Applying the 3C Model to Groupware Development

HUGO FUKS[1], ALBERTO B. RAPOSO[2], MARCO A. GEROSA[1] & CARLOS J. P. LUCENA[1]
[1] Software Engineering Laboratory (LES) – Computer Science Department
Catholic University of Rio de Janeiro (PUC-Rio)
R. Marquês de São Vicente, 225, Gávea, Rio de Janeiro – RJ, 22453-900, Brazil
{hugo, gerosa, lucena}@inf.puc-rio.br
Phone: +55-21-3114-1500 x4304 Fax: +55-21-3114-1530

[2] Computer Graphics Group (Tecgraf) – Computer Science Department
Catholic University of Rio de Janeiro (PUC-Rio)
R. Marquês de São Vicente, 225, Gávea, Rio de Janeiro – RJ, 22453-900, Brazil
abraposo@tecgraf.puc-rio.br
Phone: +55-21-2512-5984 Fax: +55-21-3114-1848

Abstract. This paper introduces an approach based on the 3C (communication, coordination and cooperation) collaboration model to the development of collaborative systems. The 3C model is studied by means of a detailed analysis of each of its three elements, followed by a case study of a learningware application and the methodology of a web-based course, both designed based on this model. Moreover, this paper describes a component-based system architecture following this 3C approach.

Keywords: groupware development, collaboration, communication, coordination, cooperation.

1. INTRODUCTION

Collaboration may be seen as the combination of communication, coordination and cooperation. Communication is related to the exchange of messages and information among people; coordination is related to the management of people, their activities and resources; and cooperation, which is the production taking place on a shared space. This model, which we call *3C model*, was originally proposed by Ellis et al. [1991], with some terminological differences. Cooperation, which Ellis denominates "collaboration", here characterizes a joint operation in a shared space.

The 3C model appears frequently in the literature as a means to classify collaborative systems, e.g. as done by Borghoff & Schlichter [2000]. In the present paper, we explore the 3C model, as a means to represent a groupware application domain and also to serve as a basis for groupware development.

1.1. Instantiating the 3C Model

The relationship among the 3Cs of the model is a guidance to understand a groupware application domain. For example, one may consider Media Spaces [Mackay, 1999], which are multimedia-enhanced spaces aimed at informal communication among people. In the Media Spaces domain, the 3C model may be instantiated according to Figure 1.

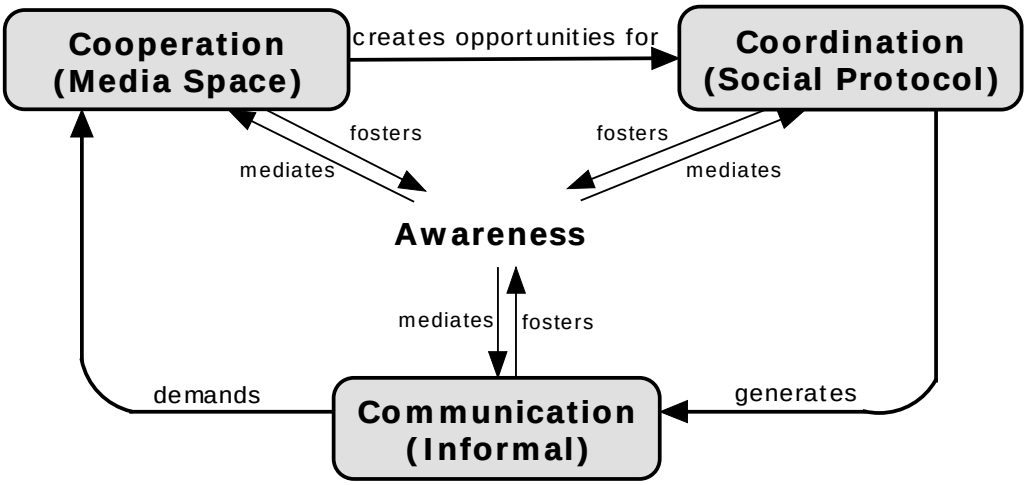

Figure 1. 3C collaboration model instantiated for the Media Space domain

The Media Space itself is the shared space. Since it is aimed at informal communication, its main goal is actually to create opportunities for informal meetings, which are coordinated by the standing social protocol, for example, by accessing the availability of remote colleagues. These meetings generate conversation, which may occur using the media provided by the system or any other available means, such as telephones.

In this paper, we are focused on another groupware domain; group work, represented in Figure 2. According to this instantiation of the 3C model, while communicating, people negotiate and make decisions. While coordinating themselves, they deal with conflicts and organize their activities in a manner that prevents loss of communication and of cooperation efforts. Cooperation is the joint operation of members of the group in a shared space, seeking to execute tasks, generating and manipulating cooperation objects. The need for renegotiating and for making decisions about non-expected situations that appear during cooperation may demand a new round of communication, which will require coordination to reorganize the tasks to be executed during cooperation.

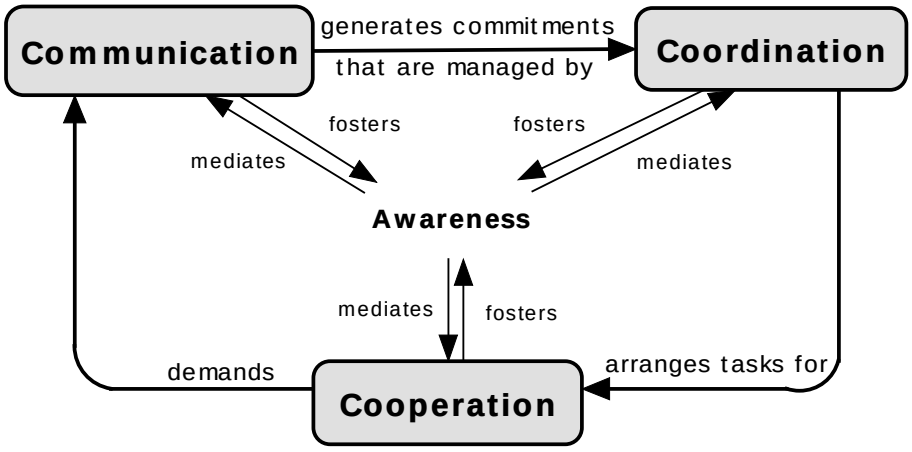

Figure 2. 3C collaboration model instantiated for group work

This cycle shows the iterative nature of collaboration. The participants obtain feedback from their actions and feedthrough from the actions of their companions by means of awareness information related to the interaction among participants [Gerosa et al., 2003].

1.2. Groupware Development

Although the 3C model is commonly used for classifying groupware, a few attempts have been made to use it in the context of groupware implementation. A notable example is the Clover design

model, which defines three classes of functionalities, namely communication, coordination and production (cooperation in the present paper) [Laurillau & Nigay, 2002]. These three classes of services appear in each functional layer of the model and, during the system design phase, they "must be identified and their access harmoniously combined in the user interface".

The Clover model shares the same usefulness of the 3C model in terms of groupware functional specification, because both deal with the three classes of functionalities that a groupware application may support. However, differently from the Clover model, the 3C model is here mapped to a groupware component-system architecture (Section 6).

In this paper, we show how the 3C model has been applied to the development of the AulaNet environment and to the dynamics of a course entitled Information Technology Applied to Education (ITAE), currently in its fourteenth edition. In the Section 2 of this paper, AulaNet and the ITAE course are introduced. The following sections detail each aspect of the 3C collaboration model, namely communication (Section 3), coordination (Section 4) and cooperation (Section 5), using AulaNet and the ITAE course as case study. In Section 6, design and implementation issues are discussed by means of the proposal of a component-based architecture. Finally, Section 7 concludes this paper.

2. THE AULANET LEARNING ENVIRONMENT

The manner in which people work has changed with the advent of the connected society. Accustomed to the paradigm of command and control that is taught in classrooms and widely disseminated on factory floors—or, rather, conditioned by it—, workers are not up to the new demands of the connected society. They are trained to react to clear orders, well-defined procedures and specific activities of individual preference. Their understanding of communication is vertical—memorandums come down from above and reports are sent up the line. Thus, as in a classroom, horizontal communication—communication with a shift colleague—besides being hardly well thought of is also given no technological support. Knowledge workers, on the other hand, constantly interact with their colleagues in order to carry out their tasks, creating a situation where workspaces and learning spaces converge. The organization that was imposed top-down in the command and control paradigm loses effectiveness and is replaced by one that is peer-to-peer like, where communication, coordination and cooperation predominate.

AulaNet is a freeware web-based environment for teaching and learning. It has been under development since June 1997 by the Software Engineering Laboratory of the Catholic University of Rio de Janeiro (PUC-Rio). Besides Portuguese, AulaNet is also available for download (http://www.eduweb.com.br) in English and Spanish versions.

In its first versions, AulaNet resources were subdivided into administrative, assessment and didactic services, a common approach in educational tools [Edutools, 2005]. Unfortunately, this approach led teachers who were using the environment to teach in the traditional vertical way: broadcasting information with a low degree of learner-teacher interaction and no interaction among learners at all. Collaborative learners are expected to have a high degree of interaction among themselves and with their teachers, who are supposed to act as coordinators or mediators rather than as information deliverers. Hence, the services were reorganized based on the 3C collaboration model, which is suitable to a collaborative learning approach [Fuks, 2000].

The AulaNet environment's services are currently subdivided into communication, coordination and cooperation services, as can be seen in Figure 3. The communication services provide tools for forum-style asynchronous text discussion (*Conferences*), chat-style synchronous text discussion (*Debate*), instant message exchange between simultaneously connected learners (*Instant Messaging*) and individual electronic mail with the mediators (*Message to Participants*) and with the whole class, in a list-server style (*Message to the Class*).

