Integrating Groupware Notations with UML

William J. Giraldo[1], Ana I. Molina[2], Manuel Ortega[2], Cesar A. Collazos[3]

1 Systems and Computer Engineering, University of Quindío, Quindío, Colombia.
wjgiraldo@uniquindio.edu.co
2 Department of Information Technologies and Systems. Castilla – La Mancha University.
{AnaIsabel.Molina,Manuel.Ortega,Miguel.Redondo }@uclm.es
3 IDIS Research Group, University of Cauca, Popayán, Colombia.
ccollazo@unicauca.edu.co

Abstract. In this paper we introduce a notation integration proposal. This proposal supports the user interface design of groupware applications enabling integration with software processes through UML notation. We introduce our methodological approach to deal with the conceptual design of applications for supporting group work, called CIAM. A study case (the design of a *Conference Resiew System*) is presented to describe our proposal. The integration process proposed is supported by a software tool called CIAT.

Keywords: GUI development, groupware design, interaction design.

1. Introduction

The groupware system design integrates disciplines such as Software Engineering (SE), CSCW, and Usability Engineering (UE), therefore, it requires the interaction of multiple *stakeholders* by using their own specific *workspaces* [1, 2]. Typically, these workspaces support modelling diagrams using different notations. It is necessary that the specified information on each workspace could serve as a complement for the modelling on other workspaces. The *integration* of approaches of model-based design and development with UML notation is conceptually possible to relate main concepts of Human Computer Interaction (HCI) to the classic ones in SE [3]. The approach of the fields of the HCI and the SE are taking a great importance and attention in the last years [4, 5]. On the one hand, the SE begins to consider *usability* like a quality attribute that must be measured and promoted [6]. On the other hand, if the proposed techniques in HCI want to gain solidity within the SE field they should clearly indicate how to integrate their techniques and activities within the process of software development. Nowadays, there is a growing number of proposals for the development of collaborative systems, however, there is still a gap between the development process of the functionality of these systems and the development of their user interface, particularly, proposals that combine group work applications and interactive aspects. CIAM (*Collaborative Interactive Applications Methodology*) is a proposal to assist designer with methodological support for modelling systems for group work [7]. CIAM proposes a specific notation called CIAN [8], which promotes modelling

collaboration, communication and coordination. CIAN adequately supports modelling collaboration, but does not allow modelling the system functionality. In this sense we need UML. Similarly, neither UML nor RUP are intended for the design of interactive system considering usability features. In order to complete the development process of groupware systems, modelling the interaction and collaboration, supported by CIAN, this process must be supplemented adequately to improve the modelling of the functionality, which is based on the use of standard UML notation. Our aim is to integrate the information specified with CIAN with the information gathered in the UML models, and so, try to reduce the gap between the development of the interface and the software development process, as well as the mapping between the two types of notations. This purpose is achieved by specifying a *taxonomy* to define methods, rules, principles and terms for classifying and organizing all necessary information for the specification of groupware systems.

This paper is organized in the following way: section 2 introduces our methodological approach for designing interactive groupware applications, presenting a brief explanation of its stages and the aspects that can be specified in each one. Also, some aspects of the CIAN notation are described in this section. Section 3 introduces the integration proposal, especially the taxonomy. Section 4 presents an example which a case study is used. Finally, the conclusions and further work is presented.

2 CIAM: A Methodological Approach for User Interface Development of Collaborative Applications

In this section we present the stages in our methodological approach. CIAM is an approach based on Model Driven Development (MDD), which promotes the use of models to simplify the complexity of groupware design. CIAM is supported for a notation called CIAN [8] (*Collaborative Interactive Applications Notation*). CIAM considers the interactive groupware modelling in two ways: the group-centred modelling and the process-centred modelling. Initially, the social relations are studied and an organizational scheme is specified. Next, the group work is modelled. The modelling process is more user-centred when we go deeper into the abstraction level, in which interactive tasks are modelled, that is, the dialog between an individual user and the application is modelled. In this way, collaborative aspects (groups, process) and interactive (individual) modelling problems are tackled jointly. The stages on this proposal (**Fig. 1.**) and their objective are enumerated as follows:

