Computer Software Support for Collaborative Learning

Patrick Jermann, Amy Soller, Alan Lesgold

Computer Software Support for Collaborative Learning

Patrick Jermann, Amy Soller, and Alan Lesgold

Introduction

In this chapter, we discuss two approaches to supporting collaborative learning activities in higher education through technological means: structuring and regulating collaboration (Dillenbourg, 2002). Structuring approaches aim to create favorable conditions for learning by designing and scripting the situation *before* the interaction begins. They attempt to define the structure of the learning experience by varying the characteristics of the participants (e.g. the size and composition of the group, or definition and distribution of roles), the availability and characteristics of tools and communication media, and the nature of the task (e.g. writing, problem-solving). Regulation approaches support collaboration by taking actions *after* the interaction has begun. They compare the dynamically changing state of student interaction to a model of "desired" interaction, and intervene when discrepancies between these two states are discovered (Jermann, Soller, & Muehlenbrock, 2001).

Following these definitions, the notions of structuring and regulating collaboration qualify pedagogical approaches rather than distinct categories of tools. Simply put, the same tool can be used in a structuring or a regulating approach: the introduction of a task manager into a learning environment corresponds to a structuring approach if it takes place before the interaction (it is part of the initial conditions) or to a regulating approach if it takes place after the beginning of the interaction. However, the task manager does not hold the same place in these two situations. In the context of the structuring approach, it is the primary means of intervention and we focus on the characteristics of the task manager and their potential effect upon the learner's behavior. Typically, the question is, what features of the task manager make it a useful tool in a particular situation for a particular learning outcome? In the context of the regulating approach, the main focus is not on the task manager itself but rather on software tools that help diagnosing the situation. The task manager is introduced into the learning environment as a remedial action that follows a diagnosis of the situation. The question is, what role can software tools play in the regulation process? As a summary, following the structuring approach, we describe how the choice and design of tools affects interaction and learning. Following the regulating approach, we illustrate how technological means can be used to help diagnose the situation.

Although the main focus of this contribution is on computer software, we must not forget that our overall goal is to encourage and enhance learning. It is not the mere presence or complexity of the technology that improves a student's learning experience, but the quality of the match between the tools and the learning processes they structure or regulate. This relationship can be described by the notion of educational affordances (Kirschner, 2002). "Educational affordances are those characteristics of an artifact […] that determine if and how a particular learning behavior could possibly be enacted in a given context (e.g., project team, distributed learning community)" (*op.cit*, p.19). Although this definition applies to a broad

range of educational tools and settings (e.g. physical tools, sign systems such as human language, Nardi, 1996), we restrict our discussion to computer software artifacts and collaborative learning processes such as socio-cognitive conflict, internalization, and explanation (for an overview, see Dillenbourg & Schneider, 1995).

In face-to-face situations, teachers structure and regulate student interaction by, first, preparing the lesson plans and setting up the group work, and then, intervening in the collaboration when they feel it is necessary. With regard to computer tools, a structuring approach might involve choosing or designing artifacts that offer affordances for the learner to discover, understand, and use in her own thinking (Kirschner, 2002; Stahl, this volume). Two powerful ideas support this notion of computer support. First is the idea that the tools we use influence the way we think, learn, and act (Leont'ev, 1981; Vygotsky, 1985; Jonassen, 1992). Second is the idea that we can purposefully design tools that enable or facilitate certain desirable actions (Salomon, 1988; Salomon, 1990). Kristine Lund gives more detail about the subjacent concepts of mediation and interiorisation in chapter 3 of this volume.

Three different types of systems may be used to structure collaboration. The first class of systems includes standard productivity tools (word processors, spreadsheets and databases), and communication tools (email, newsgroups, hypertext). Communication tools are often packaged with student management and evaluation tools in learning management systems (LMS) such as WebCT© and Blackboard©. These systems are available off-the-shelf, and can be installed easily and used immediately, without modification, to support collaborative learning. It is important, however, to understand how these tools are designed to support various collaborative learning behaviors and practices, and to choose the most appropriate tool. In this chapter, we will take a look at how Guzdial and his colleagues' (1999) used a hypertext production system called CoWeb to support learning groups.

Our second class of structuring systems includes software that was specifically designed to enhance the quality and effectiveness of collaborative interactions. For example, we describe the use of sentence openers in semi-structured dialogue interfaces (McManus & Aiken, 1995; Baker & Lund, 1997), and the effects of representational guidance through visual languages (Suthers, 2001). In the third class of structuring systems, the computer supports the learners by encouraging them to interact according to a collaboration *script*, or predefined scenario (O'Donnel & Dansereau, 1992; Dillenbourg, 2002). We illustrate this approach by presenting the Online Planning and Study Environment (OPSE), an environment developed to organize collaborative unit planning in the context of teacher professional development.

In addition to structuring the students' learning, a computer system might attempt to regulate the interaction as it unfolds. This is the second possible approach to supporting collaborative learning. Because student interaction is dynamic and somewhat unpredictable, it can be very difficult for a computer to analyze, assess, and dynamically coach group learning. Some researchers have taken steps to understand the processes of collaborative learning, and the possible methods for supporting them computationally. For example, Jermann, Soller, and Muelhenbrock

(2001) present a framework for computationally supporting the regulation of learning interaction based on the metaphor of a cybernetic feedback loop. Following this metaphor, regulation is a four step process that starts with the collection of raw data about the behavior of participants (e.g. mouse clicks, textual messages). In the second step, raw data is aggregated into a set of pedagogically meaningful indicators that constitute the state of interaction. In the third step, the current state is compared to a representation of the desired state of interaction. Then, if there is a discrepancy between these two states of interaction, remedial actions are proposed in the fourth step.

Computers may offer support for any or all of these four steps. For example, support for the first two steps might be provided by mirroring tools, which assist learners and teachers in the collection of data (e.g. Donath, Karahalios & Viégas, 1999). Support for the third step might be provided by metacognitive tools, which assist learners or tutors in diagnosing the interaction through visualizations (e.g. Jermann, 2002; Zumbach, Muehlenbrock, Jansen, Reimann, & Hoppe, 2002). Support for the fourth step might be provided by guiding systems, which propose remedial actions based on an assessment of the situation. In this chapter, we provide several examples of systems that apply a guiding approach (e.g. Barros & Verdejo, 2000; Soller & Lesgold, in press).

In the two main sections of this chapter, we discuss in more detail the structuring and regulating approaches to supporting collaborative learning. We illustrate how various software applications can be used to structure or regulate student interaction, and provide some insight into why they affect learning in a positive way. In the last section we present a case study of the EPSILON system (Soller, 2000; Soller & Lesgold, ibid ?) to illustrate the integration of structuring and regulating approaches as well as the complexity and challenges of automatic interaction analysis.

1. Structuring Collaboration

From a tools perspective, we distinguish between tools used to structure and tools that help regulate. From a more theoretical perspective, however, it may be difficult to distinguish between the various notions of structuring that drive tool development. We already briefly introduced three notions of structuring collaboration, corresponding to the three classes of structuring systems. In this section, we define the idea of structuring collaboration more precisely, and introduce a fourth strategy – that of structuring the interaction in a way that enables and facilitates computer assisted interaction regulation.

First, there is the notion of considering the properties of artifacts that, by design, structure the actions they enable and mediate. This way to structure interaction consists of *taking advantage of natural affordances.* The idea that a user's behavior and actions are influenced by the tools he chooses is a central stance in the ecological theories of action, which have recently become popular in the CSCL field. In particular, the distributed cognition approach (Hutchins, 1991; Salomon, 1993) states that in order to understand how knowledge is created, one has to observe the interaction between tools and actors, rather than the individual mind in

isolation. Wood (1993) clarifies the role of tools in this cognitive partnership: "[...] cognition is never simply 'amplified' or 'externalized', but rather cognition is 'mediated' through the external artifacts and collaborators such that the new cognitive system which is formed has a radically different character, structure and functionality than the cognition of the unsupported individual" (p. 2). The concept of affordance (Gibson, 1979; Gaver, 1991; McGrenere & Ho, 2000; Kirschner, 2002; Norman, 1988) describes the interaction between the characteristics of the tools, and the characteristics of the actors who use them. From a CSCL system designer's point of view, the goal is to choose (or in the next case, design) software that matches the needs of the learners, and the desires of the teachers. The tools should address the types of collaborative interaction, and the behaviors that promote learning.

