PUC

ISSN 0103-9741

Monografias em Ciência da Computação
n° 01/04

Applying the 3C Model to Groupware Engineering

Hugo Fuks
Alberto Barbosa Raposo
Marco Aurélio Gerosa
Carlos José Pereira de Lucena

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
RUA MARQUÊS DE SÃO VICENTE, 225 - CEP 22453-900
RIO DE JANEIRO - BRASIL

PUC RIO - DEPARTAMENTO DE INFORMÁTICA ISSN 0103-9741

Monografias em Ciência da Computação, Nº 01/04
Editor: Carlos J. P. Lucena Janeiro, 2004

Applying the 3C Model to Groupware Engineering *

Hugo Fuks
Alberto Barbosa Raposo
Marco Aurélio Gerosa
Carlos José Pereira de Lucena

* This work has been sponsored by the Ministério de Ciência e Tecnologia da Presidência da República Federativa do Brasil.

Applying the 3C Model to Groupware Engineering

Hugo Fuks **Alberto Barbosa Raposo**
Marco Aurélio Gerosa **Carlos José Pereira de Lucena**

Laboratório de Engenharia de Software – LES
Laboratório de Computação Gráfica – TecGraf
Departamento de Informática
Pontifícia Universidade Católica do Rio de Janeiro – PUC-Rio
R. Marquês de São Vicente, 225, 22453-900. Rio de Janeiro, Brasil.
http://www.les.inf.puc-rio.br/groupware

[hugo,gerosalucena]@inf.puc-rio.br; abraposo@tecgraf.puc-rio.br

PUC-RioInf.MCC01/04. Janeiro de 2004.

RESUMO

Este artigo introduz uma abordagem baseada no modelo de colaboração 3C (comunicação, coordenação e cooperação) ao desenvolvimento de sistemas colaborativos. Esta abordagem é denominada Engenharia de Groupware, que por sua vez é baseada na Engenharia de Software, aprimorada com conceitos originários das áreas de CSCW e outras relacionadas. O modelo 3C é estudado por meio de uma análise detalhada de cada um de seus três elementos, acompanhado do estudo de caso de um learningware e uma metodologia de um curso baseado na Web. Além disto, o artigo descreve uma arquitetura baseada em componentes que segue a abordagem 3C.

Palavras-chaves: modelo de colaboração, comunicação, coordenação, cooperação, engenharia de groupware, learningware.

ABSTRACT

This paper introduces an approach based on the 3C collaboration model (communication, coordination and cooperation) to the development of collaborative systems. This approach is denominated Groupware Engineering, which is based on Software Engineering, enhanced by concepts originated from the field of CSCW and related areas. The 3C model is studied by means of a detailed analysis of each one of its three elements, accompanied by a case study of a learningware together with the methodology of a web-based course, both designed based on the model. Moreover, the paper describes a component based system architecture following this 3C approach.

Keywords: collaboration modelling, communication, coordination, cooperation, groupware engineering, learningware.

1. INTRODUCTION

Software Engineering, which has advanced substantially in the development of single-user applications but only recently started addressing the human factor problem [DeMarco et. al., 1999], fails to support group aspects so needed in collaborative applications [Grudin, 1994]. Groupware Engineering formulates systematic and disciplined approaches to the development and maintenance of groupware[1], based on Software Engineering principles enhanced by concepts originated from the fields of CSCW and related areas.

Collaborative systems are especially prone to failure [Grudin, 1989]; hence demand iterative evaluation during their development. Ideally, groupware should be prototyped [Schrage, 1996]. However, given the excessive cost of throwing code away, as demanded by "pure" prototyping [Brooks, 1975], an incremental model is more adequate, leading to the development of more advanced prototypes in the subsequent cycles. The groupware engineering cycle is based on the spiral software development model [Boehm, 1988], which combines the classical sequential model and the iterative behaviour of incremental prototyping. This model covers the conceptual development and the product development, enhancement and maintenance. The phases of groupware development are shown in Figure 1 listing the topics that are the objects of study of this research project (inside the arrows).

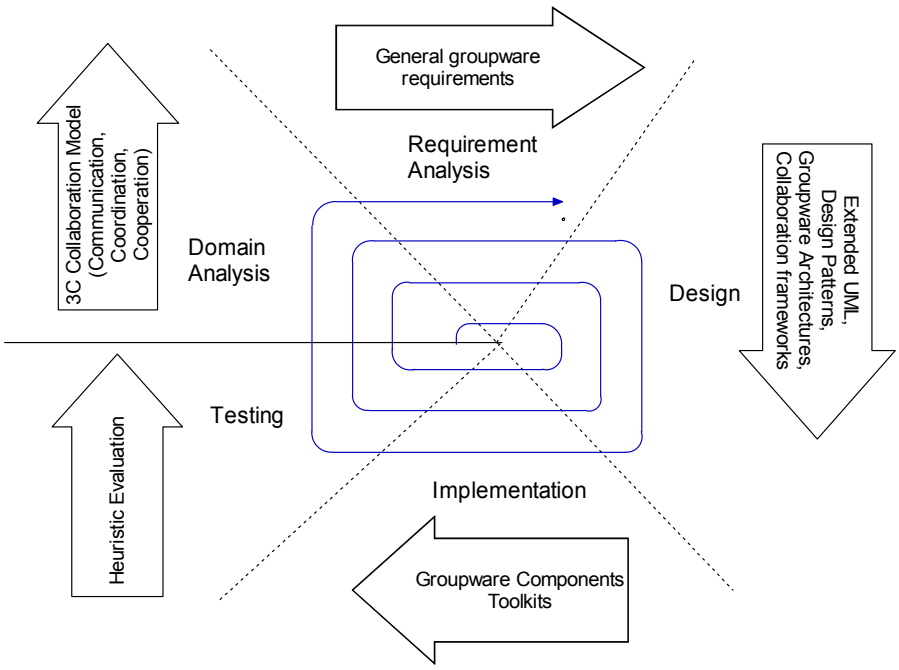

Figure 1. Groupware Engineering development cycle

The domain analysis phase of Groupware Engineering is supported by the 3C collaboration model, which is based upon the concepts of communication, coordination and cooperation. This model is detailed throughout this paper.

[1] This is an adaptation of the definition of Software Engineering of the IEEE Standard Glossary of Software Engineering Terminology [IEEE, 1991].

General groupware requirements [Schmidt & Rodden, 1996][Mandviwalla & Olfman, 1994] that are elicited in the requirement analysis phase seldom are clear enough to enable a precise specification of system behaviour. This is due to the fact that "we have only a sketchy knowledge of how people collaborate, and translating what we know into effective designs is difficult" [Gutwin & Greenberg, 2000]. Incremental prototyping makes it possible to constantly evaluate and validate the design and implementation, thus counterbalancing the necessity of having a complete set of requirements to start the design.

There are different techniques suitable for the design phase, namely, groupware design patterns [Groupware Patterns Swiki, 2003][Alejandro et al, 2002] for reusing common approaches of design; UML extensions for representing groupware specific 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.

For the implementation phase, toolkits [Roseman & Greenberg, 1997] and 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. Groupware heuristics [Baker, Greenberg & Gutwin, 2001] guide experiments to test the system. The heuristics help evaluators focus their attention on aspects that are often sources of trouble, guiding the detection of usability problems.

In this paper the 3C collaboration model, which is the basis of the proposed Groupware Engineering approach, is refined. The paper also discusses the application of this model to the development of the learningware entitled AulaNet and to the dynamics of the Information Technology Applied to Education course (ITAE), currently in its eleventh edition. In Section 2 of this paper, the use of the 3C collaboration model is justified and the 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 the ITAE course as a case study. In Section 6, development issues are discussed, detailing how they are applied to the AulaNet environment. Finally, Section 7 concludes the paper.

2. WHY THE 3C COLLABORATION MODEL?

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—or, rather, conditioned by it—in the classroom and widely disseminated on the factory floor, workers are not up to the new demands of the connected society. Workers are taught 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 the classroom, horizontal communication—communication with a shift colleague—besides being hardly well thought of also is given no technological support.

Knowledge workers, on the other hand, constantly interact with their work colleagues in order to carry out their tasks. The organisation that was imposed from top down in the

command and control paradigm loses effectiveness and is substituted for by one that is peer-to-peer like, where communication, coordination and cooperation predominate.

Successful communication results in commitments assumed by the receiver, by the transmitter or by both. Coordination deals with conflicts and organises the group in a manner that avoids that communication and cooperation efforts are lost. Cooperation is the joint operation of members of the group in a shared space, seeking to complete the tasks, generating and manipulating cooperation objects. The necessities of renegotiating and of making decisions about non-expected situations that appear during cooperation demands a new round of communication, which in turn will modify and generate more commitments that will need coordination to reorganise the tasks that are executed during cooperation. This cycle shows the interactive nature of collaboration. The participants obtain feedback from their actions and feedthrough from the actions of their companions through awareness elements. The awareness elements dispose the awareness information related to participants' interaction [Gutwin & Greenberg, 2002]. Through them, individuals become aware of changes that have taken place in the environment and can redirect their actions and anticipate future requirements.