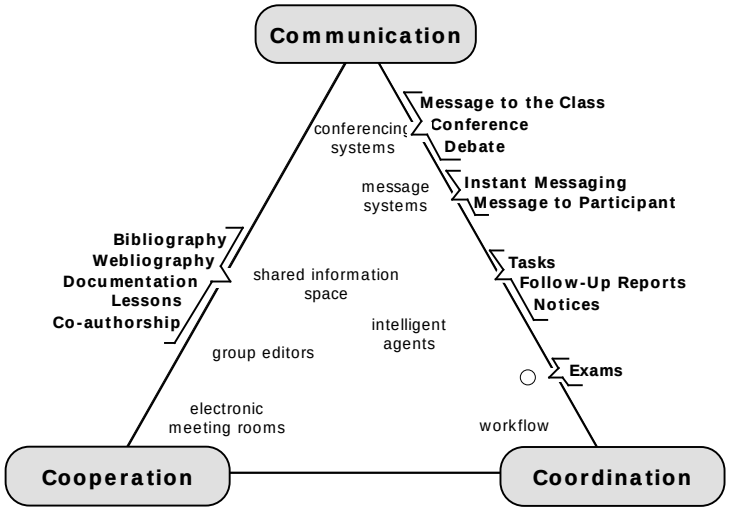

Figure 3. Classification of AulaNet services based on the 3C Model.
The 3C triangle appears in Borghoff & Schlichter [2000]

Coordination services support the management and the enforcement of group activities. In AulaNet, coordination services include tools for notification (*Notices*), evaluation (*Tasks* and *Exams*) as well as a tool that allows monitoring group participation (*Follow-Up Reports*). Cooperation services in AulaNet include *Lessons* and *Documentation*, a list of course references (*Bibliography* and *Webliography*) and course co-authoring support, both for teachers (*Teacher Co-Authoring*) and for learners (*Learner Co-Authoring*).

The Information Technology Applied to Education (ITAE) course has been taught since 1998 as one of the courses of the Computer Science Department at PUC-Rio entirely online, using the AulaNet environment.

The course's methodology was envisaged to change the behavior of students who used to be passive receivers into learners who actively generate knowledge. This process seeks to lead learners to look for their own sources of information, to deal with information overload and to collaboratively turn information into knowledge. Learners are graded for their contributions that add value to the group and not only for their individual activities [Fuks et al., 2002].

In the first part of the course, learners study the contents according to the weekly topic, watching the contents present in the course repository, and discussing the topic asynchronously in the AulaNet *Conference* service and synchronously in the *Debate* service (Figure 4). In the ITAE course, learners conduct the discussion in these services. Learners take turns performing the Conference Leader and Debate Moderator roles throughout the course. All learners are expected to contribute, discussing arguments with their colleagues, developing and refining new concepts, and having their work observed, commented upon and evaluated by their peers [Benbunan-Fich & Hiltz, 1999].

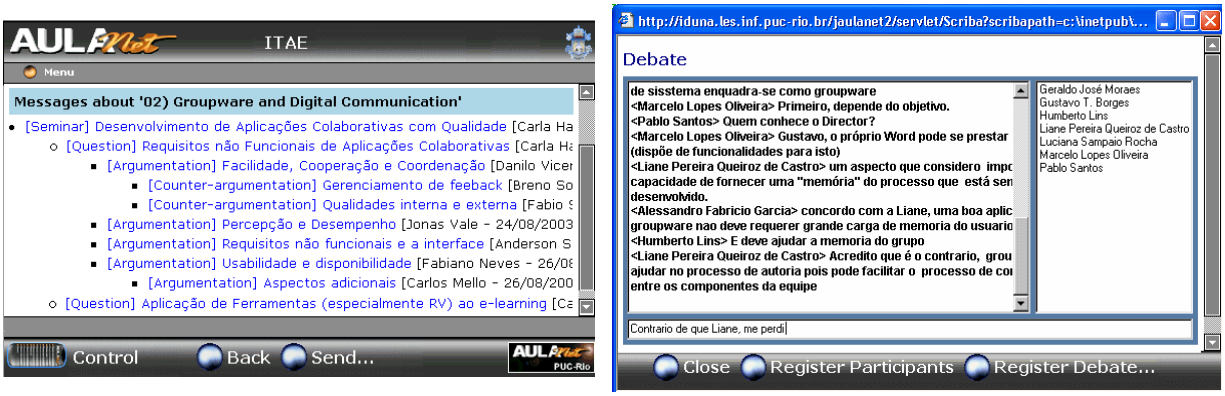

Figure 4. Portion of dialogue from the (a) *Conference* and (b) *Debate* services.

In the second part of the course, learners develop educational multimedia interactive contents, working in small groups of two or three that are formed according to the profiles they previously submitted. Based on this information, AulaNet suggests group formations that best satisfy the criteria defined by mediators (degree of skill, interest and competency) [Cunha et al., 2003]. After the initial submission, a period of collaborative peer review begins. Members of at least three other groups evaluate each group's content. This evaluation takes place asynchronously in *Conferences* created specifically for this purpose, where learners discuss problems they found in the prototypes. Once this period is over, groups are given a new deadline to present a revised version that incorporates the contributions of their colleagues.

Further detail about the course and the AulaNet environment is given in the following sections based on a case study for communication, coordination and cooperation.

3. COMMUNICATION: ARGUMENTATION FOR ACTION

In the command and control paradigm, communication is considered successful when the sender is informed that the receiver received the message. On the other hand, the success of communication in the collaboration paradigm entails the understanding of the message by the receiver. The only way to obtain indications about the receiver's understanding is by observing his actions and reactions, since they are guided by commitments assumed during communication. The receiver reads the message and interprets it, changing his knowledge and commitments in a certain way. That will prompt him to reason about the new acquired knowledge and react. This way, sender and receiver move into an argumentation process in which they reason about their actions [Schön, 1983].

The designer of a communication tool defines the communication elements that will define the communication channel between the interlocutors, taking into consideration the specific usage that is being planned for the tool (time, space, purpose, dynamics and types of participants) and other factors such as privacy, development and execution restrictions, information overload, etc. Then, these elements are mapped onto software components that provide support to the specific needs. These elements are presented below and, in Section 6, their corresponding software components are presented.

The first communication element that must be considered is the choice of **media**. They can be textual, spoken, pictorial (when images are used as part of the discourse, as it occurs with emoticons) or gestured, e.g., in a video or avatar form. The media adopted restrict and influence the vocabulary used in the conversation, which is also influenced by the context (e.g., the participants' culture, language and background).

The **transmission mode** defines whether the information is transmitted in blocks or continuously. In an audio or videoconference and in some chat tools (e.g., Unix talk) the information is transmitted continuously as it is generated. In asynchronous and in other chat and messenger tools the information

is transmitted in blocks: the author edits the message and it is only sent with an explicit command. Asynchronous communication tools are used when one wants to enhance reflection by the interlocutors, since they will have more time before they act, while synchronous tools are used for communication bursts. **Restrictions policy** is another communication element. For example, it is possible to restrict the text's size, characters allowed, bit rate (for video and audio), allowed vocabulary, etc. These restrictions are used in some tools to reduce information overload and to save bandwidth.

Meta-information complements the information being transmitted in the body of the message. Common meta-information available in communication tools are the subject of the message, its date, priority and category. **Category** can be used to define the semantics of the conversation. The sender selects a category from a pre-defined set when posting a message. The semantics of the categories should be known by all interlocutors as a means to enforce dialog dynamics, detect conflicts, identify task resolution and organize information [Gerosa et al., 2001].

Conversation structure defines how messages are structured, which can be in a linear, hierarchical or network form. The conversation structure makes the relations between messages, which are usually implicit within the text, visually explicit, as can be seen in Figure 5.

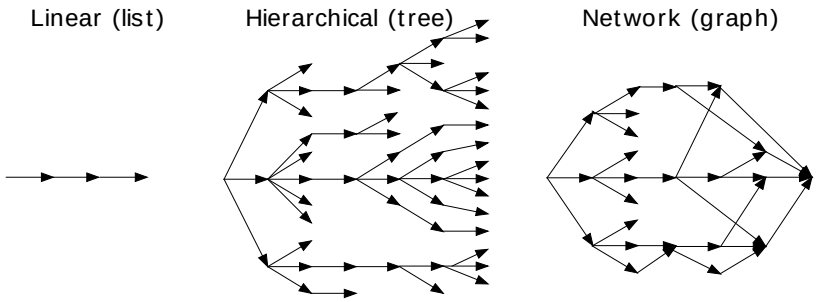

Figure 5. Examples of conversation structure

The linear form is used when there are not so many message interconnections, the chronological order of the messages is relevant and fluency in the conversation is desired [Kirschner et al, 2003]. Hierarchical structuring is appropriate when the relationships between messages, such as questions and answers, need to be quickly identified. However, as there is no way to link messages from two different branches, the tree can only grow wide and, thus, the discussion takes place in diverging lines [Stahl, 2001]. Network structuring can be used to seek convergence in the discussion.

Conversation paths can be used to restrict the possible directions the conversation can take. An example of a communication tool with conversation paths is the Coordinator [Winograd & Flores, 1987], in which some conversation stages and paths are defined. Conversation can only flow through the pre-defined paths. Conversation paths formalize the conversation and it is not recommended when fluency is desired.

In order to provide proper support to communication, the designer should also take into account coordination and cooperation elements. Coordination elements deal with access policies to the communication channel, while cooperation elements deal with information rendering and registration. These elements are discussed in Sections 4 and 5. In the following section, communication elements are illustrated within the AulaNet environment.

3.1. Communication in the ITAE course

In a learning situation it is particularly important to align the arguments of the learners in order to promote a common body of knowledge. During the argumentation process, learners have to attack and defend concepts, and collect and validate information that supports or goes against them. In the ITAE course ideas, viewpoints and arguments are expressed and understood without necessarily reaching a

single solution for the questions or arriving at agreement or consensus. It is expected that the learners' arguments differ and that the learners are mature enough to accept the positions and the arguments of others, learning something from them. In the ITAE, argumentation that is generated from the confrontation between different ideas is valued. It is expected that learning derives from these arguments and the respective aligning of ideas and not from harmonization and consensus.

Typical workers were not taught to deal with the variety of media available in the Internet and normally use written language to communicate and share information. In order to focus their attention on the content of the *Conference* discussion, only textual **media** is used. Moreover, in the last part of the course, while preparing the interactive multimedia educational content, learners act as designers, writers, actors, directors, cameramen etc.

The *Conference* service, which is an asynchronous communication service, uses a hierarchical **conversation structure** and is used in ITAE to discuss the course's subjects in greater depth. The hierarchical structuring of the messages makes it possible to organize the discussion in threads, thus preventing a message from one sub-topic from mixing with those of another one.

The *Conference* service uses subject, date and category as **meta-information**. The author of the message chooses the **category** that is most appropriate according to the content being developed, providing a semantic aspect to the relationship between messages. The categories adopted in the ITAE conferences were originally based on the IBIS' node types [Conklin, 1988]. Currently, the categories being used are: Seminar, for the root message of the discussion; Question, Argumentation, Counter-Argumentation and Clarification. Figure 6 presents a portion of a dialogue from a conference showing numbered messages mapped to a tree.

