From Mirroring to Guiding: A Review of State of the Art Technology for Supporting Collaborative Learning

Patrick Jermann[1], Amy Soller[2], and Martin Muehlenbrock[3]

[1]TECFA
University of Geneva
1205 Genève, Switzerland
Patrick.Jermann@tecfa.unige.ch

[2]Learning Research and Development Center and Intelligent Systems Program
University of Pittsburgh
Pittsburgh, PA 15260-5159 USA
soller@pitt.edu

[3]COLLIDE research group
University of Duisburg
47048 Duisburg, Germany
muehlenbrock@informatik.uni-duisburg.de

Abstract: We review systems that support the management of collaborative interaction, and propose a classification framework built on a simple model of coaching. Our framework distinguishes between mirroring systems, which display basic actions to collaborators, metacognitive tools, which represent the state of interaction via a set of key indicators, and coaching systems, which offer advice based on an interpretation of those indicators. The reviewed systems are further characterized by the type of interaction data they assimilate, the processes they use for deriving higher-level data representations, and the type of feedback they provide to users.

Keywords: CSCL systems, interaction management, mirroring, coaching collaboration

1 Introduction

Over the past decade, we have seen an explosion of network-based technologies that enable traditional and non-traditional distance learners alike to learn collaboratively. These environments enhance traditional distance learning curricula by giving students the opportunity to interact with other students and share ideas. But especially for domains in which teamwork is critical, do these environments measure up to traditional classroom group activity? Classrooms provide supportive environments where teams of students learn in the presence of an instructor who helps to manage and guide the collaboration, providing clear goals as to what is expected from the group process. It is too early to tell whether or not we will ever be able to offer the supportiveness of a traditional classroom, online; however, a few research projects have begun to explore the possibilities of enriching CSCL environments with tools to support collaborative interaction. In this paper, we attempt to develop a conceptual framework to describe the array of capabilities these tools offer. At the core of this framework lies a hypothetical model of coaching that mimics simple process control.

1.1 Four phases of the coaching process

Coaching collaborative interaction means supporting or managing the group members' metacognitive activities related to the interaction. For example, one might help students manage their interaction by assigning roles, detecting conflicts and misunderstandings, or proposing suitable tasks for each participant, given their level of expertise.

We have drawn upon work by Barros and Verdejo (2000) to develop a framework that describes the process of collaboration management. Barros and Verdejo (2000) distinguish between the performance level, in which actions are recorded, and the analysis level, which defines a set of attributes characterizing the interaction. Attributes are computed by analyzing the actions users take on the interface. An advisor module then evaluates the attributes' values and sends feedback to the learners. Finally, the effects of the advisor's interventions on the students are studied.

In our terms, collaboration management follows a simple homeostatic process that continuously compares the state of interaction with a target configuration. Actions are taken whenever a perturbation arises, in order to bring the system back to equilibrium. This model is a convenient way of classifying systems rather than a reflection of reality, as the definition of the desired state can itself change in the course of activity. This note of caution aside, collaboration management can be described as a repetitive cycle containing the following phases (see Figure 1):

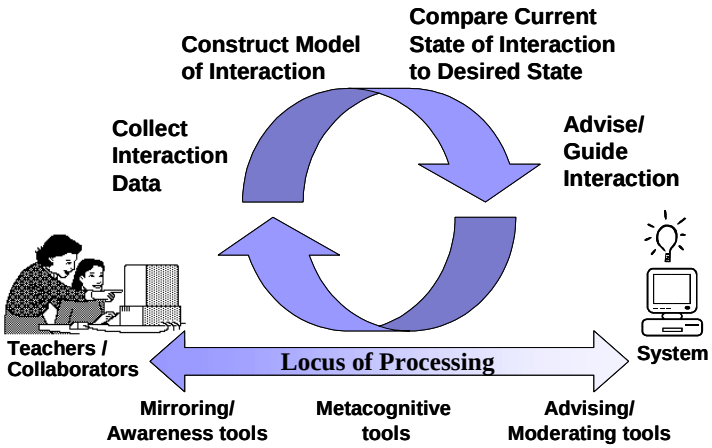

Figure 1. The collaboration management cycle, showing points at which the responsibility for analyzing and guiding the interaction might shift from the collaborators to the system.