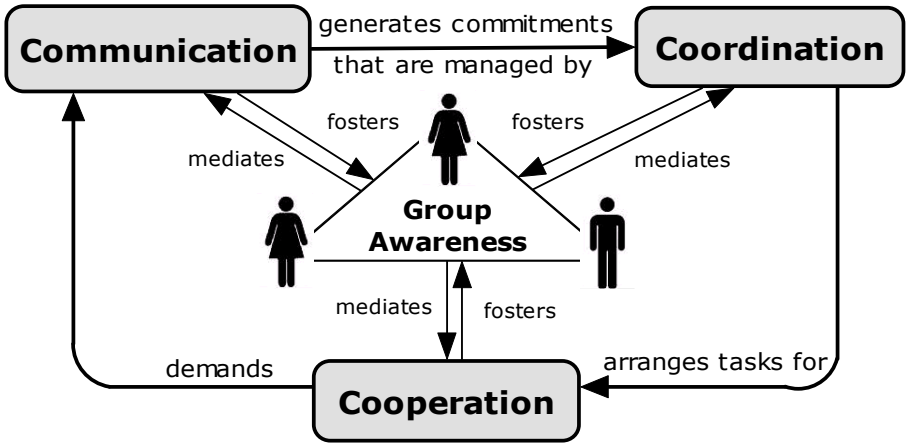

Figure 2. Overview of the 3C collaboration model

The diagram shown in Figure 2 summarizes the main concepts discussed in this paper. This diagram is based on models found in the literature, such as the 3C model proposed by [Ellis et al., 1991] and the Clover design model [Laurillau & Nigay, 2002]. Different than the model sketched by Ellis, the 3C model presented in this paper is used as a basis to groupware development, therefore, each one of the C's is deeply analysed. There is also a terminology difference, because cooperation, which Ellis denominates collaboration, in this paper characterises the joint operation in a shared space and collaboration comprises the 3C's. Regarding the Clover design model, three classes of functionalities are defined, namely communication, coordination and production (the same as cooperation in this paper). The system architecture described in this paper does not derive from the Clover architectural metamodel, which is based on Arch, Dewan's and PAC* architectural models. However, it can be seen as another alternative to a 3C-based implementation.

Finally, Sections 3 to 5 detail the main elements of the 3C model. It should be pointed out that despite the separation of these concepts for the purpose of analysis; it is not always

possible to consider them completely in an isolated manner in real applications, since they are intimately dependent and inter-related.

2.1. Collaborative Learning Using AulaNet

The 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). More than 4,000 units of AulaNet have been distributed in Brazil so far. Besides Portuguese, AulaNet is also available for download (http://www.eduweb.com.br) in English and Spanish versions.

In AulaNet courses, teachers can have three different roles, which may be assumed by the same person. The coordinator's role is to design the course by configuring the course's shared space, defining and configuring the services that are disposed to learners. The author's role is to generate and insert educational content into the course repository. It is worth to point out that the AulaNet does not contemplate content authoring. Teachers develop educational content using their habitual tools, and the environment supports learners' navigation around the course shared space. The mediator's role is to animate the group, maintaining order, motivating and assessing learners' participation. The mediator is also responsible for ensuring that collaborative learning activities get done as well as for evaluating the course as a whole. The coordinator uses this evaluation to put into practice improvements in future editions of the course.

In its first versions, AulaNet resources were subdivided into administrative, assessment and didactic services, which is a common approach in educational tools [Edutools, 2003]. 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 now are supposed to act as coordinators or mediators rather than as information deliverers. Hence, services were reorganised based on the 3C collaboration model, which is suitable to a collaborative learning approach [Fuks, 2000].

The AulaNet environment 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), instantaneous message exchange between simultaneously connected learners (Messages to Participants) and individual electronic mail with the mediators (Contact with Teachers) and with all of the class, in a list-server style (Discussion List).

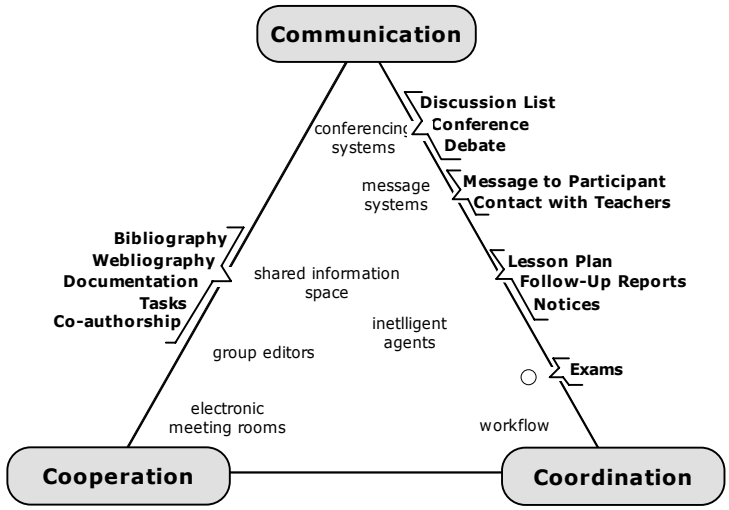

Figure 3. Classification of AulaNet services based on the 3C Model.
The 3C triangle appears in [Borghoff & Schlichter, 2000].

Coordination services support the management and the enforcement of group activities. On AulaNet, coordination services include tools for notification (Notices), for coordination of the flow of course work (Lesson Plan), evaluation (Tasks and Exams) as well as a tool that allows for following group participation (Follow-Up Reports).

Cooperation services on the AulaNet, include a list of course references (Bibliography and Webliography), a content list that can be transferred for offline study (Download) and course co-authoring support, both for teachers (Teacher Co-Authoring) as well as for learners (Learner Co-Authoring).

2.2. The Information Technology Applied to Education Course

The AulaNet environment is based on the collaboration paradigm and does not presuppose the use of any specific pedagogical methodology. The environment can be used to supplement the traditional classroom, although originally it was designed to support collaborative learning. The Information Technology Applied to Education Course (ITAE), which exemplifies this use, has been taught since 1998 as one of the courses of the Computer Science Department of PUC-Rio, and is entirely taught via the Internet within the AulaNet environment.

The course methodology was envisaged to change the behaviour of students accustomed to being passive receivers into learners who actively generate knowledge. This process seeks to lead learners to learn to look for their own sources of information, to deal with information overload and to collaboratively convert information into knowledge. Learners become responsible for the success of the learning experience, where they have to generate content, make the discussions more dynamic and contribute to their colleagues' learning. They are graded for their contributions that add value to the group and not for their individual activities [Fuks et al, 2002]. There is no checking up to see if they navigated around and studied the course content available through multimedia presentations. What is required of them is a constructive and participatory attitude in the course activities and quality in their contributions.

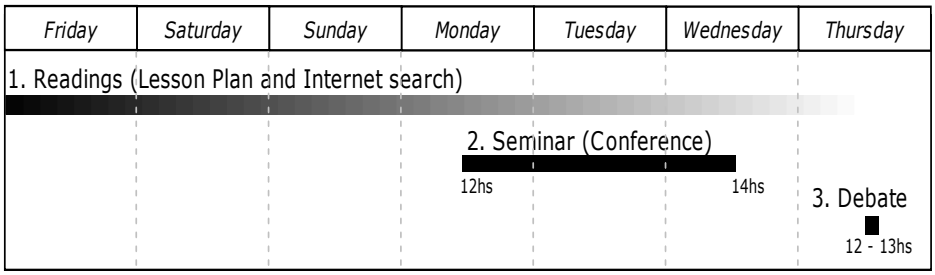

Figure 4. Sequence of learning activities of the first phase of the course.

The sequence of ITAE course learning activities is presented in Figure 4. The first phase of the ITAE course is organised by topics and lasts eight weeks. Every week, learners study the contents regarding the weekly topic, and in an appropriate time slot, all learners have to send their messages to the week's asynchronous conference. Then, all learners join the week's synchronous debate to discuss the topic studied. More detail about the course and the AulaNet environment is given in the following sections, as a case study for communication, coordination and cooperation.

3. COMMUNICATION: COMMITMENT-BASED INTERACTIVITY

In the command and control paradigm, communication is considered to have been 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 of obtaining indications about the receiver's understanding is by observing her actions and reactions, since they are guided by the commitments assumed during communication. Hence, a commitment-based interactivity is used as a means to model communication.

In this model, the management of commitments is based on the theory of Hamblin & Mackenzie [Mackenzie, 1985]. Each interlocutor has a commitments store that holds, for each stage of the conversation, a set of locutions that represents what was discussed. A interlocutor may argument and, occasionally, withdraw facts previously accepted, removing them from the commitment store. In this manner, interlocutors must interact with each other so they can negotiate an agreement acceptable for all parts. Commitments assumed during a conversation represent a verbal agreement with those involved in that conversation. These commitments may indicate a new responsibility, an obligation, a restriction or a decision, guiding future actions [Fuks, Ryan & Sadler, 1989], which will be enforced by coordination. Therefore, the representation and handling of commitments must have a computational support [Laufer & Fuks, 1995]. This computational support is different from the one offered by The Coordinator [Winograd & Flores, 1987]. There, a speech-act based workflow forces the selection among pre-structured alternatives for possible illocutionary forces limiting the available choices.

