A Multi-Agent Design of a Peer-Help Environment[*]

Julita Vassileva, Jim Greer, Gord McCalla, Ralph Deters,
Diego Zapata, Chhaya Mudgal, Shawn Grant
ARIES Lab, Department of Computer Science,
University of Saskatchewan, Canada
jiv@cs.usask.ca

Abstract: This paper presents a multi-agent approach to design of adaptive distributed collaborative and peer help environments, which addresses a number of challenges: locating appropriate human and electronic resources depending on the help-request, motivating users to help each other, and is easily extendible "in-depth" and "in-breadth". We explore two novel approaches in user modelling: modelling of user social characteristics and modelling the relationships between users to support user collaboration and peer help. We show how adaptive behaviour of a heterogeneous distributed system can arise as a result of negotiation of goal-oriented autonomous cognitive agents. We are introducing an economic model in order to motivate users to collaborate and provide peer help while protecting helpers from getting overloaded with requests. Finally, we propose using multi-agent simulation tools to test the appropriateness of various economic models for a learning environment.

1. Introduction

I-Help is an integration of previously developed ARIES Lab tools for peer help in university teaching. One of its components, CPR provides a subject-oriented discussion forum and moderated FAQ-list supporting students with electronic help. Another component, PHelpS selects an appropriate peer helper who can support the student with direct peer help via a synchronous chat environment. The selection of an appropriate discussion forum, FAQ-article or human peer helper is based on modelling learner knowledge in the context of the concept / topic structure of the subject material. See [3] for a more detailed description of the I-Help Project. This paper presents our ongoing research on distributing the centralized monolithic architecture of I-Help, by using a multi agent-architecture [9]. There are a number of reasons for adopting a multi-agent approach to developing AI-based educational systems. The rapid development of new technologies like telecommunications, networking, and mobility leads to new types of working environments where the borderline between working and learning disappears; just-in-time learning evolves into life-long learning, "hybrid societies" emerge, consisting of real persons and electronic agents. An important factor in this type of environments is that everyone and everything is connected, so possibilities emerge for peer help, collaboration, and sharing resources (computational, applications, human advice etc.). Four major consequences follow.

First, environments are needed that reduce complexity for users and allow them to concentrate on their primary goals or tasks, as well as support their learning and collaboration. Second, it becomes important to provide a source of motivation for benevolence and collaboration among users, along with protection mechanisms, ensuring security of communications, privacy of personal data, and equal chances for all to participate. Third, hard computational and software engineering challenges arise in large scale distributed learning environments, which can be overcome by an agent-based approach. Finally, the field of multi-agent systems opens a number of extremely interesting and potentially useful research avenues concerning inter-agent negotiation, persuasion and competition in agent societies. Modern society offers a rich reservoir of paradigms for modelling multi-agent systems, e.g. social roles and cultural values, norms and conventions, social movements and institutions, power and dominance distribution. We can learn from the adaptability, robustness, scalability and reflexivity of social systems and use their building blocks to come up with more powerful multi-agent technologies.

[*] Published in S. Lajoie, M. Vivet (Eds.) Artificial Intelligence and Education, Proceedings IAIED'99, Le Mans, France, IOS Press, 38-46.

I-Help provides an excellent environment for studying these issues. A distributed, agent-based architecture reflects naturally the distributed web-based environment in I-Help. If one views "live" sessions with peer helpers and electronic peer help (discussion forum postings; on-line materials) as help resources provided by some agents (human in the first case, software in the latter case), then there is no virtual difference between humans and software agents. One can draw further conclusions: it is no longer advantageous to have a monolithic ITS, which is like an almighty teacher knowing the answer to any question that may arise in the learner. Networking provides a possibility to find some narrow-focussed learning resource, suitable for the domain of the question. This resource might be some courseware or a peer helper. In this way a human enters the teaching loop to compensate for limitations related to educational software. It is no longer a necessary condition that there is a powerful diagnostic component, to perform individualised teaching. A human can "enter this loop" too and help to diagnose the reason for learners' misunderstanding. This diagnosis can be later used by an ITS or by another human, taking the role of a helper to provide the learner with advice. In this way a multi-agent architecture provides for a natural synergy between humans and software agents (ITSs, diagnostic components, pedagogical expert systems, on-line help systems, web-based pools of on-line materials, etc.)

