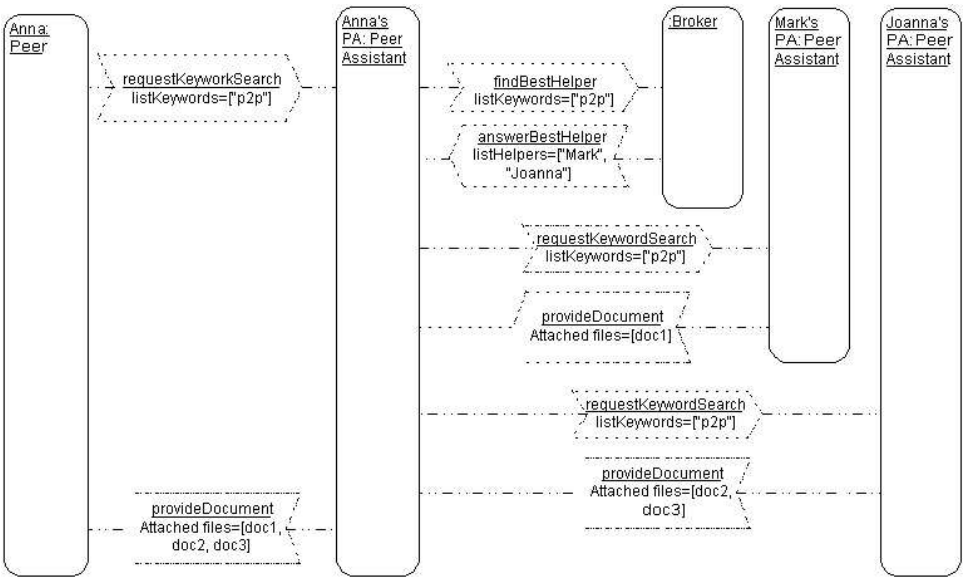

Fig. 6. Interaction Sequence Diagram, showing the interaction process when a request for a keyword search is issued by a peer

In the ISD of Fig 6, Anna requests its PA for documents about "peer-to-peer". The PA asks the Broker who are the best helpers to answer to this request. The Broker returns a list of ranked Peers to answer to the request. In this case, this list contain two Peers: Mark and Joanna. Next, Anna's PA contacts the PA of both Peers, forwarding the request for keyword search to them. The PAs search through the documents owned by their Peers, returning the available documents to Anna's PA. Finally, Anna's PA forwards the documents to Anna.

Note that the sequence shown in Fig. 4, 5 and 6 depicts just one of many possible interactions. The software engineer should make a number of ISDs in order to capture various interaction perspectives. This way he can afterwards generalize the interactions in Interaction Frame Diagrams (IFDs), which depicts the action event types and the commitment/claim types that determine the possible interactions between two agent types (or instances) [20].

Further, the interactions can be detailed in Interaction Patterns Diagrams (IPDs). These diagrams depict general interaction patterns expressed by means of a set of reaction rules, defining an interaction process type. Reaction rules are the chosen component by AOR to show the agent's reactive behavior, and they can be represented both graphically and textually. Figure 7 depicts an example of this type of diagram.

The IPD of Fig. 7 depicts only the two agents involved in this specific process: the PA and the DS. When the DS receives a checkIfExistingExplanation message, it immediately reacts, checking if the sent Question can be found in the Explanation Case (i.e. if the question is similar enough to one or more previously asked ones, according to DS's internal algorithms).

In the affirmative case, the DS sends back the respective answer to the PA. Otherwise, it simply "says no". This is modeled with the rule R1, which is textually represented (See Table 1).

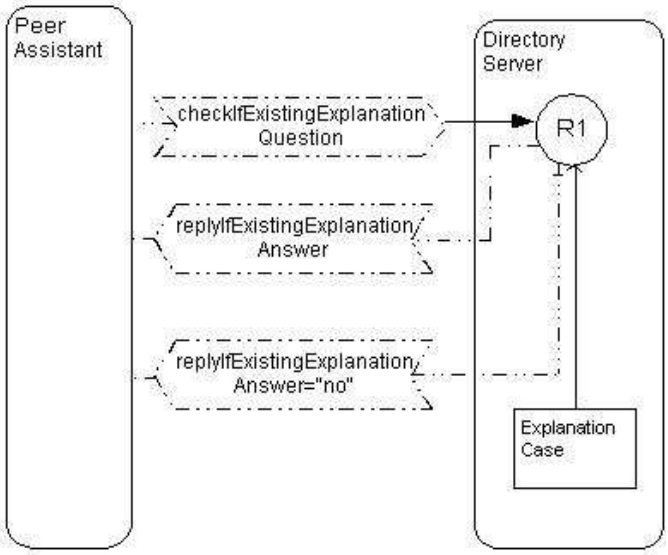

Fig. 7. Interaction Pattern Diagram, showing the PA's internal behavior when receiving a HelpItem on behalf of its user