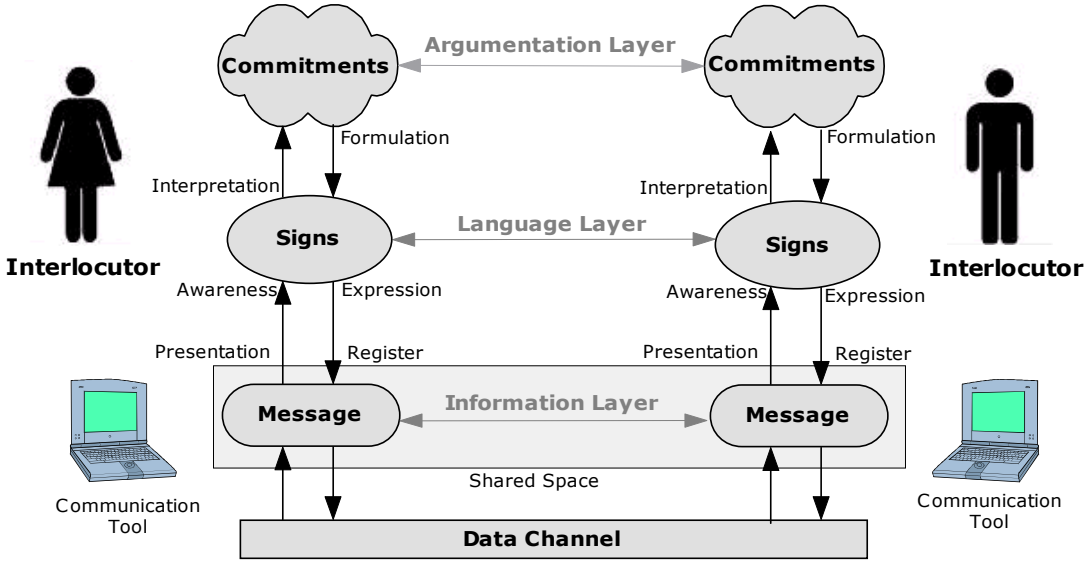

Figure 5. Modelling the communication.

In order to communicate, as illustrated in Figure 5, interlocutors use elements available in the environment. After formulating the message in a language that is proper for the conversation, the sender encodes it using available expression elements. This way, the communication tool registers the information and transmits the data necessary to restore the information on the receiver side. Then, the receiver reads the message, interprets it, changing his commitments and knowledge in a certain way. That will prompt him to react and negotiate the commitments he has assumed. This way, sender and receiver move into an argumentation process where they negotiate commitments and, therefore, their knowledge.

3.1. Communication in the ITAE course

Given that the objective of the ITAE course is to train educators to work with new information technologies for teaching and learning, the reversal of roles between teachers and learners is encouraged so that they can experience collaboration first hand [Schön, 1983]. Two special roles are assigned to learners: Conference leader and Debate moderator. Learners take turns performing these roles throughout the course, encouraging the argumentation process. All learners are expected to contribute, discussing arguments with their colleagues. This way, learners reflectively develop new concepts, refining them, and also get more motivated since their work is being observed, commented upon and evaluated by their peers [Benbunan-Fich & Hiltz, 1999].

"Arguments are presented in language. Language is a system of communication between those who use it." [Mackenzie, 1985]. In a learning situation it is particularly important to align the arguments of the learners to promote a common body of knowledge. During the argumentation process, learners have to attack and defend concepts and find and validate information that support or go against them. The argumentation log stands for the knowledge exchanged during discussion. So, using this communication model, this log stands for the learner's commitment store. According to Mackenzie [1985], transitive closure is not applicable to the facts and rules in the commitment store, thus avoiding omniscience in learning. Moreover, a commitment store could be temporally in a paraconsistent state until the other party demands a resolution. The same applies to knowledge when the novice learner

lives with contradictory understandings until another learner or the mediator clarifies the situation. Figure 6 illustrates the instantiation of the communication model for the ITAE conference.

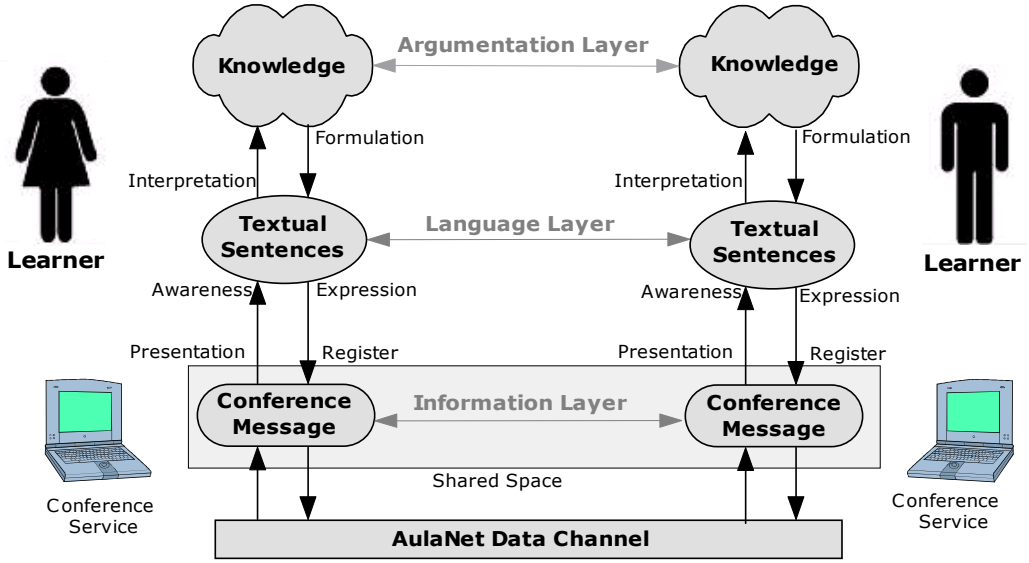

Figure 6. Instantiation of the communication model for the Conference service.

In the conference service it is possible to post messages in a nested way, as can be seen in Figure 7. The learner that plays the conference leader role initiates the weekly conference by sending a seminar message, where an aspect of the weekly topic is discussed, directing the subsequent argumentation. Besides this initial message, the conference leader also posts three messages with the questions that the other learners will discuss during the week. During this argumentation phase, the conference leader is responsible for animating the conference and making sure it is a lively collaborative learning activity.

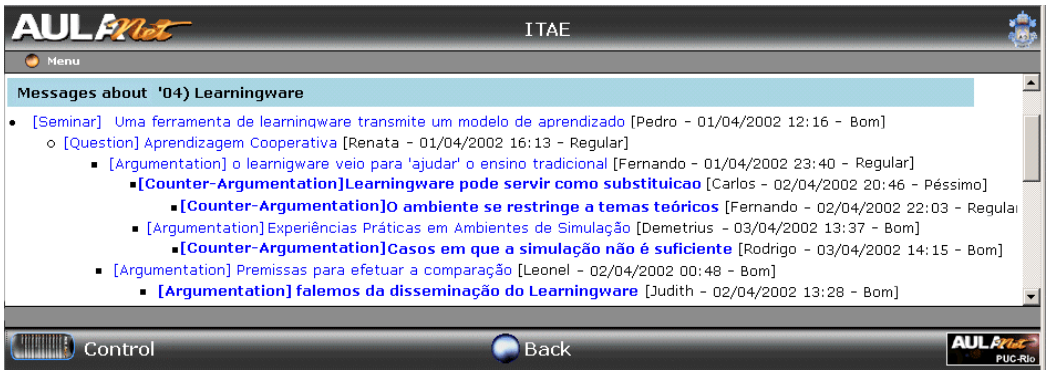

Figure 7. Portion of a dialogue from a conference service.

The topic of the week is also discussed synchronously on a textual chat tool (the Debate service). As in the Conference, a special role for learners is also defined: moderator of the debate. The moderator's job is to mediate the debate that follows the end of the conference. Learners take turns in this role during the course.

In the ITAE course, all content self-study and part of the communication are conducted asynchronously. In asynchronous events, learners can participate at a time and place

convenient to them and appropriate to the task, having more time to reflect before composing their messages. However, by reducing the pressure to answer, it is easier for a student to drop out of the group [Graham et al, 1999]. Mediators demand regular contributions in an appropriate timeframe to avoid dispersion. Coordination support provided by the Follow-Up reports helps to identify who is participating or not.

It is necessary to coordinate the ensuing activities in order to enforce the fulfilment of the commitments assumed during communication. Coordination organises the group in a manner that avoids the loss of communication and cooperation efforts and ensures that the tasks are carried out in the correct order, at the right time and in compliance with the restrictions and objectives [Raposo et al., 2001].

4. COORDINATION: MANAGING TASKS INTERDEPENDENCIES

Karl Marx defined collaborative work as multiple individuals working together in a planned manner in production processes that are linked to each other (cited in [Bannon & Schmidt, 1991]). In CSCW the notion of planning present in Marx's definition is carried out by the so-called articulation work, which is the additional effort that is necessary for achieving collaboration from the sum of individual labour. In a more ample definition, coordination is synonymous with articulation work.

Coordination involves the pre-articulation of the tasks, their management and post-articulation. Pre-articulation involves actions that are necessary to prepare collaboration, normally concluded before collaboration begins, such as the identification of the objectives, the mapping out of these objectives into tasks, the selection of participants, the distribution of tasks among them—all derived from commitments assumed during communication. The post-articulation phase occurs after the end of the tasks and involves the evaluation and analysis of the tasks that were carried out and the documentation of the collaborative process. The management of the carrying out of the tasks is the most dynamic part, needing to be renegotiated in an almost continuous fashion throughout collaboration. Looking just at this specific aspect of coordination, it can be defined as the act of managing interdependencies between tasks that are carried out to achieve an objective [Malone & Crowston, 1990].

In this paper a collaborative activity is defined as a set of tasks carried out by several members of the group in order to achieve a common objective. Tasks are the building blocks of the collaborative activities, linked by interdependencies. Tasks can be atomic or composed of sub-tasks. A group of sub-tasks can be considered to be a task on a higher abstraction level when it does not present interdependencies with tasks that are external to this group. This ensures the modelling of collaborative activities on several levels of abstraction [Raposo & Fuks, 2002].

As illustrated in Figure 8, commitments generated during communication entail collaborative learning activities, and coordination mechanisms manage the interdependencies between the tasks carried out by the members of the group. A coordination mechanism is defined as “a specialized software device, which interacts with a specific software application so as to support articulation work” [Schmidt & Simone, 1996].