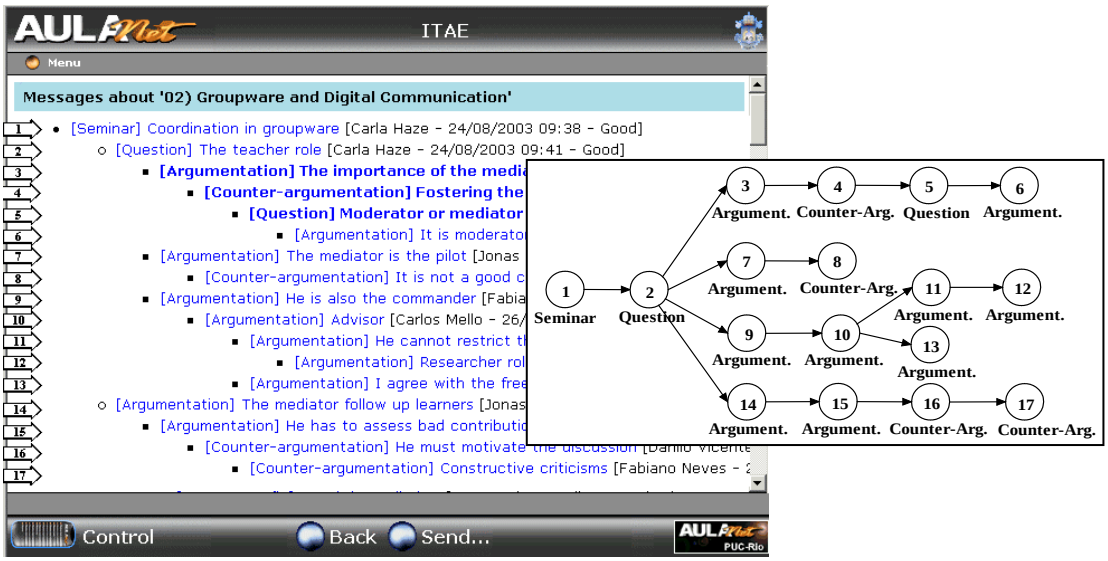

Figure 6. Tree derived from a conference with the message categories.

Categories clearly help to understand the relation between messages without having to inspect their content, thus complementing the information provided by the message structure and helping to identify the direction that the discussion is taking. For example, in a tree or a branch that only contains argumentation messages, there is no confrontation of ideas taking place, while an excessive number of counter-argumentations may indicate that the group has got into a deadlock or there may be interpersonal conflicts taking place [Gerosa et al., 2004].

The hierarchical structure is also useful to provide indications about the evolution of the class and to identify discussions that moved out from the expected pattern. For example, in the resulting trees presented in Figure 7, it is possible to observe the declining interaction in the 2002.1 edition of the ITAE course. In the first four conferences of this edition, the average level of the tree was 3.0 and the percentage of unanswered messages (leaves) was 51%; in the last four conferences, the average tree

level was 2.8 and the number of leaves was 61%. This illustrates that the conversation structure is useful to detect undesired characteristics, such as the low level of the trees and high number of leaves, which, in this case, indicate a low level of interaction among learners.

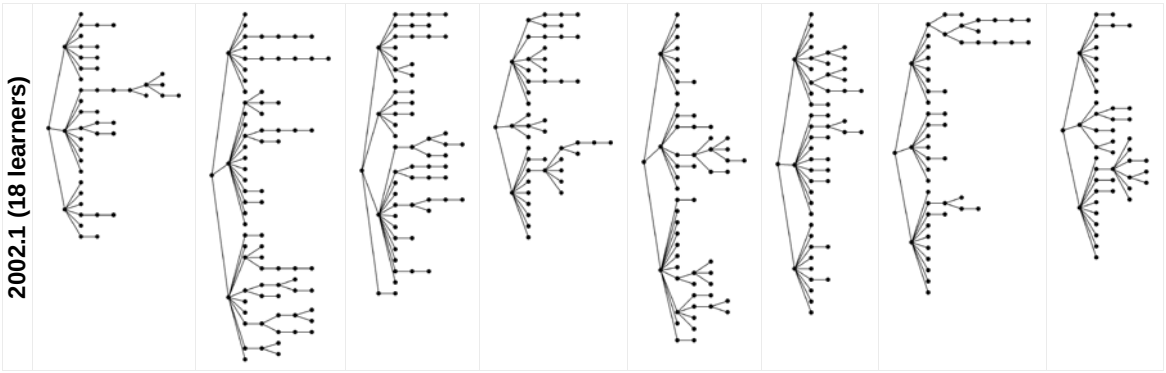

Figure 7. Conversation structure of some ITAE conferences.

Table 1 summarizes the mapping of the communication elements used in the AulaNet communication services.

Communication Element	Service				
	Message to Participants	**Message to the Class**	**Conference**	**Debate**	**Instant Messaging**
Media	Textual	Textual	Textual	Textual	Textual
Conversation structure	Hierarchical	Linear	Hierarchical	Linear	Linear
Category	-	Available	Available	-	-
Meta-information	Subject, date	Subject, date	Subject, date	Time	-
Conversation paths	-	-	-	-	-
Transmission mode	In blocks	In blocks	In blocks	In blocks	In blocks
Restrictions policy	-	-	-	-	-

Table 1. Communication elements used in the AulaNet communication services.

In the last part of the course, the *Conference* service is used to peer-evaluate the prototypes produced by each group. Learners negotiate among themselves the revisions that are necessary to the prototypes and then each group produces a final version taking into consideration the commitments assumed. *Message to Participant*, *Message to the Class* and *Instant Messaging* are communication services used with the purpose of coordinating people.

4. COORDINATION: ACTION PLANNING

Coordination may be viewed as the link connecting the other two C's in order to enforce the success of collaboration. This is more clearly observed when we analyze the elements that need to be coordinated, namely people, resources and tasks. The coordination of people, for example, is deeply related to communication and context. The coordination of resources, on the other hand, is related to the shared space (i.e., cooperation). For this reason, coordination aspects also appear in the discussion about the other C's of the model. In this section, we focus on the coordination of tasks.

Coordination involves the pre-articulation of the tasks, their management and post-articulation. Pre-articulation includes actions that are necessary to prepare collaboration, which are usually concluded before cooperation begins, such as the identification of goals, the mapping of these goals into tasks, the selection of participants and the distribution of tasks among them. The post-articulation stage occurs after the end of the tasks and involves the evaluation and analysis of tasks and the documentation of the collaborative process. The management of the tasks being carried out consists in

managing interdependencies between tasks that are carried out to achieve a goal [Malone & Crowston, 1990].

Some computer-supported collaborative activities, the so-called *loosely integrated* collaborative activities, such as those in chats or audio and videoconferences, are deeply associated with social relations and are generally satisfactorily coordinated by the standing social protocol, which is characterized by the absence of any computer-supported coordination mechanism—"a specialized software device, which interacts with a specific software application so as to support articulation work" [Schmidt & Simone, 1996]—among the tasks, trusting the users' abilities to mediate interactions. Coordination, in these situations, is contextually established and strongly dependent on mutual awareness. Through awareness information, the participants detect changes in plans and understand how the work of their colleagues is getting along: what was done, how it was done, what needs to be done until it is finished, what are the preliminary results, etc. [Gutwin & Greenberg, 2002] [Dourish & Belloti, 1992].

However, there are also the so-called *tightly integrated* collaborative activities, whose tasks are highly interdependent, as the name suggests. They require sophisticated coordination mechanisms in order to be supported by computer systems. The great challenge in designing coordination mechanisms in groupware is to achieve flexibility without losing the regulation, which is necessary in some situations in which the social protocol is not enough. The system should not impose rigid work or communication patterns, but rather offer the user the possibility to use, alter, or simply ignore them. Thus, coordination flexibility and accessibility should be pursued by groupware designers. Flexibility is related to the possibility of dynamically allowing redefinition and temporary modifications in the coordination scheme. Accessibility is related to exposing the coordination mechanisms to system users rather than having them deeply embedded in the system's implementation.

Coordination can take place on the temporal and on the object levels [Ellis & Wainer, 1994]. On the temporal level, coordination defines the sequence of tasks that makes up an activity. On the object level, coordination describes how to handle the sequential or simultaneous access of multiple participants through the same set of cooperation objects.

Temporal interdependencies establish the relative order of execution between a given pair of tasks. In the model used in this paper, the set of temporal interdependencies extends the temporal relations defined by J. F. Allen [1984]. A task's time interval is characterized by two events, which in turn are associated with time instants. The first event is the initial time of an interval A, denoted *ia*, and the other event is the final time of the same interval, denoted *fa*, always with $ia<fa$. To avoid different interpretations of a single interdependency in Allen's model, some additional operators were defined [Raposo & Fuks, 2002].

The first adaptation deals with active and passive interpretations by means of two operators: *enables* and *forces*. Suppose, for example, that an interdependency establishes that tasks A and B start and finish at the same instant, and task A is ready to begin. This situation may be interpreted in two different ways. In the passive interpretation, the execution of task A is blocked until task B is ready. In the active interpretation, the beginning of task A forces the start of task B to ensure that the interdependency will be observed. The *enables* operator represents the passive interpretation, while *forces* represents the active one. These operators may be applied to the initial and final instants of each interdependent task.

In a different situation, suppose that task A occurs before task B, and task B is ready but task A is not. The coordination mechanism may block task B until the end of task A, or it may allow the execution of task B, blocking future executions of task A, which would violate the relation. To deal with this situation, it was necessary to create the *blocks* and *unblocks* operators. For example, the relation *ib blocks ia* imposes a restriction to the execution of task A, which may no longer be executed if task B has already started its execution.

4.1. Coordination in the ITAE course

ITAE's learning activities exemplify a typical situation that requires pre- and post-articulation to define and refine the coordination itself. ITAE has been continuously evolving based on the feedback provided in previous editions. Learning activities that had been planned in advance for an edition were analyzed and afterwards reformulated in the light of the results obtained in that edition.

Figure 8 shows the sequence of tasks planned for the ITAE course. After the initial introductions, there are eight topics for content studies, each of them comprising content reading, asynchronous conference and synchronous debate. Finally, there is the content-generation activity, which is also subdivided into three sub-tasks, and the course finalization, when the final grade is announced.