- The data collection phase involves observing and recording the interaction. Typically, users' actions (e.g. 'user1 clicked on I agree', 'user1 left the room') are logged and stored for later processing.
- The next phase involves selecting one or more high-level variables, termed *indicators*, to represent the current state of interaction. An indicator's values may be obtained by instantiating a model of interaction with the data obtained in the first step. For example, an agreement indicator might be derived by comparing the problem-solving actions of two or more students, or a symmetry indicator might result from a comparison of participation indicators.
- The interaction can be "diagnosed" by comparing the current state of interaction to an ideal model of interaction. In this paper, we define an ideal model as a set of indicators describing desirable and undesirable interaction states. For instance, we might want learners to be verbose, agree frequently, and participate equally.
- Finally, if there are discrepancies between the current state of interaction (as described by the indicator values) and the desired state of interaction, some remedial actions might be proposed. Simple remedial actions (e.g. 'You have not participated enough') might result from analyzing a model containing only one indicator (e.g. word or action count), which can be directly computed from the data, whereas more complex remedial actions might require a more sophisticated computational model.

Who is managing this process? It might be a teacher, or the group members themselves who observe the interaction and propose roles or activities, or it might be a computer system that diagnoses the state of interaction and proposes remedial actions. In the next section, we describe systems that support these options.

1.2 From Mirroring to Guiding

Systems that support collaboration generally adopt one of two approaches. The first approach, which we do not cover here, structures the situation in which the collaboration takes place. Learning situations can be structured by requiring the students to use a set of structured software tools, structuring the group itself (i.e. selecting group members based on some criteria), or structuring the task (e.g. by a learning scenario). These factors may encourage group members to engage in certain types of interaction such as argumentation or peer tutoring via external means. The second approach involves structuring the collaboration itself through coaching or self regulation, as illustrated by the collaboration management cycle. As the collaboration progresses, the state of interaction is evaluated with respect to a desired state, and remedial actions may be proposed to reduce discrepancies between these states. Structuring and coaching are not exclusive approaches, as structuring interaction might take place during interaction as a remedial action.

We now distinguish between two approaches to guiding learning interaction. In the first case, the system gathers data about the students' interaction, and shows some visualization of this information to the user, possibly aside information about what an ideal interaction might look like. It is then up to the students or teacher to interpret the visualization and decide what actions (if any) to take. In the second case, the model of interaction and the system's assessment of the current state is hidden from the students. The system uses this information to make decisions about how to moderate the group.

Fundamentally, these two paradigms are the same, in that first data is collected, then a model of interaction is constructed and instantiated to represent the current state, and possibly the desired state, and finally, some decisions are made about how to proceed. The difference between these two approaches lies in the locus of processing (see Figure 1). Systems that collect interaction data and construct visualizations of this data place the locus of processing at the user level, whereas systems that advise process this information directly.

The benefits of coaching student interaction (via human or computer) are clear, given a correct diagnosis and appropriate remedial actions. But what can students learn when presented with visualizations of data, or indicators? Students who view and analyze indicator values may learn to understand and improve their own interaction, for example by relating specific indicator configurations to successful completion of a task. (e.g. - We performed well when we were all participating actively). Students might, however, lack the understanding to interpret the visualizations correctly, leading them to take unnecessary actions. Without the time and understanding to develop their own models of interaction, students may naturally rely on implicit social norms (status, equality) to manage the interaction. For example, a student might remain silent allowing his more knowledgeable partner to perform a difficult task, or partners may try to participate equally, thinking that equal participation leads to equal credit. Collaborative learners, guided by indicator displays, may need to follow a more introspective process to develop an understanding of their interaction than when they are guided by an advisor.

2 A Review of Systems for Supporting Collaborative Learning

In this section, we provide examples of three types of supportive collaborative learning systems, in the context of the collaboration management model. Systems that reflect actions, termed mirroring systems, collect raw data in log files, and display it to the collaborators. Systems that monitor the state of interaction, termed metacognitive tools, model the state of interaction and provide collaborators with visualizations that can be used to self-diagnose the interaction. These visualizations typically include a set of indicators that represent the state of the interaction, possibly alongside a set of desired values for those indicators. Finally, coaching or advising systems guide the collaborators by recommending actions students might take to improve their interaction. Figure 1 shows the stages of the collaboration management cycle that each of these three system types carry out. The next three subsections will review systems that fall into these categories.

