

Yannis A. Dimitriadis
Ilze Zigurs
Eduardo Gómez-Sánchez (Eds.)

LNCS 4154

Groupware: Design, Implementation, and Use

12th International Workshop, CRIWG 2006
Medina del Campo, Spain, September 2006
Proceedings



Springer

Commenced Publication in 1973

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison

Lancaster University, UK

Takeo Kanade

Carnegie Mellon University, Pittsburgh, PA, USA

Josef Kittler

University of Surrey, Guildford, UK

Jon M. Kleinberg

Cornell University, Ithaca, NY, USA

Friedemann Mattern

ETH Zurich, Switzerland

John C. Mitchell

Stanford University, CA, USA

Moni Naor

Weizmann Institute of Science, Rehovot, Israel

Oscar Nierstrasz

University of Bern, Switzerland

C. Pandu Rangan

Indian Institute of Technology, Madras, India

Bernhard Steffen

University of Dortmund, Germany

Madhu Sudan

Massachusetts Institute of Technology, MA, USA

Demetri Terzopoulos

University of California, Los Angeles, CA, USA

Doug Tygar

University of California, Berkeley, CA, USA

Moshe Y. Vardi

Rice University, Houston, TX, USA

Gerhard Weikum

Max-Planck Institute of Computer Science, Saarbruecken, Germany

Yannis A. Dimitriadis Ilze Zigurs
Eduardo Gómez-Sánchez (Eds.)

Groupware: Design, Implementation, and Use

12th International Workshop, CRIWG 2006
Medina del Campo, Spain, September 17-21, 2006
Proceedings

Volume Editors

Yannis A. Dimitriadis
Eduardo Gómez-Sánchez
University of Valladolid
ETSI Telecomunicación
Camino del Cementerio s/n, 47011 Valladolid, Spain
E-mail: {yannis,edugom}@tel.uva.es

Ilze Zigurs
University of Nebraska at Omaha
College of Information Science and Technology
6001 Dodge Street, PKI 284E, Omaha, NE 68182-0116, USA
E-mail: izigurs@mail.unomaha.edu

Library of Congress Control Number: 2006932263

CR Subject Classification (1998): H.5.2, H.5.3, H.5, K.3.1, K.4.3, C.2.4

LNCS Sublibrary: SL 3 – Information Systems and Application, incl. Internet/Web and HCI

ISSN 0302-9743
ISBN-10 3-540-39591-1 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-39591-1 Springer Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

Springer is a part of Springer Science+Business Media

springer.com

© Springer-Verlag Berlin Heidelberg 2006
Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11853862 06/3142 5 4 3 2 1 0

Preface

This volume constitutes the proceedings of the 12th International Workshop on Groupware (CRIWG 2006). The conference was held in Medina del Campo, Spain. The historic and scenic venue provided an excellent environment to continue the traditions of the workshop. The size of the conference was relatively small, the discussions were lively and constructive during and between sessions, and the level of collaboration was high both socially and in making new connections for research ideas.

The previous eleven CRIWG workshops were held in Lisbon, Portugal (1995), Puerto Varas, Chile (1996), El Escorial, Spain (1997), Buzios, Brazil (1998), Cancun, Mexico (1999), Madeira, Portugal (2000), Darmstadt, Germany (2001), La Serena, Chile (2002), Autrans, France (2003), San Carlos, Costa Rica (2004), and Porto de Galinhas, Recife, Brazil (2005).

This 12th CRIWG received a record number of submissions, attesting both to the continuing importance of groupware as a field and the many interesting issues for research that surround it. Groupware researchers from 21 different countries submitted a total of 99 papers. Each paper was double-blind reviewed by at least three members of an internationally known Program Committee supplemented with additional reviewers. We appreciate all their work. The Program Chairs performed a “meta review” of all the reviews, to ensure that each paper got the best and fairest chance in its assessment. Based on reviewer recommendations and the Program Chairs’ judgements, 34 papers were accepted, 21 of which were long papers representing mature work, and 13 short papers describing work in progress. The accepted papers were grouped into the following clusters: computer-supported collaborative learning, groupware development frameworks and toolkits, mobile collaborative work, collaborative applications and group interaction, Web-based cooperative environments, collaborative workspaces, languages and tools supporting collaboration, group awareness, and collaborative design.

In addition to the papers, we were very pleased to have groupware pioneer Clarence *Skip* Ellis of the University of Colorado-Boulder to provide the keynote talk for the conference. A doctoral colloquium was also held the day before the conference began.

CRIWG 2006 would not have been possible without the work and support of a great number of people. First we thank all those who submitted to the conference and who continue to support it from year to year with leading-edge research in groupware. We also extend very special thanks to the Program Committee and additional reviewers for their diligent and constructive reviewing. We are grateful to the CRIWG Steering Committee for their on-going advice and support, as well as all members of the Local Organizing Committee. Special acknowledgement and thanks go to our sponsors: the Spanish Ministry of Education and Science, the Regional Government of Castilla and León, the University of Valladolid

and its Computer Science Department and Signal Theory, Communications, and Telematics Engineering Department.

Finally, we thank the attendees for their enthusiastic commitment and we hope that all participants and readers of these proceedings continue to find excitement and learning opportunities in their continuing work on groupware.