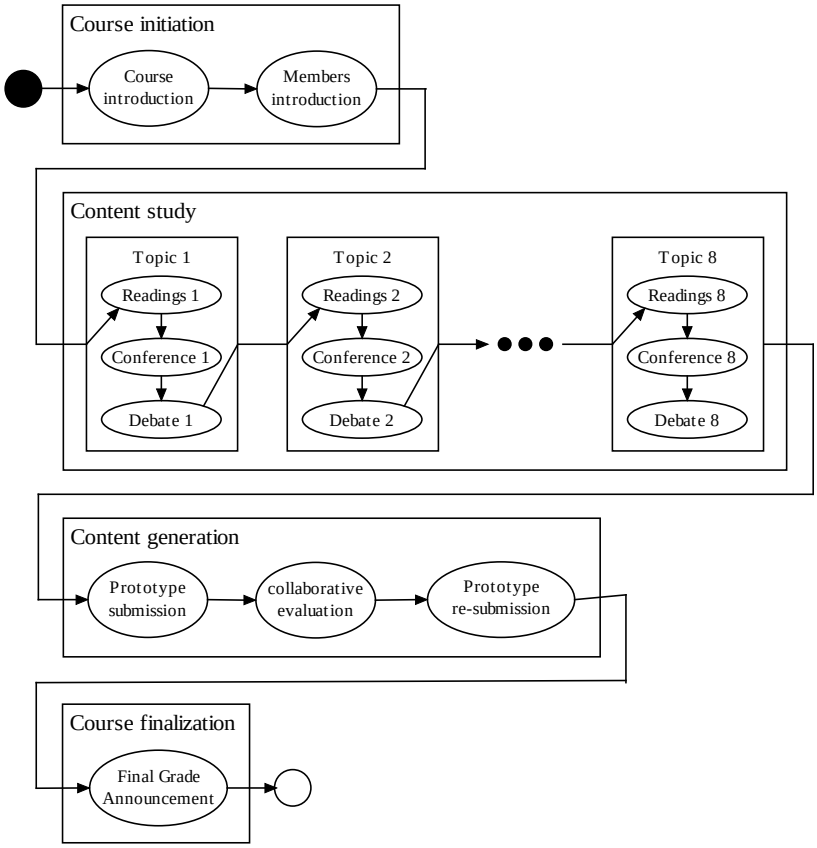

Figure 8. ITAE flow of activities

Figure 9 presents the expanded coordination protocol for the weekly conference. This collaborative learning activity involves three roles. The mediators select the learner who will be the Conference Leader during that week and initializes the conference session. The Leader must then submit the seminar message to the conference and post a number of questions for discussion. Learners post messages developing an argumentation about the questions proposed. The mediators, who also finalize the session, evaluate each one of the messages.

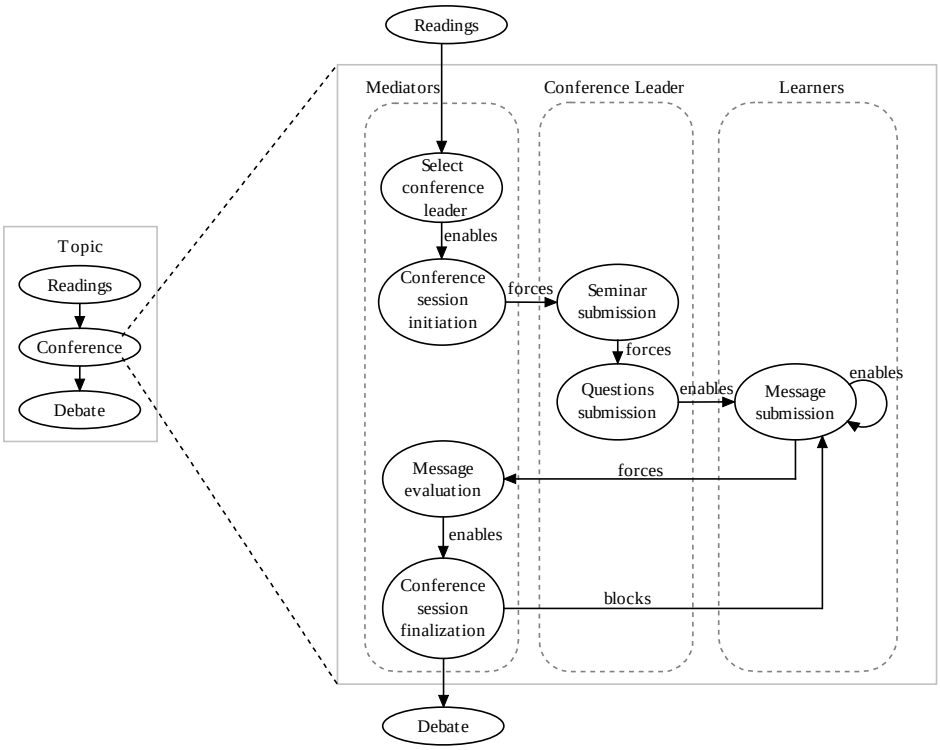

Figure 9. ITAE conference coordination protocol

The interdependencies among the tasks presented in Figure 9 are expressed by the *enables*, *forces*, and *blocks* operators. The selection of the Conference Leader, for instance, enables the session's initialization, meaning that the mediators cannot initialize a session without previously selecting a Leader; however, the selection of the Leader does not force the mediators to initialize a session. The same relation takes place between the Leader's question submission and learners' message submission. Learners are not able to submit messages before the Leader's questions, but these questions do not force learners to submit messages. Although the non-participation of a learner may have a negative impact on her degree, it does not necessarily harm the procedural flow of the conference. The *forces* operator is used, for example, to ensure that the session's initialization by the mediators force the Leader to submit her seminar, followed by questions. The *blocks* operator is used in Figure 9 indicating that learners cannot submit messages after the mediators finalize the conference session.

The tasks that learners have to perform during the conference are reading and posting messages that constitute arguments, answer an argument or prepare a rebuttal. The interdependencies among theses tasks are characterized by the schedule and the proper order of the messages. It is up to the mediators to enforce the correct use of categories in the messages and their correct positioning.

To avoid contributions that do not add value to the group, each message is individually assessed and commented upon. Follow-Up reports make clear who is not participating or who is participating at an inadequate level. The problems detected in a contribution are commented in the message itself, generally visible to the entire class to enable learners to understand where they have room to improve and what they have gotten right [Fuks et al, 2002]. This also helps to ensure the netiquette and the use of a common language.

Previous editions of the ITAE course have taught us that most of the content should be self-studied and that most of the discussion should be conducted asynchronously in order to enhance reflection. However, by reducing the time pressure to respond, it is easier for a learner to drop out of the group [Graham et al, 1999]. Each conference session lasts 50 hours: from 12 noon Monday to 2 pm

Wednesday. Until the 2003.2 edition, there was a burst during the last five hours of the conference. In some cases, more than 50% of the messages were sent during this period. This phenomenon of students waiting until the last possible moment to carry out their tasks is well known and has been dubbed "Student Syndrome" [Goldratt, 1997]. The act of sending contributions near the deadline disturbs in-depth discussions, for those last-minute messages will neither be graded nor be answered during the discussion.

In order to avoid this unwelcome behavior mediators have to encourage the earlier sending in of contributions. Unfortunately, our experience with this course has shown that this encouragement does not work. In the 2004.1 edition, the following experiment was conducted. The last 4 conferences had a different assessment rule than the first 4 conferences, this different assessment being that if until the 25th hour the learners had not sent half of the expected amount of messages, the grade of all the messages sent during the following 25 hours would be divided by 2.

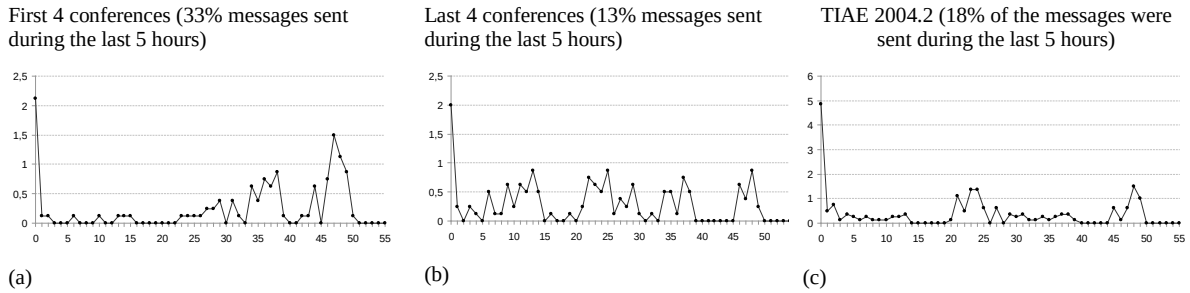

Figure 10. Average hourly rate of messages in conferences

In Figure 10b, the chart does not show the 50th hour message burst, which indicates that the rule has worked. The percentage of messages sent during the last 5 hours of conference fell from 33% in the first half of the course to 13% in the second half. Nevertheless, there are lower 25th and 50th hour peaks. However, now mediators and learners have room to access and answer the first batch of messages. The same can be seen in Figure 10c, where all 8 conferences of the 2004.2 edition were assessed based on the aforementioned rule. In this edition, an average of 18% of the messages was sent during the last 5 hours.

Communication and coordination, although crucial, are not enough: "it takes shared space to create shared understandings" [Schrage, 1995]. Given that coordination is required to manage the tasks, according to the 3C model it is also necessary to provide a shared workspace where cooperation will take place. In the specific case of ITAE, where knowledge production is the main goal, the example for shared space is the *Conference* service; another choice would be the *Debate* service.

5. COOPERATION: PRODUCTION IN THE SHARED WORKSPACE

Cooperation is the joint operation during a session within a shared workspace. Group members cooperate by producing, manipulating and organizing information, and by building and refining cooperation objects, such as documents, spreadsheets, artwork etc. The shared workspace provides a number of tools for managing these artifacts, such as the recording and the recovery of previous versions, access control and permission. By recording the information exchanged, the group is able to count on collective memory, which can be consulted whenever necessary to recover the history of a discussion or the context in which a decision was made.

There are a number of tools in the literature that use hypertext to organize group memory [Shum, S.B & Hammond, 1994]. Some of these tools make it possible to use cooperation objects within a shared information space, explaining their links with the interaction that they set off and those that originated them. As a result, the context of the cooperation objects and the interaction are preserved, facilitating understanding and subsequent recovery, which may serve as input for coordination in its post-articulation stage.