2.1 Systems that Reflect Actions

The most basic level of support a system might offer involves making the students or teachers aware of the participants' actions. Actions taken on shared resources, or those that take place in private areas of a workspace may not be directly visible to the collaborators, yet they may significantly influence the collaboration. Raising awareness about such actions may help students maintain a representation of their teammates' activities.

Some systems in this category represent actions along a timeline. For example Plaisant, Rose, Rubloff, Salter, and Shneiderman (1999) describe a system in which students learn the basics of vacuum pump technology through a simulation. As the learner manipulates the controls of the simulation, a history of actions is displayed graphically beneath the target variable (e.g. pressure). It consists of stripes and boxes that represent the user's actions as well as the system's messages. The data displayed to the student does not undergo any processing or summarizing, but directly reflects the actions taken on the interface. These graphical records of actions can then be sent to a tutor or a peer, or replayed by the learner to examine his own performance. Although Plaisant and colleagues did not design the system to be used by two persons at the same time, the learning history might be used to mirror the collaborative situation by displaying the actions of the learners side-by-side, and offering a representation of concurrent actions, thus helping students coordinate their actions.

Other systems reflect actions but are not geared specifically toward learning, and hence will not be covered in much detail here. For example, one of the awareness tools (Gutwin et al., 1995) in the Groupkit system (Roseman and Greenberg, 1992) consists of a shared scrollbar to display the section of a text each participant is looking at, allowing students to locate their partner's focus of attention. There exists many groupware systems that provide users with information such as where other users are located (if the system uses a room-based paradigm), or what objects other users are viewing or manipulating. See NCSA Habanero, CuseeMe, Collaborative Virtual Workspace, and Microsoft NetMeeting for some examples.

2.2 Systems that Monitor the State of Interaction

Systems that monitor the state of interaction fall into two categories: those that aggregate the interaction data into a set of high-level indicators, and display them to the participants, and those that internally compare the current state of interaction to a model of ideal interaction, but do not reveal this information to the users. In the former case, the learners are expected to manage the interaction themselves, having been given the appropriate information to do so. In the latter case, this information is either intended to be used later by a coaching agent, or analyzed by researchers in an effort to understand and explain the interaction.

Our first group of systems models the state of interaction via a set of indicators that are displayed to the users. Jermann (work in progress) has developed a system that displays participation rates to the collaborators while they solve a traffic light tuning problem. One indicator shows the number of messages and another shows the number of problem-solving actions. Such tools might have a positive impact on a group's metacognitive activities by aiding in the construction and maintenance of a shared mental model of the interaction. A mental model may encourage students to discuss and regulate their interaction explicitly, leading to a better coordination of the joint effort to reach a solution. Taking these ideas one step further, we might imagine a system whose model of desired interaction is displayed to the students next to the actual state of interaction. The model might also change during the learning process, causing the target values of the indicators to be dynamically updated, encouraging the learners to improve in different ways.

In situations where more than two persons interact, social networks may be used to represent the exchange patterns among participants in a discussion (Nurmela, Lehtinen, and Palonen, 1999; Wortham, 1999). A social network typically consists of a network of nodes in which each node represents a participant. The thickness of an edge connecting any two nodes represents the amount of discussion between two participants. Simoff (1999) proposes an interesting way to merge the graphical representation of participation rates, and the potential for learning. His system visualizes discussion threads with nested boxes. The thickness of the boxes' edges represents the number of messages produced in response to the opening message for a particular thread. In an educational environment, thicker boxes might mean deeper conversations, hence deeper understanding.

Some indicators are implicitly contained in the tools used by the students. In Sharlock II (Ogata, Matsuura, and Yano, 2000), a special tool called a Knowledge Awareness Map graphically shows who is discussing or manipulating the knowledge pieces users have posted. In this case, the distance between users and knowledge elements on the map indicates the degree to which users have similar knowledge.

The systems we have discussed so far refrain from interpreting the content of the interaction, and instead focus on quantitative aspects of the interaction. Analyzing participation rates involves counting words or messages, whereas indicators such as acknowledgement rate and delay (how often users respond to incoming messages, and how long this takes) or role distribution (what kind of actions are taken by whom) require more sophisticated computation (e.g. advanced modeling or natural language processing techniques). Studying more complex variables often involves analyzing the semantic aspects of interaction and the patterns of student actions. A structured interface may facilitate the interpretation of actions by the system. For example, users may be required to select a dialog act (e.g. propose, encourage, question) when they send messages to each other. MArCo (Tedesco and Self, 2000) is a dialog-oriented system for the detection of meta-cognitive conflicts. The system adopts a dialog game approach with a limited set of possible dialog moves. User utterances must be formulated in a formal language that enables the conversation to be mapped onto a belief-based model (BDI). The analysis mechanism detects disagreements and conflicts between users' beliefs and intentions.

