Development of Groupware Based on the 3C Collaboration Model and Component Technology

Marco Aurélio Gerosa, Mariano Pimentel, Hugo Fuks,
and Carlos José Pereira de Lucena

Software Engineering Laboratory (LES), Computer Science Department
Pontifícia Universidade Católica do Rio de Janeiro – PUC-Rio
R.M.S. Vicente, 225, Rio de Janeiro, RJ, 22453-900, Brasil
{gerosa, mariano, hugo, lucena}@inf.puc-rio.br

Abstract. Groupware is evolutionary and difficult to develop and maintain. Thus, its code becomes unstructured and difficult to evolve. In this paper, a groupware development approach based on components organized according to the 3C collaboration model is proposed. In this model, collaboration is analyzed based on communication, coordination and cooperation. Collaboration requirements, analyzed based on the 3C model, are mapped onto software components. These components aid developers to assembly groupware. The RUP-3C-Groupware, which is a groupware development process, is used for that purpose. This process is a RUP extension focused on groupware domain, and is the result of 8 years of experience with the development of collaborative services for the AulaNet Project. The proposed approach is applied as a case study to the development of the new version of the AulaNet environment. In order to instantiate the environment's communication services, 3C based component kits were developed for the case study. The components allow composition, re-composition and customization of services to reflect changes in the collaboration dynamics.

Keywords: groupware, component software, collaboration model, groupware development process.

1 Introduction

Douglas Engelbart [1968] pointed out the relevance of applications for office automation, hypertext and groups. Today the first two are widely available, used and commercially accepted, while groupware technology is still perceived to be unstable and commercially risky, generating few products [Greenberg 2006]. In most companies, computational support for collaboration is limited to systems for exchanging messages or filing documents.

Groupware technology has not fulfilled its potential yet. Although, research at CSCW is now at a fairly advanced stage, it still lacks a manner of simplifying the programming of collaborative systems and promoting a critical mass of users. Groupware development requires qualified programmers trained to deal with protocols, connections, resource sharing, distribution, rendering, session management,

Y.A. Dimitriadis et al. (Eds.): CRIWG 2006, LNCS 4154, pp. 302 – 309, 2006.

etc. This limits the number of developers active in the area and misplaces the creativity and efforts of these developers, taking their attention out from the creation of solutions to the solving of low-level technical problems, disrupting the investigation of collaboration support [Greenberg, 2006]. In addition, groupware development lacks a process to guide the developer while generating related artifacts.

These groupware development problems are experienced in the development and maintenance of the AulaNet environment [Fuks et al. 2006]. AulaNet is a web-based groupware solution for teaching and learning. AulaNet has been under development since 1997 and is widely used. The system has grown through prototyping, while its functions have been implemented in an evolving fashion. The constant changes required by collaboration and the evolution that forces the changes in technology made the application code strongly linked and with a low level of cohesiveness. Technical aspects permeate the entire code, mixed with the collaboration support, diverting this way the developer's attention.

This article proposes the use of 3C based components as a means of developing extendable groupware whose assembly is determined by collaboration needs. By analysing the problem from the viewpoint of the 3C model and using a component structure designed for this model, changes in the collaboration dynamics are mapped onto the computational support. This way, the developer has a workbench with a component-based infrastructure designed specifically for groupware, based on a collaboration model. In addition, the developer is provided with a process designed specifically for this approach.

The proposed approach is being applied to the re-development of the AulaNet environment. The new version of AulaNet is being developed with the capability to recompose the environment, reuse its services in various situations and reconfigure them to accompany the evolution of the work processes and group characteristics. A layered architecture is defined comprising component frameworks and collaboration components.

2 A Component-Based Infrastructure Based on the 3C Collaboration Model to Groupware Development

The 3C collaboration model is based on the idea that to collaborate, members of a group communicate, coordinate and cooperate. The 3C model derives from the seminal article by Ellis et al. [1991]. The model proposed by Ellis et al. is used to classify computational support for collaboration. In this article, the 3C model is used as a basis for modeling and developing groupware. There is also a difference in terminology; the joint operation in the shared workspace is denominated collaboration by Ellis, while it is denominated cooperation in the 3C model. The 3C model is also similar to the Clover model [Laurillau & Nigay 2002], where cooperation is called production.