Production is dependent on how the shared workspace is structured to present the cooperation objects and the interaction that is taking place there. In a face-to-face situation, a large part of how we maintain a sense of who is around and what is going on is related to being able to see and hear events or actions such as people arriving or leaving, phones ringing, conversations etc. with little conscious effort [Fitzpatrick et al., 2002]. On the other hand, in a computer-supported workspace, awareness support is less effective since the means for making information available to sensory organs are limited; however, irrelevant information can be filtered in a way that reduces distractions that usually affect face-to-face collaboration.

Individuals seek the awareness information necessary to create a shared context and to anticipate actions and requirements related to their collaboration goals. Thus, it becomes possible to interpret the intentions of the members of the group in such a way that one can provide assistance in terms of their work whenever it is convenient and needed [Baker et al., 2001].

The designer of a digital environment must identify what awareness information is relevant, how it will be obtained, where it is needed and how to display it. Excessive information can cause overload and disrupt the collaboration flow. To avoid disruption, it is necessary to balance the need to supply information with care to avoid distracting the attention required to work. The supply of information in an asynchronous, structured, filtered and summarized form can accomplish this balance [Kraut & Attewell, 1997]. The big picture should be supplied and individuals could select which parts of the information they want to work with, leaving further details to be obtained when required. There must also be some form of privacy protection. The shared space must be conceived in a way that group members could seamlessly move from awareness to work.

5.1. Cooperation in the ITAE course

Knowledge and multimedia content are the ultimate production of the ITAE course. Given that the latter is created offline and outside AulaNet, we are focusing on the former, which takes place in the *Conference* service. It is worth pointing out that AulaNet does not contemplate content authoring. Learners develop educational content using their habitual tools, and the environment supports navigation around the course's shared space.

In AulaNet, services are available through a menu that resembles a remote control unit, which transfers to the learner—up to a certain point—control over the learning process. The control traditionally wielded by the teacher in a classroom is replaced by the coordination of the teacher in charge of mediating the class through knowledge. Learners begin to work in a manner that is similar to what is currently expected of them in the professional world.

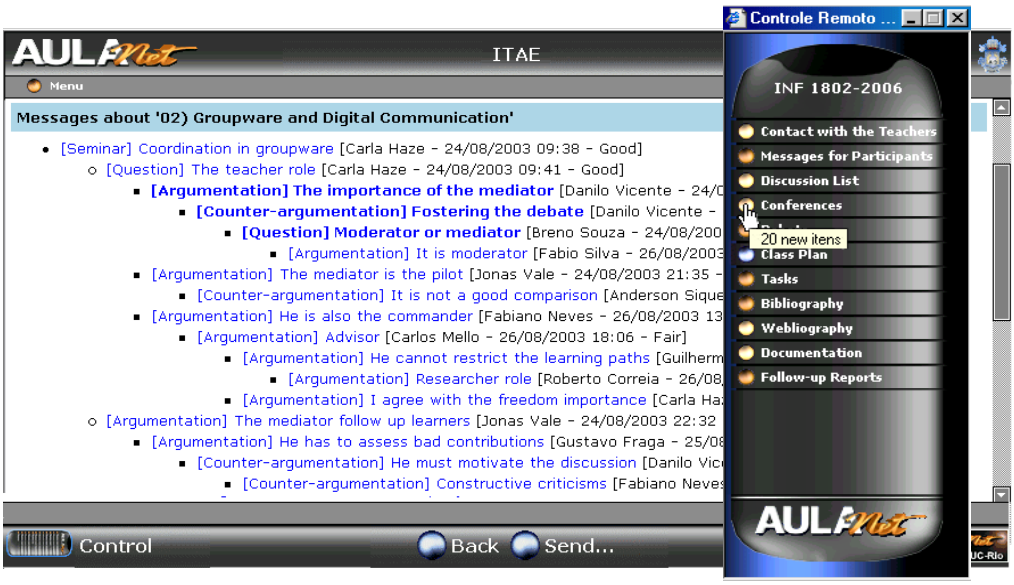

Figure 11. Conference window and remote control providing awareness information

On the upper part of the remote control, the course's code provides contextual information. The remote control items make learners aware of the services available at a given moment. Next to each menu item there is a circular button, whose lightness changes to provide awareness information about the services, indicating the service that the learner has selected, the service where no changes have taken place since her last access or a service for which some action should be taken, thus taking the learner from awareness to work. As highlighted in Figure 11, upon moving the mouse over the button of the *Conference*, the remote control shows that there are 20 unread messages for that learner.

The *Conference* service provides a shared workspace where learners cooperate by producing and refining knowledge by means of an argumentation process. In ITAE the conference session lasts 50 hours. Learners generate new cooperation objects, in this case conference messages. Learners also construct and handle conference messages (the cooperation objects) and receive feedback from their actions and feedthrough for their colleagues' actions by means of awareness information. In the conference shared space, awareness information about the cooperation objects is displayed, including their authorship, date, category, subject and the assessment made by course mediators. The 3Cs for the conferences are shown in Figure 12.

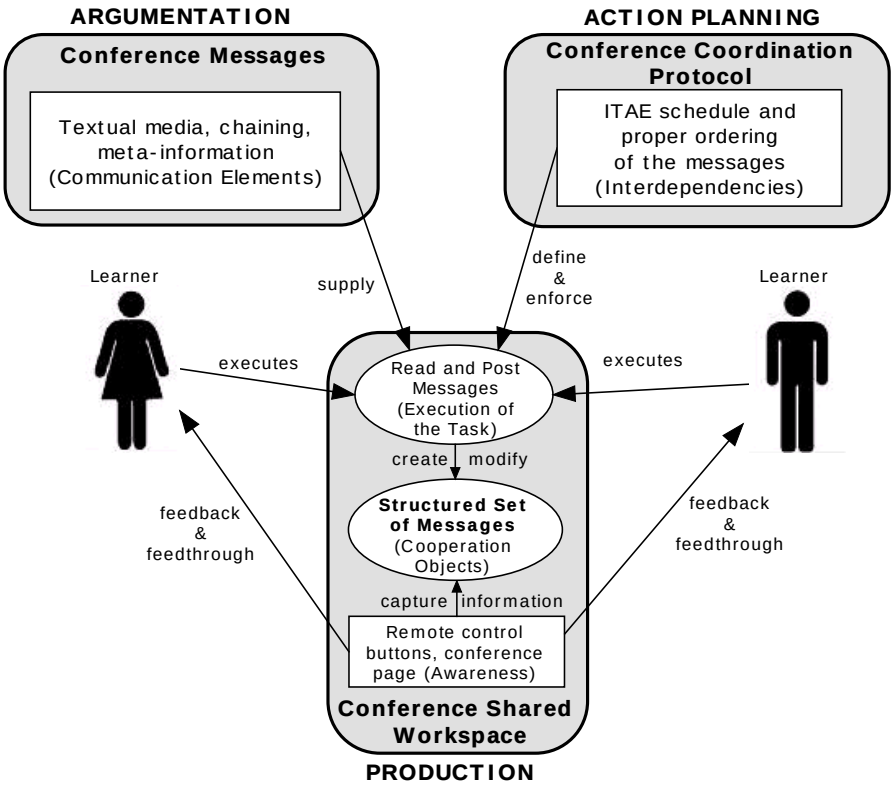

Figure 12. Production in the *Conference* shared workspace

The register of group interactions is filed, catalogued, categorized and structured within cooperation objects. This is how group memory is saved in AulaNet. Ideas, facts, questions, points of view, conversations, discussions, decisions, etc. are retrievable, providing a history of the collaboration and the context in which learning took place [Kanselaar et al., 2003].

6. DESIGN AND IMPLEMENTATION ISSUES

During the design and implementation of groupware, the designer must have in mind that collaborative applications must be flexible enough to adapt to group characteristics and to the evolution of work processes. Although there is no way to foresee all features that will be demanded from a groupware application, different groupware products share a number of characteristics.

This scenario is suitable for the application of component-based development techniques, which provide the flexibility needed in projects with changing requirements [Szyperski, 1999]. Groupware services can be seen as groupware components that are plugged and unplugged from the system. The system's architecture comprises component frameworks, which define overall invariants and protocols for plugging components. "Without an adequate architectural framework, the construction of groupware and general interactive systems is hard to achieve, the resulting software is difficult to maintain and interactive refinement is hard to obtain" [Calvary et al., 1997].

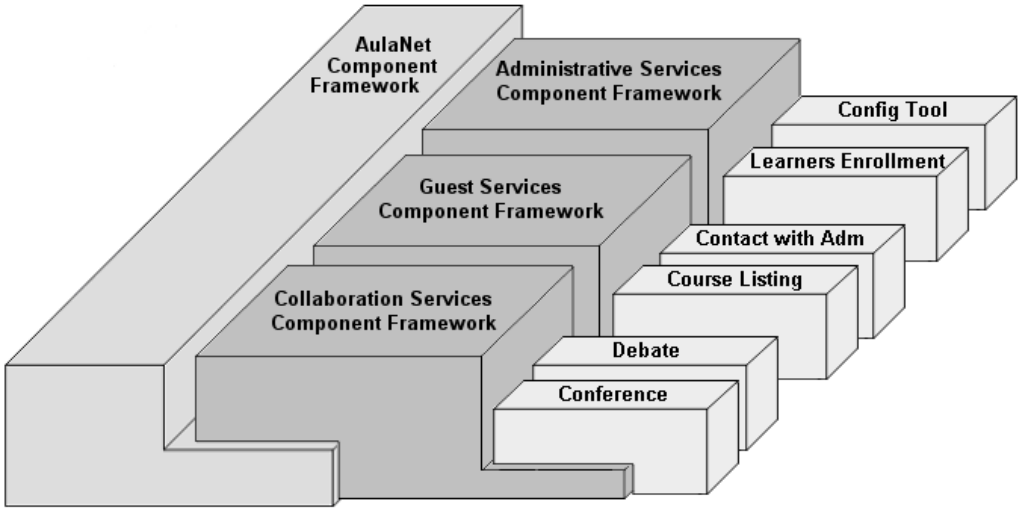

Figure 13. AulaNet's system architecture (for the sake of clarity only a few services were listed).

In AulaNet's architecture, the AulaNet component framework defines the general functionalities common to all services, such as service interaction management and data sharing. Currently, there are three different families of services: collaboration, administrative and guest services, which correspond to component frameworks that deal with characteristics specific to each service (Figure 13). The first family of services is used by teachers and learners to support collaborative learning activities; the second is used by the environment administrator to manage and configure its functionalities and data; and the third aggregates the functionalities that are used by visitors, such as FAQ, contact with the administrator, bug report, course listing and enrollment, etc.