Conversational acts may be considered in isolation, or in the temporal context of other acts. Muehlenbrock and Hoppe (1999) were one of the first to propose actions in shared workspaces as a basis for a qualitative analysis. Unlike dialog tags, actions on external representations are not only interrelated on a temporal dimension, but also on a structural dimension, i.e. concerning their context of application. This approach has been termed *action-based collaboration analysis* (Muehlenbrock, 2000) and is implemented as a plug-in component in the generic framework system CARDBOARD/CARDDALIS, which is for collaboration by means of shared workspaces with structured external representations (visual languages) and for the provision of intelligent support. Action-based collaboration analysis derives higher-level descriptions of group activities, including conflicts and coordination, based on a plan recognition approach.

One reason for not displaying a visualization of the model of interaction to the students or the teacher is that the evaluation of complex variables contains a margin of error; hence it may be more appropriate to abstract the relevant aspects of the model and present them to the users. The models of interaction developed by the next two systems are intended to be used by a coaching agent (in the

future) in advising and guiding the group interaction. HabiPro (Vizcaino, Contreras, Favela, and Prieto, 2000) is a collaborative programming environment that both displays the students' participation statistics, and models more complex interaction variables. The system includes a group model, and an interaction model, which includes a set of "patterns" describing possible characteristics of group interaction (e.g. the group prefers to look at the solution without seeing an explanation). During the collaborative activity, the group model compares the current state of interaction to these patterns and proposes actions (such as withholding solutions until the students have tried the problem).

EPSILON (Soller and Lesgold, 2000) monitors group members' communication patterns and problem solving actions in order to identify situations in which students effectively share new knowledge with their peers while solving object-oriented design problems. In the first phase of the collaboration management cycle (Figure 1), the system logs data describing the students' speech acts (e.g. Request Opinion, Suggest, Apologize) and actions (e.g. Student 3 created a new class). In the second phase, the system collects examples of effective and ineffective knowledge sharing, and constructs two Hidden Markov Models which describe the students' interaction in these two cases. A knowledge sharing example is considered effective if one or more students learn the newly shared knowledge (as shown by a difference in pre-post test performance), and ineffective otherwise. In the third phase, the system dynamically assesses a group's interaction in the context of the constructed models, and determines if the students need mediation.

2.3 Systems that Offer Advice

This section describes systems that analyze the state of collaboration using a model of interaction, and offer advice intended to increase the effectiveness of the learning process. The coach in an advising system plays a role similar to that of a teacher in a collaborative learning classroom. This actor (be it a computer coach or human) is responsible for guiding the students toward effective collaboration and learning. Since effective collaborative learning includes both learning to effectively collaborate, and collaborating effectively to learn, the facilitator must be able to address social or collaboration issues as well as task-oriented issues. Collaboration issues include the distribution of roles among students (e.g. critic, mediator, idea-generator) (Burton, 1998), equality of participation, and reaching a common understanding (Teasley and Roschelle, 1993), while task-oriented issues involve the understanding and application of key domain concepts. The systems described here are distinguished by the nature of the information in their models, and whether they provide advice on strictly collaboration issues or both social and task-oriented issues. We begin by taking a look at systems that focus on the social aspects of collaborative learning.

A classroom teacher might mediate social interaction by observing and analyzing the group's conversation, and noting, for example, the levels of participation among group members, or the quality of the conversation. A CSCL system that can advise the social aspects of interaction therefore requires some ability to understand the dialog between group members. Barros and Verdejo's (2000) asynchronous newsgroup-style system, DEGREE, accomplishes this by requiring users to select the type of contribution (e.g. proposal, question, or comment) from a list each time they add to the discussion. This data satisfies the first phase of the collaboration management cycle. The system's model of interaction (phase 2 of the collaboration management cycle) is constructed using high-level attributes such as cooperation and creativity (derived from the contribution types mentioned above), as well as low-level attributes such as the mean number of contributions. In the third phase of the collaboration management cycle, the system rates the collaboration between pairs of students along four dimensions: initiative, creativity, elaboration, and conformity. These attributes, along with others such as the length of contributions, factor into a fuzzy inference procedure that rates students' collaboration on a scale from "awful" to "very good". The advisor in DEGREE elaborates on the attribute values, and offers students tips on improving their interaction.