Second, structuring collaboration may refer to a pedagogical approach in which tools are designed specifically to enable and foster particular behaviors. This notion takes advantage of both upon the natural affordances of tools, and the potential to purposefully *design situations and tools to create new affordances*, or make some existing affordances more salient. For example, a teacher might construct a lesson plan in which students are assigned roles and asked to debate a topic via email rather than chat, because of the ways that the email medium influences communication (e.g. longer messages with more contextual information). This scenario corresponds to a structuring approach in which a standard communication tool was chosen, and a situation was designed to take advantage of its affordances. Alternatively, the teacher could have had the students use customized tools that create new affordances geared specifically toward the learning objectives. For example, communication interfaces may feature sentence-openers (buttons that suggest phrase stems such as, "I propose that…" or "Do you know why…", or diagrammatic conversation tools (i.e. graphical notes that can be created and linked to build networks of arguments). Although these interfaces may change the way in which users naturally communicate, they are designed to foster certain types of interaction believed to promote learning. Both standard and custom built tools structure communication through their very design and affordances, corresponding to Jonassen's (2000 as cited in Kirschner & Wopereis, 2003) notion of mindtools: "[...] computer-based tools and learning environments that have been adapted or developed to function as intellectual partners with the learner in order to engage and facilitate critical thinking and higher order learning." (p. 9).

Third, structuring collaboration may refer to the use of a collaboration script, in which the learners are encouraged to interact according to a predefined scenario (O'Donnel & Dansereau, 1992; Dillenbourg, 2002), in other terms *structuring via scripting*. The script defines a set of rules and characteristics for each collection of activities. It also specifies the sequence of activities, and for each activity, it specifies how group members are expected to collaborate to complete the exercise. With regard to computer support, the script can be implemented in a learning management system (LMS), in which the learning activities are organized through navigation tools, and visualizations enable the teacher and the learners to track their progress.

Finally, collaboration may be structured with the intention of *facilitating the human or computer-based analysis, assessment, and eventual regulation of the interaction.*

An important side effect of custom built tools as we described them above, is that they may provide information about the interaction that would not be available otherwise. For instance, sentence-opener based communication interfaces allow learners to self-categorize their messages. The user need only consider and select the first few words of her utterance. The system then translates the selected phrase into a conversational intention or *speech act* (i.e. Inform, Request, Acknowledge). This information may be valuable to a system that attempts to abstract meaning from sequences of conversational intentions, in order to automatically assess the quality of conversation (Soller, 2002; Soller & Lesgold, in press). Another example of structuring with the intention of facilitating regulatory processes is the use of diagrammatic conversation tools. When posting notes in an LMS (described earlier in the introduction), students might be asked to choose the type of note that best matches their communicative intention. The notes may then be linked to each other in a specified way, enabling the system to identify the structure of the students' arguments, and propose potential improvements (e.g. can you find some evidence for this claim?). In the next two sections, we discuss the various roles that the computer might play these structuring and regulating approaches, and the types of interaction they support.

We now describe a few tools to illustrate the first three notions underlying the idea of structuring collaboration: taking advantage of natural tool affordances, creating new affordances through design, and structuring via scripting.

1.1 Taking advantage of natural tool affordances

Communication is at the center of collaboration, be it to exchange points of view, debate disagreements, or explain difficult concepts. Technology that enables and facilitates the communication and transmission of information provides natural affordances for knowledge exchange, contributing towards the development of mutual understanding and learning. In distance collaborative learning environments, the first (although not necessarily primary) role the computer plays is that of an information transmitter. This is an important role, because without it, learning mechanisms such as appropriation, and mutual regulation (discussed by Stahl in this volume), are not possible. It is important to remember, however, that the use of communication tools does not necessarily lead to positive learning outcomes. The transmission of information should be complemented by a well planned curriculum, meaningful tasks, assistance by a facilitator or coach, and a sound social space to allow trust (also see chapter 1 in this volume). Three of the four examples we discuss in this section illustrate how researchers have taken advantage of the natural affordances offered by communication technology to support collaborative learning activities. The fourth example illustrates how one group of researchers took advantage of the affordances offered by simulation technology to support collaborative learning communication processes.

1.1.1 Four examples

Simple communication tools, such as email, can mediate communication in subtle ways, while having a large impact on the form and frequency of interaction. For example, email was originally designed as an asynchronous communication tool.

This means that after users send a message to one or more recipients, they may or may not receive a response. If they do receive a response, their choice to use an asynchronous communication medium requires that they understand a certain period of acceptable time may elapse – hours, days, weeks, or even months! If they were to choose a synchronous medium, (for example, the telephone, or an online chat system), then they might reasonably expect a response within seconds or minutes. Although chat systems are traditionally considered synchronous communication tools, and email asynchronous systems, it is possible for learners to exchange emails at a higher rate than they would reply to each other in a chat room. The possibility to use email rapidly, simulating a chat (not intended by the designers of email), also depends on the technological infrastructure. A slow, low bandwidth telephone connection to a mail server affords a very different pattern of email communication than a broadband cable, or LAN connection. From the users' point of view, the choice of email or chat depends on the type of available connection, and whether or not an immediate response is needed. Instructional designers, teachers, and students need to evaluate the affordances offered by such communication tools in light of their learning goals, and choose the most appropriate medium.

The next example illustrates how the intrinsic properties of communication channels influence their use. Dillenbourg and his colleagues (Dillenbourg, Traum, & Schneider, 1996) describe an experiment in which dyads use a whiteboard (a multiuser drawing application) and MOO (a sophisticated chat system) to collect evidence for discovering the assassin in a murder mystery problem-solving task. The designers of the experiment intended the MOO to be the primary communication tool used by students. It turned out, however, that students switched between using the whiteboard and MOO, depending on the type of information they wanted to exchange. They tended to use the MOO when exchanging non-persistent information (e.g. inferences), and the whiteboard when exchanging persistent information (e.g. facts). These students apparently assessed the 'persistence of information' to determine which communication channel to use, and found that the whiteboard afforded the exchange and storage of persistent information better than the MOO.

The third example also illustrates how affordances may influence the activities of learners. The CoWeb project, by the Collaborative Software Lab at Georgia Tech (Guzdial et al., 1999), exploits a special kind of hypertext called a swiki. The swikis in the CoWeb system allow users to create and edit webpages through standard web browsers (e.g. Netscape Navigator™ or Microsoft Internet Explorer™) without requiring them to enter a password or to know the HTML description language. Guzdial and his colleagues describe a wide range of uses for their system. In the CoOl Studio project (Collaborative Online Studio), CoWeb was used as a platform for design review. The project aimed at providing architecture students with a more authentic learning experience than they would obtain in a traditional design studio, in which only their teachers and peers critique their projects (Zimring, Khan, Craig, Haq, & Guzdial, 2001). Using CoOl Studio, students learned about the various perspectives in everyday architectural practice by collaborating with experts outside of the school to develop their architectural designs. The outside reviewers acted as critiques, helping groups of students by pointing out problems with their designs,

and suggesting ideas and references. The asynchronous nature of these tools gave the critics more time to prepare their comments than they had in the previous arrangement, in which students presented posters in a face to face environment. The process of preparing projects for online presentation also improved the students' achievement.

In some cases, the technical limitations afforded by standard communication tools may produce unexpected positive effects. For example, because some experts in the CoOL project had only limited bandwidth, the students needed to learn how to construct simple, low resolution drawings that still encompassed their design ideas. This process of choosing an appropriate representation that would work with the available media forced students to take a closer and more critical look at their project (Zimring et al., 2001).

The three previous examples illustrate the effects of the natural affordances of communication technology upon the communicative behavior of students. Support for learning communication, however, can also be provided through tools that do not enable the transmission of information. For example, imagine two students, sitting face-to-face, in front of one computer, solving a physics problem together by proposing hypotheses to test on a virtual simulation. The design of the simulator orients the way in which students will experience the physical phenomenon together. Can the students select among different views for displaying the phenomenon? Does the system include a schematic view of the situation as well as a dynamic chart showing the key variables (e.g. speed, acceleration)? The design of the simulator also impacts the students' problem solving strategies. For example, does it allow the students to change several parameters at a time, or does is enforce an experimental approach by allowing changes to only one parameter at a time? When the computer structures the interaction between the learners and the task in these ways, it serves as an external support for communication.