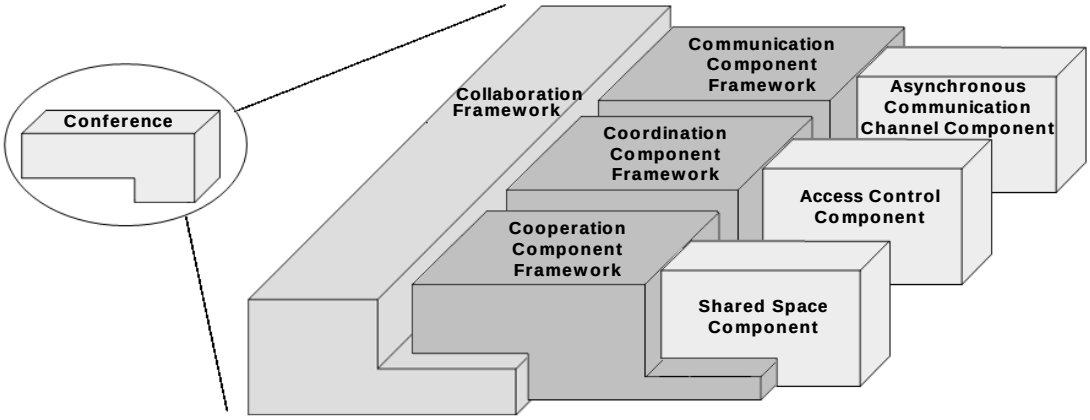

Figure 14. Architecture of a collaboration service.

Current AulaNet services were also developed using a component-framework-based architecture, as can be seen in Figure 14. There is a common structure implemented by the collaboration framework, which defines the skeleton of the services, and plugged to this framework there are the communication, the coordination and the cooperation component frameworks, which support each C. Class frameworks are used to implement components, which are plugged to the corresponding C-framework and implement the specific functionalities of the service.

For example, using the communication class framework, the developer implements components for synchronous and asynchronous communication, message transmission, commitment management, etc. Using coordination class framework, components for task management, participation follow-up, workflow, etc. are implemented. Using cooperation class framework, components for managing the

shared space and its awareness elements, version management, among others are implemented. In the next subsection, the AulaNet *Debate* service is used to exemplify the composition of the service using the instantiated components.

The AulaNet component framework comprises six main components: Service Manager, Security Manager, Session Manager, Interface Component Library, Domain Object Library and Infrastructure Component Library. The Service Manager is responsible for version control, name control, installation of new services, service aggregation, communication between services, remote localization, etc. The Security Manager handles identification, authentication and access control for each user, as well as data cryptography. The Session Manager keeps a given set of values persistent from one call to another among those made by the same user. The Interface Component Library provides interface elements that are used in the services, facilitating the update of an element, enforcing interface consistency and standardization, making the use of specific components for mobile devices possible and supporting the change of the entire environment's skin. This library also stores service interface texts, because of the use of multiple languages and text customization. The Domain Object Library contains objects that represent the data model that is shared by services, enabling the reuse of the data model and supporting the persistence of shared objects and tools. This library contains mechanisms for replication and concurrency control. To facilitate service implementation and to group features that are common to different services, AulaNet has an Infrastructure Component Library, which supports the sending of messages and error handling, among others.

A thin-client strategy was adopted on AulaNet's architecture, that is, most of the processing takes place in the server, leaving to the client the task of displaying the user interface. This strategy is especially useful in a learning environment, because its users are not necessarily from technology-related areas and this approach requires minimum installation and configuration on the client side. Using a web browser, users interact with AulaNet and its services. Some services—such as *Debate*—require components being executed on the client side. In these cases, the architecture supplies the client/server communication resources by means of applets running on the client machine.

Component-based development allows us to have a flexible set of services that can be used within AulaNet. The environment provides a standard set of services that can be adapted to each AulaNet server. It is also possible to develop new services, even without knowing the internal implementation of AulaNet. There is only one set of interfaces that should be maintained to enable communication between AulaNet and its services. This also allows for the use of tools that were not originally developed to be used with AulaNet by implementing some classes that satisfy the required interfaces and mediate the communication between the environment and the tools.

6.1. The Debate Service Case Study

AulaNet services are composed of components plugged to component frameworks. For example, a previous version of the AulaNet *Debate* service, denominated Mediated Chat 1.0 (MC1), was implemented with a communication component, which implemented synchronous communication protocols, and a cooperation component, which implemented the shared space, as can be seen in Figure 15 [Pimentel et al., 2003]. The communication component implements the *Debate*'s communication elements shown in Table 1, namely textual media, linear conversation structure, meta-information and block transmission mode.

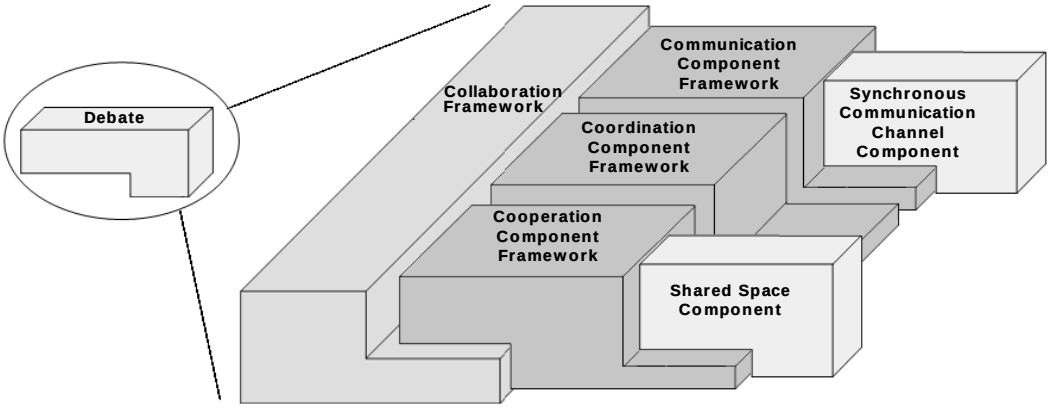

Figure 15. Implementation of the *Debate* component: Mediated Chat 1.0 (MC1).

The MC1 user interface was presented in Figure 4b. It is a plain chat tool that provides a text box, where learners type their messages, a text area, where messages are posted, and another area where the learners present in that chat session are listed.

Given the evolutionary nature of the ITAE course, the debate procedure changed and, unfortunately, the MC1 tool could not support it anymore. Figure 16 shows the new debate coordination protocol. The tasks are organized as follows. The mediators declare the debate session initiated. Then, the Moderator—role performed by one of the learners—posts a summary of the conference's discussion followed by a question related to it. Next, each learner posts a contribution commenting on that question. After all learners have sent their comments, the group chooses one contribution to be further discussed. Then, they all become involved in a free discussion. Finally, they draw their conclusions. This cycle—question, comments, vote, free discussion and conclusions—is repeated for each new question. Concluding the debate, the mediators declare the end of the session and later on appraise learners' participation.

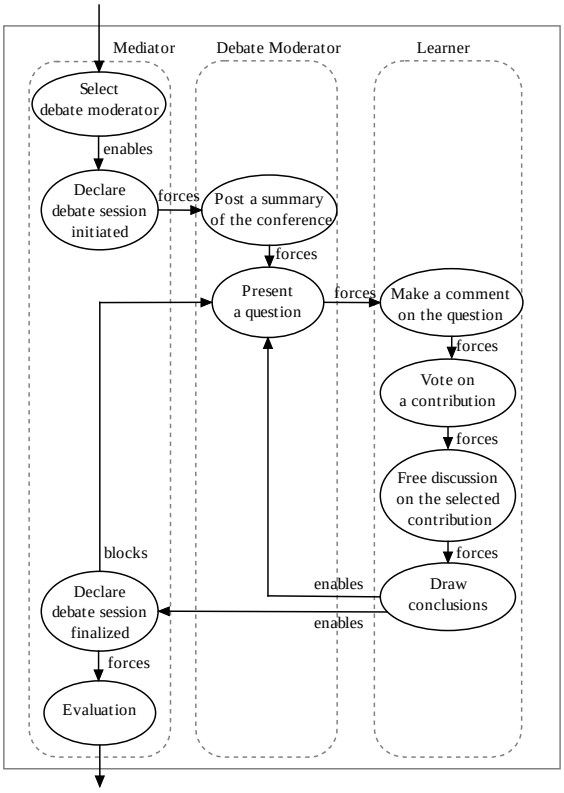

Figure 16. ITAE debate coordination protocol.

For that purpose, in the current version of the *Debate* service, denominated Mediated Chat 2.0 (MC2) and presented in Figure 17, the shared space was enhanced with new awareness information, such as session title, message time stamp, identification of mediators and participation order. Coordination mechanisms incorporated into the *Debate* service are used to implement the coordination protocol modeled in Figure 16.

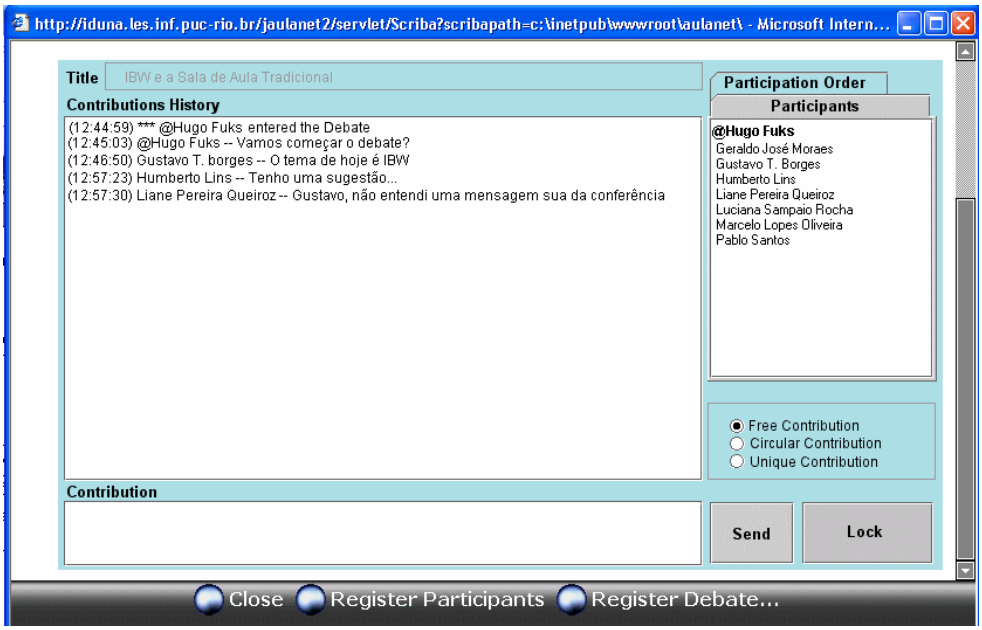

Figure 17. Mediators interface of the MC2 *Debate* service.