A limitation of the DEGREE approach might be its dependence on users' ability to choose the correct contribution type (proposal, comment, etc.). An alternative way of obtaining this information is to have users select *sentence openers*, such as "Do you know", or "I agree because" to begin their contributions. Associating sentence openers with conversational acts such as Request Information, Rephrase, or Agree, and requiring students to use a given set of phrases, may enable a system to understand the basic flow of dialog without having to rely on Natural Language parsers. Most sentence opener approaches make use of a structured interface, comprised of organized sets of phrases. Students typically select a sentence opener from the interface to begin each contribution.

McManus and Aiken (1995) take this approach in their Group Leader system. Group Leader builds upon the concept that a conversation can be understood as a series of conversational acts (e.g. Request, Mediate) that correspond to users' intentions (Flores, Graves, Hartfield, and Winograd, 1988). Like Flores et al.'s Coordinator system, Group Leader uses state transition matrices to define what

conversation acts should appropriately follow other acts, however unlike the Coordinator, users are not restricted to using certain acts based on the system's beliefs. Group Leader compares sequences of students' conversation acts to those recommended in four finite state machines developed specifically to monitor discussions about comments, requests, promises, and debates. The system analyzes the conversation act sequences, and provides feedback on the students' trust, leadership, creative controversy, and communication skills.

The success of McManus and Aiken's Group Leader (1995) began a proliferation of systems that take a finite state machine approach to modeling and advising collaborative learners. One year later, Inaba and Okamoto (1996) introduced iDCLE, a system that provides advice to students learning to collaboratively prove geometry theorems. This system infers the state of interaction by comparing the sequences of conversation acts to one of four possible finite state machines. Advice is generated through consideration of the dialog state and the roles of each group member.

The next three collaborative learning systems interact with students via a set of specialized computer agents that address both social and task-oriented aspects of group learning. GRACILE (Ayala and Yano, 1998) is an agent-based system designed to help students learn Japanese. The system maintains user models for each of the students, and forms beliefs about potential group learning opportunities. Group learning opportunities are defined as those that promote the creation of *zones of proximal development* (Vygotsky, 1978), enabling a student to extend her potential development level. GRACILE's agents assess the progress of individual learners, propose new learning tasks based on the learning needs of the group, and cooperate to maximize the number of situations in which students may effectively learn from one another.

The models of interaction employed by LeCS (Rosatelli, Self, and Thirty, 2000), and COLER (Constantino-Gonzales and Suthers, 2000) also integrate task and social aspects of interaction. LeCS is similar to GRACILE in that a set of computer agents guide students through the analysis of case studies. The agents monitor students' levels of participation, and track students' progression through the task procedure, while addressing students misunderstandings and ensuring group coordination. COLER uses decision trees to coach students collaboratively learning Entity-Relationship modeling, a formalism for conceptual database design. For example, the coach might observe a student adding a node to the group's shared diagram, and might notice that the other group members have not offered their opinions. The coach might then recommend that the student taking action invite the other students to participate. The system also compares students' private workspaces to the group's shared workspace, and recommends discussion items based on the differences it finds.

3 Discussion

In the first half of this paper, we developed the collaboration management cycle from a systems perspective. This cycle describes the actions a system can take to support online collaborative learning interaction. In the second half of this paper, we reviewed an array of systems that instantiate the three stages of this model: mirroring, monitoring, and advising.

Mirroring systems record and reflect input data, while monitoring and advising systems process this input data to obtain a higher-level representation which is then either displayed to the collaborators (in the case of indicator-based systems), or used by the system (in the case of advising systems). This higher-level, *derived* representation may be quantitative or qualitative in nature. A quantitative derivation process might entail counting, for instance, the number of dialog or workspace actions a user has taken. A qualitative derivation process requires taking relational information into account, such as interdependencies between actions or between actions and application context. Table 1 summarizes a number of systems we have reviewed in this paper by the type of interaction data they assimilate, the derivation mechanism they use to produce higher-level (derived) data representations, the derived data representation, and the way in which they attempt to achieve or maintain equilibrium (ideal collaboration).