Roschelle and Teasley's envisioning machine (1990, 1993), or the rocket simulator described by Stahl (this volume), are examples of simulation environments that learners use to test scientific hypotheses. In both systems, the interaction takes place outside of the system, and the system provides a test bed that serves as a reality check for the learners (Van Bruggen, Kirschner, & Jochems, 2002). As the students discuss the experiments represented on the computer interface, the natural affordances of the simulation influence the students' interaction. For instance, the envisioning machine was specifically designed to represent a physicist's mental model of velocity and acceleration, and this model is grounded in the physicist's professional *community of practice* (Lave & Wenger, 1991). So, structuring collaborating around a particular representation also means helping students learn to communicate about artifacts that are grounded in a community of practice (i.e. learning to speak the "language" of the practice). The mediation of the student's activities by the software tool helps them in entering the community.

1.2 Creating new affordances through design

In the previous subsection, we saw how standard communication tools can be used to structure the interaction between the learners. These tools might even be

improved by observing learners' actions, and adapting the software to promote those actions that facilitate learning. For example, Craig et al. (2000) reported that many participants using CoWeb (described in the previous section) did not seem to post comments in a coherent way, attaching messages to the end of the page rather than close to the referring statements. So, the designers included an in-line text editing box on the CoWeb web pages, encouraging students to post comments near the most relevant statements. This intervention, geared towards augmenting coherence, represents a deliberate approach to designing tools in order to structure collaboration, our second notion of structuring collaboration. In the next two subsections, we present two other types of systems that provide affordances to deliberately structure communication: graphical argumentation tools, and structured dialogue interfaces.

1.2.1 Graphical argumentation tools

Representational tools support group learning by providing a shared context in which students can discuss the problem at hand. Collaborators construct external representations by selecting from a limited set of objects and relations, and adhering to certain rules regarding their use and combination (Van Bruggen, Boshuizen, & Kirschner, 2003). These objects help students structure, externalize, and coordinate their ideas, and then serve as the medium for communication. They support and structure collaborative problem solving through the way in which they represent the students' discussions and arguments. In this section, we focus on one type of representational tool: those that support collaborative graphical argumentation. These systems allow people to build graphical representations out of a set of primitive components. They are characterized by the following aspects:

- Ontology. What primitive components are available? What kinds of knowledge can these components represent. For example, a system that permits diagrams to contain only data and hypotheses and links between them will lead to very different representations, and presumably very different group learning than a system that has only a single object type – such as concept mapping – and relies on link forms to convey low-level meaning. The choice of an ontology depends on the designers' analysis of the students' capabilities and the worldview they want to transmit through the language.
- Perspective. Systems support different kinds of perspectives on a discussion. For example, one system might focus attention on the temporal sequence of contributions to a discussion while another might focus all attention on relationships between assertions and supporting evidence, while a third might focus attention on the most central underlying assumptions in an argument. A view may be multivalent, and related to other views from functional, behavioral or physical perspectives (see De Jong et al, 1998 for examples). Stahl (2001) also uses this term to describe different conceptualizations of a problem.
- Specificity and precision. Stenning and Oberlander (1995, p. 98) defined this as, "the demand of a system of representation that information in some

class be specified in any interpretable representation." A graphical display language will be more useful if it permits users to specify arguments in as much detail and with as much precision as possible. This will likely require a number of capabilities, including the ability to provide both microscopic detail for parts of arguments when that is useful and also more generalized statements as well, so that users can overview arguments and also "drill down" and deal with details.

- Modality describes the form of expression used for displaying information such as text, animation, and graphs. The modality corresponds to the way the representational notation is implemented in the representational tools (Van Bruggen, Boshuizen, & Kirschner, 2003). Systems differ, for example, in whether they have simple boxes with labels or multiple shapes of text boxes with each shape – or even a more meaningful icon – denoting the kind of information that is in the box.

Belvedere (Suthers, Weiner, Connelly, & Paolucci, 1995) is a graphical argumentation system that facilitates and structures different aspects of student communication. The system was initially developed to encourage individual students to consider competing scientific theories, and was then extended to support group discussion about these theories. The environment includes a specialized drawing window where students can create boxes and circles that represent 'Principles', 'Hypotheses', and 'Claims', and connect them via links such as 'explains', 'justifies', or 'supports'. The Belvedere system has recently been redesigned to focus on the evidential relations between data and hypotheses, and the graphical representations are now considered more as a resource for conversations, than as the medium of conversation.

Suthers (2001) describes two ways in which external representations may impact the collaborative learning process. First, the representation may *constrain* the language that students use to express themselves, since the language must be consistent with the representation. For example, the constraints imposed by a particular notation are a powerful way to structure the students' understanding of the task. And, the names of the objects that students manipulate may impact the meaning that the students give them. Second, the representation may encourage students to discuss those aspects of the problem that are most *salient* in the representation, and it may suppress discussion on those aspects that are not salient, or not represented. The salience of concepts in a representation also impacts the ease with which students can remember or refer to them. For example, Suthers (2001) explains that a double entry table, which maps hypotheses to data, makes missing mappings more salient than a textual representation of the same information. In the double entry table, it is easy to detect the empty cells, while in the textual representation, the missing relationships may not necessarily be perceptually visible. Suthers (2001, ibid ?) describes these two aspects as the constraint and salience of representational guidance. Although constraints are often thought of as the aspects of technology that we try to minimize, the constraints imposed by a particular notation may in fact provide powerful means to structure the way that students perceive the task.

1.2.2 Structured Dialogue Interfaces

Structured dialogue interfaces structure learners' interactions in a different way – they provide specialized widgets that students can use to compose messages. These interfaces became popular in the late 1990s, after McManus and Aiken (1995) showed the potential benefits of using structured chat interfaces to enhance educational activities. Reports by Baker and Lund (1997), Robertson (1998), and Soller (1998) describe sentence opener-based interfaces to either encourage students to engage in certain types of interaction, or structure the interaction to facilitate computational analysis. To illustrate these ideas, we describe the work done by Baker, de Vries, Lund, and Quignard (1997, 2001) on the C-CHENE system. A snapshot of the structured dialogue interface used by Soller in her research can be found in the case study at the end of this chapter (Figure 4).

The C-CHENE software (Baker et al., 1997) contains a graphical workspace where students can build energy chains on a shared workspace, and chat through a separate communication tool. Two versions of the communication tool were developed. The first was a plain chat-box, and the second a dedicated button interface. Students using the dedicated interface clicked on the buttons to either send a message directly to their peer (e.g. 'Ok'), or bring up a prepared phrase (sentence opener) that they could finish however they liked (e.g. "Because …"). They could also compose their entire contribution using a menu (e.g. "I propose to … <create a reservoir>". These menu based sentence openers are of special interest because they integrate references from the task (e.g. <the reservoir>) into the communication interface, and thereby combine task and communication support in one widget.

Baker et al.'s (1997) results show that providing the right degree of constraint on typewritten CMC can in fact promote reflection on the fundamental concepts at stake. We do not yet understand the full effects of semi-structured interfaces. From our experiences, some students perceive that semi-structured communication interfaces make the expression of ideas more difficult (because the "right" button is not available), while for others, the mere presence of the labeled buttons encourages them to generate more messages that use those labels. For example, some semi-structured dialogue interfaces are designed specifically to encourage the generation of specific kinds of messages (e.g. reflection, explanation). The degree of structure (and hence flexibility from the user's perspective) also impacts the degree to which students are able to generate off-task messages, that were not intended by the designer (see Vizcaino, 2001, for an example of a system that detects these).

Later in this chapter (in the guiding systems section), we illustrate how a structured communication interface can help a system assess the quality of student interaction (Soller, 2002; Soller & Lesgold, in press). In this research, the structured interface takes advantage of the fact that students self-categorize their contributions, thereby facilitating the automatic interaction analysis.