The *circular contribution* is used in the "Commenting on the question" task; *unique contribution* is used in "Vote on a contribution"; and *free contribution* is used in the "Free discussion" and "Conclusions" tasks. For the tasks "Select debate moderator", "Declare debate session initiated", "Declare debate session finalized" and "Evaluation", mediators lock the shared space, obtaining exclusive access to it and thus avoiding parallel conversations.

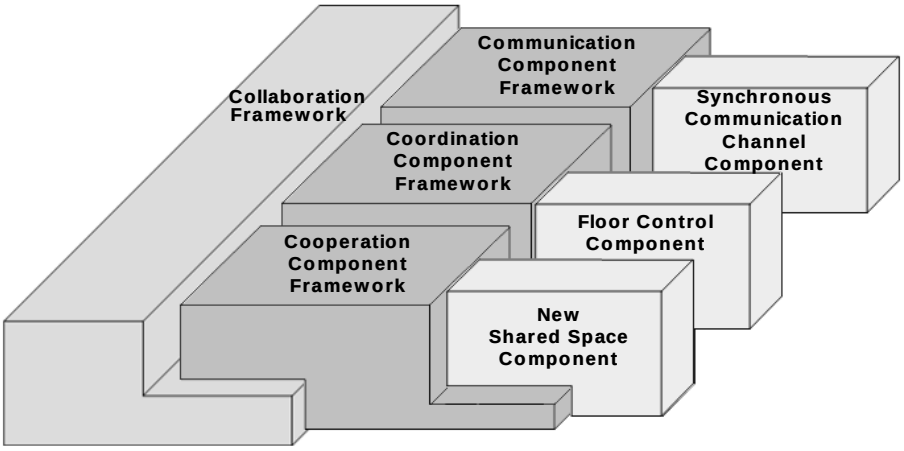

Figure 18. Implementation of the *Debate* component: Mediated Chat 2.0 (MC2).

MC2 was implemented with the same communication component as MC1, which implements synchronous communication protocols, a new cooperation component, which implements the shared space enhanced by new awareness information, and a coordination component, which implements the floor control techniques, as can be seen in Figure 18. This way, a communication service was enhanced without changing any communication element, only dealing with the other elements of the 3C collaboration model.

7. CONCLUSION

Groupware systems are evolutionary because the composition and the characteristics of workgroups change with time, as well as the tasks that need to be executed. For this reason, even if a groupware designer is able to develop an "optimal" application for a group, it will eventually become inadequate due to new situations and problems that certainly will appear.

Ideally, groupware should be prototyped [Schrage, 1996] because collaborative systems are especially prone to failure [Grudin, 1989], hence demanding iterative evaluation during their development. However, given the excessive cost of throwing code away, as demanded by "pure" prototyping [Brooks, 1975], an incremental model can be considered more adequate, leading to the development of more advanced prototypes in the subsequent cycles.

The AulaNet environment has been developed by means of prototyping and its functionalities have been implemented evolutionarily. The evolution in the collaboration support offered by the environment has deemed the application's code tightly integrated and with low cohesion. Technical aspects of synchronism and sharing were present throughout the whole code, being mixed with collaboration support. Changes in the environment were reflected in diverse parts of the code and caused undesired side effects, thus making the environment's evolution difficult.

In face of construction and maintenance difficulties, the groupware developer spent more time dealing with technical difficulties than moderating and providing support to the interaction among users. Such problems led to the need to create a quicker and more effective way to develop groupware in which low-level complexities resulting from distributed and multi-user systems were encapsulated into infrastructure components of the architecture. Besides, the concepts of the domain's modeling

should permeate all other activities and artifacts of the application's development. This way, the modeling done during the domain analysis could be mapped to implementation, thus increasing productivity in groupware development and maintenance and making the applications more adequate to the users' collaboration needs.

The 3C collaboration model defines three types of services that a groupware may support. The concepts and representation models described in this paper can be used to guide the functional specification and to provide a common language for representing and describing the collaboration aspects of a workgroup. The application of the 3C model was illustrated throughout this paper using the AulaNet learning environment and the ITAE course. The groupware component system architecture used in the AulaNet environment mirrors the 3C model. Communication, coordination and cooperation functionalities were directly mapped to the implementation of AulaNet's collaboration services. The redesign of the AulaNet *Debate* service illustrates this mapping and the modularity achieved using the component system architecture.

The 3C model and component-based architecture may be instrumental in the incremental groupware development cycle, which could be based on the spiral software-development model [Boehm, 1988] and combine the classical sequential model with the iterative behavior of incremental prototyping (Figure 19).

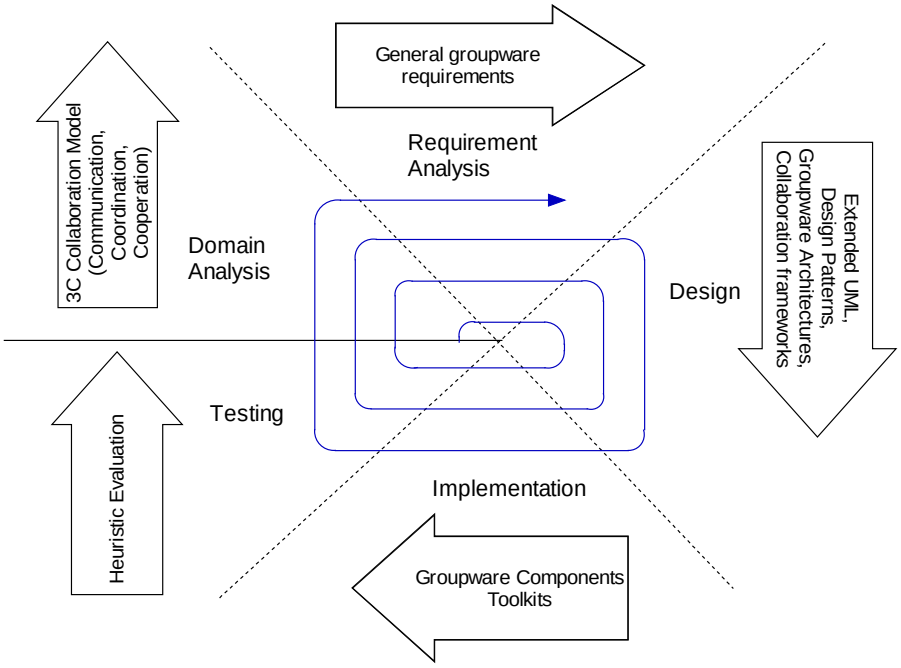

Figure 19. Groupware development cycle.

Considering domain analysis as the process of identifying, collecting, organizing and representing relevant information in a domain based on the study of existing systems and their development histories, knowledge captured from domain experts, underlying theory and emerging technology within a domain [Hess et al, 2000], it could be supported by the 3C collaboration model. The 3C model also guides the subsequent stages of groupware development.

General groupware requirements [Schmidt & Rodden, 1996] [Mandviwalla & Olfman, 1994] that are elicited in the requirement-analysis stage are seldom clear enough to enable a precise specification of system behavior. Incremental prototyping makes it possible to constantly evaluate and validate the design and implementation, thus counterbalancing the need to have a complete set of requirements to start the design. We are currently investigating how groupware requirements could be elicited from the 3C model viewpoint.

There are different techniques suitable for the design stage, namely groupware design patterns [Groupware Patterns Swiki, 2005] for reusing common design approaches, UML extensions for representing specific groupware aspects of the software, groupware architectures [Tietze, 2001] [Marsic & Dorohonceanu, 2003] and groupware-related frameworks [Marsic & Dorohonceanu, 1999] [Tarpin-Bernard et al., 1998] for reusing code and infrastructure. In Section 6 of this paper we have presented a component-based architecture and collaboration framework based on the 3C model that facilitates the task of programmers, who can reuse and extend data structures provided by frameworks, leaving to the infrastructure provided by the groupware architecture the support of some specific multi-user aspects, such as data synchronization, distributed resource sharing and inter-component communication.

For the implementation stage, communication, coordination and cooperation components could be plugged to the collaboration component frameworks, as shown in the case study presented here. Toolkits [Roseman & Greenberg, 1997] and other kinds of groupware components [Hummes & Merialdo, 2000] [Stiemerling & Cremers, 2000] [Blois & Becker, 2002] are alternatives for building collaborative systems.

Given its complex interactive nature, groupware testing has not yet achieved its maturity. We are currently investigating how the 3C model may help evaluators focus their attention on the communication, coordination and cooperation aspects, guiding the detection of usability problems. Groupware heuristic evaluation [Baker, Greenberg & Gutwin, 2001] and task-based analysis [Pinelli et al, 2003] could be used to guide groupware testing.

ACKNOWLEDGMENTS

The AulaNet project is partially financed by Fundação Padre Leonel Franca and by the Ministry of Science and Technology through its Program Multi-Agent Systems for Software Engineering Project (ESSMA) grant nº 552068/2002-0. It is also financed by individual grants awarded by the National Research Council to: Carlos José Pereira de Lucena nº 300031/92-0, Hugo Fuks nº 303055/02-2, Alberto Barbosa Raposo nº 305015/02-8 and Marco Aurélio Gerosa nº 140103/02-3. Thanks to Prof. Marcelo Gattass, Head of Tecgraf/PUC-Rio, a group mainly funded by Petrobras, Brazilian Oil & Gas Company.

REFERENCES

Allen, J. F. (1984): Towards a General Theory of Action and Time. *Artificial Intelligence*, Vol. 23, 1984, pp. 123-154.

Baker, K., Greenberg, S. & Gutwin, C. (2001): Heuristic Evaluation of Groupware Based on the Mechanics of Collaboration. *8th IFIP International Conference (EHCI 2001)*. Lecture Notes in Computer Science Vol. 2254, pp. 123-139, Springer-Verlag.