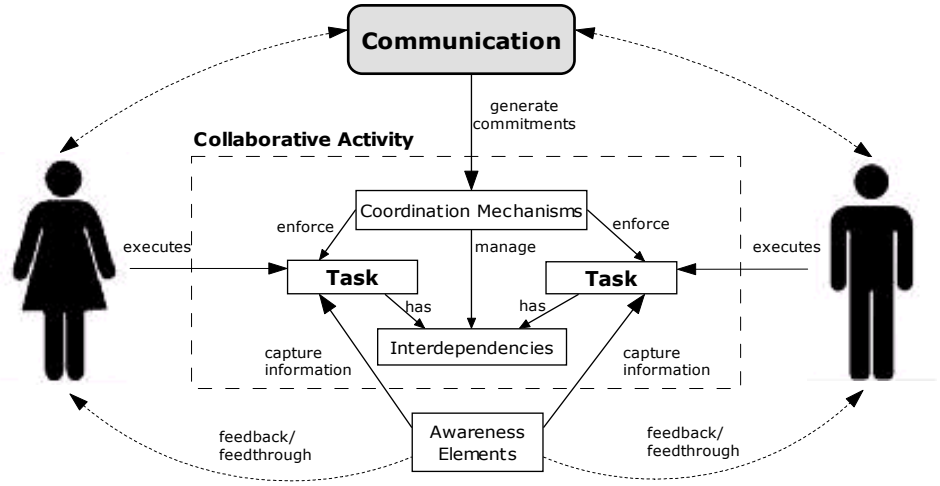

Figure 8. Modelling the coordination.

Some computer supported collaborative activities, so called loosely integrated collaborative activities, such as those realized by means of chats or audio and videoconferences, are deeply associated with social relations and generally are satisfactorily coordinated by the standing social protocol, which is characterized by the absence of any computer supported coordination mechanism among the tasks, trusting users' abilities to mediate interactions. Coordination, in these situations, is culturally established and strongly dependent on mutual awareness. Through awareness information, the participants detect changes in plans, 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 tightly integrated collaborative activities whose tasks are tightly interdependent, as the name suggests. They require sophisticated coordination mechanisms in order to be supported by computer systems. A possible way to deal with tightly integrated collaborative activities is to separate "the work devoted to activity coordination and coordinated work, i.e., the work devoted to their articulated execution in the target domain" [Simone, Mark & Giubbilei, 1999]. One of the advantages of the separation of task and interdependency is the possibility of altering coordination policies by simply altering the coordination mechanisms for the interdependencies, without the necessity of altering the core of the collaborative system. Additionally, interdependencies and their coordination mechanisms may be reused. It is possible to characterise different kinds of interdependencies and identify coordination mechanisms to manage them [Malone & Crowston, 1994].

The coordination can take place on the temporal and object levels [Ellis & Wainer, 1994]. On the temporal level, the coordination defines the sequencing of the tasks that make up an activity. On the object level, the 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 pair of tasks. The set of temporal interdependencies extends the temporal relations defined by J. F. Allen [Allen, 1984]. He proved that there is a set of seven primitive and mutually exclusive relations that could be applied over time intervals. The adaptation of Allen's primitives to the context of collaborative activities takes into account that any task will take some time to be

performed. A task 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 the Allen 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 guarantee that the interdependency will be respected. The *enables* operator represents the passive interpretation, while *forces* represents the active one. These operators may be applied on 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 not task A. 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 in the execution of task A, which may not be executed anymore if task B has already started its execution. There is no restriction on the execution of B (B does not have to wait for the execution of A, as would happen with the situation given by *fa enables ib*).

Resource-related interdependencies may be represented by combinations of temporal relations. However, considering resource management dependencies independently of temporal ones, a more flexible model is created, allowing the designer to deal with each kind of dependency separately. This kind of interdependency deals with the distribution of resources among the tasks. Three basic resource management dependencies are defined:

Sharing – a limited number of resources must be shared among several tasks;

Simultaneity – a resource is available only if a certain number of tasks request it simultaneously. It represents, for instance, a machine that may only be used with more than one operator;

Volatility – indicates whether, after the use, the resource is available again. For example, a printer is a non-volatile resource, while a sheet of paper is volatile.

This coordination model may be used to create coordination mechanisms that encompass both temporal and resource interdependencies, like the example using Petri Nets presented in [Raposo & Fuks, 2002].

4.1. Coordination in the ITAE course

During the conference argumentation phase in the ITAE course, the conference leader shares the coordination duties with the course mediator, encouraging group members to post messages to the conference. Figure 9 shows the instantiation of the coordination model to the conference activity.

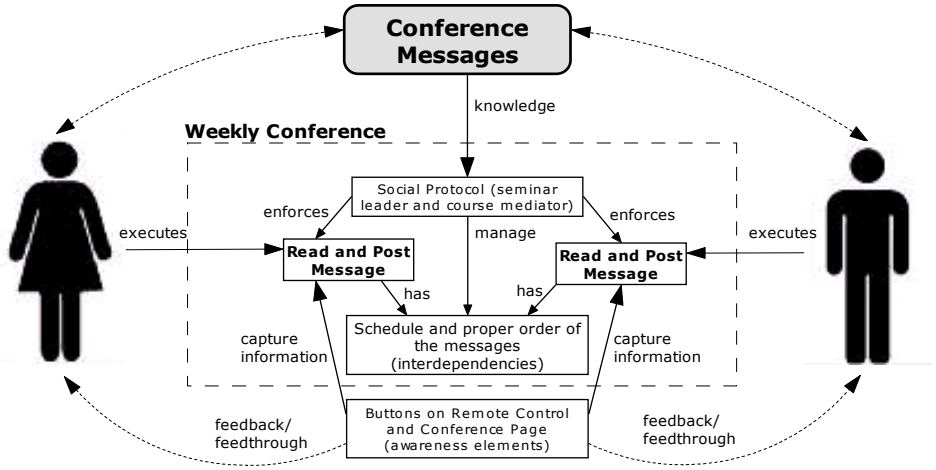

Figure 9. Instantiation of the coordination model for ITAE.

The tasks that learners have to perform during the Conference are to read and post messages. The interdependencies among theses tasks are characterized by the schedule and the proper order of the messages. There is no computer supported coordination mechanism in this learning activity. It is up to the mediator to enforce the right use of categories on the messages and their correct positioning. To carry out this task, the mediator has specific expression elements to change the categories, hence modifying the cooperation objects.

To avoid contributions that do not add value to the group, each message is individually assessed and commented upon. Follow-Up reports make it clear who is not participating or who is participating at an inadequate level. The problems detected in the contributions are commented on 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.

Figure 10 shows the sequence of tasks of the ITAE course from a learner point-of-view. It starts with the course and members introduction. Then, there are eight periods of 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.

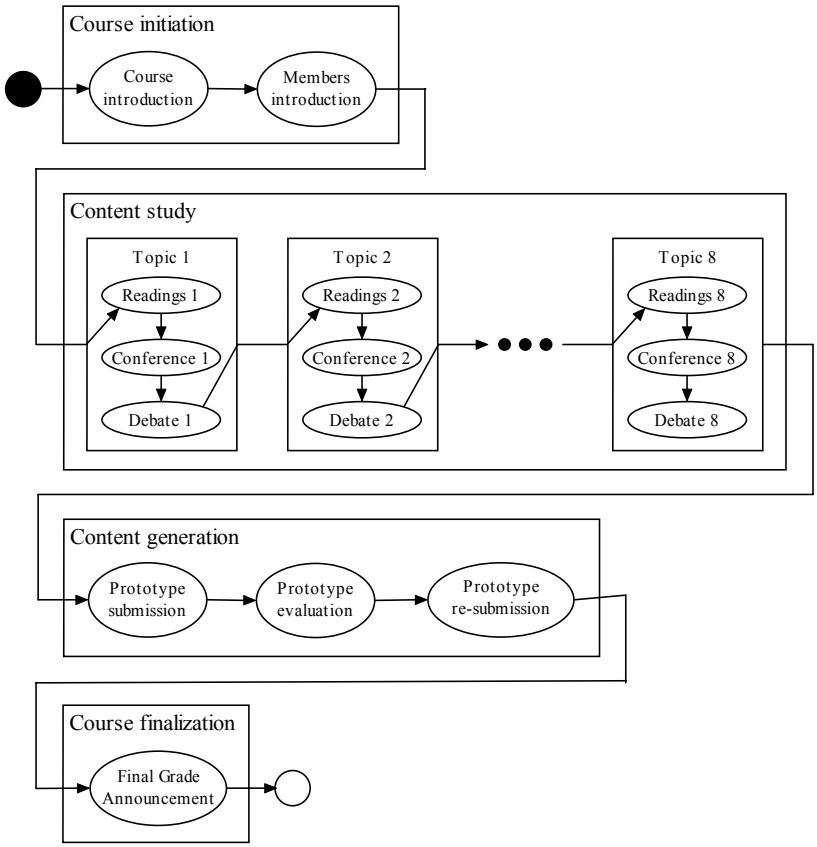

Figure 10. Workflow of ITAE learning activities: Overall view.

The overall view of the course workflow indicates that it is necessary to have a hierarchical representation of tasks. In Figure 10, the composed tasks represented inside the boxes are internally subdivided into more detailed sub-tasks. For example, content generation by learners has three sub-tasks, namely, prototype submission, its evaluation by other learners, and prototype re-submission. Reaching an even lower abstraction level, it is necessary to decompose some tasks into atomic ones and detail the interdependencies among these tasks.

Before presenting the subdivision of the conference activity into atomic tasks, it is necessary to raise an issue that appears in the workflow of Figure 10. When working with conventional workflows, the non-execution of a previous task by the person who is supposed to do it forces the workflow to stop in a certain state. In the ITAE course, there is an additional timing factor that determines when a task will be declared finished. For example, if a learner cannot participate in the learning activities of a given week, she will miss a topic. In the following week, the next topic will be assigned to her workflow.