Table 1. Textual Representation of the R1 Reaction Rule

ON	Event	RECEIVE checkIfExistingExplanation (?Question) FROM ?PeerAssistant
IF	Condition	ExplanationCase (?Question,?Answer)
THEN	Action	SEND replyIfExistingExplanation (?Answer) TO ?PeerAssistant
ELSE	Action	SEND replyIfExistingExplanation (?Answer="no") TO ?PeerAssistant

After the external model has been completed, the modeling can proceed to the design stage, in which, for each type of agent system to be designed, the external model is *internalized* according to the perspective of the respective agent, and subsequently further refined. For instance, an *action event*, if created by the agent to be designed, is turned into an *action*, while it is turned into an *event* if it is perceived by it. Using such an internal perspective and the corresponding indexical terms (such as *actions* and *outgoing messages* versus *events* and *incoming messages*), leads to a natural terminology for designing and implementing agents. H&L internal models will be the subject of future publications.

6 Related Work

Regarding Help&Learn, it is important to mention other initiatives on developing peer-to-peer architectures to support knowledge sharing. One of these initiatives is the EDUTELLA project [12], which aims at providing a peer-to-peer networking infrastructure to support the exchange of educational material. In order to accomplish

this, peers can make their documents available in the network, specifying metadata information as a set of RDF statements.

Bonifacio et al. [2] have developed KEx, a peer-to-peer system to mediate distributed knowledge management. KEx allows each individual or community of users to build their own knowledge space within a network of autonomous peers. Each peer can make documents locally available, along with their context, i.e. a semantic representation of the documents' content. When searching documents from other peers, a set of protocols of meaning negotiation are used to achieve semantic coordination between the different representations (contexts) of each peer. Both EDUTELLA and KEx are specifically concerned with the exchange of documents and do not address peer collaboration through the exchange of messages, which is one of the targets of H&L.

On the other hand, the work proposed by Vassileva [18] proposes a peer-to-peer system to support the exchange of messages between students. A student needing help can request it through his agent, which finds other students who are currently online and have expertise in the area related to the question. As in H&L, there is a centralized matchmaker service, which maintains models of the users competences and matches them to the help-requests. This work is particularly concerned with user motivation to collaborate. Thus, the system rewards users who contribute to the community, by providing them with a better quality of service.

Concerning agent-oriented modeling, we should mention AUML [14] and Message/UML [4] since both propose UML extensions to model agent-based systems. AUML has especially extended UML sequence diagrams to model interaction protocols involving agent roles. Message/UML proposes 5 views: Organization, Goal/Task, Agent/Role, Interaction and Domain views, each of them modeling a specific aspect of the multi-agent system. In comparison with AORML, these two approaches do not target domain modeling, being both design-oriented. Besides, both of them lack the 'mentalistic' concepts (commitments, claims and beliefs) presented by AORML.

It is also important to acknowledge the efforts of Molani et al. [11] in the direction of providing a system analysis and design methodology specific for the Knowledge Management domain. They claim 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. Like AORML, they take an agent-oriented approach to model the domain. The major difference when compared to AORML is the adopted i* framework. Instead of the AORML constructs of agents, objects, relationships, messages, commitments, etc., this framework models the organization as a set of actors, goals, 'soft goals', dependencies, tasks and resources.

7 Conclusions

In this paper, we have described our work in progress with Help&Learn, a peer-to-peer agent-oriented architecture, aimed at providing its users with a rich environment for both collaborative and individual use of knowledge. In order to do so, results on collaborative learning [6,10,15] and KM related research [5,7,8,13] have been used in the conceptualization and modeling of H&L. We take an agent-oriented perspective on system architecture, where agents play a crucial role in supporting the effectiveness, flexibility and personalization of the whole process. Following, we

apply an agent-oriented modeling approach (AORML), which proved to be an effective modeling language for our purposes. On the one hand, AOR models have led us thoroughly to this specification of H&L, aiding us on a system's requirements specification, analysis and initial design cycles. On the other hand, this experimentation has also provided us with feedback on how AORML can be extended, adding new constructs to facilitate agent-oriented modeling.

