

ConferenceXP-Powered I-MINDS: A Multiagent System for Intelligently Supporting Online Collaboration

Adam Eck, L.D. Miller, Leen-Kiat Soh, and

Hong Jiang

Department Of Computer Science and Engineering
University of Nebraska - Lincoln
256 Avery Hall, Lincoln, NE 68588-0115 USA
E-mail: {aeck, lksoh, lmille, knobel, jiang}@cse.unl.edu

Tim Chou

Microsoft Research
Microsoft Corporation
One Microsoft Way, Redmond, WA 98052-6399 USA
E-mail: timchou@microsoft.com

Abstract

In this paper, we describe a multiagent system designed for intelligently supporting online human collaboration, built on top of the ConferenceXP platform developed by Microsoft Research. Many current collaborative systems are passive in nature and do not provide active, intelligent support to users. A multiagent system can be used to track user behavior, perform automated tasks for humans, find optimal collaborative groups, and create and present helpful processed information based on data mining without detracting from the rest of the collaborative experience. Our ConferenceXP-powered I-MINDS application currently offers five different components for enhancing collaboration and supporting moderator decision making by giving each user a personal agent that works with other agents to further support the entire system. These capabilities include two modes for group-based discussions, one for question/answer pairs between users and moderators, a search engine for retrieving tracked data, and a centralized classroom/team management system for quickly accessing user performance. CXP+I-MINDS has been successfully deployed to support an interactive business course where its intelligent activities assisted the professor in teaching, and we are working on delivering it to support a wireless classroom.

Introduction

One of the main enabling technologies for geographical-independent conferencing is online collaborative software. This type of application provides users with many different modes for discussion and collaboration: text-based chatting, audio/video streams, shared web browsing, shared whiteboards, etc. These offerings greatly benefit users because the software can contain all of their conferencing needs in only one package. Unfortunately, most of the current online collaboration software lacks one key feature: active and intelligent support of users. While providing different means for communication is very useful, computers have a vast wealth of computational power at their disposal. As computing power becomes cheaper and faster, more and more of this power could be used to drive artificial intelligence (AI) to monitor participants' discussions, learn from their work, and actively provide support for improving users' productivity, learning, and collaboration. Passive systems, like most that are used today, do not offer these features.

Some efforts to provide intelligent support to online collaboration have been made and many have centered around the use of multiagent systems, such as I-Help (Bull et. al., 2001) and DALIS (Mühlenbrock, Tewissen, and Hoppe, 1998). Using agent intelligence, these systems have the ability to model users and find potential aid when users need help. Other multiagent systems have been designed to support individual users, including CALO (Berry et. al., 2006) and Electric Elves (Chalupsky et. al., 2001), where agents work as personal assistants to take care of mundane tasks, allowing the users to focus on higher level activities, such as collaborating with other users.

However, these systems often do not take full advantage of the intelligence available to a multiagent system. Because agents are intelligent units actively learning about their own individual environments, each agent has a distinct perspective. Just as humans collaborate to draw on the strengths of each individual's history and abilities, a good multiagent system can rely on the same behavior from its agents through communications and negotiations. If one agent is struggling to consistently model its user correctly, it can receive help from other agents in identifying its problems. If an agent comes to a difficult situation it has never faced before, it can ask the other agents for guidance. However, since agents are also autonomous, they do not need to fully rely on the other agents and can make local decisions based on local knowledge to actively support their users. This might result in actions such as automatically presenting relevant stored data, providing intelligent tutoring, and matching their assigned user with potential collaborators based on the strengths and compatibility of all participants.