Figure 11 presents the expanded workflow for the weekly conference. This collaborative learning activity involves three roles. The mediator selects the learner who will be the conference leader in that week and initialises 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 mediator, who also finalises the session, evaluates each of the messages.

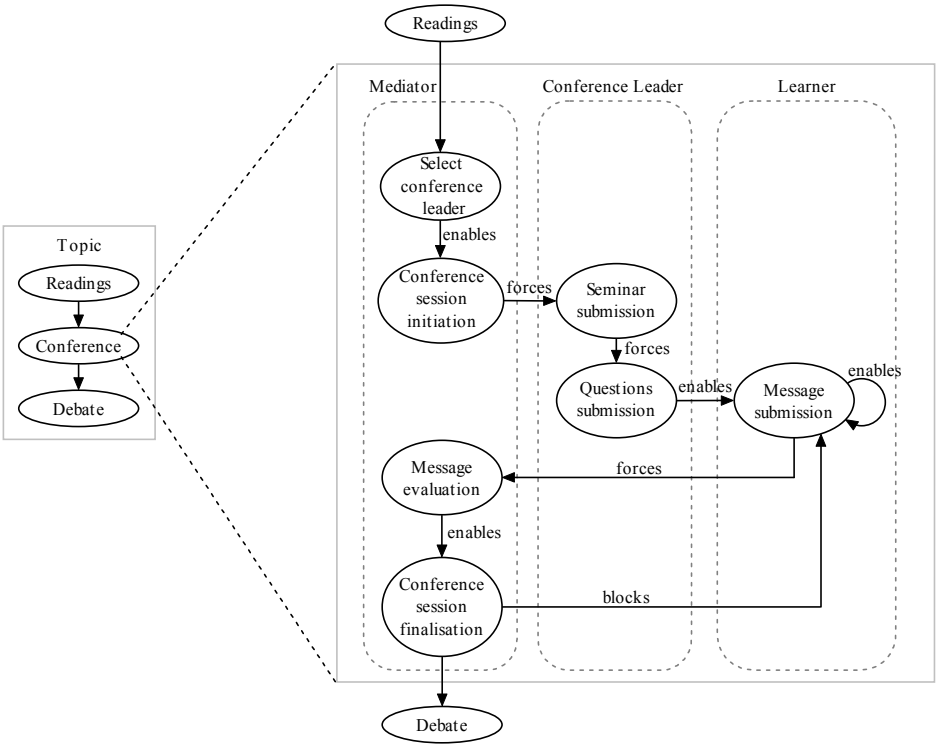

Figure 11. Expanded workflow of an ITAE conference.

The interdependencies among the tasks presented in Figure 11 are expressed by *enables*, *forces*, and *blocks* operators. The selection of the conference leader, for instance, enables the session initialisation, meaning that the mediator cannot initialise a session without previously selecting a leader; however, the leader selection does not force the mediator to initialise 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 initialisation by the mediator forces the leader to submit her seminar, followed by questions. The *blocks* operator is used in Figure 11 indicating that learners cannot submit messages after the mediator finalises the conference session.

Applying the same coordination approach to the AulaNet service Lesson Plan, it is possible to create different workflow paths for different learners or groups over a single course. For example, a more advanced class may skip introductory contents, while novices should study them. Another possible example is to offer the same course for undergraduates and postgraduates, assigning more complex tasks for the latter.

Communication and coordination, although vital, are not enough: "it takes shared space to create shared understandings" [Schrage, 1995]. Given that commitments were assumed during communication, and coordination was needed to manage the tasks needed to realize the commitments, according to the 3C model, it is also necessary to offer a shared workspace where cooperation will take place.

5. COOPERATION: FROM AWARENESS TO WORK

Cooperation is the joint operation in the shared workspace. Group members cooperate by producing, manipulating and organising information and building and refining cooperation objects, such as documents, spreadsheets, artwork, etc. Expression elements are the means for acting upon cooperation objects, while awareness elements display the results of a participant action (feedback) and the action of their colleagues (feedthrough), as shown in Figure 12.

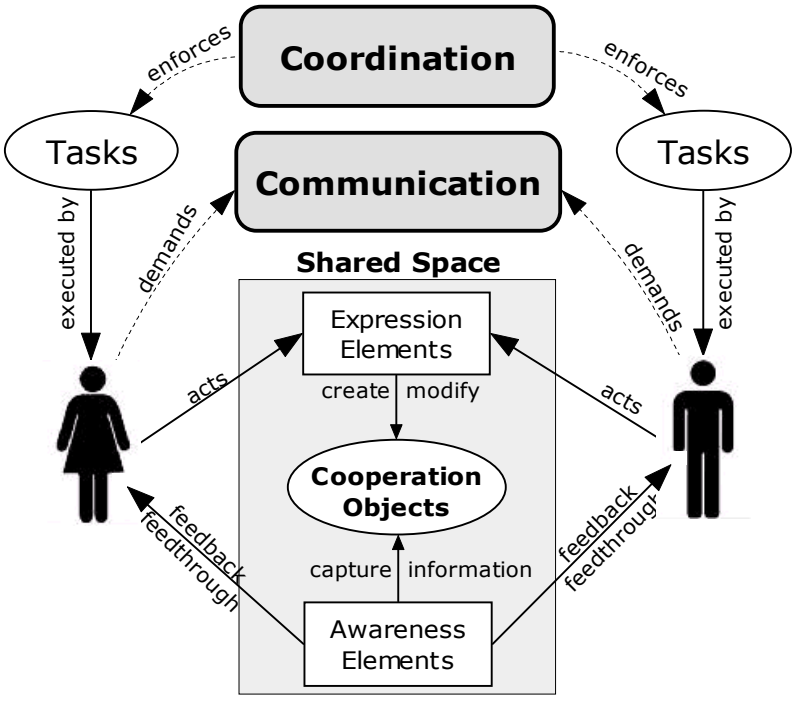

Figure 12. Modelling cooperation.

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 by being able to see and hear events or actions such as people arriving or leaving, phones ringing, sighs of frustration, conversations, etc. "These are all things we can potentially use to coordinate our work and play together and we often do so with little conscious effort" [Fitzpatrick et al., 2002]. On the other hand, in a digital environment, 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 distraction that affects face-to-face collaboration.

Individuals seek in awareness elements the necessary information 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 as to make it possible to provide assistance in terms of their work whenever it is convenient and needed.

The designer of a digital environment must identify what awareness information is relevant, how it will be obtained, where awareness elements are needed and how to display and give individuals control over them. Excessive information can cause overload and disrupt the collaboration flow. To avoid disruption, it is necessary to balance the need to supply

information with the care to avoid distracting the attention required to work. The supply of information in an asynchronous, structured, filtered and summarized form accomplishes this balance [Kraut & Attewell, 1997]. The big picture should be supplied and individuals may 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

On AulaNet, services are available through a menu that resembles a remote control unit (Figure 13), which intents to domesticate the computer and transform it into a home appliance for learning purposes. The remote control unit also transfers to the learner—at least up to a certain point—control over the learning process. The control traditionally wielded by the teacher in the classroom is substituted 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 13. Learner interface of the ITAE course on the AulaNet environment.

On the upper part of the remote control, the course code offers an individual awareness element for localization and context. 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, which changes colour to provide information about services. A blue button indicates the service that the learner has selected. A dark orange button indicates a service where no changes have taken place since her last access. Upon moving the mouse over a light orange button (highlighted in Figure 13), the number of items upon which some action should be taken (items not read or not solved) appears.

Awareness interconnects the three C's of the 3C model [Gerosa et al, 2003]. For example, one may use the Message to Participants service to know which learners are currently

connected to the environment and to start a conversation with one of them. In another situation, the course coordinator is notified when learners submit an educational content through the *Learner Co-authorship* cooperation service. Eventually, the coordinator will evaluate and, perhaps, incorporate the new learning object into the course repository. During this process the coordinator may exchange messages with learners to clarify some specific point or to ask for improvements.

Normally, the educational content submission is done at the end of the first part of the course, when learners are organised into small groups (two to three learners), based upon profiles they previously submitted. In the profile, learners select their skills and interests in each one of the topics on the course. Based upon this information, AulaNet suggests group formations that best satisfy the criteria defined by mediators (degree of skill and interest) [Cunha, Fuks & Lucena, 2003]. Groups organise themselves in order to collaboratively generate interactive multimedia educational content about a topic defined by the mediator and submit a content prototype by a given deadline. Subsequently, a period of collaborative peer-evaluation begins. Members of at least three other groups evaluate each group's content. This evaluation takes place in Conferences created specifically for this purpose, where learners discuss problems they found in the prototypes. Once this period is concluded, groups are given a new deadline to present a revised version that incorporates the contributions of their colleagues.

The Conference service provides a shared space where learners cooperate by producing and refining knowledge through an argumentation process. Learners generate new cooperation objects, in this case, conference messages, acting on expression elements, such as the ones shown in Figure 14. In the Conference shared space, awareness elements display information about the cooperation objects, including their authorship, date, category, subject and the assessment made by course mediators. Acting by means of expression elements, learners select the category and fill in the subject and the body of the message.

