Proc. of the 9th International Conference on Computer Supported Cooperative Work in Design (CSCWiD), p. 329-333. Coventry, U.K., May 2005.

Towards an Engineering Approach for Groupware Development: Learning from the AulaNet LMS Development

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

Catholic University of Rio
gerosa@inf.puc-rio.br

Abstract

This paper presents the AulaNet learning management system, its architecture and the collaboration model that guided its development and that was refined during this process. A case study of an online course indicates the necessity to have an architectural support for collaboration aspects and a collaboration-based engineering approach to groupware development. This approach, Groupware Engineering, is based on Software Engineering and on concepts originated in the field of CSCW.

Keywords

Collaboration model, groupware engineering, groupware architecture, component framework.

1. Introduction

Although the Internet offers advantages and facilities for teaching/learning, there are also many difficulties associated with its use. To create interactive Web-based content, for instance, teachers must deal with technologies that sometimes they don't master. To reduce these difficulties the LMS may separate content from navigation, allowing teachers to concentrate on the production of content using their preferred tools such as commercial text editors or slide presentation software, while leaving learner navigation support to the system. Additionally, integrated communication, coordination and cooperation services may be made available by the LMS to be available to courses, and reports may also be made available to facilitate learner participation follow up.

The AulaNet was created based on the above mentioned features. It is a freeware web-based LMS. Besides Portuguese, it is also available for download (http://www.eduweb.com.br) in English and Spanish versions. The AulaNet interface is presented in Figure 1.

Figure 1. AulaNet learner interface

In its first versions, AulaNet resources were subdivided into administrative, assessment and didactic services, which is a common approach in educational tools. Unfortunately, this approach led teachers who were using the system to teach in the traditional way: broadcasting information with no interaction among learners. Hence, services were reorganized based on the 3C collaboration model, which seems to be suitable to a collaborative learning approach. The AulaNet services are currently subdivided into communication, coordination and cooperation services, as can be seen in Figure 2 (The 3C triangle appears in [3]).

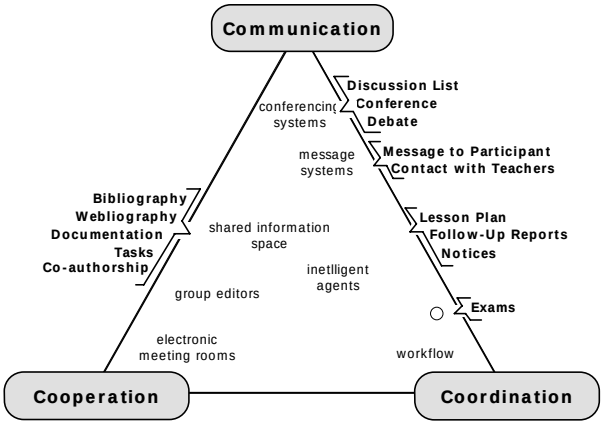

Figure 2. Classification of AulaNet services

The AulaNet has been developed through prototyping. Its developers are doctorate and masters degree candidates and undergraduate students at the Catholic University of Rio, who, besides maintaining it, use it in their theses, dissertations and monographs, implementing and testing concepts for their work. The AulaNet has grown and its features have been implemented on demand, making it hard to make any extension to the code and in need of a code restructuring.

Developing environments to support collaborative work is a complex job that requires an understanding of a number of fields of knowledge, such as information technology, psychology, sociology, cognition, etc. Environments are quite susceptible to breakdowns, in view of the fact that work processes between individuals are quite specific and evolve over time, and on the top of that, the adoption of groupware requires changes in attitudes and work habits [7]. Although there is no way to foresee all of the future demands, different groupware products share a number of characteristics.

This paper presents the groupware engineering approach used to re-structure AulaNet. This approach is based on component based development and on the 3C collaboration model.

2. The 3C Collaboration Model

The diagram shown in Figure 3 summarizes the main concepts of the 3C Collaboration Model. This diagram is an extension of models found in the literature, such as the model proposed by Ellis et al. [4] and the Clover design model [9].