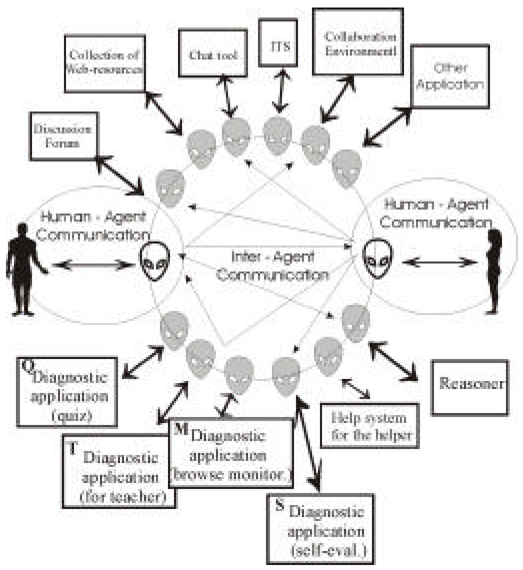
Collection of Web-resources
Chat tool
ITS
Collaboration Environment
Other Application
Discussion Forum
Human - Agent Communication
Inter - Agent Communication
Human - Agent Communication
Q Diagnostic application (quiz)
T Diagnostic application (for teacher)
M Diagnostic application (browse monitor)
S Diagnostic application (self-eval.)
Help system for the helper
Reasoner

Figure 1: Agent-based architecture

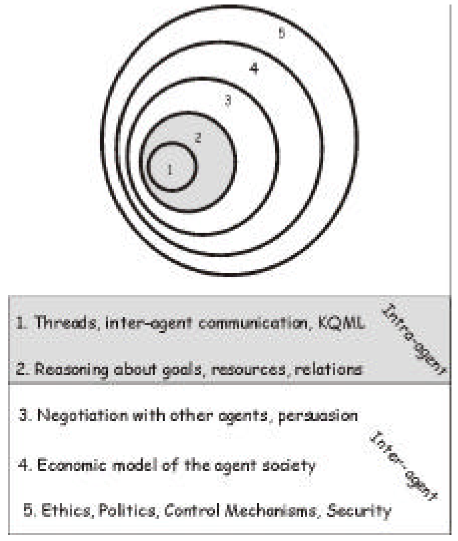
5
4
3
2
1
1. Threads, inter-agent communication, KQML
2. Reasoning about goals, resources, relations
Intra-agent
3. Negotiation with other agents, persuasion
4. Economic model of the agent society
5. Ethics, Politics, Control Mechanisms, Security
Inter-agent

Figure 2. A multi-level structure of an agent society.

2. Multi-level multi-agent architecture of I-Help

The adaptation within the multi-agent I-Help system is based on models of human users and models of involved software applications. These are maintained by two classes of agents (see Figure 1): personal agents (of human users) and application agents (of software applications). These agents use a shared taxonomy (ontology) and a communication language. Each agent manages specific resources of the user / application it represents, for example the knowledge resources of the user on certain tasks, topics or concepts or web-based materials belonging to an application. The agents use their resources to achieve the goals of their users, their own goals, and goals of other agents. Thus all the agents are *autonomous* and *goal-driven*. The agents are *cognitive* - they can plan the achievement of goals by means of decomposing them into sub-goals and relating them to resources. In their goal pursuit the agents can also use resources borrowed from other agents, i.e. they are *collaborative*. For this they have to negotiate and become involved in persuasion and conflict resolution. Each agent possesses a model of its inter-agent relationships, some of which reflect relationships between human users. Finally, the agents are *mobile* i.e. they can travel from one computer to another, thus optimising resources and bandwidth. In this way, we achieve a complex (multi-user, multi-application) adaptive (self-organised) system that supports users in locating and using resources (other users, applications, and information) to achieve their goals. A relatively detailed description of the multi-agent architecture we are using is presented in [9, 10].