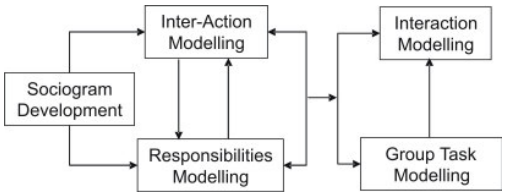

Fig. 1. CIAM methodological proposal stages

The **Sociogram Development** is the fist stage of the methodological approach. In this stage, the organization structure is modelled, as well as the relationship between its members. Organization Members belong to these categories: *roles, actors, software agents*; or in associations of them, forming *groups* or *work teams*. The elements in those diagrams can be interconnected by means of three kinds of basic relationships (*inheritance, acting* and *association*). In the **Inter-Action Modelling** stage, the main tasks that define group work performed in the previously defined organization are described. For each task, the roles involved, the data manipulated and the products generated are specified. Each task must be classified in one of the following categories: *group* or *individual tasks*. Tasks will be interconnected by means of several kinds of relationships (temporal or data relationships). In **Responsibilities Modelling** stage, the individual and shared responsibilities of each role are modelled. In **Group Tasks Modelling** stage the group tasks identified in the previous stage are described in a more detailed way. There are two different kinds of tasks, which must be modelled in a differentiated way: ***c****ooperative* and *collaborative tasks*. *Collaborative Tasks* modelling includes specification of the roles involved, as well as the data model objects manipulated by the work team (that is, the *shared context* specification). Once the objects that make up the *shared context* have been decided, it is necessary to fragment this information into three different parts: the objects and/or attributes manipulated in the *collaborative visualization area*, the ones which appear in the *individual visualization area* and the ones that make up the *exclusive edition segment* (a subset in the data model that is accessed in an exclusive way for only one application user at the same time). Finally, in the **Interaction Modelling** stage, interactive aspects of the application are modelled using the notation. An *interaction model* for each individual task detected in the diverse stages of the gradual refinement process is created. An *interactive tasks decomposition tree* in CTT [9] is developed. In the case of collaborative tasks, the interactive model is directly derived from the shared context definition. Our methodological approach includes the way of obtaining this model from the shared context modelling [8].

3 The integration proposal.

Our integration proposal is based on the assumption that an interactive groupware system can be classified and, therefore, modelled through one or more abstraction layers and using several families or sets of specifications. This idea, expressed graphically in Fig. 2., leads to the definition of our proposal. Each layer could be a stand alone software component. A *layer* is a set of diagrams organized according to a particular criterion, for example: diagrams modelled with the same notation, diagrams representing a particular abstraction, diagrams representing a quality indicator, etc. In this paper our interest is centered in the integration of some models in CIAN and UML; however, our integration proposal can be applied to a large number of notations, each one appropriate to specify different aspects of the system.

The integration layer we propose is based on the Zachman Framework [10]. This Framework proposes a systematic ***taxonomy*** that allows us associating concepts that describe the real world with those who describe their information system and its subsequent implementation. This taxonomy is defined in two dimensions organized in ***perspectives*** and ***views***. We use only the *business* model, *system* model and

technology model perspectives and the *data*, *function*, *network* and *people* views. The intersection of views and perspectives leads to 12 Modelling cells, (Fig. 2.). Each cell provides a container for models that address a particular perspective and view. The Framework provides a representation from different points of view, different levels of granularity, generality and abstraction.