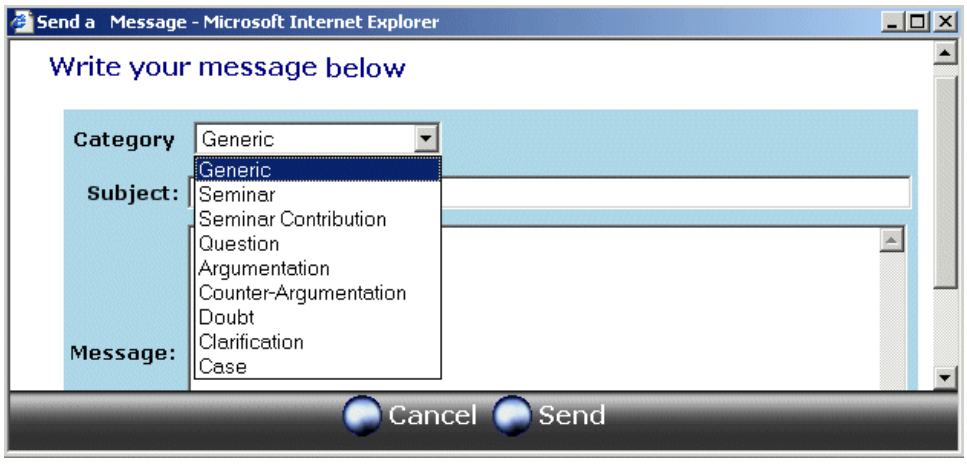

Figure 14. Posting a message to the conference.

The instantiation of the cooperation diagram to the ITAE course's conference collaborative learning activity is shown in Figure 15. Learners act on the expression elements to construct and handle conference messages (the cooperation objects) and receive feedback from their actions and feedthrough of their colleagues' actions by awareness elements. The activity is enforced by the conference leader and by the course mediator.

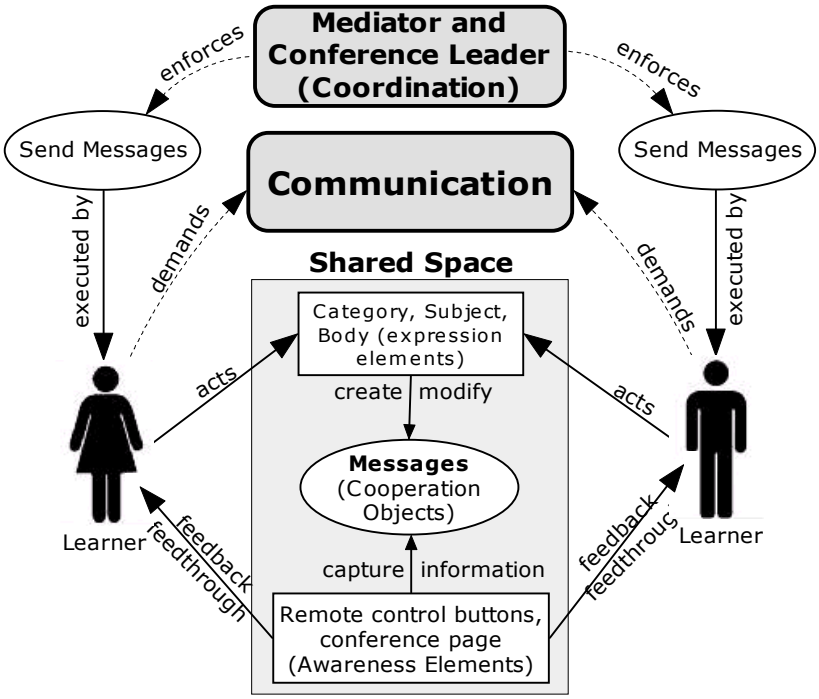

Figure 15. Instantiation of the cooperation model for ITAE.

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

6. DEVELOPMENT ISSUES

After refining the 3C collaboration model for supporting the domain analysis phase, the study of the groupware engineering development cycle (Figure 1) is resumed in this section with a view to AulaNet development. Regarding the requirements analysis phase, given the incremental development of AulaNet, requirements were elicited based on the use of previous versions. The same may be said about the testing phase. The widespread use of the environment propitiated failure detection and usability improvement. The ITAE course, which requires intensive collaboration support from AulaNet services, was instrumental for both phases.

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

This scenario is suitable for the application of component-based development techniques, which provides 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 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, Coutaz & Nigay, 1997].

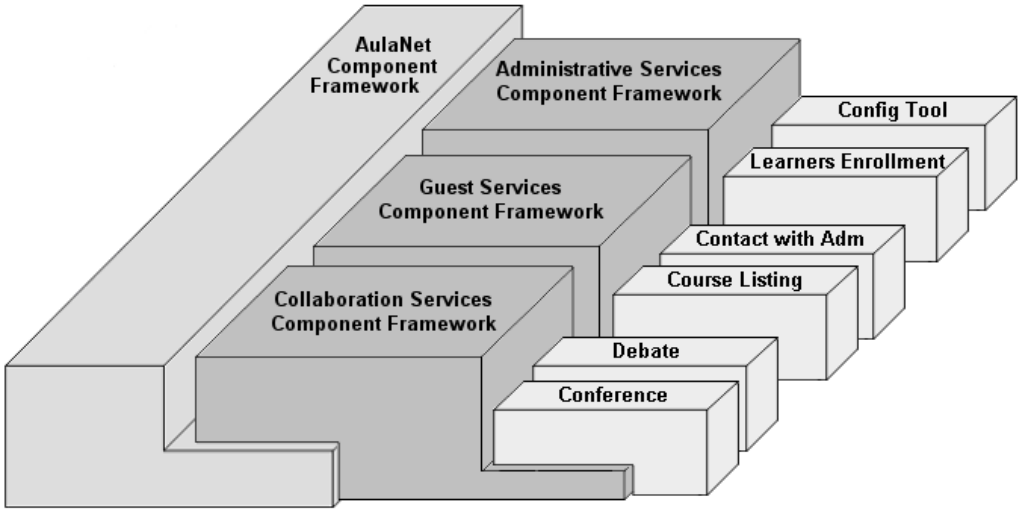

Figure 16. AulaNet system architecture (for the sake of clarity only a few services were listed).

In the AulaNet architecture, the AulaNet component framework defines the general functionalities common to all services, like the management of services interaction and data sharing. Currently, there are three different families of services: collaboration, administrative and guest services, which corresponds to components frameworks that deal with characteristics specific to each service (Figure 16). 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, like the environment FAQ, contact with the administrator, bug report, course listing and enrolment, etc.

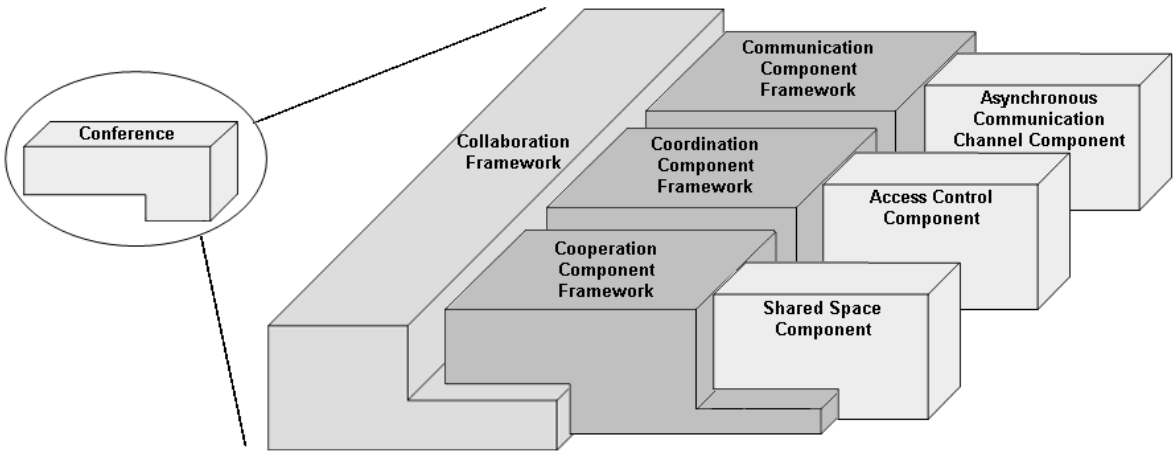

Figure 17. Architecture of a collaboration service.

Current AulaNet services are also developed using a component framework-based architecture, as can be seen in Figure 17. There is a common structure implemented by the collaboration framework, which defines the skeleton of the service, and plugged into this framework, there are the communication, the coordination and the cooperation component frameworks, which gives support to each C. Class frameworks are used to implement components, that are plugged into the corresponding C-framework and implement the specific functionalities of the service.

For example, among communication class frameworks, the developer implements components for synchronous and asynchronous communication, message transmission, commitment management, etc. Using coordination class frameworks, components for task management, participation follow-up, workflow, etc. are implemented. Using cooperation class frameworks, components for managing the shared space and its awareness elements, versions management, among others are implemented. In the framework instantiation, the groupware developer implements the frameworks' hot spots specific to each kind of component. In the next subsection, the AulaNet Debate service is used to exemplify the service composition using the instantiated components.

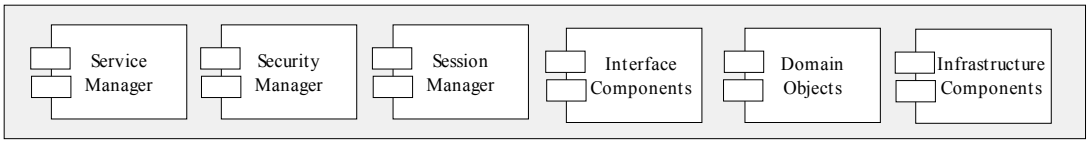

Figure 18. AulaNet component framework.