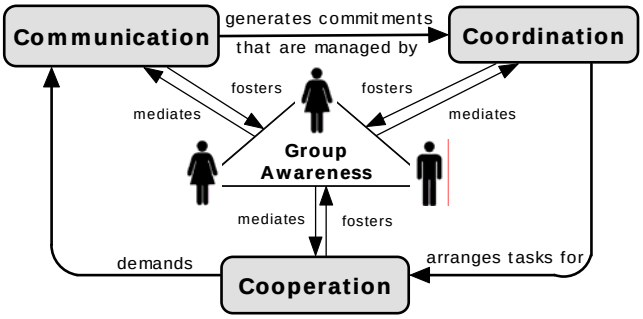

Figure 3. Overview of the 3C collaboration model

The communicational apparatus transmits and registers the information. Then, the group interprets the message, forcing an update their commitments and knowledge. Then, the group moves into an argumentation phase where they negotiate commitments and, therefore, their knowledge.

Next step is to coordinate the ensuing activities designed to enforce the fulfillment of the commitments. Coordination organizes the group to avoid the loss of communication and cooperation efforts and to ensure that the tasks are carried out in the correct order, at the right time and in compliance with the restrictions and objectives [12].

Coordination involves the pre-articulation of tasks, their management and post-articulation. Pre-articulation involves actions that are necessary to prepare collaboration, normally concluded before cooperation begins, such as the identification of the objectives and the mapping out of these objectives into tasks. 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 act of managing interdependencies between tasks that are carried out to achieve an objective [10].

Cooperation is the joint operation within the shared workspace. Group members cooperate by producing, manipulating and organizing information and building and refining cooperation objects. 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).

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. The shared space must be conceived in a way that group members could seamlessly move from awareness to work.

3. The AulaNet Architecture

The evolving nature of groupware make it suitable for the application of component-based development techniques, which provides the flexibility needed in projects with changing requirements [15]. In this situation, groupware services could be plugged and unplugged from the system.

The system architecture comprises component frameworks, which define overall invariants and protocols for plugging components.

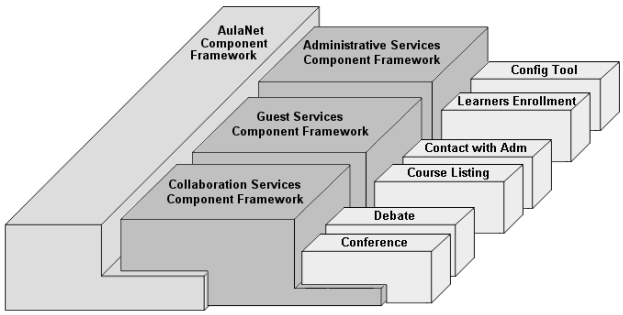

Figure 4. AulaNet system architecture

In the AulaNet architecture (Figure 4), the AulaNet Component Framework defines the general functionalities common to all services, like the management of services and data sharing. 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.

Moreover, AulaNet services are also developed using a component framework-based architecture. 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, each one giving support to each C. Class frameworks are used to implement components, that are plugged into the corresponding C-framework that implement the specific functionalities of the service.

For example, in a previous version of the AulaNet LMS, the Debate service was a plain chat tool, holding an expression element, where learners could type their messages; and awareness elements, where learners participating at the chat session were presented, as can be seen in Figure 5.

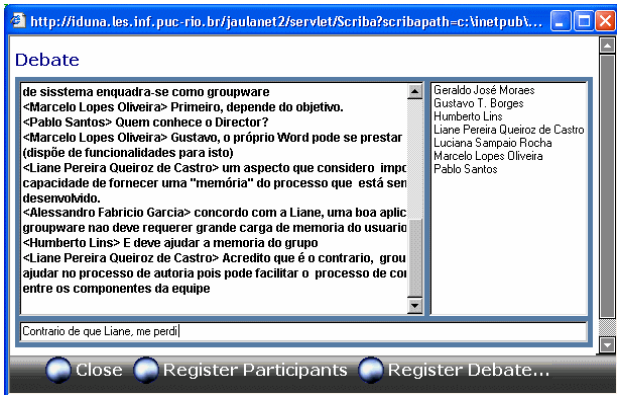

Figure 5. Debate Interface

This version of the Debate was implemented using a communication component, which implements synchronous communication protocols, and a cooperation component, which implements the shared space, as can be seen in Figure 6.

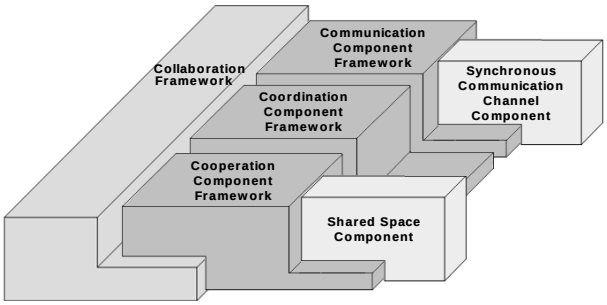

Figure 6. Expanded view of the Debate component plugged into the Collaboration Service Component Framework show in Figure 3

This version of the Debate service gives no computer support to coordination, leaving it to the standing social protocol. However, this is not always the case, because some courses use a well defined procedure to the debate activity, like the one shown in Figure 7, which represents the procedure adopted in a course [5].