The multi-agent architecture of I-Help can be viewed as a society of autonomous intelligent agents. It involves various levels of organization, including intra- and inter-agent organization. A schematic representation of the multi-level organization of the I-Help architecture (and the agent society structure) is shown in Figure 2.

2.1. Basic Agent Infrastructure

The environment in which all personal and application agents "live" guarantees a safe existence and fair chance of getting computational resources for all agents. It provides resource management, detects and limits or eliminates faulty and malicious agents, i.e. agents that consume too many resources. It also takes care of appropriate agent migration for the purpose of optimal use of resources. The agent environment consists of several self-explainable modules: monitor, resource manager, communication module, transporter (for agent migration) and magistrate (to make decisions in case of resource conflicts).

The agents are implemented as a set of tasks, which are mapped on Java threads. Each agent consists of two or more tasks: a *Kernel* task containing information about the state of the agent, a *Message* task that ensures that the agent can communicate with other agents, and one or more *Working* tasks that allow the agent to pursue specific goals. The anatomy of an agent at Level 1 is shown in Figure 3. An agent is an instantiation of these classes of threads. There can be multiple instantiations of such threads within one machine, therefore multiple agents can co-exist on one machine. Considering a pool of networked machines, a society of agents is virtually independent of the physical location of the threads of the individual agents. There can be an agent with a Message thread running on one machine and Working threads running on other machines.

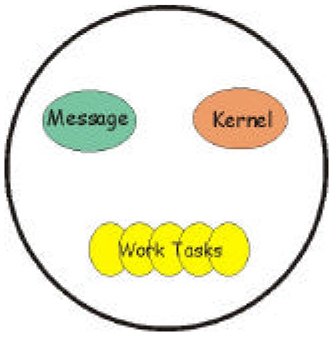

Figure 3. Anatomy of an agent on Level 1 (each bubble is a thread).

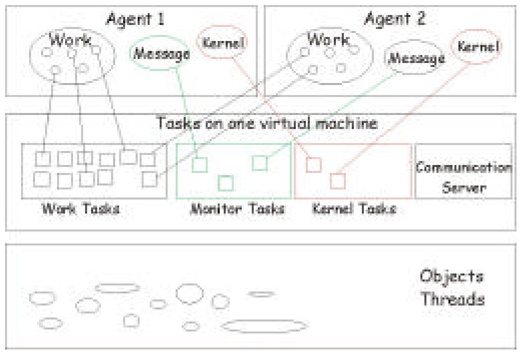

Figure 4. Different abstract views over the threads.

Agents can be "put to sleep" at moments when they have no active Message or Working threads. In this case, the Kernel thread is the only one that remains active, after storing information of the agent's last state in a database. In order to optimize resources, we have to provide a mechanism that allows an agent that has free Working or Message threads to offer them to other agents to use. This can be achieved by introducing an abstract "task level" between the agents and the actual processes running on the Java machine (see Figure 4). The upper two levels in Figure 4 are abstractions; only the lowest level exists in software consisting of Java objects and threads. However, these objects and threads can be viewed as belonging to one task, or to one agent, or to several cooperating / collaborating agents at the same time, since collaborating agents can be viewed as one agent (sharing Work threads, Message threads and the corresponding tasks). Agents communicate by sending messages (objects) to each other. Communication among agents is implemented by a set of communication primitives, which is a subset of KQML (extended with specific primitives). Upon receiving a speech act primitive from another agent, an agent has to analyze it and respond to it depending on how the speech act corresponds to the agent's goals and negotiation plan (strategy). Two different communication forms have been implemented:

- **Peer communication**: the agent uses an address book to generate a channel to the desired agent. The channel takes care to send the message to the target agent. The use of a channel enables migrations of agents during communication, which may be forced by computational resource limitations. A channel offers a bi-directional communication line, which is useful for private communication among agents.
- **Bus communication**: it offers communication among several agents. An agent can create, join, leave, or send a message to a bus. Every message is broadcast to all members. A bus can be itself a member of a bus. It is useful for sending requests among agents, but it may lead to a flood of answers that the requesting agent will have to process.