In order to build upon the work of previous multiagent systems research, we have developed the Intelligent Multiagent Infrastructure for Distributed Systems in Education (I-MINDS), a framework to actively support online collaboration in both classroom and business settings. I-MINDS provides each user her own personal agent for automated support which tracks and profiles user behavior, finds useful collaborative partners, and can discover and deliver useful content to assist current group work. All of the agents within I-MINDS also have the ability to assist one another in their duties, with available actions such as sharing user modeling techniques and collectively creating the best collaboration groups. Recent I-MINDS publica-

tions describe our latest experiments with the framework as implemented in the Java programming language (Khandaker and Soh, 2006), our VALCAM algorithm for coalition formation (Soh et. al, 2006a), and also some results as to the effectiveness of our system (Soh et. al, 2006b).

A persistent weakness with our Java version of I-MINDS is the lack of support for multiple modes of online collaboration (e.g., audio and video streaming), automated content archival, and a reliable network infrastructure. As a result, we have recently looked to Microsoft Research’s ConferenceXP (CXP) platform to help us strengthen the underlying collaborative infrastructure for I-MINDS while we focus on its intelligent components. ConferenceXP is an open collaborative framework developed by Microsoft Research which offers reliable networking and numerous extendable multimedia capabilities. The new ConferenceXP-powered I-MINDS application, called CXP+I-MINDS, which has been successfully deployed to support classroom collaboration, is the focus of this paper.

In the following, we will first discuss the I-MINDS framework and its Java prototype, which includes some preliminary results through actual classroom deployments. Then we will describe the CXP+I-MINDS system. This section includes a description of the CXP platform and the intelligent components of I-MINDS. We will then present briefly the deployment results of CXP+I-MINDS before concluding.

I-MINDS Framework and Prototype

Framework

The Intelligent Multiagent Infrastructure for Distributed System in Education (I-MINDS) is a framework to support both individual users and groups in collaborative environments (Liu et. al., 2003; Soh et. al., 2006a). While this system was originally designed for computer supported collaborative learning (CSCL), it is also applicable to any collaborative scenario, such as group projects in industry and managerial meetings in business.

I-MINDS is a heterogeneous multiagent system with three main types of agents: user agents, moderator agents, and group agents. Every user is assigned their own individual user agent who performs automated tasks for the user and also acts to provide support in the best interests of its user.

A typical I-MINDS topology is characterized in Figure 1, where different users and a moderator interact between several modes (e.g., chatting, drawing, audio/video). These modes allow users to collaborate with one another and can either be shared or local. Shared modes are visually identical and share the same content, like a whiteboard upon which each user can make additions. Local modes, like text-based chatting, allow users to send data back and forth, but are not necessarily in the same state for each user (e.g., one user might have three topics she is discussing in a chat program, while another only has two). The user

agents gather data from these modes based on which modes their user is currently engaged with, the content of information exchanged between users, and the behaviors exhibited by each participant. For every group formed between humans, a group agent is also assigned which gathers data from the modes as well as the individual user agents in order to make decisions and perform actions intended to support their assigned group. The moderator agent tracks data from the moderator’s actions and can also receive information from the individual and group agents. This allows the moderator agent to act as a leader for all of the agents, mirroring the leadership role of the moderator human. However, each user and group agent retains its own autonomy, giving a federated nature to the agent hierarchy.

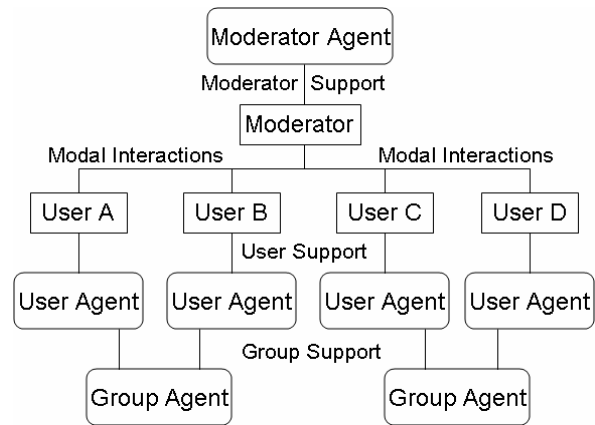


Figure 1: I-MINDS System Topology

To actively support both users and groups, the agents must have some of the same core functionalities, but they differ in how they process information and what types of support they provide. The I-MINDS agent infrastructure is depicted in Figure 2.