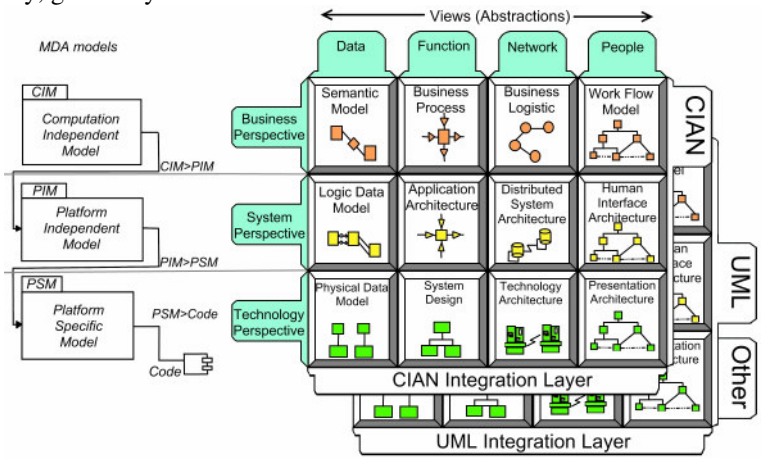

Fig. 2. Integration layer structure and MDA mapping.

A ***perspective*** is an architectural representation at a specific abstraction level and represents a set of logical or physical constraints that may affect the development of a system at that level. This classification by using perspectives enable designers to establish independence between different levels of abstraction, however, it is necessary to have a solid architecture that allows its subsequent integration. MDA (*Model Driven Architecture*) [11] is an architecture that promotes design guided by models and, as can be seen in Fig. 2., there is a relationship between the perspectives and levels of MDA. The concept of ***view***, or abstraction, is a mechanism used by designers to understand a specific system aspect. A key issue in software architectures (perspective) is the support to handle different levels of abstraction. The abstraction is the tool that enables software developers to manage the complexity of their developments. During development we focus, first, on abstractions, and later on implementations that are derived from these abstractions. With the aim at obtaining *integrity*, *uniqueness*, *consistency* and *recursion* of the information specified, taxonomy defines a series of rules. Therefore, the seven rules of the Zachman Framework has been adopted and refined [12]. Examples of these rules are: (R2) All of the cells in each column-view-is guided by a single metamodel. (R5) The composition or integration of all models of the cells in a row is a complete model from this perspective. (R7) The logic is recursive.

4 Case Study: a Conference Review System

In this section a brief example of the application of this method for integrating CIAN and UML is presented. This proposal is supported by a tool, called CIAT

(*Collaborative Interactive Applications Tool*). CIAT [13] is an Eclipse-based tool that helps developers to specify models using CIAN. Eclipse Framework provides tools for guiding the software modelling by using metamodel concepts [14]. By using the EMF (*Eclipse Modelling Framework*) and GMF (*Graphical Editing Framework*), we design the CIAT tool as an Eclipse Plug-in. We have chosen a *Conference Review System* as a case study, extracted from [15].