The AulaNet component framework (Figure 18) comprises six main components: Service Manager, Security Manager, Session Management, Interface Component Library, Domain Object Library and Infrastructure Component Library. The Services Manager takes care of version control, name control, new services installation, services aggregation, communication between services, remote localization, etc. The Security Manager handles identification, authentication and access control of 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 services, facilitating the update of an element; enforcing interface consistency and standardization; making use of specific components for mobile devices possible; and allowing for changing the entire environment skin. This library also stores services interface texts, because of multiple languages and text customisation. The Domain Object Library contains objects that represent the data model that is shared by services, enabling the reuse of the data model and also persistence of shared objects and tools. This library contains mechanisms for replication and concurrency control. To facilitate the services implementation and to group features that are common to different services, AulaNet has an Infrastructure Component Library, which supports the sending of messages, error handling, among others.

A thin client strategy was adopted on AulaNet architecture; that is, most of the processing takes place on the server, leaving to the client the task of exhibiting 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 Web browser, users interact with AulaNet and its services. Some services—such as the Debate—require components being executed on the

client side. For these cases, the architecture supplies the client/server communication resources through applets running on the client machine.

J2EE[2] provides resources for components integration and distribution in more than one AulaNet server, improving environment scalability. Remote connectivity is implemented by J2EE technologies, like JNDI, JMS and RMI. In addition, J2EE provides resources for user authentication, permissions controlling and session management, serving as a base to component frameworks implementation.

Component-based development makes it possible to have a flexible set of services that can be used within AulaNet. The environment offers 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 a set of interfaces that should be respected to enable communication between AulaNet and its services. This also allows for using tools that were not originally developed to be used with AulaNet by implementing some classes that satisfy the required interfaces and mediates the communication between the environment and tool.

6.1. The Debate Service Case Study

AulaNet services are composed of components plugged into 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 implements synchronous communication protocols, and a cooperation component, which implements the shared space, as can be seen in Figure 19.

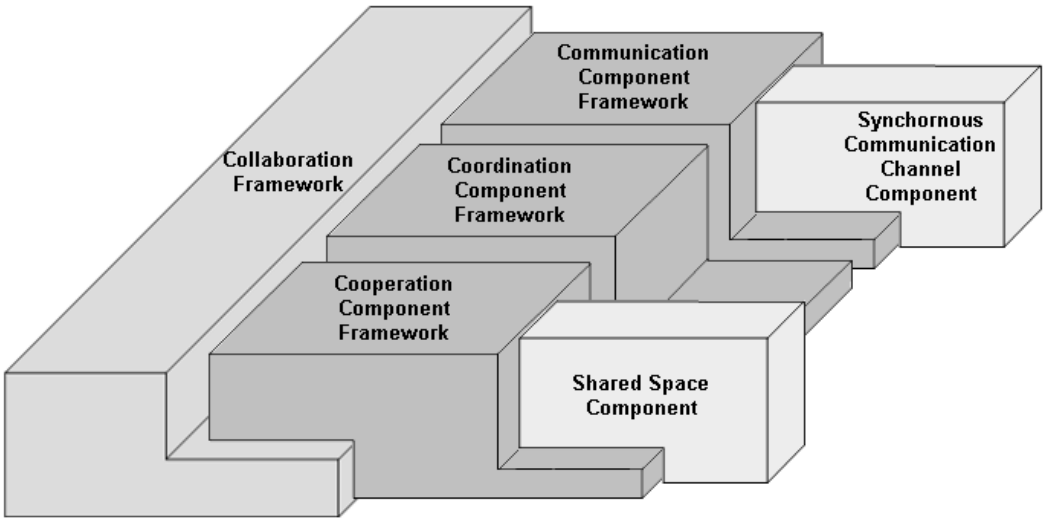

Figure 19. Implementation of the Debate component: Mediated Chat 1.0 (MC1).

In Figure 20, the MC1 user interface is presented. It is a plain chat tool, holding an expression element, where learners type their messages; and awareness elements, where messages are posted and the listing of learners present in that chat session.

[2] http://java.sun.com/j2ee

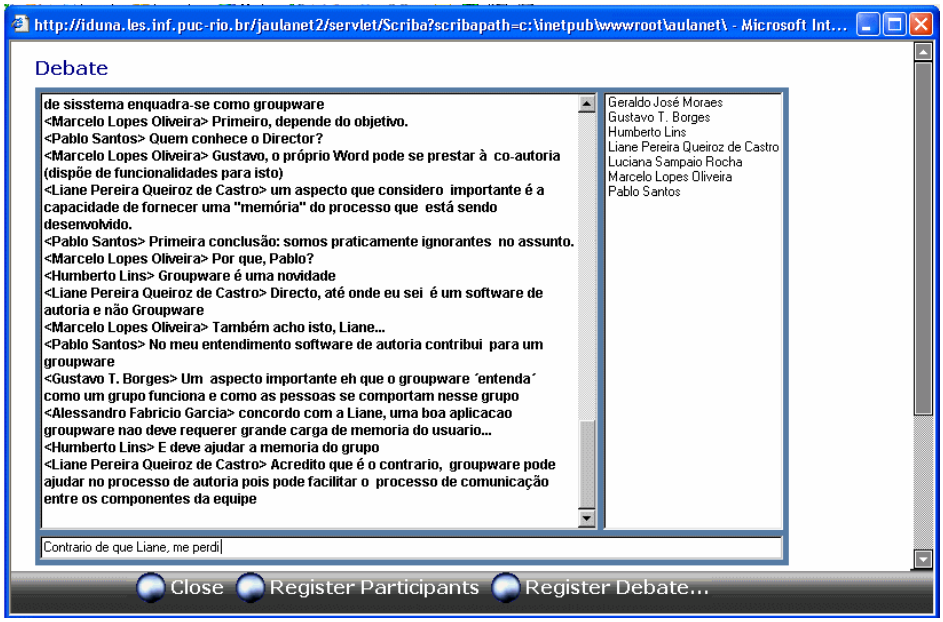

Figure 20. MC1 Debate service mediator interface.

Given the evolutionary nature of the ITAE course, the Debate procedure changed and, unfortunately, the MC1 tool could not support it anymore. Figure 21 shows the changed workflow of a Debate. The tasks are organised as follows. The mediator declares 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, mediators declare the end of the session and later on appraise learners' participation.

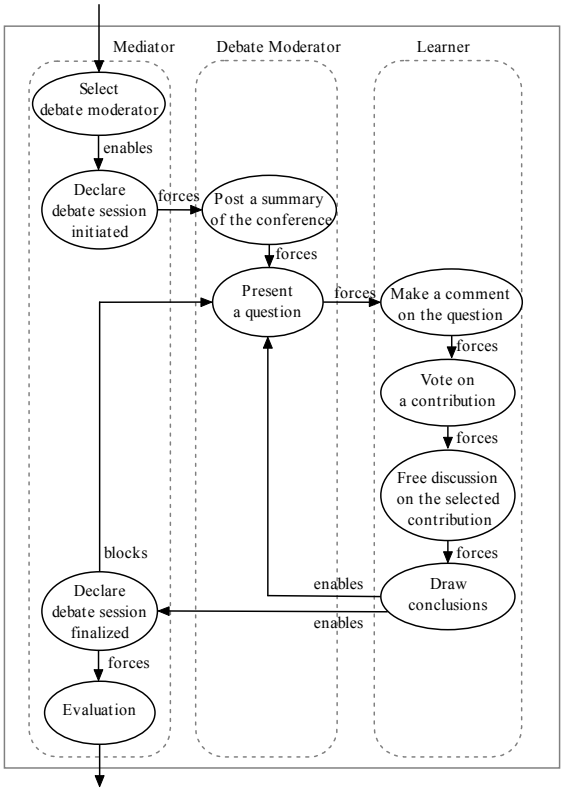

Figure 21. Expanded workflow of a debate.

For that purpose, in the current version of the Debate service presented in Figure 22, denominated Mediated Chat 2.0 (MC2), the shared space was enhanced by new awareness elements, like session title, time stamp of the messages, identification of mediators and participation order. Coordination mechanisms incorporated into the Debate service are used to implement the workflow modelled in Figure 21.

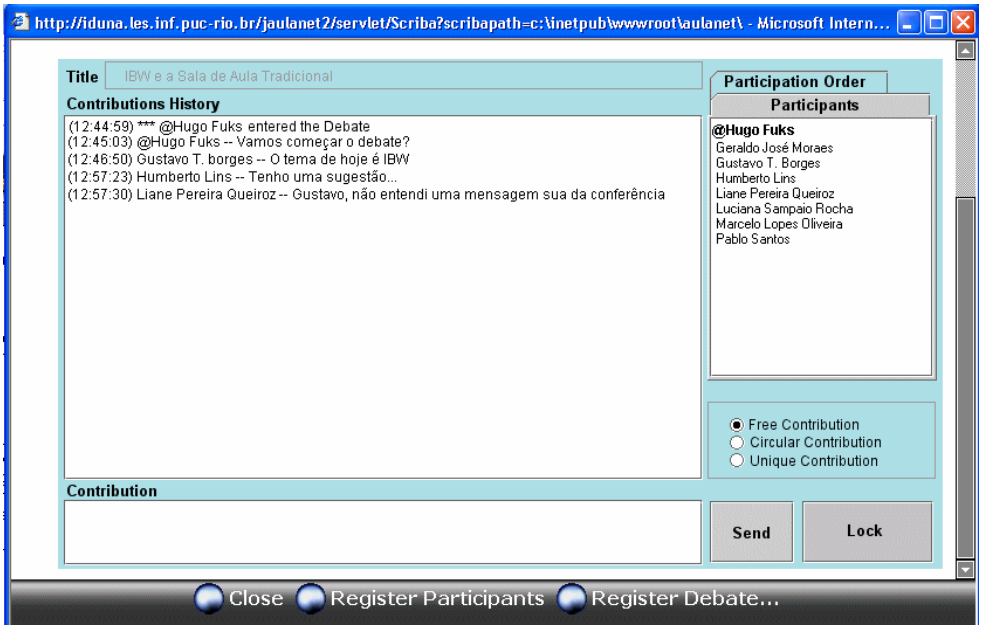

Figure 22. MC2 Debate service mediator interface.