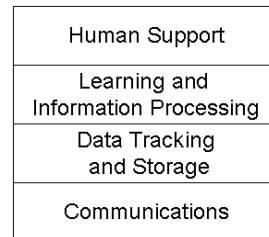


Figure 2: I-MINDS Agent Infrastructure

One of the most fundamental features needed by the agents is the ability to track, store, and retrieve data, as well as the ability to process this data into information that can be used to determine future supportive actions. For a user agent, this might mean keeping track of all the keywords entered by their user and also tracking certain behaviors like response time and number of questions asked of the moderator. For group agents, it is important to keep track of the relationships between the group members, including instances of agreement or disagreement. For moderator agents, this tracked data is especially useful for com-

computing statistics for each individual user to determine the strengths and weaknesses of the participants and make reasoned decisions about how to improve the overall state of the entire system.

Another key feature that enables agents to provide the best possible support is the ability to actively learn about and model their environments and users. Any environment inhabited by humans is noisy and dynamic in terms of both content and behavior. Set heuristics are important in determining appropriate agent behavior, but environmental change necessitates that agents be reactive in order to respond to changing circumstances and adapt to help their users remain successful under the new conditions. In addition, agents must also be proactive and have the ability to make predictions in order to successfully affect their environments and guide the users to their goals. For user agents, this means that the agents must be able to predict how their user's performance will be affected by a change in her working group, as well as be able to identify when a user is struggling so they can find an appropriate source of assistance. Group agents must be able to learn what types of communication are best utilized by their group members and also be able to identify when the group is failing to meet its goals. Moderator agents should be able to identify the projects, topics, and users that are having the most difficulties and alert the moderator of these problems, as well as take autonomous action to resolve the situation themselves, where appropriate.

Finally, just as each user can receive guidance from her peers, no agent is alone and must be able to assist its brethren. For example, when an agent is having difficulties modeling a specific characteristic of a user, possibly caused by incomplete data or inconsistent behavior, it should be able to ask other agents for help. The other agents can autonomously choose whether or not to assist. If they choose to do so, they will respond based on their own perspectives and experiences, which may be different from the requesting agent's. This feature is one of the most important strengths of multiagent systems: even though each agent's experience is limited to its own environment, the agents can draw from the differences in each other's experiences to provide support based on information *not otherwise available*. Also, because I-MINDS is a heterogeneous system of agents, each type of agent possesses its own set of abilities, so by working with a different type of agent, *each can gain from and use functionality that it does not possess*.

Java Prototype and Initial Results

To test our I-MINDS framework, we have partially implemented it in the Java programming language as a research prototype for a computer-supported collaborative learning (CSCL) system. Currently, Java I-MINDS consists of three main applications: manager, student interface, and teacher interface. The manager works as a centralized administrator, taking care of classroom registration, user login, and other server administration duties. The student

interface is used by all students and displays any presentations, audio, or video streams used by the instructor. It also has a pane for sending and receiving messages from other students or for asking questions of the teacher. The student interface also contains a shared whiteboard to allow students to collaborate with freeform drawing, a flow-chart editor, and an equation editor (in progress). Inside the interface resides a user agent which handles all information flow from the instructor to the students, profiles the students based on their messages and whiteboard activities, and automatically finds other students for their student to work with.

The teacher interface of Java I-MINDS allows an instructor to send a set of presentation slides to students, along with audio and video streams. It also contains a pane that displays questions from students, allowing instructors to answer or discard questions, along with finding similar questions in the database. The interface also provides a form for displaying information about each student, including key statistics such as motivation level and responsiveness. In addition, the teacher interface contains a moderator agent which automatically evaluates all questions received from students, scores them based on quality and relevance to the current topics discussed in class, and sorts them for the instructor, helping the teacher know which questions to focus on. The agent also profiles each student based on these questions to generate displayable statistics. Finally, the agent also handles all communication streams and insures that students receive the proper ones.