2.2. Autonomous Cognitive Agents

Users, learners, applications and learning environments are represented by autonomous goal-based social agents who communicate, co-operate, and compete in a multi-system and multi-user distributed environment (see Figure 1). Each agent maintains three distinct models of the user / application that it represents:

- a model of the user's/application's goals (in the case of human users – the goals and the preferences of the user),
- a model of the resources available to the user / application, which in the case of a user, can include his / her cognitive resources (i.e. there are several domain specific models of the user's knowledge, experience, skills etc.),
- a model of the relationships of the user or application (such as personal friends, acquaintances, agents which have proven to be useful, have offered help, and other agents who the user/application "owes a favour" to).

These models allow the agent to be autonomous and reason about user goals, to plan about user resources and to search for agents of other users or applications who have the same goal and/or possess resources the user needs and doesn't have. This level involves the knowledge representation (user and application models) and reasoning mechanisms that make the agent autonomous and cognitive.

Each agent has a pool of goals of two types. **Intrinsic** are goals that are built-in the agent, or pre-programmed, such as "maximize utility", "save state periodically", "search for help-resource", "offer service", "gain more resources", or "refresh a specific resource", etc. **Extrinsic** are goals that are adopted from other agents, be it the user of the personal agent, other application agents, or personal agents, like "find me a helper or knowledge resource on topic X", "ask your user to be a helper on topic Y", "provide me with access to your resource Z".

Extrinsic goals are delegated explicitly to the agents, i.e. they don't infer the goals of other agents. The personal agents have to be told explicitly by the user on what topic help is needed. Inferring user goals by a personal agent is an attractive application of that classical diagnosis for user modelling, however, since we don't want to invest effort in developing a smart and narrow focussed mechanism for a certain domain, we let all goals be stated explicitly.

All goals of an agent are co-existing in a goal-space. The position of a goal in the goal space depends on the goal's importance. The importance of each goal is a number which is calculated /updated constantly by the reasoning mechanism of the agent. It depends on the inherent characteristics of the goal, like whether it concerns the existence of the agent, the relationship with the user (if it is a goal of a personal agent), on what type of goal it is. Extrinsic goals are usually less important, depending on the importance of the relationship with the agent from whom the goal is adopted.

The goal importance is a highly dynamic goal characteristic: with time flow or changes in the environment, some goals raise in importance and some drop in importance. The constant analysis of the importance of each goal is one of the central reasoning functions in an agent. This analysis is plan-based and reactive, i.e. it happens periodically, but it can be also triggered by opportunities, when certain changes in environment, or certain events occur. For example, if a help-request arrives (a new goal enters the goal-space), the agent has to evaluate the relative importance of the new goal with respect to the other goals, so that it can decide whether to pursue it or not. The evaluation of a goal's importance follows a set of rules that involve the parameters of a goal. We are experimenting with a simple PROLOG rule-based reasoner as the kernel of the goal-evaluation thread of the agent.

One of the important intrinsic goals of an agent is the goal to maximize utility. Utility is defined as a function of the number of goals achieved by the agent, their importance, and the amount and cost of resources expended in the pursuit of the goals. This definition of utility is different from the usual definition of utility for a rational agent, whose utility is measured usually by maximizing a certain resource (e.g. money). We can model the behaviour of such a classical rational agent by defining an important goal for the agent to acquire a certain resource (money). However, the above definition gives us a huge flexibility in defining the agent's major drive (motivation). It can be motivated, for example, to maximize the number of relationships, or to maximize some other resource, let's say general user knowledge. The major agent motivation can be set to maximize the number of user goals in which the agent helps achieving (an altruistic helpful agent). In this way, by modifying the goal-importance of different agent goals, we can achieve very different agent behaviour. The setting of the goal importance should be probably left to the user; in this way the "character" of his/her agent can be "tuned" to be altruistic, greedy, sociable, helpful, hostile etc.