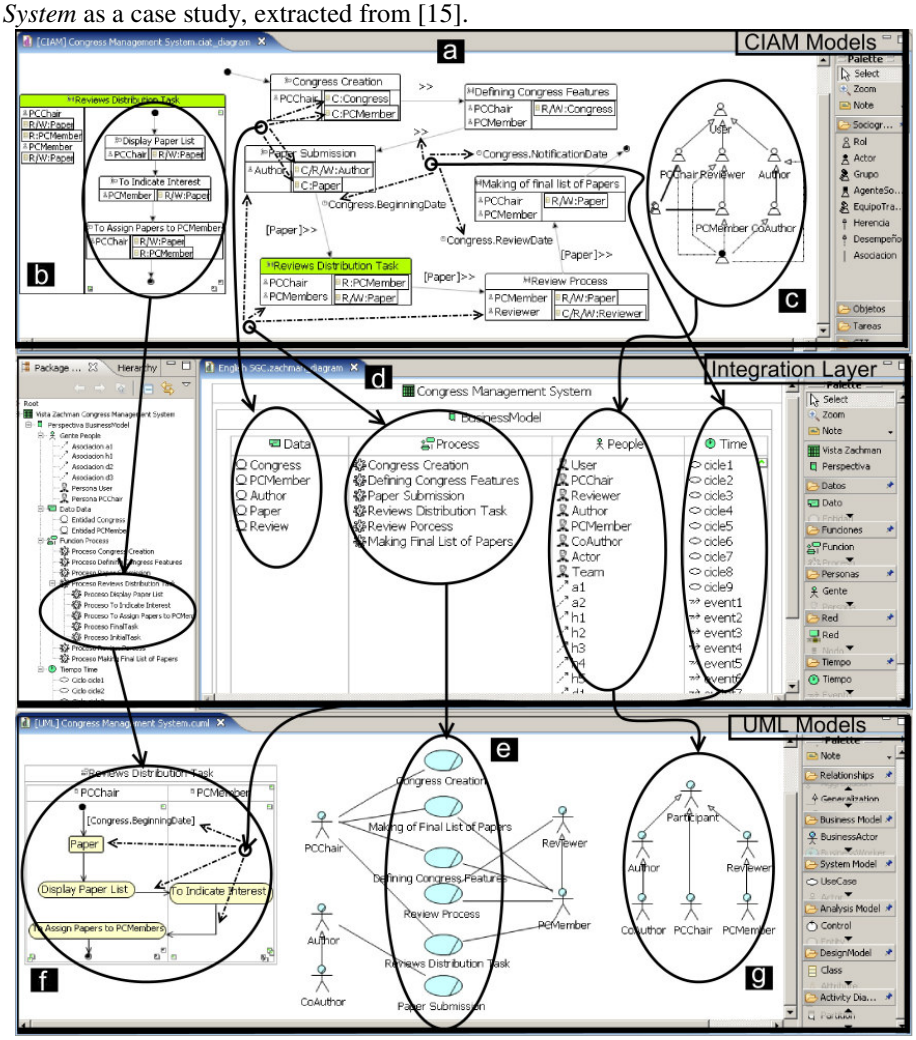

Fig. 3. Integration example between CIAN and UML by using the CIAT tool.

In this section we are going to present the transformation from CIAN to UML of the information specified in the *Sociogram Development,* the *Group Task Modelling* and the *Inter-Action Modelling* stages. As it has been indicated previously, the ***Sociogram*** is a diagram that allows representing the organizational structure, as well as the relationships that can exist among its members. In our case study we have the following roles: *PCChair*, *PCMember*, *Reviewer*, *Author* and *CoAuthor*.

The Fig. 3.(c) shows the structure of the organization in CIAN notation. In particular, the mapping process from the diagram called Sociogram of CIAN and its corresponding representation in UML notation is shown (Fig. 3.(c) to Fig. 3.(g) and Figure 4). The information regarding the roles and relationships among organization members, as it is shown in the Sociogram, is processed through the transformations to generate partial information of *Business Model* and *System Model* perspectives. This information is classified into these two perspectives for the *People* view mainly. Each actor in CIAN can represent both a *Business Actor* as a *System Actor* in UML. The first transformation generates an UML Business Actor diagram -**Fig. 4.**4(c) - from the Sociogram in CIAM -**Fig. 4.**4(a). The People column -**Fig. 4.**4(b) - contains information that relates these two models. The h4 relationship -**Fig. 4.**4(d) - establishes the inheritance relationship on Author and CoAuthor. The relationships dependency and association do not have direct representation in UML; however, information must be stored to generate other artifacts.

The ***Inter-Action*** diagram, see Fig. 3.(a), illustrates the system macro activities and their interdependencies. This model is essential, because certain temporal information (precedence and coordination information) is represented. This information can be enriched through using information related with the domain (that is extracted from the models of the ES process). This diagram provides information about the preconditions, post conditions, messages and data that are required or generated by the activities. UML lacks a diagram of this type.