The first Java prototype was deployed in an actual classroom and was well received by both students and instructor (Liu et. al., 2003). We have continued work on the prototype since then, focusing primarily on testing theories and functionalities for improving the multiagent intelligence found in the system. Recently, we developed an auction-based, coalition formation algorithm called VALCAM which allows the agents to form new human collaboration teams through negotiations. Another proto-type of I-MINDS was deployed using VALCAM and we found that over time, students rated their peers in groups formed by the agents higher, as compared to a control group which did not use I-MINDS (Soh, et al. 2006b). We also found that students' achievements in post-test activities were consistently at least as good as, or in many instances even better, than those from the control section.

However, as we worked on the Java prototype of I-MINDS, we discovered several key problems that were difficult to address. For example, we had to build all the multimedia functionality and content archival ourselves, as well as develop reliable network support for connecting multiple geographical locations, some of which were multicast enabled while others were not. Rather than spending an extensive amount of time on these issues, we looked for possible existing solutions. We came upon the ConferenceXP platform, which fit perfectly with what we were looking for: a powerful networking base for distance collaboration along with existing multimedia functionality and automated content archiving.

CXP + I-MINDS Application

ConferenceXP

The ConferenceXP (CXP) framework, developed by Microsoft Research, is a conferencing platform designed for high-quality multimedia transmissions over high-bandwidth connections, such as those traditionally found in academic and business settings. It is released under Microsoft's Shared Source license, promoting open source development for education and research. (<http://research.microsoft.com/conferencexp/>) One of its primary goals is supporting distance education and other educational collaborative activities (Beavers et. al., 2004).

The CXP framework contains three main layers. At the base of the framework is the networking layer, which supports the Real-Time Transport Protocol (RTP) for sending and receiving streams of multimedia data over multicast User Datagram Protocol (UDP) transmissions. The second layer is the conferencing application layer, which does the bulk of the work in CXP. It keeps track of all the different venues a participant can join, all of the participants in a currently selected venue, and all of the capabilities currently running. It also serves as a translator between the networking layer and the individual capabilities.

The third and final layer of the ConferenceXP framework is the capability layer. One of the most interesting features of CXP is the fact that it is basically just a plug-in architecture, with each individual capability being a plug-in. ConferenceXP already includes several capabilities such as presentation viewers, a shared web browser, a capability for sharing a display between multiple participants, and a basic chat application. This architecture is development-friendly because it abstracts away all of the underlying details to the conference and networking layers, allowing the developer to focus solely on the application they wish to create without having to worry about how the lower-level mechanics work.

In addition to the default capabilities and framework underpinnings, ConferenceXP also has three services to facilitate conferencing: VenueService, which allows users to create virtual rooms (venues) to meet in; ArchiveService, which automatically stores all raw capability data sent by participants for future playback; and ReflectorService, which bridges the gap between multicast and unicast connectivity, allowing users with unicast networking connections to take advantage of CXP's capabilities.

ConferenceXP-Powered I-MINDS

The ConferenceXP and I-MINDS marriage has proven to be a very fruitful one, especially because there are certain features that CXP lacks which I-MINDS brings to the table, such as extensive data tracking, automated user support, and multiagent intelligence. The CXP+I-MINDS application provides several different collaboration enhancing components not found in the default CXP pack-

age, including **BuddyChat**, **StructuredChat**, **Question/Answering**, **DatabaseSearch**, and **VirtualClassroom**. A possible usage scenario is given in Figure 3.