1.3 Structuring via scripting

We now take a look at the role that software can play to complement structuring through scripting. Scripts define sequences of student activities, and advice on how students should collaborate during each activity (Dillenbourg, 2002). Online study

environments and virtual campuses reify scripts with various phases that differ depending on the modality of interaction (e.g. distance, face-to-face), the size of the group, and the task. Reifying a script means breaking an extensive domain down into manageable pieces. Some virtual campuses provide a set of generic tools (such as email, discussion boards, chat, online notebooks), that students can use to supplement learning activities that were developed outside the virtual environment. In these cases, discussion tools tend to be the focus of the technological support. Other online study environments, such as the Online Planning and Study Environment (OPSE) that we describe next, do include customized online learning activities that assist learners as they build representations and objects. Yet other systems primarily provide a project management functionality to the students by representing the goals and tasks that they have to accomplish as well as their progress.

1.3.1 Representing the script

A graphical representation of the phases that constitute the script is common to all these systems, regardless of whether they provide computer based learning activities or not. The script might be simply represented as a navigation tool or as a dedicated planification tool.

As an example, the Online Planning and Study Environment (OPSE hereafter) is an online environment that embodies a theory of unit planning and lesson study as defined by the Institute for Learning (Learning Research and Development Center, University of Pittsburgh). The environment supports teacher professional development using a unit planning approach, and includes provisions for interactive activities, both face-to-face and at a distance. Scripts like this have shown promising results in similar settings (Derry, Siegel, Stampen, & the STEP Research Group, 2002; Jermann, Dillenbourg, & Brouze, 1999; Steinkuehler, Derry, Woods, & Hmelo-Silver, 2002). Teachers who follow the script in the OPSE learn to develop a common unit plan. This unit plan is incrementally built and refined during five sessions, each consisting of online work, and a face-to-face meeting. In between meetings, teachers work online in one of four sections (learning standards, guiding questions, content and skills, and assessment). Each section contains two different types of activities that contribute to planning the total unit: planning activities, which are production oriented, allowing teachers to build unit plan subsets, and learning activities, which present teachers with real-world examples of teaching, and prompt them to discuss their observations. The OPSE is an example of a structured environment that blends individual and collaborative learning activities, online and face to face.

The graphical representation of the script in the OPSE system serves more than just navigational purposes. The recommended order in which the activities should be completed is represented in a schema (see figure 1).

Activities in this section

The schema below shows you the activities to complete. Gray boxes mean that some work still remains to be done, Green boxes indicate that you did work on an activity before.

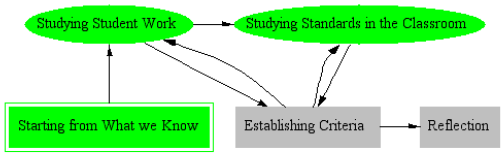

Figure 1. Process Graph, illustrating the steps in the script, and which steps have been completed

The square boxes represent planning steps (teachers plan a real unit by using software tools), and the circles represent study activities (teachers reflect about real world examples). This schema is dynamically generated when the page is requested. The boxes and circles are painted in green if the learner has visited the activity page and posted a message, or created an entry in one of the tools. The boxes and circles are painted grey if the learner has not done the activity yet. From informal observations, we can tell that the motivational effect of the schema's feedback is very strong.

2. Regulating collaboration

The structuring technologies that we described in the previous section enable a system to collect data in a format that is also suitable for automatic diagnosis. Although unconstrained human dialogue is very difficult for computers to analyze autonomously, interaction that is structured through sentence openers or formal languages may help them understand the meaning behind students' actions. Systems designed to structure learners' interaction often record coded traces of their discussions and actions in log files. These actions might be then be analyzed in the context of a computational model describing a set of variables, or indicators, that describe the possible states of the interaction (Jermann, Soller, & Muehlenbrock, 2001). In this section, we will see how a system might use such internal representations to provide support and guidance.

We recently proposed a framework for computer supported interaction regulation (Jermann, Soller, & Muehlenbrock, 2001), which is summarized in the schema shown in Figure 2. According to the schema, interaction regulation mimics a negative feedback loop and consists of three phases. Computers can play a part in the process by supporting the data collection (phase 1), the aggregation of raw data into pedagogically sound indicators (phase 2) the diagnosis of the interaction (phase 3), and the recommendation of remedial actions (phase 4).

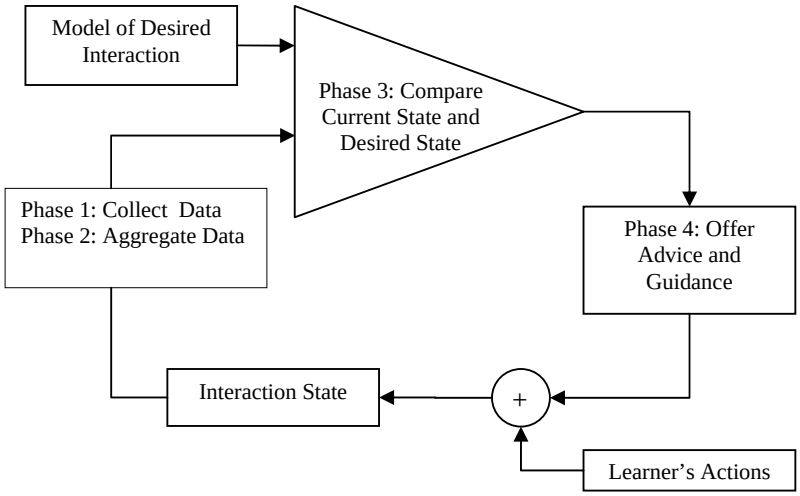

Figure 2. Interaction regulation mimics a negative feedback loop, constantly monitoring the state of the interaction for discrepancies.

As stated in the introduction, computers may offer support for any or all of these four steps. We describe systems that take over phase 1 or 2 as *mirroring* systems, because they are designed to reflect student actions. This reflection could be realized, for example, as a graphical visualization of chat contributions. Systems that support phases 1-3 are termed *metacognitive* tools because they provide the learners or human coaches with information about the state of the interaction, and aid in the analysis of the interaction (Simoff, 1999; Wortham 1999; Zumbach, Muehlenbrock, Jansen, Reimann, & Hoppe, 2002). Some metacognitive tools display the current and desired state of interaction side by side to facilitate and orient diagnosis of the interaction (Jermann, 2002). Finally, *guiding* systems perform all the phases in the regulation process, and then propose remedial actions to help the learners. In this section, we briefly discuss examples of systems that regulate interaction in each of these phases (but see Jermann, Soller, & Muehlenbrock, 2001, for more detail).

2.1 Mirroring tools

The most basic level of regulation involves making the students or teachers aware of participants' actions, without abstracting or evaluating these 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' activity.

Some systems in this category represent dialogue or 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 learners manipulate the controls of the simulation, they can view a history of their actions displayed graphically beneath each target variable (e.g. pressure). The display shows a series of boxes along a timeline, indicating the intervals in which

the users are taking actions, and the system's messages. The data displayed to the students does not undergo any processing or summarizing, but directly reflects the actions taken on the interface. 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.

Chat-based mirroring tools, such as Chat Circles (Donath, Karahalios, & Viégas, 1999), help users keep track of ongoing conversations. Chat Circles is a graphical interface for synchronous chat communication that reveals the social structure of the conversation. Each participant is represented by a coloured circle on the screen in which his or her words appear. The tool is based on an *auditory* metaphor: while one can see all the participants at once, one can only "hear" (that is, read the words) of those one is sufficiently close to. Participants' circle grows and brightens with each message that they send, and fades in periods of silence. The circles, however, do not completely disappear as long as the participants are still connected to the chat. Viewed over time, Chat Circles creates a visual record of conversational patterns. Each user is made aware of the other active, animated participants and can watch the emergence and dissolution of conversational groups.

Metacognitive technologies

Metacognitive tools help regulating interaction by aggregating the interaction data into a set of high-level indicators, and displaying these indicators to the participants along with a representation of quality that allows to evaluate interaction. Learners using these systems are expected to regulate their interaction themselves, assuming that they have been given the appropriate information to do so. For example, Simoff (1999) describes a system that visualizes discussion threads as 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. Simoff's system does not contain this normative information explicitely but it could be transmitted ad hoc as explanation for the meaning of the graphical properties of the tool.