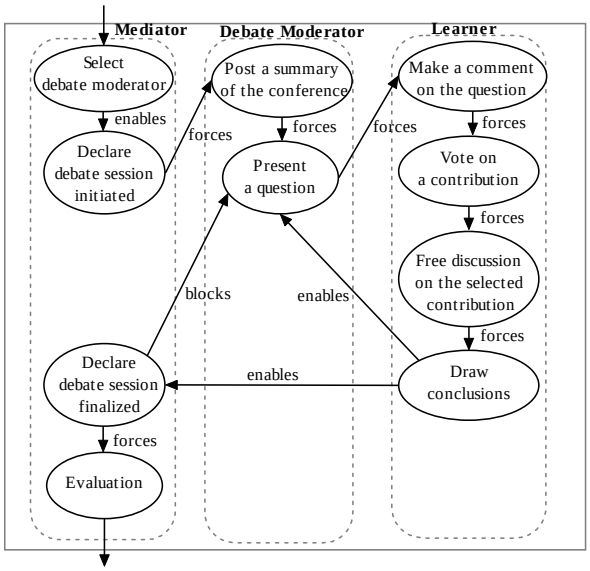

Figure 7. Expanded workflow of a debate

In this procedure, for each debate, the course mediator selects a learner to be the session moderator. It is also up to the mediator to declare the session initiated and finalized and to evaluate learners' participation. The debate moderator posts a summary of the discussion that took place during the week's conference and then poses three questions. For each question, each learner posts a comment, and after every learner has posted its comment, they vote and decide which one will be discussed. Then, a free discussion takes place. Before the moderator poses a new question, learners have to draw their conclusions.

In order to better support tightly integrated activities, like the one exemplified above, in the following version of the Debate service (presented in Figure 8), coordination mechanisms were implemented. Floor control, participation order and shared space blocking ability were added to the service. The shared space was also enhanced by new awareness elements, like session title, message timestamp and the identification of mediators.

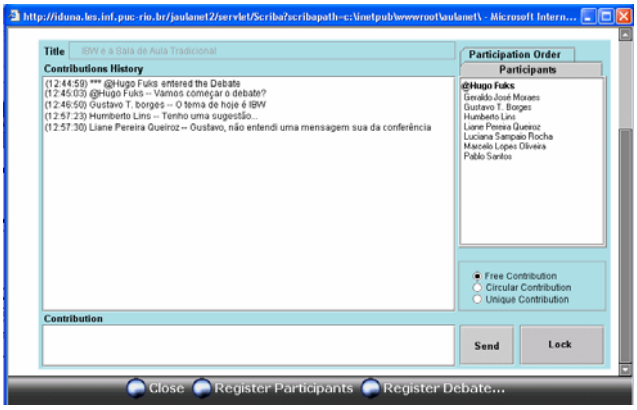

Figure 8. Debate service mediator interface

In this new Debate version, the same communication component was used, as the synchronous communication protocols and the characteristics of the messages did not change. The cooperation component, which implements the shared space, however, was enhanced by the new awareness elements mentioned above. The main difference is the insertion of a coordination component, which implements the floor control coordination mechanisms, as can be seen in Figure 9.

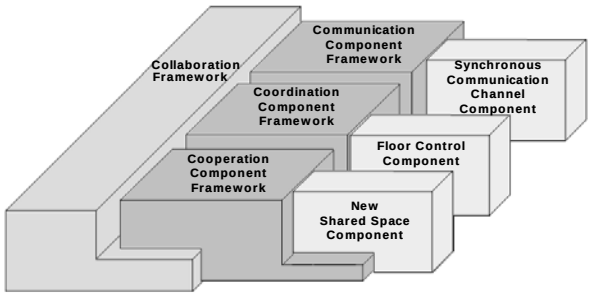

Figure 9. Implementation of the new Debate service

This example illustrates the benefits of having a component-based architecture that deals with the three Cs of the collaboration model, namely, communication, coordination and cooperation. Groupware Engineering combines the systematic development approach provided by Software Engineering together with the domain analysis given by the 3C model originated from the CSCW field.