BuddyChat and StructuredChat are two modes which allow users to collaborate in standard freeform chat, similar to many instant messaging (IM) clients like MSN Messenger and AIM. All dialogs are organized by topic, allowing the user to quickly see how each conversation they belong to is progressing. Also, all new messages and their containing topics are flagged to help users stay on top of their conversations. The only difference between these two modes is their groups: in BuddyChat, a user is allowed to determine their own working partners (called "buddies"), while in StructuredChat, the groups are set by the moderator. This latter type is very useful for structured activities like those found in classrooms or for group projects in industry. For the user version of these two components, dialogs are only displayed if they were started by a partner, but for the moderator version, every dialog is displayed so the moderator can always see everything that is going on.

The Question/Answering mode allows users to ask questions to and receive answers specifically from the moderator. We have found that this is especially useful in educational settings, allowing students to ask questions of instructors while they are lecturing. All unanswered questions are queued up, allowing the moderator to address them when they have time. The moderator then has the option to answer the question to the asking user, broadcast the answer to everyone (useful for answers to a question that everyone might have or for illustrating a specific point), or discard the question. Also, if the moderator has something specific they would like to say to the users, they have the option of broadcasting messages to any combination of users without being tied to any question.

All messages sent between BuddyChat, StructuredChat, and Question/Answering are automatically stored with their content and relevant metadata (e.g., timestamp, sender, type) in a relational database. To facilitate retrieving information from this database, we have included DatabaseSearch within CXP+I-MINDS, a search engine which allows each individual user to search through tracked messages to find previous content. This capability supports searching by any combination of tracked metadata and content to allow for easy information retrieval.

The final capability, available only to moderators, is VirtualClassroom. Obviously, this capability is centered around educational use, but like the rest of I-MINDS, it is equally applicable to any collaboration scenario. Using all the data we track from the BuddyChat, StructuredChat, and Question/Answering capabilities, the moderator agent can compute several statistics that describe the users and their behaviors, like responsiveness, dialog improvement, average question quality, etc. The values of these statistics are then displayed to the moderator in graphical form, along with information about current dialogs, to allow them to quickly assess the current status of their classroom

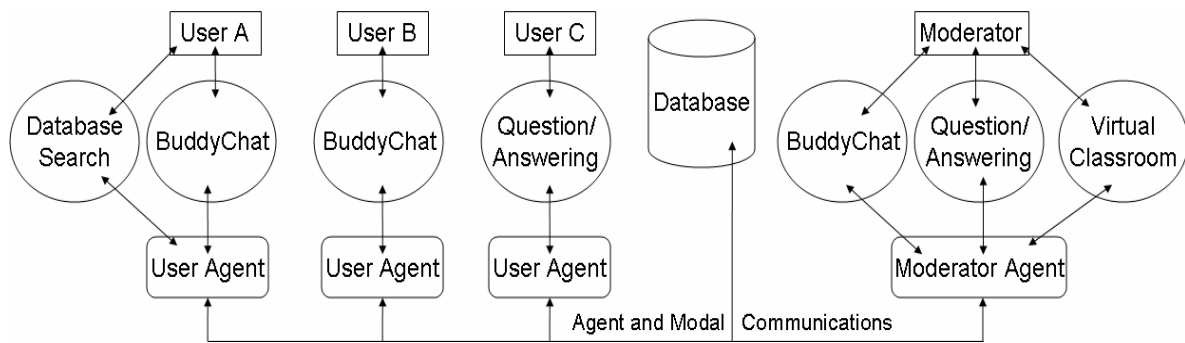


Figure 3: User Scenario for CXP+I-MINDS

Intelligent Support

Active, intelligent support in collaborative software provides many important opportunities that would otherwise be absent or difficult for humans to optimally perform. For example, an intelligent system can find good partners when a user needs assistance, based on tracked data and models generated for each user (Bull et. al., 2001). In an educational environment, an intelligent system can choose assignments and problems for specific individuals based on their strengths and weaknesses in order to promote learning and achievement (Mühlenbrock et. al., 1998). Learned models can also be used to select roles in a group or participation in meetings (Chalupsky et. al, 2001). Moderators in charge of many users cannot always monitor everyone's performance, but an intelligent system can track everyone and their activities and alert the moderator when problems occur (Soh et. al., 2006b).