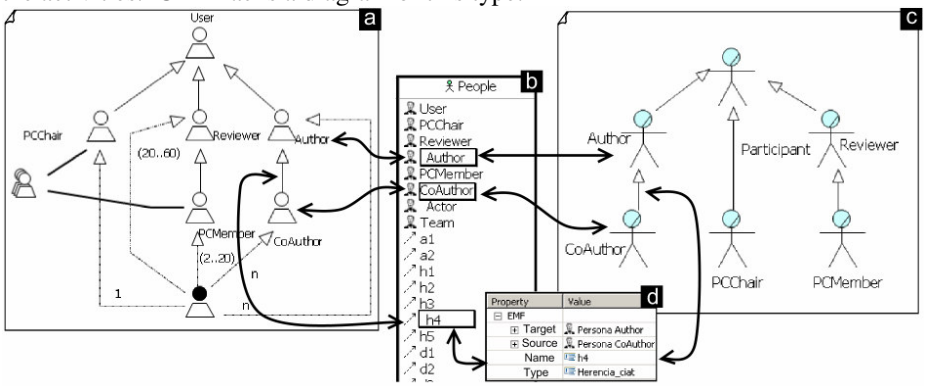

Fig. 4. transformation process by using CIAT.

The Inter-Action diagrams are very rich in information to populate the integration layer. The Fig. 3.(d) illustrates the information extracted from this diagram. The transformations separate information as follows: **(1)** The Inter-Action activities are associated with business use cases. The cooperative activities are transformed into diagrams activity. **(2)** The interdependencies are associated with preconditions, post conditions and events among various activity diagrams. **(3)** The domain objects are associated with business entities. A business object diagram is derived from the information in each activity, which is related with roles and objects.

Fig. 5.5 shows in a more detailed way the mapping between the Inter-Action model (**Fig. 5.**5(b)) and UML diagrams that specifies the same information (business uses cases, **Fig. 5.**5(c), and the activity diagram, **Fig. 5.**5(g)). The integration is based

on information from the *Process* column (function) -Fig. **5.**5(a)- and the *Time* column -Fig. **5.**5(d)- into the integration layer. The variables cicle4, event4 and event5 have the information needed to build the activity diagram in UML. See Fig. **5.**5(e,f,h), respectively. The variables of type event become preconditions or postconditions of business use cases. In Fig. **5.**5(g) is observed as the event4 and event5 are transformed into the guard [Congress.Beginning.Date] and the object node "Paper". Similarly, the variable "Reviews Distribution task" stores the information required to relate the business use case with their respective Actors - Fig. **5.**5(i).

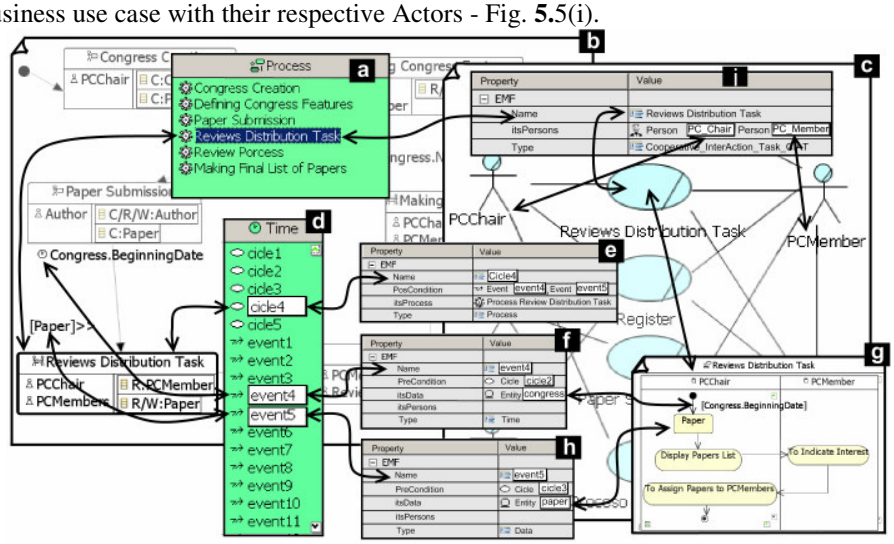

Fig. 5. Detailed integration example between CIAN and UML.