The *circular contribution* is used on "Commenting on the question" task; *unique contribution* is used on "Vote on a contribution"; and *free contribution* is used on "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, thus avoiding parallel conversations.

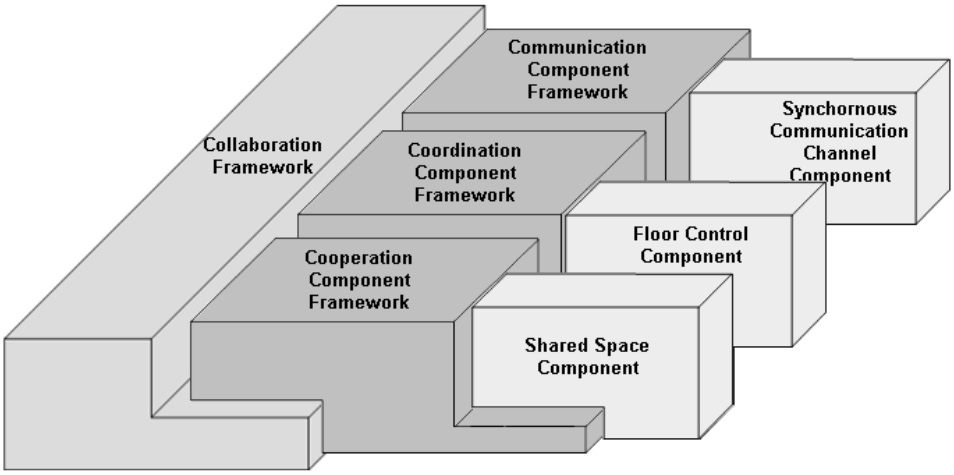

Figure 23. Implementation of the Debate component: Mediated Chat 2.0 (MC2)

The Mediated Chat 2.0 (MC2) was implemented with the same communication component of Mediated Chat 1.0, which implements synchronous communication protocols; a new cooperation component, which implements the shared space enhanced by new awareness elements; and a coordination component, which implements the floor control techniques, as can be seen in Figure 23.

7. CONCLUSION

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 provide a common language for representing and describing the collaboration aspects of a group. Using a groupware system architecture and component frameworks 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 resources sharing, inter-components communication, etc.

It should be remarked that, as pointed out by [Laurillau & Nigay, 2002], although bringing the separation of the aspects of the collaboration model to the software architecture is a good choice, doing the same to the software interface can lead to user confusion and misunderstandings. The collaboration elements should be harmoniously combined in the user interface, positioned near the objects that they may affect.

The application of the 3C collaboration model is 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 collaboration model. Communication, coordination and cooperation functionalities are directly mapped into the

implementation of AulaNet collaboration services. The redesign of the AulaNet Debate service illustrates this mapping and the modularity achieved using the component system architecture.

Finally, in the ITAE course, learners are encouraged to work in groups, to seek updated information, to argue, to take on and accomplish commitments—at the end of the day, to communicate, coordinate and cooperate.

ACKNOWLEDGMENTS

The AulaNet project is partially financed by the Fundação Padre Leonel Franca and by the Ministry of Science and Technology through its 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. Thanks also to Mariano Gomes Pimentel and Leonardo Magela Cunha.

REFERENCES

Alejandro, F., Torsten, H., Jessica, R. & Till, S. (2002): Three Groupware Patterns from the Activity Awareness Family, *Proceedings of Seventh European Conference on Pattern Languages of Programs – EuroPLoP 2002*, Irsee, Germany.

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

Baker, K., Greenberg, S. & Gutwin, C. (2001): Heuristic Evaluation of Groupware Based on the Mechanics of Collaboration. In M.R. Little and L. Nigay (Eds) *Engineering for Human-Computer Interaction*, 8th IFIP International Conference, EHCI 2001, Toronto, Canada, Lecture Notes in Computer Science Vol 2254, p123-139, Springer-Verlag.

Bannon, L.J. & Schmidt, K. (1991): CSCW: Four Characters In Search of A Context. In J.M. Bowers and S.D. Benford (eds): *Computer Supported Cooperative Work*, North-Holland: Elsevier Science Publishers B. V., Holland, pp. 3-16.

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. Springer-Verlag, p. 134-146

Boehm, B.W. (1988): A Spiral Model of Software Development and Enhancement, *IEEE Computer*, V. 21, N. 5, p. 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, 1975.

Calvary, G., Coutaz, J., Nigay, L. (1997): From Single-User Architectural Design to PAC*: a Generic Software Architectural Model for CSCW. *Proceedings of CHI'97*, Atlanta, March 1997, pp 242-249.

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, June 30 - July 2, Rhodes - Greece, pp 321-326.

Demarco, T. and Lister, T. (1999): *Peopleware: Productive Projects and Teams*. Dorset House Publishing, USA.

Dewan, P. (1999): Architectures for Collaborative Applications, *Computer Supported Cooperative Work*, Beaudouin-Lafon (ed), John Wiley & Son Ltd, p. 169-194.

Dourish, P. & Belloti, V. (1992): Awareness and Coordination in Shared Workspaces. In J. Turner and R. Kraut (eds): *Proceedings of The Conference on Computer Supported Cooperative Work*, Toronto, Ontario, October 1992, ACM Press, USA, pp. 107-114.

Edutools (2003): http://www.edutools.info (date 10/10/2003)

Ellis, C.A. & Wainer, J. (1994): A Conceptual Model of Groupware, In T. Malone (ed): *Proceedings of The Conference on Computer Supported Cooperative Work*, Chapel Hill, NC, October 1994, ACM Press, USA, 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, V. 11, N. 3-4, Shmidt, K., Heath, C. & Rodden, T. (eds), Kluer Academic Pub, 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., Ryan M., and Sadler, M. (1989): Outline of a Commitment Logic for Legal Reasoning. Proceedings of 3rd International Conference on Logics, Informatics and Law, V2, Florence, Italy, 391-405.

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

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*, V. 14, No. 3, AACE, USA, pp. 315-332.

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, May, 1999, pp. 32-45.

Groupware Patterns Swiki (2003), http://swiki.darmstadt.gmd.de/gw-patterns (date 10/26/2003)

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

Grudin, J. (1994): Groupware and Social Dynamics: Eight Challenges for Developers. *Communications of The ACM*, vol. 37, no. 1, pp. 92-105.

Gutwin, C. & Greenberg, S. (2000): The Mechanics of Collaboration: Developing Low Cost Usability Evaluation Methods for Shared Workspaces. *IEEE 9th International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises -WETICE (2000)*, p. 98-103.

Gutwin, C. & Greenberg, S. (2002): A Descriptive Framework of Workspace Awareness for Real-Time Groupware, Computer Supported Cooperative Work, V. 11, N. 3-4, Shmidt, K., Heath, C. & Rodden, T. (eds), Kluer Academic Pub, pp. 411-446

Hummes, J. & Merialdo, B. (2000): Design of Extensible Component-Based Groupware, *Computer Supported Cooperative Work (CSCW)* 9(1): 53-74; Jan 2000.

IEEE (1991): IEEE Standard Glossary of Software Engineering Technology, USA, 1991.

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.), Cap 3, Springer-Verlag, London, 2003.

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,

Laufer, C. & Fuks. H. (1995): ACCORD: Conversation Clichés for Cooperation, *Proceedings of COOP'95 – First International Workshop on the Design of Cooperative Systems*, Antibes-Juan-les-Pins, France, INRIA Press, 351-369.

Laurillau, Y. & Nigay, L. (2002): Clover architecture for groupware, Proceedings of the ACM Conference on Computer Supported Cooperative Work, Louisiana, USA, p. 236 - 245

Mackenzie, J. (1985): No Logic before Friday, *Synthese*, V.63, pp 329-341.

Malone, T.W. & Crowston, K. (1990): What is Coordination Theory and How Can It Help Design Cooperative Work Systems?. In F. Halasz, (ed): *Proceedings of The Conference on Computer Supported Cooperative Work*, Los Angeles, USA, October 1990, ACM Press, USA, pp. 357-370.

Malone, T.W. & Crowston, K. (1994): The Interdisciplinary Study of Coordination. *ACM Computing Surveys*, vol. 26, no. 1, pp. 87-119.

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, September 1994, 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*, January 5-8, 1999, Maui, Hawaii.

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

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.

Raposo, A.B., Magalhães, L.P., Ricarte, I.L.M. & Fuks, H. (2001): Coordination of Collaborative Activities: A Framework For The Definition of Tasks Interdependencies. In M. R. S. Borges, J. M. Haake and U. Hoppe, (eds.): *Proceedings of The 7th International Workshop on Groupware - CRIWG*, Darmstadt, Germany, September 2001. IEEE Computer Society, USA, pp. 170-179.

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

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

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, p. 157-176

Schmidt, K. & Simone, C. (1996): Coordination mechanisms: Towards a conceptual foundation of CSCW systems design. *Computer Supported Cooperative Work,* 5(2-3), 1996, 155-200.

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.

Simone, C., Mark, G. & Giubbilei, D. (1999): Interoperability as a Means of Articulation Work. *Proceedings of WACC: Int. Conf. on Work Activities Coordination and Collaboration*, 39-48.

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, Heraklion, Crete, Greece. IFIP Conference Proceedings 150, Kluwer, p. 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*. Addison-Wesley, USA.