Social network analysis (SNA) (Moreno, 1951; Scott, 2000) provides a set of methods and measurements for studying the relationships within and among groups. SNA tools allow the researcher to manipulate and discover the properties of the group as a whole or of particular participants within the (social) network. Common indicators associated with this technique include variables such as the number of messages sent from one user to another or to a group, and the centrality of various actors within the network. Ogata, Matsuura, and Yano (2000) have extended the notion of social network analysis through a special tool called a Knowledge Awareness Map. This tool can be seen as a specialized social network that also includes "knowledge pieces" describing information that is linked to participants. The Knowledge Awareness Map graphically shows users who else is discussing or manipulating their knowledge pieces. In this case, the distance between users and knowledge elements on the map indicates the degree to which users have similar

knowledge. As in the previous example, the key to interpret the distance between users and knowledge elements is not explicitely given to the users but might be inferred by students based on social problem-solving heuristics.

A third example for metacognitive tools is Jermann's system (2002) that displays to a dyad of subjects a graphical representation of the balance between problem-solving actions and participation in dialogue. The subjects have to tune the four traffic lights in a small simulated town. Every minute, the system computes a ratio based on the number of actions (changes to the traffic lights) and the number of words produced by the participants. This indicator is displayed in an interaction meter that looks like a speedmeter in a car. One end of the scale corresponds to exclusive talking while the other end corresponds to exclusive problem-solving. The interaction meter's color is red on the side of problem-solving actions and green on the side of talking. This normative information is used by the subjects of the experiment to regulate their problem-solving behavior. The dyads in the condition with interaction meters talk more, produce more numerous and more precise plans for action than the dyads in the control condition.

2.2 Computer-Based Coaches and Facilitators

The automatic analysis of interaction and group learning through a distance collaborative learning system is at the forefront of educational technology research. This is the guiding role of the computer, and probably the most challenging function to program. It demands some ability to computationally understand and assess the interaction, as well as a set of diagnostic rules that recommend remedial action. Assessing peer interaction requires an understanding of the factors that influence the collaboration process. Because our knowledge of these factors and their effects is still limited, research in computationally processing and guiding learning teams is ongoing. Many opportunities exist for studying both the behavior of teams and the ways in which we might program computers to play roles such as "smart" facilitators. The systems that fall into this category take advantage of the structuring technologies described earlier in this chapter.

Systems that play a guiding role in supporting collaborative learning generally act in one of three ways: (1) by taking over a portion of the task, hence making the task easier and lessening the cognitive load of the students, (2) by playing the role of teacher or facilitator, offering task-based or social oriented guidance, or (3) by playing a particular social role in the group (e.g. motivator, challenger).

Systems that support collaborative learning by primarily processing information usually operate "behind the scenes", by sending analysis results or recommendations to an online teacher or computer-based coaching agent. They can be seen as black boxes that receive transcriptions of student and group interaction, and output the results of analysing these transcriptions (see Figure 3). The collaborating students need not be concerned with the processing in this black box (shown as a shaded box in the figure); they only need to be aware of its effects, which they would see as the coach's recommendations and advice. The idea is that the black box, or processing engine, should output something that an online human teacher or computer-based coach might find useful in guiding the students towards learning.

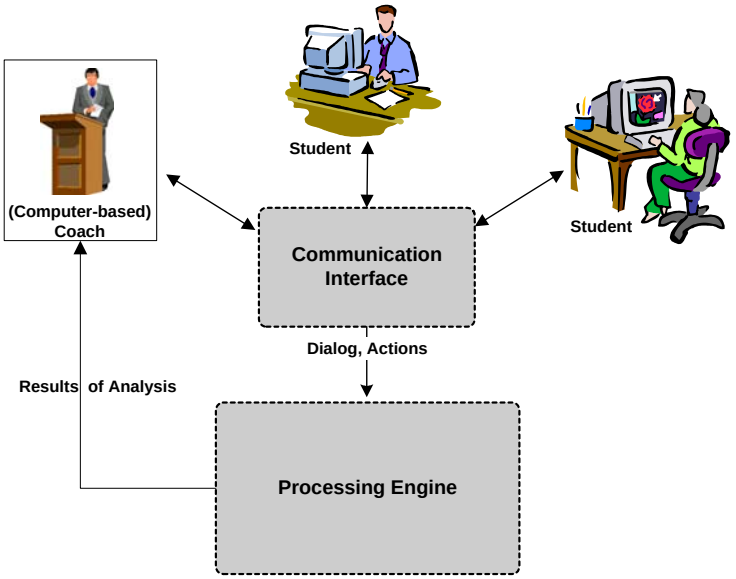

Figure 3: "Black Box" framework for the system-as-processor role

Our first coaching system, Barros and Verdejo's (2000) DEGREE, is an asynchronous newsgroup-style system. It requires 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 is sent to the processing engine (not seen by the students), which constructs a model of interaction using high-level attributes such as cooperation and creativity (derived from the contribution types), as well as low-level attributes such as the mean number of contributions. The engine performs a fuzzy inference procedure on the model and data, and outputs ratings (on a scale from "awful" to "very good") describing the effectiveness of collaboration between pairs of students. Collaboration is assessed along four dimensions: initiative, creativity, elaboration, and conformity. The ratings are sent to DEGREE's computer-based advisor agent, who elaborates on the attribute values, and offers students tips on improving their interaction. In this system, the computer plays the role of an instructor for social interaction.

In our next example, the system plays the role of a task-oriented instructor. GRACILE (Ayala & 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 processing engine assesses the progress of individual learners, and sends this assessment to a set of agents that propose new learning tasks based on the learning needs of the group. The agents also cooperate to maximize the number of situations in which students may effectively learn from one

another.

Our third example, COLER (Constantino-Gonzales & Suthers, 2001), coaches students who are collaboratively learning Entity-Relationship modeling, a formalism for conceptual database design. In this system, students are asked to provide their opinions on their peers' actions. These opinions are sent to the processing engine along with the students' actions on their individual, private workspaces, and their shared, group workspaces. This initial phase provides structure to the students' interaction, beliefs, and actions so that the processing engine can apply decision trees to analyse them, and recommend advice. 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. The coach in COLER plays the role of a facilitator, offering both task-based and social interaction tips.

In these three systems, the automated coach 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 regulating the interaction, and guiding the students towards 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), equality of participation, and reaching a common understanding (Teasley & Roschelle, 1993), while task-oriented issues involve the understanding and application of key domain concepts

The EPSILON System: A Case Study in Guiding Knowledge Sharing Interaction

To close this chapter, we propose a final case illustrates the relationship between tools that structure collaborative learning, and those that regulate collaboration. The EPSILON system follows the black box framework for guiding interaction (cf. Figure 3), implementing the framework's processing engine, but not the actual coaching agent. It differs from the systems described in the previous sections because it is designed to structure, analyse, and (eventually) regulate student knowledge sharing.

Imagine a group of students, who gather around a table to solve a problem, and begin to exchange the knowledge that each brings to bear on the problem. Each group member brings to the table a unique pool of knowledge, grounded in his or her individual experiences. The combination of these experiences, and the group members' personalities and behaviors will determine how the collaboration proceeds, and whether or not the group members will effectively learn from and with each other (Brown & Palincsar, 1989; Dillenbourg, 1999; Webb & Palincsar, 1996). Group members that do not effectively share the knowledge they bring to the learning situation may have a difficult time establishing a shared understanding, and co-constructing new knowledge. These difficulties ultimately lead to poor learning outcomes (Jeong, 1998; Winquist & Larson, 1998), making research efforts to

understand and support student knowledge sharing activities essential. One such endeavor is the EPSILON project – a research initiative to analyse student knowledge sharing interaction, and dynamically assist an automated coach or online instructor in supporting the group.

The EPSILON system is composed of a front end communication interface, and a back end analysis engine (not seen by the students). The communication interface (Figure 4) is comprised of a shared graphical workspace (shown on the top half of the figure), and a chat area (shown on the bottom half of the figure). The graphical workspace allows students to collaboratively solve object-oriented design problems. Objects on the shared workspace can be selected, dragged, and modified, and changes are reflected on the workspaces of all group members.