Communication involves the exchange of messages and the negotiation of commitments. Coordination enables people, activities and resources to be managed so as to resolve conflicts and facilitate communication and cooperation. Cooperation is the joint production of members of a group within a shared space, generating and manipulating cooperation objects in order to complete tasks [Fuks et al. 2005].

Despite their separation for analytic purposes, communication, coordination and cooperation should not be seen in an isolated fashion; there is a constant interplay between them. Groupware such as chat, for example, which is a communication service, requires communication (exchange of messages), coordination (access policies) and cooperation (registration and sharing).

A groupware environment normally offers the participant a set of collaborative services that are used in different moments of collaboration. Most of them offer mail, discussion list, forum, chat, messenger, agenda, etc. Very similar services are used in groupware environments and each service is relatively independent within the environment. These characteristics are well suited to the application of component-based development. In a component-based environment, the developer selects the services most suited to the group's collaboration needs. Services are classified according to their purposes and characteristics of the 3C model: communication, coordination and cooperation.

The same rationale that was used for environment and its services, may be used for services and their functionalities. Almost every chat possesses a shared area where messages are displayed, a list of connected participants and an area for writing messages. By using a component-based architecture, these characteristics can also be reused. Other developers can use them to select the functionalities best suited to the groups and activities in question.

This analysis leads to the adoption of software components at two levels. The first level comprises the components that implement the communication, coordination and cooperation services, used to offer computational support to the collaboration dynamics as a whole. The second level comprises the components used to assemble the aforementioned services, providing specific support to communication, coordination and cooperation within the dynamics of a particular service. The components that implement the collaborative services are called *services* and the components used to implement the computational support for service collaboration are called *collaboration components*.

2.1 The Collaboration Component Kit

This approach provides the developer with component kits to be used in assembling groupware solutions and collaborative services. Domain engineering aims to provide components that implement the concepts of a software domain and may be reused to implement new applications on this domain. In this paper, the domain analysis, the first step of domain engineering, was based on the literature and on the knowledge accumulated by the AulaNet development group, which has eight years of experience in developing tools for collaboration. The domain analysis was restricted to communication services, which in addition to their communication elements present a representative cross-section of coordination and cooperation elements. Even a communication service, as for example, a discussion forum, besides the communication components, also uses coordination and cooperation components. Communication is related to the media [Daft & Lengel 1986], message categorization [Gerosa et al., 2001], dialog structure [Stahl 2001] and transmission mode. Support for coordination in a communication service is related to channel access policies, task

Table 1. Collaboration Component Kit

COMMUNICATION	COORDINATION	COOPERATION
MessageMgr	AssessmentMgr	CooperationObjMgr
TextualMediaMgr	RoleMgr	SearchMgr
VideoMediaMgr	PermissionMgr	VersionMgr
AudioMediaMgr	ParticipantMgr	StatisticalAnalysisMgr
PictorialMediaMgr	GroupMgr	RankingMgr
DiscreteChannelMgr	SessionMgr	RecommendationMgr
ContinuousChannelMgr	FloorControlMgr	LogMgr
MetaInformationMgr	TaskMgr	AccessRegistrationMgr
CategorizationMgr	AwarenessMgr	TrashBinMgr
DialogStructureMgr	CompetencyMgr	
ConversationPathsMgr	AvailabilityMgr	
CommitmentMgr	NotificationMgr	

and participant management. Support for cooperation in a communication tool is related to the recording and handling of the information.

A component kit is a collection of components designed to work as a set [D'Souza & Wills 1998]. A family of applications can be generated from a component kit, using different combinations and developing other components on demand. Component kits are extendable, allowing new components to be absorbed as necessary. Software components are refined repeatedly until they reach the desired maturity, reliability and adaptability. With the aim of providing tools for the groupware developer, a Collaboration Component Kit is provided, for using collaboration components to assemble services. The components are shown in Table 1.