There are several different ways we use artificial intelligence to support users in CXP+I-MINDS. In the BuddyChat and StructuredChat modes, the user agent monitors each topic to see how much the user is participating in the discussion. If the agent detects that too many messages are unread in a specific topic, it automatically alerts the user and asks them if they would like to go to the topic. If the user responds affirmatively, the agent switches the topic for the user, but if the user responds negatively, the agent uses reinforcement learning to learn that the user doesn't want to be bothered about unread messages. If the user's contribution level is too low compared to the other participants of the dialog, the agent will once again alert the user and give them an option of switching to the relevant discussion.

Also in the BuddyChat mode, if the user's agent determines that another participant could make a good buddy based on their contributions to topics shared by the local user and the potential buddy, the agent will prompt their user, recommending the other person as a good buddy. If the user decides that they would like to take the agent's recommendation, the agent automatically sends a buddy request to the other person to become buddies. If the local user doesn't want to take the agent's advice and chooses not to add the potential buddy, the agent once again uses reinforcement learning to learn that their user doesn't want to add other participants based on agent advice. Also, the

user agents work together to make sure the buddy relationships are maintained at all times, which is especially useful in a lossy wireless environment or when users start CXP+I-MINDS.

For the Question/Answering mode, the moderator agent reads every message and learns keyword/weight pairs based on the frequency of use of the keywords. Using these weights combined with ones optionally assigned by the moderator and an extra weighting factor based on how many questions each individual user has asked, the agent then scores every question for relevance to current topics. Next, the questions are sorted based on score and displayed with higher scored questions at the top. This allows the agent to recommend questions to the moderator to answer first, which helps busy moderators know what to focus on, hopefully allowing them to spend more time on the most relevant questions.

In VirtualClassroom, all of the statistics are grouped into several different categories, and the moderator can set expected thresholds for each statistic. If any statistic dips below its expected value, the agent flags it as a weakness. The agent then creates a pie-chart for each user, consisting of segments for each category. If any statistic in a category is considered a weakness, the segment is colored red for easy identification, else the segment is colored blue. The agent then sorts the users based on the number of weaknesses they have and displays the pie-charts to the moderator. This color coded approach allows the moderator to quickly see how each user is doing and which ones are having the most problems. If the agent detects that more than half of the users are having difficulties with any category, they immediately notify the moderator of this situation, further helping their human know which problems to address.

Deployment and Results

In the summer of 2006, the ConferenceXP powered I-MINDS application was deployed to support an interactive business course taught at Bellevue University, the second largest online university in the U.S. The instructor required students to actively participate in every lecture, and found CXP+I-MINDS to be highly conducive to facilitating student participation. Many of the students in this class were nontraditional students, mostly consisting of working

adults taking night courses to complete a college degree. The students enjoyed using I-MINDS, and the instructor especially liked the keyword learning and message scoring functionality built into Question/Answering, which allowed him to track student performance and topic progression over time. The instructor stated that “[I-MINDS] has provided [him] with solutions for improving [his] classroom activity that were lacking in other options that [he has] explored.”

We are also working with the IT administrators at our own university to open multicasting over the wireless network so we can deploy our application in the wireless classroom environment offered by our combined honors program in computer science and business later this year. We ran an initial test in this environment, and found that CXP was able to support 30+ student participants through wireless connections. However, we also found two areas that need to be improved: (1) more efficient logging and tracking of data over the wireless network to reduce communication traffic, and (2) more reliable loss recovery through the use of the experimental Pragmatic General Multicast protocol (Network Working Group, 2001).