5 Conclusions

In this paper we have shown a brief picture of our methodological proposal for modelling interactive groupware applications and an integration proposal of the notation used in this approach (called CIAN) into the Unified Development Process (supported by the UML notation). This integration proposal is based on the definition of a integration layer (*taxonomy*) and it is supported by a tool called CIAT. We have used a case study in order to explain the integration method by using our integration layer.

The integration proposal presented can be extended to support the integration of a large number of notations. The implemented tool allows the stakeholders involved in the development of a groupware system to construct models, supported by a suitable workspace and using specific notations in their specific domains. Besides, thanks to the use of GMF, CIAT can be integrated with other tools and services available in Eclipse project.

Acknowledgements. This work has been supported by the Universidad del Quindío, Castilla–La Mancha University and Junta de Comunidades de Castilla–La Mancha in the projects AULA-T (PBI08-0069), mGUIDE (PBC08-0006-512) and M-CUIDE (TC20080552).

6 References

1. Molina, A.I., M.A. Redondo, and M. Ortega. *A conceptual and methodological framework for modeling interactive groupware applications*. in *12th International Workshop on Groupware (CRIWG 2006)*. 2006c. Valladolid. Spain: Springer-Verlag (LNCS).
2. Gutwin, C. and S. Greenberg. *Design for Individuals, Design for Groups: Tradeoffs between power and workspace awareness*. in *ACM CSCW'98*. 1998. Seattle: ACM Press.
3. Artim, J., et al., *Incorporating work, process and task analysis into industrial object-oriented systems development.* SIGCHI Bulletin, 1998. **30**(4).
4. Granollers, T., et al. *Integración de la IPO y la ingeniería del software: MPIu+a*. in *III Taller en Sistemas Hipermedia Colaborativos y Adaptativos*. 2005. Granada España.
5. Ferré, X. and A.M. Moreno. *Integración de la IPO en el Proceso de Desarrollo de la Ingeniería del Software: Propuestas Existentes y Temas a Resolver*. in *V Congreso Interacción Persona-Ordenador (Interacción 2004)*. 2004. Lleida, España.
6. Ferre, X., N. Juristo, and A.M. Moreno. *Improving Software Engineering Practice with HCI Aspects*. in *SERA*. 2004: Springer-Verlag.
7. Molina, A.I., et al., *CIAM: A methodology for the development of groupware user interfaces.* Journal of Universal Computer Science(JUCS), 2007.
8. Molina, A.I., M.A. Redondo, and M. Ortega. *A conceptual and methodological framework for modeling interactive groupware applications*. in *12th International Workshop on Groupware (CRIWG 2006)*. 2006. Valladolid. Spain: Springer-Verlag (LNCS).
9. Paternò, F., C. Mancini, and Meniconi. *ConcurTaskTree: A diagrammatic notation for specifying task models*. in *IFIP TC 13 International Conference on Human-Computer Interaction Interact'97*. 1997. Sydney: Kluwer Academic Publishers.
10. Zachman, J.A., *A Framework For Information Systems Architecture.* IBM Ssystems Journal, 1987. **26**(3).
11. Miller, J. and J. Mukerji. *MDA Guide Version 1.0.1.* 2003 [cited 08-07-2007; Available from: http://www.appdevadvisor.co.uk/express/vendor/domain.html.
12. Sowa, J.F. and J.A. Zachman, *Extending and formalizing the framework for information systems architecture.* IBM Syst. J, 1992: p. 590-616.
13. Giraldo, W.J., et al. *CIAT, A Model-Based Tool for designing Groupware User Interfaces using CIAM*. in *Computer-Aided Design of User Interfaces VI, CADUI 2007*. 2008. Albacete España: Springer Verlag.
14. Moore, B., et al., *Eclipse Development using the Graphical Editing Framework and the Eclipse Modeling Framework*. Redbooks. 2004: ibm.com/redbooks.
15. Schwabe, D. and G. Rossi (2001) *Conference Review System in OOHDM*. International Workshop on Web Oriented Software Technology **Volume**,