In some cases, systems that monitor the state of interaction are not all that different from systems that provide advice. For example, suggesting that a student participate more does not require much more computation than displaying students' participation statistics; moreover both approaches may have the same effect. These systems begin to differ when the knowledge behind the indicators requires a great enough level of inferencing to warrant having a coach analyze the data to scaffold the learning process.

Seeing that providing advice is easy given a set of indicators, why not both advise and show indicators? This may be difficult if we do not know which indicator configuration is most favorable to learning. Furthermore, the interaction management skills students need to interpret and act upon the indicator values might transfer well to other situations. On the other hand, analyzing the indicators may increase the students' cognitive load, and some students might misinterpret the indicators.

Table 1. A summary of systems that support collaborative learning, classified according to their structure

System	Input data	Derivation mechanism	Derived data	Intervention type
Groupkit, Gutwin (1995)	interface usage	none	none	mirroring
Plaisant et al. (1999)	problem-solving actions	none	none	mirroring
Simoff (1999)	synchronous and asynchronous dialog	counting, content analysis	participation, structure of discussion	graphical visualization
Wortham (1999)	dialog	counting, social network	exchange patterns	graphical visualization
Jermann (work in progress)	shared workspace actions and dialog	counting	participation	graphical visualization
Sharlock II, Ogata et al. (2000)	user profile, web page access	counting, similarity indices	shared knowledge awareness map	graphical visualization
HabiPro, Vizcaino et al. (2000)	shared workspace actions, student preferences	matching group interaction "patterns"	ideal participation, motivation	coach in progress
Action-based Collaboration Analysis, Muehlenbrock (2000)	shared workspace actions on visual languages	activity (plan) recognition	activities, conflicts, coordination	visualization (coach in progress)
EPSILON, Soller and Lesgold (2000)	shared workspace actions, tagged dialog	Hidden Markov Models	effectiveness of knowledge sharing	coach in progress
Group Leader, McManus and Aiken (1995)	tagged dialog	finite state machines	trust, leadership, communication	coach
iDCLE, Inaba and Okamoto (1996)	tagged dialog	finite state machines	roles	none
DEGREE, Barros and Verdejo's (2000)	tagged dialog	counting and fuzzy inference	initiative, creativity, elaboration, conformity	coach and visualization
MArCo (Tedesco and Self, 2000)	dialog in formal language	BDI modeling	(meta-cognitive) conflicts	display of conflicts
GRACILE, Ayala and Yano (1998)	workspace actions, learner models	rule-based expert system	appropriate student helpers & learning tasks	coach
LeCS (Rosatelli, Self, & Thirty, 2000)	shared workspace actions,	case tree	participation, group coordination	coach
COLER, Constantino-Gonzales & Suthers (2000)	shared and private actions, dialog	decision trees	participation, agreement with group procedure	coach

4 Future Work

The concept of *supporting* (as opposed to enabling) peer-to-peer interaction in computer-supported collaborative learning systems is still in its infancy. We have not yet seen full-scale evaluations of the types of systems we have covered here. More studies are needed that test the utility of various strategies for computationally supporting online collaborative learning.

Knowledge about how students interact is useful to a system only if it can apply this knowledge to recognize specific situations that call for intervention. Classroom teachers learn to analyze and assess student interaction through close observance of group interaction, trial and error, and experience. Developing a system to analyze group conversation, however, poses its own challenges. Focused research in computational modeling of peer dialog will help in making the transition from understanding how to mediate learning groups to understanding how to train a system to mediate learning groups.

References

Ayala, G, and Yano, Y (1998). A collaborative learning environment based on intelligent agents. *Expert Systems with Applications*, 14, 129-137.

Barros, B., and Verdejo, M.F. (2000). Analysing student interaction processes in order to improve collaboration. The DEGREE approach. *International Journal of Artificial Intelligence in Education*, 11, 221-241.

Burton, M. (1998). Computer modelling of dialogue roles in collaborative learning activities. Unpublished doctoral dissertation, Computer Based Learning Unit, The University of Leeds.

Constantino-Gonzalez, M., and Suthers, D. (2000). A coached collaborative learning environment for Entity-Relationship modeling. *Proceedings of the 5th International Conference on Intelligent Tutoring Systems*, Montreal, Canada, 324-333.