Component frameworks [Syzperski 1997] are used to provide support to the management and execution of the components. In the proposed architecture, a component framework is used for each proposed component type (service, collaboration), allowing the peculiarities of each one to be met. Services are plugged into the Service Component Framework for the assembling of the groupware environment, and collaboration components are plugged into the Collaboration Component Framework for the assembling of the services. Component frameworks are responsible for handling the installation, removal, updating, deactivation, localization, configuration, monitoring, and import and export of components. The Service Component Framework manages the instances of the services and their links to the corresponding collaboration components. The same service can possess various instances independent of each other. The Component Framework manages the instances and keeps their current state, enabling restoration at a later date.

Most of the functionalities of the component frameworks are recurrent and reusable. A framework can be used for the instantiation of a family of systems. In this article, a framework is used to instantiate the component frameworks. This type of framework is called a *component framework framework* (CFF) [Szyperski 1997, p.277]. A component framework framework is conceived as a second-order component framework whose components are component frameworks. Just as a component interacts with others directly or indirectly via the component framework,

the same applies to component frameworks, whose highest level support is the component framework framework.. Extending the notion used by Szyperski [1997], Figure 1 illustrates the application architecture, including the Groupware Component Framework Framework, as a second-order component framework. The Service Component Framework interacts with the Collaboration Component Framework to enable the instantiation and the association between the instances of the components.

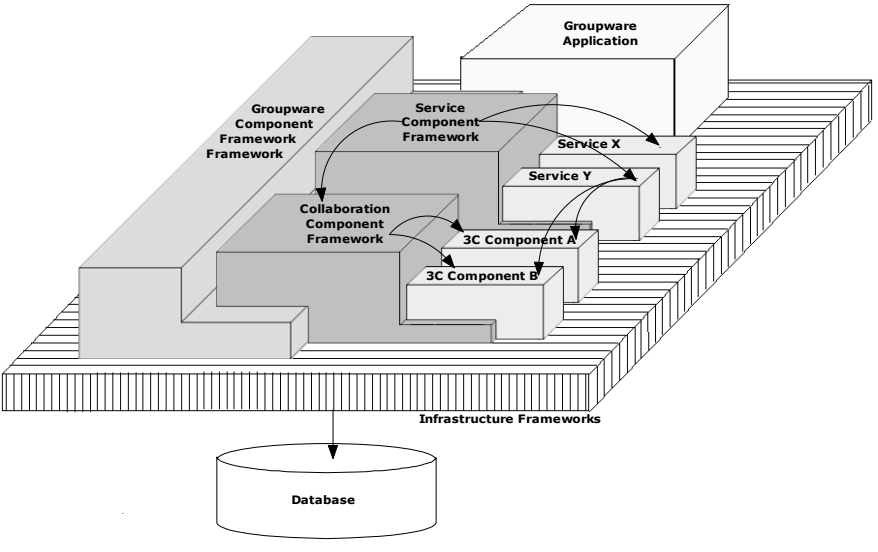

Fig. 1. The proposed architecture

The application's architecture represents a high level logical project independent of the support technology [D'Souza & Wills 1998]. The components plugged into the business layer implement the concepts of the 3C collaboration model.

3 RUP-3C-Groupware

The groupware development process RUP-3C-Groupware is a RUP extension comprising the good practices learnt during the eight years of the AulaNet Project: Component Oriented Approach, which was already discussed in the previous sections; 3C Collaboration Model to Guide the Development and Evolutionary Development Investigating One Problem per Version.

To develop software, particularly groupware, is to solve problems. A good practice is trying to solve one problem at a time [Fuks et al. 2006]. In each version a specific problem is addressed, allowing a better understanding of both the problem and the solution tried, and the identification of new problems still calling for a solution, feeding the development process back.