The state of resources of each agent reflects the resources of its user or of the application represented by the agent. The agent manages the resources of the user/application just like a broker manages the money of an investor. The resources of users are represented in user models and the resources of software applications in application models. Both types of models are represented in a database and are created by special applications (diagnostic applications) whose goal is to enter or infer the state of resources from users' input or behaviour. The resources can be characterized according to several dimensions, for example, whether they are perishable or not, whether they are rechargeable or not, whether they are lendable or not. For example, time is perishable, not rechargeable and not lendable; knowledge is (hopefully) not perishable, re-chargeable and lendable (through teaching or peer help).

Knowledge resources are organized according to a taxonomy of knowledge topics / concepts / skills depending on the domain. Each agent manages the set of knowledge resources of the user or application it represents. We say that an application possesses a certain knowledge resource (say, on topic X), if the application provides electronic source of information about this topic (web-page, article in a FAQ, discussion thread on this topic, or if the application is supposed to teach the topic). A user possesses a certain knowledge resource if the user model contains a certain knowledge level on this topic.

In the context of I-Help the key to finding appropriate peer helpers is modelling the users' ability, willingness and readiness to help. The readiness is a function of the inter-agent negotiation, as will be explained in the next section. The ability is a model of the user's cognitive resources, represented as an overlay over the concept (topic, skill, or task) structure. The willingness is a model of the user's social behaviour, including a list of resources characterizing the user's eagerness, helpfulness, and ranking in the group (class), i.e. a user "social model".

The initialization and updating of the user model is done by a number of independent diagnostic applications. Each diagnostic application is represented by its own application agent who is activated at certain times or is called by the personal agent at a certain period of time to update the state of resources of the user. A diagnostic application updating the model of the user's knowledge can be a diagnostic component of an ITS in a specific domain. In our case we have two different diagnostic applications: one is an online self-evaluation questionnaire for students to evaluate their own knowledge on the topics of the course and a set of on-line quizzes on the concepts which students have to complete as they move through the course curriculum. The diagnostic application may request help from a knowledgeable human "diagnoser" or "cognologist" to carry out the diagnosis. In this way a human can enter the diagnosis loop.

The parameters in the social model are updated by several diagnostic applications. Diagnosis of the user's eagerness is based on monitoring the user's on-line activity (for example the votes on posting of this person in the discussion forum, the posted questions and answers in the discussion forum, observation of the threads visited during discussion forum browsing, or browsing in the web-based course materials). The user's class ranking is evaluated by taking into account the user's marks on assignments. The corresponding diagnostic application provides an interface for the teacher to input grades of the students from spreadsheet. Finally, the calculation of the helpfulness parameter of the user is based on the number of times the person has given help and the feedback on the quality of peer help received by the helpee. A diagnostic application monitors these peer-help activities and the evaluation questionnaires after each session and enters values for certain parameters in the user's social model.

Time is another important user resource. The amount of time available to the user can be assigned directly by the user or inferred using a stereotype value, for example, if the user is on-line, the user is available. Currency is a universal exchange resource for the society of agents. It may be given an real value by exchanging it for goods (marks, candies or lottery chances) in the real world.

The user model employed by the agent when acting on behalf of its user or application also contains information about the user's / application's relationships with other users / applications. This representation allows an agent to know about certain interpersonal relationships existing among users and certain dependencies existing among applications (for example, a diagnostic system and a teaching system for a certain skill). The relationships are represented along several dimensions: symmetry (dominant - peer), sign (positive - negative), type (between humans, between agents, between applications). These dimensions play a role when selecting a partner for negotiation and in the negotiation itself. It is very important to model qualitatively interpersonal relationships, since they play a major role in human motivation to collaborate.

An agent needs to have at least four inference mechanisms:

- monitoring the environment (i.e. the communication channel) for new coming goals, about opportunities (unexpected resource availability, for example, the student has time which is an opportunity for the agent to pursue the goal to increase the state of its currency resource by making the student give help) for achieving some persistent, long term goals that are "sleeping" at the moment; to estimate the goal importance periodically and opportunistically;
- calculating the utility function and to decide how to maximize it by selecting a set of active goals to pursue at every moment;
- retrieving plans for goal achievement (eventually a plan generator may be integrated);
- coordinating the execution of the plan by deploying the methods (Java threads) and resources required by the goal-method (task) definitions.