Flores, F., Graves, M., Hartfield, B., & Winograd, T. (1988). Computer systems and the design of organizational interaction. *ACM Transactions on Office Information Systems*, 6(2), 153-172.

Greer, J., McCalla, G., Cooke, J., Collins, J., Kumar, V., Bishop, A., and Vassileva, J. (1998). The Intelligent Helpdesk: Supporting peer-help in a university course. *Proceedings of the 4th International Conference on Intelligent Tutoring Systems (ITS '98)*, San Antonio, TX, 494-503.

Gutwin, C., Stark, G., Greenberg, S. (1995) Support for workspace awareness in educational groupware. *Proceedings of CSCL'95. The First International Conference on Computer Support for Collaborative Learning*, 147-156.

Inaba, A., and Okamoto, T. (1996). Development of the intelligent discussion support system for collaborative learning.. *Proceedings of ED-TELECOM '96*, Boston, MA, 137-142.

McManus, M. and Aiken, R. (1995). Monitoring computer-based problem solving. *Journal of Artificial Intelligence in Education*, 6(4), 307-336.

Muehlenbrock, M. (2000). Action-based collaboration analysis for group learning. Ph.D. thesis, Department of Mathematics/Computer Science, University of Duisburg.

Muehlenbrock, M., and Hoppe, U. (1999). Computer supported interaction analysis of group problem solving. *Proceedings of the Computer Support for Collaborative Learning (CSCL) 1999 Conference*, Palo Alto, CA: Stanford University, 398-405.

Nurmela, K.A., Lehtinen, E., Palonen, T. (1999). Evaluating CSCL log files by Social Network Analysis. *Proceedings of the Computer Support for Collaborative Learning (CSCL) 1999 Conference*. Palo Alto, CA: Stanford University, 434-444.

Ogata, H., Matsuura, K., and Yano, Y. (2000). Active Knowledge Awareness Map: Visualizing learners activities in a Web based CSCL environment. *International Workshop on New Technologies in Collaborative Learning*, Tokushima, Japan.

Plaisant, C., Rose, A., Rubloff, G. Salter, R. & Shneiderman, B. (1999). The design of history mechanisms and their use in collaborative educational simulations. *Proceedings of the Computer Support for Collaborative Learning (CSCL) 1999 Conference*., Palo Alto, CA: Stanford University, 348-359.

Rosatelli, M., Self, J., and Thirty, M. (2000). LeCs: A collaborative case study system. *Proceedings of the 5th International Conference on Intelligent Tutoring Systems*, Montreal, Canada, 242-251.

Roseman, M. and Greenberg, S. (1992). GroupKit: A groupware toolkit for building real-time conferencing applications. *Proceedings of the ACM CSCW Conference on Computer Supported Cooperative Work*, Toronto, Canada, 43-50.

Simoff, S. (1999). Monitoring and Evaluation in Collaborative Learning Environments. *Proceedings of the Computer Support for Collaborative Learning (CSCL) 1999 Conference*, Palo Alto, CA: Stanford University, (Available at http://www.ciltkn.org/cscl99/A83/A83.html).

Soller, A., and Lesgold, A. (2000). Knowledge acquisition for adaptive collaborative learning environments. *Proceedings of the AAAI Fall Symposium: Learning How to Do Things*, Cape Cod, MA.

Teasley, S. and Roschelle, J. (1993). Constructing a joint problem space. In S. Lajoie and S. Derry (Eds.), *Computers as cognitive tools* (pp. 229-257). Hillsdale, NJ: Lawrence Erlbaum.

Tedesco, P. and Self, J. A. (2000). Using meta-cognitive conflicts in a collaborative problem solving environment. *Proceedings of the 5th International Conference on Intelligent Tutoring Systems*, Montreal, Canada, 232-241.

Vizcaino, A., Contreras, J., Favela, J., and Prieto, M. (2000). An adaptive, collaborative environment to develop good habits in programming. *Proceedings of the 5th International Conference on Intelligent Tutoring Systems*, Montreal, Canada, 262-271.

Vygotsky, L. S. (1978). *Mind in society: The development of higher psychological processes*. London: Harvard University Press.

Wortham, D.W. (1999). Nodal and matrix analyses of communication patterns in small groups. *Proceedings of the Computer Support for Collaborative Learning (CSCL) 1999 Conference*. Palo Alto, CA: Stanford University, 681-686.