Although H&L's general architecture has been defined, many questions remain to be answered. In fact, AOR modeling has guided us on eliciting these open questions. In the future, we intend to address issues related to: a) the structuring of the questions and respective answers, present in the Explanation Case (EC); b) the organization of personal knowledge assets owned by each peer; and c) the management of HelpItems by the Resource Managers (RMs). Targeting a), we aim at investigating, for instance, how the techniques applied in a previous work [15] can be enhanced to provide suitable structuring and retrieval of the EC's questions and answers (refer to IPD of Fig. 6). This investigation, along with some extra studies, can indicate possibilities for addressing b) and c) as well. Inspired by current research on the Semantic Web [1], we intend to incorporate Ontologies into the H&L architecture. A preliminary study suggests that these ontologies can be aimed at making knowledge explicit, supporting interaction among the system peers.

References

1. Benjamin, R., Contreras, J., Corcho, O., Gomez-Pérez. A. (2002) Six Challenges for the Semantic Web. In KR2002 Semantic Web Workshop.
2. Bonifacio, M., Bouquet, P., Mameli, G., Nori, M. (2003) Peer-Mediated Distributed Knowledge Management. In van Elst, L., Dignum, V., Abecker, A. (eds.) Agent-Mediated Knowledge Management, LNAI 2926, Springer 2004.
3. Bruckman, A. (1997) MOOSE Crossing: Construction, Community, and Learning in a Networked Virtual World for Kids. PhD Thesis, MIT Media Lab, at: http://asb.www.media.mit.edu/people/asb/thesis/
4. Caire, G. et al. (2001) Agent Oriented Analysis using MESSAGE/UML In: Proceedings of the Workshop on Agent Oriented Software Engineering.
5. Dignum, V. (2002) An Overview of Agents in KM, at: http://www.cs.uu.nl/~virginia/#Publications
6. Dillenbourg, P. (1999). Collaborative learning: cognitive and computational approaches. Amsterdam [etc.]: Pergamon Press.
7. Fischer, G., & Ostwald, J. (2001) KM - Problems, Promises, Realities, and Challenges. In IEEE Intelligent Systems, Vol. 16, No. 1, Jan/Feb'01.
8. Ferran-Urdanet, C. (1999) Teams or Communities - Organizational Structures for KM. In Proc. of ACM SIGCPR'99.
9. Freire, P. (1970) Pedagogia do oprimido (In Portuguese) (Pedagogy of the Opressed). Rio de Janeiro: Paz e Terra.
10. Maçada, D. L., Tijiboy, A. V. (1998) Aprendizagem Cooperativa em Ambientes Telemáticos (In \Portuguese) (*Cooperative Learning in Telematic Environments*). In Proc. of IV RIBIE Conference, Brasilia, Brazil.
11. Molani, A., Perini, A., Yu, E., Bresciani, P. (2003) Analyzing the Requirements for Knowledge Management using Intentional Analysis. In van Elst, L., Dignum, V., Abecker, A. (eds.) Agent-Mediated Knowledge Management, LNAI 2926, Springer 2004.
12. Nejdl, W., Wolf, B., Qu, C., Decker, S., Sintek, M., Naeve, A., Nilsson, M., Palmer, M., Risch, T. (2002) EDUTELLA: P2P Networking Infrastructure Based on RDF. In Proceedings of WWW2002, May 7-11, Honolulu, Hawaii, USA.

13. Nonaka, I. and Takeuchi, H. (1995) The Knowledge Creating Company: How Japanese Companies Create the Dynamics of Innovation. New York: Oxford University Press.
14. Odell, J., Parunak, H., Bauer, B. (2000) Extending UML for Agents. In Proceedings of the Agent-Oriented Information Systems Workshop at the 17th National conference on Artificial Intelligence.
15. Souza, R. S., Menezes, C. S. (2001) Aplicando Técnicas de RI no apoio às Interações Mútuas de uma Comunidade Virtual de Aprendizagem – Um Ambiente Orientado a Agentes (in Portuguese) (*Applying Information Retrieval Techniques to support mutual interactions in a Learning Virtual Community – An Agent Oriented Environment*). MSc. Thesis. Federal University of Espirito Santo, Brazil.
16. Tiwana, A. (2003) Affinity to Infinity in Peer-to-Peer Knowledge Platforms In: Communications of ACM, v. 46, n. 5, p. 76-80.
17. Ulbrich, A., Ausserhofer, A. (2002) Aspects of Integrating E-learning Systems with KM. In Proc. of IASTED 2002, Cancun, Mexico.
18. Vassileva J. (2002) Supporting Peer-to-Peer User Communities. In R. Meersman, Z. Tari (Eds.): CoopIS/DOA/ODBASE 2002, LNCS 2519, pp. 230-247. Springer-Verlag.
19. Viant Innovation Center: The Human Side of P2P: where technology and conversation come together, at: http://www.viant.com/downloads/innovation_p2p.pdf
20. Wagner, G. (2003) The Agent-Object-Relationship Meta-Model: Towards a Unified View of State and Behavior. *Information Systems*, 28:5.