An open question is whether all of these inference mechanisms have to be "on-board" the agent. Currently, since the mechanisms are fairly simple, they are incorporated in every agent. However, since they are the same for all agents, the virtual tasks running on the machine can be reused by other agents.

2.3. Communicative agent

In pursuing their goals, the agents require resources that they may or may not have. In the latter case, the agents can try to get access to resources of another agent, if this agent is willing to lend them these resources. For example, an agent with the goal to find help for its user on topic X will try to find agents of another user who knows about this topic, or of application that possess on-line resources on this topic. Agents can also delegate the execution of a goal to another agent who is better equipped with resources and methods for achieving the goal, for example, it can contact the agent of an application who teaches this topic. In this case the second agent may adopt the goal of the first agent, thus offering a service to the first agent. That means that agents have to offer resources, services (contracted goal adoption) or relationships (e.g. increase in the relationship importance or change the relationship sign or type) in exchange for the resources / service they receive from other agents. The agents engage in negotiation until they agree about the price (in terms of specific resource, service or relationship change. A negotiation protocol between agents has been introduced including a set of speech primitives (KQML performatives). Negotiation is a way for the agents to engage in collaboration [7]. There hasn't been as much research on negotiation among agents in multi-agent societies of deliberative and planning agents as in the field of sub-cognitive agents [2]. Most of the studies of simulated collaboration involve agents that are purely reactive [5]; [8].

2.4. Economic Model of the agent society

The location of knowledgeable helpers and appropriate on-line resources or applications could have been achieved also without a multi-agent architecture. In fact, the first version of I-Help was a centralized component. However, the evaluation of the system showed that it is hard to motivate good students to serve as helpers for a longer time after the initial enthusiasm about the new system has gone. In every learner community there are students who ask for help often and students who nearly never ask for help. These latter are exactly the students who would make good peer helpers. It is obvious that there should be some reward for these students, in order to encourage them to participate -- to "pay them" with something that they consider useful. The need for some sort of economic model is evident. However, this raises many questions: Should the economy based on currency or on barter (exchange of goods) as proposed in [1]? If based on currency exchange, what should be the real world equivalent of the fictitious currency: marks or goods? Should the economy be based on the zero-sum assumption (when one gains, the other one loses) or on accumulating some resource, like for example the global knowledge resource of all participants in the system?

The role of the economic model is to ensure a co-ordination mechanism in the agent society and a source of motivation (reward) for the participating agents to offer their resources to other agents who may need them and thus cooperate with each other. Without the notion of "currency" as a resource which agents can accumulate and exchange, and an intrinsic goal for the agents to maximize the amount of their currency, it will be hard to create a utility function that will encourage agents having excess resources to offer them to other agents (i.e. good students to spend their time in helping weaker students). By variations of the price with respect to quality of resource (helper ranking) the agent economy will regulate the number of requests to the best helpers and will provide motivation for them to participate in the system. By introducing currency, without economic control mechanisms (e.g. taxation) we can expect a chaotic emergent behaviour from the system of negotiating agents. We need to ensure stable prices, trust in the currency, to ensure that the society will not become polarized with few very "rich" agents (what the agents of good students are likely to become) and a lot of very "poor" agents of students who need help but can't afford to buy it.

2.5. Control

The purpose of the control level is to ensure fair chances for all agents to pursue their goals, and to protect the society from criminal, misbehaving agents. It involves diagnosis of potential troublemakers and punishing them by either isolating them (damaging their reputation, so that other agents avoid having deals with them), or taking away their computational resources (putting them in jail). Measures of soft control involve the ability of agents to gossip, i.e. the agents themselves would communicate with each other regarding how their help transactions went, and introducing a "Review agent" that would collate and organize preference data based on global agent experiences. Measures taken under a hard social control scheme would include the banning of unethical agents, monetary deductions, reprimands through e-mail, etc. Examples of hard social control measures are introducing a "Refund Agent" which monitors transactions, and if some sort of obvious fraud was detected, would revoke the transaction and possibly impose harsher measures on the agent who did not live up to its contractual obligations. Another example of a hard control measure would be to introduce an "Immigration/Police Agent"**.** This agent would monitor the environment, looking for agents who do not have permission to perform transaction or participate in other ways.