Benbunan-Fich, R. & Hiltz, S. R. (1999): Impacts of Asynchronous Learning Networks on Individual and Group Problem Solving: A Field Experiment, *Group Decision and Negotiation*, Vol.8, pp. 409-426.

Blois, A.P.T.B. & Becker, K.A. (2002): Component-based Architecture to Support Collaborative Application Design. *8th International Workshop on Groupware (CRIWG)*. Lecture Notes in Computer Science Vol. 2440, pp. 134-146, Springer-Verlag.

Boehm, B.W. (1988): A Spiral Model of Software Development and Enhancement, *IEEE Computer*, Vol. 21, No. 5, pp. 61-72

Borghoff, U.M. and Schlichter, J.H. (2000): *Computer-Supported Cooperative Work: Introduction to Distributed Applications*. Springer, USA.

Brooks Jr., F.P. (1975): Plan to Throw One Away. In: *The Mythical Man-Month – Essays on Software Engineering*, chap. 11, pp. 115-123. Addison-Wesley.

Calvary, G., Coutaz, J., Nigay, L. (1997): From Single-User Architectural Design to PAC*: a Generic Software Architectural Model for CSCW. *Conference on Human Factors in Computing Systems (CHI'97)*, pp 242-249.

Conklin, J. & Begeman, M. (1988) "gIBIS: A hypertext tool for exploratory policy discussion", *Conference on Computer-Supported Cooperative Work (CSCW),* pp 140-152.

Cunha, L.M., Fuks, H. & Lucena, C.J.P. (2003): Setting Groups of Learners using Matchmaking Agents. *IASTED International Conference on Computers and Advanced Technology in Education (CATE 2003)*, pp 321-326.

Dourish, P. & Belloti, V. (1992): Awareness and Coordination in Shared Workspaces. In J. Turner and R. Kraut (eds): *Conference on Computer-Supported Cooperative Work (CSCW)*, pp. 107-114.

Edutools (2005): http://www.edutools.info (date 06/04/2005)

Ellis, C.A. & Wainer, J. (1994): A Conceptual Model of Groupware, In T. Malone (ed): *Conference on Computer-Supported Cooperative Work (CSCW)*, pp. 79-88.

Ellis, C.A., Gibbs, S.J. & Rein, G.L. (1991): Groupware - Some Issues and Experiences. *Communications of the ACM*, Vol. 34, No. 1, pp. 38-58.

Fitzpatrick, G., Kaplan, S., Mansfield, T. Arnold, D. & Segall, B. (2002): Supporting Public and Accessibility with Elvin: Experiences and Reflections. *Computer Supported Cooperative Work*, Vol. 11, No. 3-4, pp. 447-474

Fuks, H. (2000): Groupware Technologies For Education In Aulanet. *Computer Applications In Engineering Education*, Vol. 8, Nos. 3-4, pp. 170-177.

Fuks, H., Gerosa, M.A. & Lucena, C.J.P. (2002), "The Development and Application of Distance Learning on the Internet", *Open Learning Journal*, Vol. 17, No. 1, pp. 23-38.

Gerosa, M.A., Fuks, H. & Lucena, C.J.P. (2001) Use of Categorization and Structuring of Messages in order to Organize the Discussion and Reduce Information Overload in Asynchronous Textual Communication Tools. *7th International Workshop on Groupware (CRIWG)*, pp 136-141.

Gerosa, M.A., Fuks, H. & Lucena, C.J.P. (2003): Analysis and Design of Awareness Elements in Collaboration Digital Environments: A Case Study in the AulaNet Learning Environment. *The Journal of Interactive Learning Research*, Vol. 14, No. 3, pp. 315-332.

Gerosa, M.A., Pimentel, M., Fuks, H. & Lucena, C.J.P. (2004) Analyzing Discourse Structure to Coordinate Educational Forums. *The 7th International Conference on Intelligent Tutoring Systems (ITS-2004).* Lecture Notes on Computer Science LNCS 3220, pp. 262-272.

Goldratt, E.M. (1997) *Critical Chain*, The North River Press Publishing Corporation, Great Barrington.

Graham, M., Scarborough, H. & Goodwin, C. (1999): Implementing Computer Mediated Communication in an Undergraduate Course - A Practical Experience. *Journal of Asynchronous Learning Networks*, Vol. 3, No. 1, 1999, pp. 32-45.

Groupware Patterns Swiki (2005), http://www.groupware-patterns.org (date 04/06/2005)

Grudin, J. (1989): Why Groupware Applications Fail: Problems In Design And Evaluation. *Office: Technology and People*, Vol. 4, No. 3, pp. 245-264.

Gutwin, C. & Greenberg, S. (2002): A Descriptive Framework of Workspace Awareness for Real-Time Groupware, *Computer Supported Cooperative Work*, Vol. 11, No. 3-4, pp. 411-446.

Hess, H. et al. (2000) *A Domain Analysis Bibliography*, Carnegie Mellon University/Software Engineering Institute, http://www.sei.cmu.edu/publications/documents/90.reports/90.sr.003.html, Special Report -90-SR-3.

Hummes, J. & Merialdo, B. (2000): Design of Extensible Component-Based Groupware, *Computer Supported Cooperative Work (CSCW)*, Vol. 9, No. 1, pp. 53-74.

Kanselaar, G., Erkens, G., Andriessen, J., Prangsma, M., Veerman, A. & Jaspers, J. (2003): Designing Argumentation Tools for Collaborative Learning, *Visualizing Argumentation: Software Tools for Collaborative and Educational Sense-Making*, Kirschner, P., Shum, S. and Carr, C. (Eds.), chap. 3, Springer-Verlag.

Kirschner, P.A., Shum, S.J.B. & Carr, C.S. (2003), *Visualizing Argumentation: Software Tools for Collaborative and Educational Sense-Making*, Springer.

Kraut, R. E., & Attewell, P. (1997): Media use in global corporation: electronic mail and organisational knowledge, *Research milestone on the information highway*, Mahwah, NJ: Erlbaum.

Laurillau, Y. & Nigay, L. (2002): Clover architecture for groupware, *Conference on Computer-Supported Cooperative Work (CSCW)*, pp. 236 - 245

Mackay, W. E. (1999): Media Spaces: environments for Informal Multimedia Interaction. In M. Beaudouin-Lafon (ed): *Computer Supported Co-operative Work (Trends in Software: 7)*. John Wiley & Sons, England, pp. 55-82.

Malone, T.W. & Crowston, K. (1990): What is Coordination Theory and How Can It Help Design Cooperative Work Systems? *Conference on Computer-Supported Cooperative Work (CSCW)*, pp. 357-370.

Mandviwalla, M. & Olfman, L. (1994): What do groups need? A proposed set of generic requirements. *ACM Transactions on Computer-Human Interaction*, Vol. 1, No. 3, pp. 245-268.

Marsic, I. & Dorohonceanu, B. (1999): An Application Framework for Synchronous Collaboration using Java Beans, *Proceedings of the Hawaii International Conference On System Sciences*.

Marsic, I. & Dorohonceanu, B. (2003): Flexible User Interfaces for Group Collaboration. *International Journal of Human-Computer Interaction*, Vol. 15, No. 3, pp. 337-360.

Pimentel, M.G., Fuks, H. & Lucena, C.J.P. (2003): Co-text Loss in Textual Chat Tools. *Proceedings of the 4th International and Interdisciplinary Conference on Modeling and Using Context* (CONTEXT), Lecture Notes in Artificial Inteligence 2680, pp. 483-490.

Pinelle, D., Gutwin, C. & Greenberg, S. (2003), Task Analysis for Groupware Usability Evaluation: Modelling Shared-Workspace task with the Mechaniscs of Collaboration. *ACM Transactions on Computer-Human Interaction (TOCHI)*, Vol. 10, Issue 4, pp. 281-311.

Raposo, A.B. & Fuks, H. (2002): Defining Task Interdependencies and Coordination Mechanisms For Collaborative Systems. In M. Blay-Fornarino, A.M. Pinna-Dery, K. Schmidt and P. Zaraté (eds): *Cooperative Systems Design (Frontiers In Artificial Intelligence and Applications Vol. 74)*. IOS Press, Amsterdam, pp. 88-103.

Roseman, M. & Greenberg, S. (1997): Building Groupware with GroupKit, *Tcl/Tk Tools*, Harrison, M. (ed), Chap. 15, pp. 535-564, O'Reilly Press.

Schmidt, K. & Rodden, T. (1996): Putting it all Together: Requirements for a CSCW Platform. In: Shapiro, D., Tauber, M., Traunmüller, R. (eds.): *The Design of Computer Supported Cooperative Work and Groupware Systems*. North Holland, Holland, pp. 157-176.

Schmidt, K. & Simone, C. (1996): Coordination mechanisms: Towards a conceptual foundation of CSCW systems design. *Computer Supported Cooperative Work*, Vol. 5, No. 2-3, pp. 155-200.

Schön, D.A. (1983), *The reflective practitioner: How professionals think in action*, Basic Books, New York.

Schrage, M. (1995): *No More Teams! Mastering The Dynamics Of Creative Collaboration*. Currency Doubleday, USA.

Schrage, M. (1996): Cultures Of Prototyping. In T. Winograd (ed): *Bringing Design To Software*, ACM Press, USA, pp. 191-205.

Shum, S.B & Hammond, N. (1994): Argumentation-based design rationale: what use at what cost?, *Human-Computer Studies* Vol. 40, pp. 603-652.

Stahl, G. (2001): WebGuide: Guiding collaborative learning on the Web with perspectives, *Journal of Interactive Media in Education*.

Stiemerling, O. & Cremers, A.B. (2000): The Evolve Project: Component-Based Tailorability for CSCW Applications. *AI And Society*, Vol. 14, pp. 120-141.

Szyperski, C. (1999): *Component Software: Beyond Object-Oriented Programming*. Addison-Wesley.

Tarpin-Bernard, F., David, B.T. & Primet, P. (1998): Frameworks and Patterns for Synchronous Groupware: AMf-C Approach, *Seventh Working Conference on Engineering for Human-Computer Interaction*. IFIP Conference Proceedings 150, pp. 225-241.

Tietze, D.A. (2001): *A Framework For Developing Component-Based Co-Operative Applications.* Ph.D. Dissertation, Computer Science, Technischen Universität Darmstadt, Germany.

Winograd, T. & Flores, F. (1987): *Understanding Computers and Cognition - A New Foundation for Design*, Adison Wesley Publishing Inc.