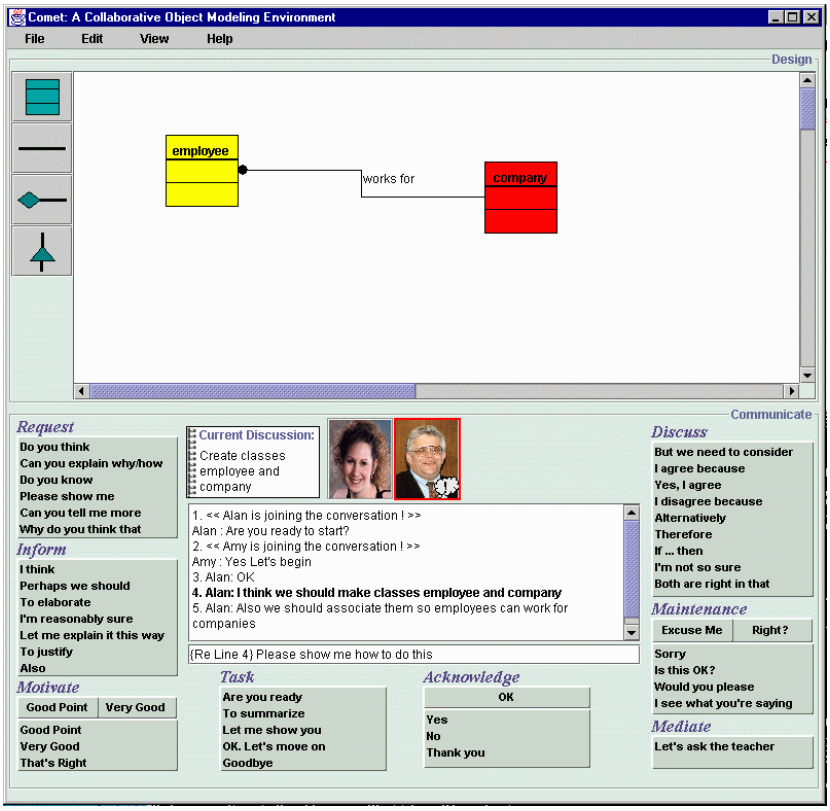

Figure 4: The shared OMT workspace (top), and sentence opener interface (bottom)

The EPSILON chat interface is shown on the bottom half of Figure 4. It contains sets of *sentence openers* (e.g. “I think”, “I agree because”) organized in intuitive categories (such as Inform or Discuss). The sentence openers are designed to help the students identify, to the system, the intentions underlying their conversational

contributions. Each sentence opener is associated with a particular conversational intention, given by a subskill and attribute. For example, the opener, "I think" corresponds to the subskill (or category) "Inform", and the more specific attribute, "Suggest". The categories and corresponding phrases on the interface represent the conversation acts that were most often exhibited during collaborative learning and problem solving in a previous study (Soller, 2001). To contribute to the group conversation in the EPSILON system, a student first selects a sentence opener. The selected phrase appears in the text box below the group dialogue window, where the student may type in the rest of the sentence. Requiring students to use a given set of sentence openers allows a system to automatically code the dialogue without having to rely on Natural Language parsers.

As the students collaborate through the EPSILON interface, the system codes the student communication and actions, and sends the sequences of coded student knowledge sharing interaction to the back-end processing engine (from Figure 3). The processing engine then determines whether or not the students are effectively sharing new knowledge with each other, and what sort of guidance might be helpful. EPSILON applies an artificial intelligence technique, Hidden Markov Modeling (Rabiner, 1989), in order to assess the sequences of knowledge sharing interaction. Hidden Markov Modeling is a probabilistic, state-based machine learning method, designed to analyse noisy, sequential data (for more information, see Soller, 2002).

In an experiment involving 12 groups of 3 students each, engaged in a total of 29 knowledge sharing conversations, the Hidden Markov Model (HMM) approach was able to differentiate between episodes of effective and ineffective knowledge sharing interaction with 74.14% accuracy (Soller, 2002). The approach involved training two different HMMs: one modelled effective knowledge sharing (we call this the effective HMM), and another modelled breakdowns in knowledge sharing (the ineffective HMM). The Hidden Markov Models were shown to be useful for identifying when students are effectively sharing the new knowledge they bring to bear on the problem, and when they are experiencing knowledge sharing breakdowns. The 25% error rate still means that the instructor or the computer-based coaching agent might miss the opportunity to offer advice to the group when it is needed. In this case, however, the data suggests that there is a good chance the system would pick up on the breakdown the next time it occurs.

Once a system has determined that the students may be having trouble, the next step is determining *why*, so that appropriate facilitation methods can be identified and tested. Multidimensional Scaling (Shepard & Arabie, 1979) was used to develop generalized models of effective knowledge sharing, and breakdowns in knowledge sharing. This procedure revealed, for example, that effective knowledge sharing patterns include situations in which the receiver probes the sharer for information, the sharer provides justification and clarification, and the receiver provides motivation and encouragement (Soller, 2002). In developing support strategies, these are some of the behaviors that we might like to encourage. Once the processing engine has determined whether or not the students are experiencing a knowledge sharing breakdown, it would send this information to the coaching module to select an appropriate support strategy.

This case study describes the sort of computational capabilities that are required to analyse collaborative learning interaction, and provide advice and guidance via an automated coach. In summary, the EPSILON software first *enables* interaction by providing a network-based infrastructure and communication interface, then *structures* interaction through the interface, and finally diagnoses the structured data representation with the goal of providing *guidance* to the students.

3. Conclusion

Developing and applying technology to support learning involves understanding the computer's role in the environment, the learning task, and the group learning process. Research shows that there are no recipes that guarantee successful use of technology to support collaborative learning. This is due to the complexity of the interactions between the features of the tools, the characteristics of the interaction that they enable or foster, and the learning outcomes that result from the interaction. As we previously said in introduction, there is no direct link between the use of a particular tool and learning. We will nevertheless attempt to formulate some recommendations that might be useful in a CSCL practitionner everyday's practice.

The use of communication technologies pervades higher education, especially distance education. Many virtual campus systems, such as Blackboard (2003) or WebCT (ref) include communication components that enable information exchange and online discussions, both synchronous and asynchronous. Teachers and online facilitators that understand these tools' affordances will be better prepared to use them to further students' learning. In the examples that we presented, we saw how even tools that do not serve to transmit information influence interactions among students, because they provide the external references that ground the dialogue.

We believe that it is preferable to start by establishing the learning task and environment and to choose or develop software to play a meaningful role that is compatible with the established goals. In situations where the software is already selected (for example by the institution) and used off-the-shelf, the software's features can guide the careful design of the learning situation and present an opportunity for pedagogical innovation.

In the section about *designing new affordances*, most tools were developed in a research context. Designing affordances means making critical decisions about the representation and degree of structure to impose on the learning situation.

Affordances do not magically work, a process of interpretation is necessary to fully take advantage of an artifact's affordances (Stahl, this volume; see also Kirschner, 2002). An affordance refers to the relationship between an object's physical properties and the characteristics of an actor (user) that enables particular interactions between actor and object (Kirschner, Strijbos, & Kreijns, 2003). These properties/artifacts interact with users and may provide strong clues about their operation. For example, Pea (1993) explains the concept of affordance by saying that "a door knob is for turning"; however, a door knob suits the "hang my vest" scheme as well as the "turning" scheme. In the design of educational technology, there is a limit to the extent to which the design of artifacts can shape the interaction. Because learners are not experts of the representations presented to them, it is

difficult to predict their behavior while using the artifacts. The effectiveness of a given representational structure lies both in its careful design, and then, in the guidance that follows, by the computer or a human coach.

The usage of graphical argumentation tools and structured dialogue interfaces requires some extra effort from the students (in comparison with plain email or chat). The students have to choose the type of message that corresponds to what they want to say before posting. Such a tool is probably less efficient in terms of raw productivity but sometimes more effective with regard to the quality of the messages produced.

Also, structuring the interface at the same time structures the data that is available for computational analysis of the interaction by the system. In the case study presented in the previous section, the system has access to the dialogue as a sequence of speech acts because they are selected by the students. It would be impractical or take much longer to identify the speech acts in dialogue if the structured interface was replaced by a simple chat or a telephone connexion.