Conclusions and Future Work

In this paper, we have described an application designed and implemented to enhance the ConferenceXP collaborative framework developed by Microsoft Research using I-MINDS. Using a multiagent system with personalized support agents, we offer five different components for user collaboration and moderator support. The system has been successfully deployed in a classroom environment at Bellevue University, where it was well received by both instructor and students. We are currently working on implementing new functionality into the system based on the work done with our Java prototype and hope to deploy it in more classrooms in both Bellevue University and the J.D. Edwards Honors Program at the University of Nebraska.

In the future, we will work on improving the communications between the agents to allow the system to take further advantage of the strengths of a multiagent system. Also, we want to improve the user modeling available to each agent, which should allow for more reactive and proactive support in a dynamic collaboration environment, in addition to more personalized and targeted support for improving different problems common to students and other users of collaboration systems. Finally, we would like to work to further integrate I-MINDS to support all of ConferenceXP’s capabilities (e.g., classroom presenter), to track the use of those capabilities by students to increase the accuracy of our modeling, combining the efforts of the entire community of developers and users that CXP has to offer with a powerful multiagent infrastructure to offer a wider range of intelligent support for online collaboration.

Acknowledgements

We would like to acknowledge the people we have worked closely with at Microsoft Research, including Todd Needham and Jason Van Eaton. Also, those who have worked on Java I-MINDS: Nobel Khandaker, Xuli Liu, Xuesong Zhang, Jameela Al-Jaroodi, and Phanivas Vemuri. The project is sponsored partially by a Microsoft Research ConferenceXP grant and a Pepsi sponsored UCARE grant at the University of Nebraska. The authors would like to thank David Levy and Gary Christensen at Bellevue University for using I-MINDS over the past year.

References

- Beavers, J., Chou, T., Hinrichs, R., Moffatt, C., Pahud, M., Powers, L., and Van Eaton, J. 2004. The learning experience project: enabling collaborative learning with ConferenceXP, Microsoft Research *Technical Report* MSR-TR-2004-42.
- Berry, P., Peintner, B., Conley, K., Gervasio, M., Uribe, T., and Yorke-Smith, N. 2006. Deploying a personalized time management agent, in *Proc. AAMAS’2006*, Hakodate, Japan, May 8-12. 1564-1571
- Bull, S., Greer, J., McCalla, G., Kettel, L, and Bowes, J. 2001. User modeling in I-Help: what, why, when, and how, in *Proc. 8th Int. Conf. User Modeling*, Sonthofen, Germany, July 11-17, 117-126.
- Chalupsky, H, Gil, Y., Knoblock, C., Lerman, K., Oh, J., Pynadath, D., Russ, T., and Tambe, M. 2001. Electric elves: Applying agent technology to support human organizations, in *Proc IAAI’2001*, Seattle, WA, August 7-9.
- Khandaker, N. and Soh, L-K. 2006. Student learning and team formation in a structured CSCL environment. in *Proc. 14th Int. Conf. Computers in Education*. Beijing, China, Nov. 30-Dec. 4, 185-192.
- Liu, X, Zhang, X., Soh, L-K., Al-Jaroodi, J., and Jiang, H. 2003. I-MINDS: An application of multiagent system intelligence to on-line education, in *Proc. IEEE Int. Conf. Systems, Man, and Cybernetics*, Washington, D.C., October 5-8, 4864-4871
- Mühlenbrock, M., Tewissen, F., and Hoppe, H.U. 1998. A framework system for intelligent support in open distributed learning environments. *Int. J. Artificial Intelligence in Education*, 9, 256-274.
- Network Working Group. 2001. PGM reliable transport protocol specification. <http://www.ietf.org/rfc/rfc3208.txt>.
- Soh, L-K., Khandaker, N., Liu, X. and Jiang, H. 2006a. Computer-supported cooperative learning system with multiagent intelligence, in *Proc. AAMAS’2006*, Hakodate, Japan, May 8-12, 1556-1563.
- Soh, L-K., Khandaker, N., and Jiang., H. 2006b. Multi-agent coalition formation for computer-supported cooperative learning, in *Proc. IAAI’2006*, July 17-20, Boston, MA, 1844-1851.