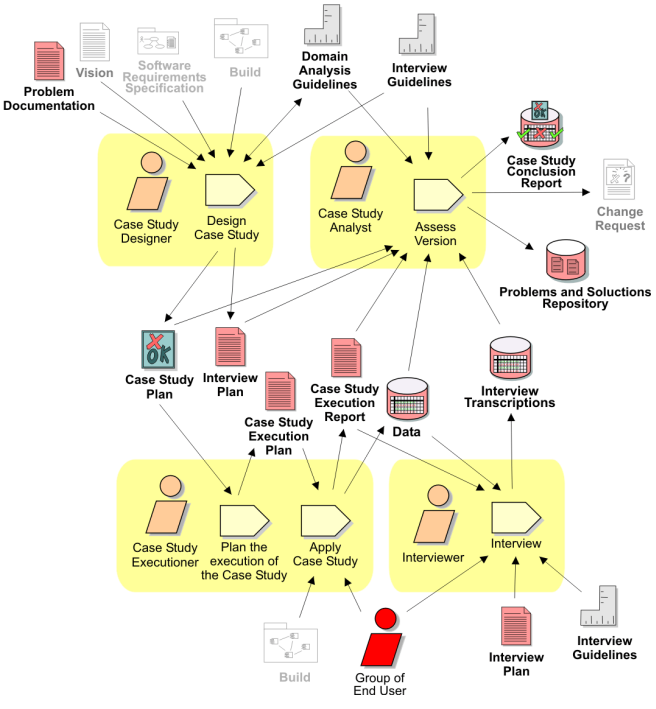

Fig. 2. Case Study activity of RUP-3C-Groupware (RUP's original artifacts are dimmed)

In the RUP-3C-Groupware, the "Case Study" activity was defined, Figure 2, which aims to verify whether the solution implemented in the version solves the problem that is being addressed. A Case Study Plan is developed considering the expected results. Then, the version is used by a group. Data is collected and analysis is carried out to evaluate the version. This version can then be deemed adequate and released for use, and the Deploy discipline is initiated. Otherwise, from the evaluation of the version, new problems may be identified, initiating a new cycle in the development process. Other diagrams and activities of RUP process were also adapted.

4 Case Study in the AulaNet Environment

An early version of the Debate service was implemented using a communication component, tailored for synchronous communication protocols, and a cooperation component, which implements a plain shared space. This version of Debate is a typical chat service, containing an expression element, where learners type their messages, and awareness elements, where messages from learners taking part in the chat session are displayed, as shown in Figure 3.

The early version provided no support for coordination, leaving it to the standing social protocol. However, some courses that use a well-defined procedure for the

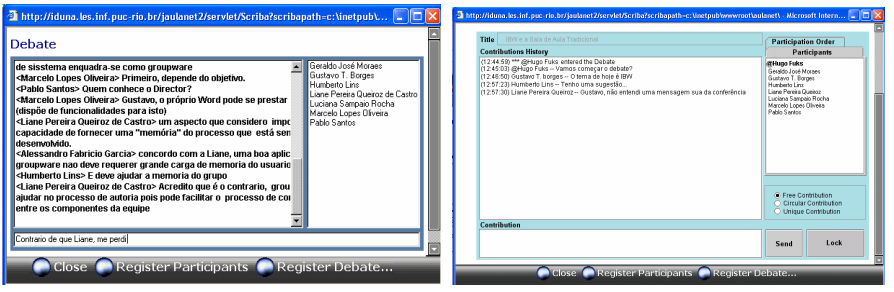

Fig. 3. Early Debate interface (left) and current Debate interface (right)

debate activity, such as the one shown in Figure 3, need effective coordination support. Floor control, participation order and shared space blocking ability were added to the service. The shared space was also enhanced with new awareness elements, like session title, timestamp and identification of mediators.

The same communication component was used for the new version of Debate, given that the synchronous communication protocols and the message characteristics remained the same. The cooperation component, which implements the shared space, was also enhanced with new awareness elements.