Scripting is a common approach when interactive activity is mediated in person, for example when a group facilitator helps a large group plan future activities or conduct a self-assessment. Trained facilitators develop specific schemes that have proven successful in their past experience, and they then use these schemes in new situations to shape group activity, including learning activity. Generally, specific scripts work better than unfocused activity for a variety of learning situations. For example, programs for teaching children how to read that use strict scripting of the classroom process tend to result in improvements when used by teachers who are otherwise not systematic enough or well enough trained to figure out how to deal more specifically with the problems of particular classes or particular students. We would expect, similarly, that scripting would result in improvements when the baseline level of group activity is unfocused or disturbed by various pathologies (such as one speaker taking all of the group time). On the other hand, scripting is not specifically adaptive and hence should not be expected to work as well as approaches that are tailored to the specific situations occurring in a particular group or to the specific level of collaborative skill of a group's members.

Dillenbourg (2002) cautions us about the risk of over-scripting CSCL, by hard-coding the succession of activities and modes of interaction into the system. Customized scripts come with a steep increase in software development costs.

Regulating technologies aim to promote effective interaction by analysing the interaction, and using the results of the analysis to provide feedback to the students (Jermann, Soller, & Muehlenbrock, 2001). These tools provide various levels of support, along a continuum starting with data collection and reflection (phase 1 and 2), moving through various levels of complexity in diagnosing the interaction (phase 3), and ending with the recommendation of remedial actions (phase 4).

Guiding technologies differ most from structuring, and other regulating technologies such as mirroring and metacognitive tools, through the addition of their processing engines. The processing engine is intended to inform the decisions of the (human or computational) coach by analysing and evaluating the student interaction. The better

the processing engine is able to understand the interaction, the more informed it will be to advise the facilitator agent. Understanding student interaction, however, has been a major challenge in CSCL research, not unlike the challenge that understanding natural language poses to the field of artificial intelligence. Furthermore, correctly diagnosing the state of interaction does not guarantee that students will heed the advice the system proposes, and that this advice will have the intended effect (Crook, 1994). For these reasons, CSCL systems tend to provide minimal advice based on the analysis of student actions taken on workspaces, and statistics of messages that student send to each other (e.g. frequencies of different types of messages, such as Requests or Acknowledgements).

References

Baker, M., & Lund, K. (1997). Promoting reflective interactions in a computer-supported collaborative learning environment. *Journal of Computer Assisted Learning*, 13, 175-193.

Baker, M.J., de Vries, E., Lund, K. & Quignard, M. (2001). Computer-mediated epistemic interactions for co-constructing scientific notions: Lessons learned from a five-year research programme. In P. Dillenbourg, A. Eurelings & K. Hakkarainen (Eds.) *Proceedings of EuroCSCL 2001: European Perspectives on Computer-Supported Collaborative Learning* (pp. 89-96), Maastricht.

Barros, B., & Verdejo, M.F. (1999). An approach to analyse collaboration when shared structured workspaces are used for carrying out group learningprocesses. *Proceedings of the Ninth International Conference on Artificial Intelligence in Education*, 449-456.

Blackboard, Inc. (2003). Blackboard Learning and Community Portal Systems™ [On-line], Available: http://www.blackboard.com.

Brown, A., & Palincsar, A. (1989). Guided, cooperative learning and individual knowledge acquisition. In L. Resnick (Ed.), *Knowing, learning, and Instruction: Essays in honor of Robert Glaser*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Cahn, J. & Brennan, S (1999). A psychological model of grounding and repair in dialog. *Proceedings of the AAAI Fall Symposium: Psychological Models of Communication in Collaborative Systems*, Cape Cod, MA, 25-33.

Clark, H., & Schaeffer, E. (1989). Contributing to Discourse, *Cognitive Science*, 13, 259-294.

Constantino-Gonzalez, M. & Suthers, D. (2001). Coaching collaboration by comparing solutions and tracking participation. *Proceedings of Euro-CSCL 2001*, Maastricht, The Netherlands.

Craig, D. L., Haq, S., Khan, S., Zimring, C., Kehoe, C., Rick, J., & Guzdial, M. (2000). Using an unstructured collaboration tool to support peer interaction in large college classes. *Proceedings of ICLS 2000*, Ann Arbor, MI

Crook, C. (1994). Computers and the Collaborative Experience of Learning. London, Routledge.

De Jong, T., Ainsworth, S., Dobson, M., van der Hulst, A., Levonen, J., Reimann, P., Sime, J. -A., Van Someren, M. W., Spada, H., & Swaak, J. (1998). Acquiring knowledge in science and mathematics: the use of multiple representations in technology-based learning environments. In M. W. Van Someren, P. Reimann, H. P. A. Boshuizen, & T. de Jong (Eds.), *Learning with multiple representations* (pp. 9-40). Amsterdam: Pergamon.

Derry, S. J., Siegel, M., Stampen, J., & the STEP Research Group (2002). The STEP system for collaborative case-based teacher education: Design, evaluation & future directions. *Proceedings of the Computer Support for Collaborative Learning (CSCL) 2002 Conference*, Boulder, CO, 209-216.

Dillenbourg, P. (2002). Over-scripting CSCL: The risks of blending collaborative learning with instructional design. Paper presented at *Three Worlds of CSCL: Can We Support CSCL?* P. Kirschner, Inaugural Address, Open Universiteit Nederland.

Dillenbourg, P. & Schneider, D. (1995). Mediating the Mechanisms Which Make Collaborative Learning Sometimes Effective. International Journal of Educational Telecommunications, 1(2/3), 131-146.

Dillenbourg, P., Traum, D. & Schneider, D. (1996). Grounding in multi-modal task-oriented collaboration. In *Proceedings of the European Conference on AI in Education*, 1996.

Donath, J., Karahalios, K., & Viégas, F. (1999). Visualizing conversation. *Journal of Computer-Mediated Communication*, 4(4).

Fisher, L., & van Belle, G. (1993). *Biostatistics: A methodology for the health sciences*. New York: John Wiley & Sons, Inc.

Gaver, W. (1991). Technological Affordances. *Proceedings of the CHI'91 Conference on Computer and Human Interaction* (pp. 79-84). NY: ACM, 1991,.

Gibson, J. J. (1979). The Ecological Approach to Visual Perception. Boston: Houghton Mifflin.

Guzdial, M., Realff, M., Ludovice, P., Morley, T., Kerce, C., Lyons, E., et al. (1999). Using a CSCL-driven shift in agency to undertake educational reform. In C. M. Hoadley & J. Roschelle (Eds.), *Proceedings of the Computer Support for Collaborative Learning (CSCL) 1999 Conference* (pp. 211-217). Mahwah, NJ: Lawrence Erlbaum.

Hutchins, E. (1991). The social organization of distributed cognition. In L. B. Resnick , J. M. Levine & S.D. Teasley (Eds.), *Perspectives on Socially Shared Cognition*.

Jeong, H. (1998). *Knowledge co-construction during collaborative learning*. Unpublished Doctoral Dissertation. University of Pittsburgh, PA.

Jermann, P., & Schneider, D. (1997). Semi-structured interface in collaborative problem solving. *Proceedings of the First Swiss Workshop on Distributed and Parallel Systems*, Lausanne, Switzerland.

Jermann, P., Dillenbourg, P., & Brouze, J. C. (1999). Dialectics for collective activities: an approach to virtual campus design. In S. P. Lajoie and M. Vivet (Eds.) *Artificial Intelligence in Education* (pp. 570-577). IOS Press.

Jermann, P., Soller, A., & Muehlenbrock, M. (2001). From mirroring to guiding: A review of state of the art technology for supporting collaborative learning. *Proceedings of the First European Conference on Computer-Supported Collaborative Learning*, Maastricht, The Netherlands, 324-331.

Jonassen, D.H. (1992). What are Cognitive Tools ? In P.A.M. Kommers & al. (Eds.), Cognitive Tools for learning, NATO ASI Series. Berlin : Springer.

Kirschner, P. A. (2002). Cognitive load theory. *Learning and Instruction, 12* (1), 1-10.

Kirschner, P. A., Strijbos, J. W., & Kreijns, K. (2003). Designing integrated electronic collaborative learning environments. In W. M. G. Jochems, & J. J. G. van Merriënboer (Eds.), Integrated e-learning.