4. Groupware Engineering

Collaborative systems are especially prone to failure [7]; hence demand iterative evaluation during their development. Ideally, groupware should be prototyped [14], but given the excessive cost of throwing code away, as demanded by "pure" prototyping, an incremental model is more adequate. The groupware engineering cycle is based on the spiral software development model [2], which combines the classical sequential model and the iterative behavior of incremental prototyping. The Groupware Engineering cycle is presented in Figure 10.

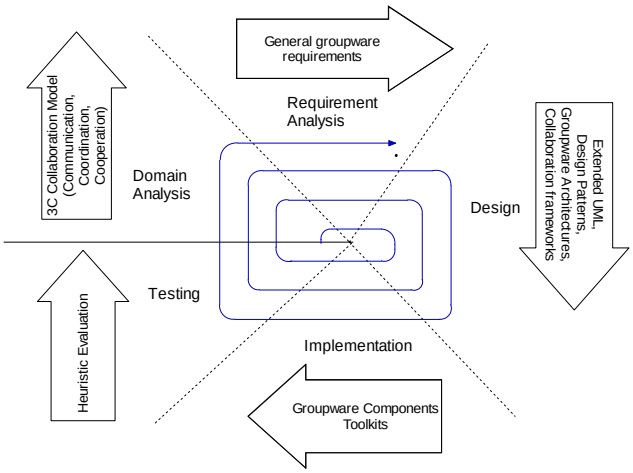

Figure 10. Groupware development cycle

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

General groupware requirements that are elicited in the requirement analysis phase seldom are clear enough to enable a precise specification of system behavior. 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" [8]. 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 of the design.

There are different techniques suitable for the design phase, namely, groupware design patterns [6] for reusing common approaches of design; UML extensions for representing groupware specific aspects of the software; groupware architectures [16] and groupware-related frameworks [11] for reusing code and infrastructure. For the implementation phase, toolkits [13] and groupware components [16] are alternatives for building collaborative systems. Groupware heuristics [1] guide experiments to test the system.

5. Conclusion

Based on the 3C model, in order to collaborate, individuals must debate ideas (communicate), be in tune with other participants of the group (coordinate) and operate together in a shared space (cooperate). Successful communication results in commitments assumed by the group. Coordination enforces the group tasks to avoid that communication efforts are lost. Cooperation is the joint operation of members of the group in a shared space, seeking to accomplish the tasks that are needed to fulfill the commitments.

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.

The example shown in this paper illustrates the benefits of having a component-based architecture that deals with the three Cs of the collaboration model. Groupware Engineering combines the systematic development approach provided by Software Engineering together with the domain analysis given by the 3C model originated from the CSCW field. 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, etc.

The use of component-based techniques is a way of facilitating the development of groupware so that it becomes more flexible. These techniques seek to develop modular systems composed of software components that can be adapted and combined as needed, always having maintenance in mind.

6. References

[1] Baker, K., Greenberg, S. & Gutwin, C. (2001): Heuristic Evaluation of Groupware Based on the Mechanics of Collaboration. *8th IFIP International Conference*, EHCI 2001, LNCS V. 2254, p123-139, Springer-Verlag.

[2] Boehm, B.W. (1988): A Spiral Model of Software Development and Enhancement, *IEEE Computer*, V. 21, N. 5, p. 61-72

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

[4] 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.

[5] 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, pp. 23-38.

[6] Groupware Patterns Swiki (2003), http://swiki.darmstadt.gmd.de/gw-patterns

[7] Grudin, J. (1989): Why Groupware Applications Fail: Problems in Design and Evaluation. *Office: Technology and People*, vol. 4, no. 3, pp. 245-264.

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

[9] Laurillau, Y. & Nigay, L. (2002): Clover architecture for groupware, CSCW 2002, Louisiana, USA, p. 236 - 245

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

[11] Marsic, I. & Dorohonceanu, B. (1999): An Application Framework for Synchronous Collaboration using Java Beans, *Proceedings of the HICSS*, 1999, Maui, Hawaii.

[12] Raposo, A.B. & Fuks, H. (2002): Defining Task Interdependencies and Coordination Mechanisms For Collaborative Systems. *Cooperative Systems Design*, IOS Press, Amsterdam, pp. 88-103.

[13] Roseman, M. & Greenberg, S. (1997): Building Groupware with GroupKit, *Tcl/Tk Tools*, Cap.15, pp 535-564.

[14] Schrage, M. (1996): Cultures of Prototyping: *Bringing Design To Software*, ACM Press, USA, pp. 191-205.

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

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