3. Current state of the project

The multi-agent approach allows our system to be developed simultaneously bottom-up and top-down. Currently we have implemented the basic agent infrastructure, and we are experimenting with agents with a very limited set of goals and resources (i.e. our user models are fairly simple). All resources (user models) are kept in a database and the personal agents only have access to the user models of their own users. The database is updated by several independent diagnostic applications. One of them (diagnostic application denoted with "Q" in Figure 1) provides a quiz on the concepts involved in the course. Another one ("M") monitors the user's activities (browsing, postings, voting), checks time-stamps and updates the user's eagerness. A third diagnostic application ("T") allows a teacher to enter marks on assignments, which are used to calculate the ranking of the student in the class (a part of the user model). A fourth diagnostic application ("S") allows the user to fill a self-evaluation form to initialize the knowledge component of the model. The reasoning of the agents is currently very simple; it matches resources to goals in a 1:1 fashion, and decides which goals to pursue by trying to maximize its utility function. Certain parameters of the utility function can be defined by the user, thus defining the "character" of the personal agent (greedy, altruistic, friendly, generous etc.). Currently the agents have no planning capabilities and no knowledge about goal-decompositions. The agents' negotiation capabilities are also limited. The only "reasoning" happening at this level involves determining the price for a resource / service based on the state of resources of the agent, the ranking of the agent in the list of potential helpers and the relationship with the user asking for help. Both of these are implemented as functions of a number of parameters.

At the same time we are working on the upper levels of the multi-agent architecture, in a "top-down" fashion, by using a multi-agent based simulation tool, SWARM [6] to predict the global behavior of the system when multiple users and applications are negotiating and trading resources and services. In this simulation, the real users and applications are replaced with "random" functions producing certain input and the simulation concentrates only on the multi-agent society and the economic activities in it. We want to predict what would be the behaviour of the system depending on different distributions of agents with utility functions of various types. Will the economy function better with more "greedy" or with more "friendly" agents? Under what circumstances it is likely that "crooks" appear (e.g. cliques of agents who exchange fake help among each other in order to maximize their currency resources)? How can we discourage such behaviour with economic measures, and if not possible, what types of control / enforcement measures are necessary? Based on SWARM, different ways of control will be simulated (soft - based on "reputation" of agents and hard - based on forceful measures like taking away resources from agents). It would be interesting to see what influence different simulated negotiation strategies have on the flow of currency in the society of agents and to correlate this with the gained knowledge in the system. Two student projects [4] and [11] are underway to investigate these issues.

4. Discussion

I-Help is an user-adaptive distributed system deployed to support a collaborative community of learners to locate help resources (electronic resources and human peer helpers) in a University setting within a large course of introductory computer science. Adaptivity of the system is achieved through multi-agent architecture with autonomous, cognitive, and collaborative agents.
Motivation is an important issue in learning environment, and especially so in collaborative and peer help environments. By introducing an economic model we address this issue, hoping that it will increase motivation for participation and collaboration among the users / learners. A multi-agent architecture provides a natural testbed for implementing different kinds of economic models and experimenting with them to test under which conditions such economic models will work out to be beneficial for the learning of all participants in the system.

An important issue in systems for peer help is to protect good helpers from being overused. The multi-agent approach allows regulating the offer and demand naturally by varying the price for help. The personal agent of the helper can decide, depending on the priorities of the user, what the price would be. This further depends on the load on the helper; the urgency of the request for the helpee; the assumed quality of help provided (based on the helper's ranking in the list of helpers for the topic). Of course the user (helper) makes the final decision.