Kirschner, P. A., & Wopereis, I. G. J. H. (in press). Mindtools for teacher communities: A European perspective. *Technology, Pedagogy and Education, 12*(1).

Lave, J., &Wenger, E. (1991). *Situated Learning: Legitimate Peripheral Participation.* Cambridge: Cambridge University Press.

Leontiev, A.N. (1981). The problem of activity in psychology. In J.V. Wertsch (Ed.), The concept of activity in Soviet psychology. Armonk, NY: Sharpe.

McGrenere J., Ho W. (2000). Affordances: Clarifying and Evolving a Concept. *Proceedings of Graphics Interface.* (http://www.graphicsinterface.org/proceedings/2000/177/)

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

Moore, M. G. (1989) Three Types of Interaction. The American Journal of Distance Education, 3(2).

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

Moreno, J. L. (1951). *Sociometry, experimental method and the science of society: An approach to a new political orientation.* Ambler, PA: Beacon House, Inc./Horsham Foundation.

Nardi, B., ed. (1996). Context and Consciousness: Activity Theory and Human-Computer Interaction, Cambridge: MIT Press.

Norman, D. A. (1988). The Psychology of Everyday Things. New York: Basic Books.

Ngwenyama, O., & Lyytinen, K. (1997). Groupware Environments as Action Constitutive Resources: A Social Action Framework for Analysing Groupware Technologies. Computer Supported Collaborative Work: The Journal of Collaborative Computing, 6, 71-93.

Ogata, H., Matsuura, K., & 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.

O'Donnel, A.M., & Dansereau (1992). Scripted cooperation in student dyads: A method for analyzing and enhancing academic learning and performance. In R. Hertz-Lazarowitz and N. Miller (Eds.), Interaction in cooperative groups: The theoretical anatomy of group learning (pp. 120-141). London, Cambridge University Press.

Pea, R. (1993). Practices of Distributed Intelligence and Designs for Education. In G. Salomon (Ed.), *Distributed cognitions: Psychological and educational considerations*. Cambridge: Cambridge University Press.

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.

Rabiner, L. (1989). A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2), 257-286.

Resnick L.B., J. M. Levine & S.D. Teasley (Eds.) (1991) - Perspectives on Socially Shared Cognition. Washington: American Psychological Association.

Robertson, J., Good, J. & Pain, H. (1998). BetterBlether: The Design and Evaluation of a Discussion Tool for Education. *International Journal of Artificial Intelligence in Education*, 9, 219-236.

Roschelle, J. & Teasley, S. (1995). The construction of shared knowledge in collaborative problem solving. In O'Malley, C.E., (ed.), Computer Supported Collaborative Learning. pages 69--97. Springer-Verlag, Heidelberg.

Salomon, G. (Ed., 1993) Distributed cognitions. New York: Cambridge University Press.

Scardamalia, M., & Bereiter, C. (1991). Higher levels of agency for children in knowledge-building: A challenge for the design of new knowledge media. *Journal of the Learning Sciences*, *1*(1), 37-68.

Scott, J. (1991). *Social Network Analysis. A Handbook (2nd edition).* London: Sage Publications.

Shepard, R., & Arabie, P. (1979). Additive Clustering: Representation of similarities as combinations of discrete overlapping properties. *Psychological Review*, 86(2), 87-123.

Simoff, S. (1999). Monitoring and Evaluation in Collaborative Learning Environments. *Proceedings of Computer Support for Collaborative Learning (CSCL) 1999*, Palo Alto, CA: Stanford University.

Soller, A. (2002). *Computational analysis of knowledge sharing in collaborative distance learning*. Unpublished Doctoral Dissertation. University of Pittsburgh, PA.

Soller, A., & Lesgold, A. (in press). Modeling the process of knowledge sharing. In U. Hoppe & M. Ikeda (Eds.) *New Technologies for Collaborative Learning*. Kluwer Academic Publishers.

Soller, A., Goodman, B., Linton, F., & Gaimari, R. (1998). Promoting Effective Peer Interaction in an Intelligent Collaborative Learning Environment. *Proceedings of the Fourth International Conference on Intelligent Tutoring Systems (ITS 98)*, San Antonio, TX, 186-195.

Steinkuehler, C.A., Derry, S.J., Woods, D.K. & Hmelo-Silver, C.E. (2002). The STEP Environment for Distributed Problem-Based Learning on the World Wide Web. *Proceedings of Computer Support for Collaborative Learning* 2002, Boulder, CO.

Stenning, K., & Oberlander, J. (1995). A cognitive theory of graphical and linguistic reasoning: logic and implementation. *Cognitive Science*, 19, 97-140.

Stahl, G. (2001). WebGuide: Guiding collaborative learning on the web with perspectives. *Journal of Interactive Media in Education.* Retrieved February 8, 2003, from http://www-jime.open.ac.uk/2001/1/stahl-01-1.pdf

Stolcke, A., Ries, K., Coccaro, N., Shriberg, E., Bates, R., Jurafsky, D., Taylor, P., Martin, R., Van Ess-Dykema, C., & Meteer, M. (2000). Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26(3).

Suthers, D., Weiner, A., Connelly, J., & Paolucci, M. (1995). Belvedere: Engaging students in critical discussion of science and public policy issues. *Proceedings of the 7th World Conference on Artificial Intelligence in Education*, Washington D.C., 266-273.

Suthers, D. (2001). *Towards a Systematic Study of Representational Guidance for Collaborative Learning Discourse*. Journal of Universal Computer Science, *7*(3), 254-277.

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

Tedesco, P., & Self, J. (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.

Van Bruggen, J. M., Boshuizen, H. P. A., & Kirschner, P. A. (2003). A framework for cooperative problem solving with argument. In P. A. Kirschner, S. J. Buckingham Shum, & C. S. Carr (Eds.), *Visualizing Argumentation: Software Tools for and Educational Sense-Making* (pp. 25-47). London: Springer-Verlag.

Van Bruggen, J. M., Kirschner, P. A., & Jochems, W. (2002), *External representations of argumentation in CSCL and the management of cognitive load. Learning and Instruction*, *12* (1), 121-138.

Vizcaíno, A. (2001). Negative situations in collaborative environments: Can a simulated student avoid them? *Proceedings of AIED'01, The 10th International Conference on Artificial Intelligence in Education*, San Antonio, Texas, 610-612.

Vygotsky, L. (1978). *Mind in Society: The development of higher psychological processes*. Harvard University Press. Cambridge, MA. Collection of work edited by Michael Cole, Vera John-Steiner, Sylvia Scribner, & Ellen Souberman.

Webb, N. (1992). Testing a theoretical model of student interaction and learning in small groups. In R. Hertz-Lazarowitz and N. Miller (Eds.), *Interaction in Cooperative Groups: The Theoretical Anatomy of Group Learning* (pp. 102-119). New York: Cambridge University Press.

Webb, N., & Palincsar, A. (1996). Group processes in the classroom. In D. Berlmer & R. Calfee (Eds.), *Handbook of Educational Psychology* (pp. 841-873). New York: Simon & Schuster Macmillan.

Webb, N., Troper, J., & Fall, R. (1995). Constructive activity and learning in collaborative small groups. *Journal of Educational Psychology*, *87*(3), 406-423.

Winquist, J. R. & Larson, J. R. (1998). Information pooling: When it impacts group decision making. *Journal of Personality and Social Psychology*, *74*, 371-377.

Wood, C.C. (1993). A cognitive dimensional analysis of idea sketches. *Cognitive Science Research Paper* CSRP 275, The University of Sussex, School of cognitive and computing Sciences, Falmer, Brighton.

Zimring, C., Khan, S., Craig, D., Haq, S., & Guzdial, M. (2001). *CoOL Studio: Using Simple Tools to Expand the Discursive Space of the Design Studio*. Automation in Construction , *10*(6), 675-685.

Zumbach, J., Muehlenbrock, M., Jansen, M., Reimann, P., & Hoppe, H. U. (2002). Multidimensional Tracking in Virtual Learning Teams. In G. Stahl (Ed.), *Computer Support for Collaborative Learning: Foundations for a CSCL Community* (pp. 650-651). Hillsdale, NJ: Erlbaum.