The collaborative service was extended to follow the evolution of the work dynamics. The use of the 3C model allowed an isolated analysis of the necessities and difficulties of each collaboration aspect. Based on this analysis a more suitable service was assembled, mapping collaboration necessities onto software components, both of them organized according to the 3C collaboration model.

5 Conclusion

Many groupware environments found in the literature use a component-based architecture, namely FreEvolve [Won et al. 2005], DACIA [Litiu & Prakash 2000], CoCoWare platform [Slagter & Biemans 2000] and GroupKit [Roseman & Greenberg 1996]. However, none of them use the 3C collaboration model as a basis for designing and organizing software components and the development process.

By designing and developing the collaboration services as software components, the developer has the means to assemble a specific groupware environment tailored for the collaboration needs of the group. The services are selected from a component kit based on the 3C model. These components encapsulate business implementations and rules on collaboration, provided by experts and also obtained from experimentation. A component-based architecture provides a working environment with the capability to evolve.

"Without an adequate architecture, the construction of groupware and interactive systems in general is difficult to maintain and iterative refinement is hindered" [Calvary et al. 1997]. A component-based architecture allows components to be selected to assemble a groupware solution meeting a group's specific interests. The components are customized and combined as required, keeping in mind future maintenance. The use of this approach enables prototyping and experimentation,

which are fundamental in CSCW, given that the success cases are very few and poorly documented. However, it is worth stressing that the proposed solution does not eliminate the need for an aware developer who is knowledgeable about the subject in question, since it is not enough to link the components randomly to produce an effective collaborative system.

References

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.

D'Souza, D.F. & Wills, A.C. (1998) Objects, Components and Frameworks with UML: The Catalysis Approach. Addison Wesley, ISBN 0-201-31012-0, 1998.

Daft, R.L. & Lengel, R.H. (1986). Organizational information requirements, media richness and structural design. Management Science 32(5), 554-571.

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.

Engelbart, D. & English, W. (1968) Research Center for Augmenting Human Intellect, Proc. Fall Joint Computing Conference, AFIPS Press, 395-410

Fuks, H., Raposo, A.B., Gerosa, M.A. & Lucena, C.J.P. (2005) Applying the 3C Model to Groupware Development. International Journal of Cooperative Information Systems (IJCIS), v.14, n.2-3, Jun-Sep 2005, World Scientific, ISSN 0218-8430, pp. 299-328.

Fuks, H., Pimentel, M. & Lucena, C.J.P. (2006) "R-U-Typing-2-Me? Evolving a chat tool to increase understanding in learning activities", International Journal of Computer-Supported Collaborative Learning, Volume 1, Issue 1. ISSN: 1556-1607 (Paper) 1556-1615 (Online). Springer: Mar 2006. pp. 117-142.

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", CRIWG 2001, September 6-8, Germany, pp 136-141.

Greenberg, S. (2006) "Toolkits and Interface Creativity", Journal of Multimedia Tools and Applications, Special Issue on Groupware, Kluwer. In Press. Disponível em http://grouplab.cpsc.ucalgary.ca/papers

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

Litiu, R. & Prakash, A. (2000) "Developing Adaptive Groupware Applications Using a Mobile Computing Framework", Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW'00), pp. 107-116.

Roseman, M. & Greenberg, S. (1996) "Building real time groupware with GroupKit, a groupware toolkit". ACM Transactions on Computer-Human Interaction, 3, 1, p. 66-106.

Slagter, R.J. & Biemans, M.C.M. (2000) "Component Groupware: A Basis for Tailorable Solutions that Can Evolve with the Supported Task", ICSC Conference on Intelligent Systems and Applications 2000.

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

Szyperski, C. (1997) Component Software: Beyond Object-Oriented Programming, Addison-Wesley.

Won, M., Stiemerling, O. & Wulf, V. (2005) "Component-Based Approaches to Tailorable Systems", End User Development, Kluwer, pp. 1-27.