We believe that it is very important to take into account the interpersonal relationships among users when creating a peer help environment and we suspect that this will turn out to be even more important in a collaboration environment. The interpersonal relationships (friends, "favours owed" or "Bozo list") play a role in decisions to help or to refuse. In our multi-agent I-Help system, the personal relationships of every user are represented as a part of his/her user model and managed as a specific type of user resource by the personal agent. They are taken into account in calculating the price of service or resources that the personal agent offers to other agents, i.e. there is a specific discount for friends and an extra-charge for "Bozos". People who have been helped successfully (positive evaluation by the helpee) can be added to the helper's list of relationships as "friends" or "favours owed". At first glance it may seem that this would prevent people from collaborating, by for example, making people help only their friends and not help people they don't know (which makes the whole idea of I-Help senseless). However, the agents have other drives, such as maximizing their profit, (which makes them help unknown people who they can charge more), as well as maximizing the number of their relationships, which will pay back later since the agents of "friends" will charge less in case the user asks them for help). This makes modelling relationships a powerful motivation catalyst for cooperation and collaboration. To our best knowledge this is a new idea in user modelling which we believe will have impact on the design of collaboration environments.

The multi-agent approach which we adopted in the design of a peer-help and collaborative learning environment I-Help offers software engineering advantages, like easy extendibility and inter-operability. We can add new on-line materials, new applications (diagnostic systems, teaching systems, discussion forums, collaboration environments), and add more users on line. Such a system is less vulnerable when certain applications don't work, since it is essentially distributed: every participant can be taken away without causing the overall system to crash.

Another software advantage is that the overall system works independently of the intelligence of each individual agent. We start with fairly simple agents, with simple user models (limited resources) and limited reasoning about goals and ways to achieve them, with very limited negotiation strategies and gradually extend the agents capabilities by providing new applications offering "reasoning" services, "negotiation" services etc. Gradually, with the development of more sophisticated reasoning and negotiation mechanisms tuned to fit with functions controlling the simulation so that the system behaves in a desired way we hope to achieve a powerful distributed peer-help and collaboration learning environment.

Acknowledgement: We would like to acknowledge the TeleLearning NCE for financial support and to thank members of the ARIES Lab "Agents" group for their input.

References*:*

1. Boyd G. (1997) Providing Real Learning with Virtual Currency, Proceedings of the International Conference on Distance Education, Penn State Univ., June 1997.
2. Castelfranchi C. and Conte, R. 1992. Emergent functionality among intelligent systems: Cooperation within and without minds. *AI & Society*, 6, 78-93.
3. Greer, J., McCalla, G., Cook, J., Collins, J., Kumar, V., Bishop, A. and Vassileva, J. (1998) The Intelligent HelpDesk: Supporting Peer Help in a University Course, Proceedings ITS'98, San Antonio, Texas.
4. Kostuik, K. (to appear) Simulating a decentralized economy in a multi-agent learning environment, CMPT400 Honours Project, Dept. of Computer Science, University of Saskatchewan.
5. Mataric, M. (1992) Designing Emergent Behaviors: From Local Interactions to Collective Intelligence. In *Simulation of Adaptive Behavior 2*. MIT Press. Cambridge
6. Minar, M., Burkhart, R., Langton, C., Askenazy, M. 1996, The Swarm Simulation System: A Toolkit for Building Multi-agent Simulations. Santa Fe Institute.
7. Mudgal, Ch, (to appear). Negotiation and conflict resolution in multi-agent systems: a game theoretic approach, MSc Thesis Proposal. Dept. of Computer Science, University of Saskatchewan.
8. Steels, L. (1990) Cooperation between distributed agents through self-organization. In Y. Demazeau and J.P. Mueller (eds.) *Decentralized AI* North-Holland, Elsevier
9. Vassileva J. (1998) Goal-Based Autonomous Social Agents Supporting Adaptation and Teaching in a Distributed Environment, Proceedings of ITS'98, San Antonio, Texas..
10. Vassileva J., Deters R., Greer J., McCalla G., Kumar V., Mudgal C. (1998) A Multi-Agent Architecture for Peer-Help in a University Course, Proc. Workshop on Pedagogical Agents at ITS'98, San Antonio, Texas, 64-68.
11. Winter, M. (to appear) Simulated Social Control in a multi-agent based learning environment, CMPT405 Project, Dept. of Computer Science, University of Saskatchewan.