September 2006

Yannis A. Dimitriadis
Ilze Zigurs
Eduardo Gómez-Sánchez

Conference Organization

Program Committee Chairs

Yannis A. Dimitriadis, University of Valladolid, Spain
Ilze Zigurs, University of Nebraska at Omaha, USA

Program Committee Members

Pedro Antunes, Universidade de Lisboa, Portugal
Jaco Appelman, Delft University of Technology, The Netherlands
Juan I. Asensio-Pérez, University of Valladolid, Spain
Marcos Borges, Federal University of Rio de Janeiro, Brazil
Patrick Brézillon, Université Paris 6, France
Traci Carte, University of Oklahoma, USA
Wei Qin Chen, University of Bergen, Norway
César A. Collazos, Universidad del Cauca, Colombia
Kevin Crowston, Syracuse University, USA
Atanasi Daradoumis, Open University of Catalonia, Spain
Bertrand David, Ecole Centrale de Lyon, France
Gert-Jan de Vreede, University of Nebraska at Omaha, USA
Dominique Decouchant, LSR-IMAG, Grenoble, France
Henrique J.L. Domingos, Universidade Nova de Lisboa, Portugal
Thomas Erickson, IBM T. J. Watson Research Center, USA
Cléver Farias, Universidade de São Paulo, Brazil
Jesus Favela, CICESE, Mexico
Christine Ferraris, Université de Savoie, France
Hugo Fuks, Catholic University of Rio de Janeiro, Brazil
Matt Germonprez, University of Wisconsin-Eau Claire, USA
Werner Geyer, IBM T. J. Watson Research, Cambridge, USA
Luis A. Guerrero, Universidad de Chile, Chile
Jörg M. Haake, FernUniversität in Hagen, Germany
Andreas Harrer, University of Duisburg-Essen, Germany
H. Ulrich Hoppe, University of Duisburg-Essen, Germany
Deepak Khazanchi, University of Nebraska at Omaha, USA
Ned Kock, Texas A&M International University, USA
Stephan Lukosch, FernUniversität in Hagen, Germany
Alberto L. Moran, UABC, Mexico
Bjørn Erik Munkvold, Agder University College, Norway
Leandro Navarro, Polytechnic University of Catalonia, Spain
Fred Niederman, St. Louis University, USA

Yvan Peter, University of Lille 1, France
José A. Pino, Universidad de Chile, Chile
Nuno Pregoça, Universidade Nova de Lisboa, Portugal
Alberto Raposo, Catholic University of Rio de Janeiro, Brazil
Symeon Retalis, University of Piraeus, Greece
Jeremy Roschelle, Stanford Research International, USA
Nicolas Roussel, Université Paris-Sud & INRIA Futurs, France
Flavia Maria Santoro, UNIRIO, Brazil
Till Schümmer, FernUniversität in Hagen, Germany
Choon Ling Sia, City University of Hong Kong, Hong Kong
Carla Simone, University of Milan, Italy
José Valdeni de Lima, Universidade Federal do Rio Grande do Sul, Brazil
Aurora Vizcaíno-Barceló, Universidad de Castilla-La Mancha, Spain
Jürgen Vogel, European Media Laboratory (EML) GmbH, Germany
Jacques Wainer, State University of Campinas, Brazil
Martin Wessner, Fraunhofer IPSI, Germany
Vance Wilson, University of Wisconsin-Milwaukee, USA
Volker Wulf, Fraunhofer FIT, Germany

Additional Reviewers

Alanah Davis, University of Nebraska at Omaha, USA
Miguel L. Bote-Lorenzo, University of Valladolid, Spain
Eduardo Gómez-Sánchez, University of Valladolid, Spain
Davinia Hernández-Leo, University of Valladolid, Spain
Alejandra Martínez-Monés, University of Valladolid, Spain
Matt Payne, University of Nebraska at Omaha, USA
Terry Schoonover, University of Nebraska at Omaha, USA
Halbana Tarmizi, University of Nebraska at Omaha, USA
Guillermo Vega-Gorgojo, University of Valladolid, Spain
Chi Zhang, University of Nebraska at Omaha, USA

Doctoral Colloquium Chairs

Traci Carte, University of Oklahoma, USA
Andreas Harrer, University of Duisburg-Essen, Germany

Organizing Committee Chair

Eduardo Gómez-Sánchez, University of Valladolid, Spain

Organizing Committee Members

Miguel Bote-Lorenzo, University of Valladolid, Spain

Miguel Ángel Gómez-Hernández, University of Valladolid, Spain

Iván M. Jorrín-Abellán, University of Valladolid, Spain

Alejandra Martínez-Monés, University of Valladolid, Spain

Sponsoring Institutions

Ministerio de Educación y Ciencia, Spain

Junta de Castilla y León, Spain

Universidad de Valladolid, Spain

Dept. Teoría de la Señal, Comunicaciones, e Ingeniería Telemática, Universidad de Valladolid, Spain

Departamento de Informática, Universidad de Valladolid, Spain

Table of Contents

Collaborative Applications and Group Interaction

| | |
|--|----|
| Task Analysis Based Methodology for the Design of Face to Face Computer Supported Collaborative Learning Activities | 1 |
| <i>Maria Francisca Capponi, Miguel Nussbaum, Maria Ester Lagos</i> | |
| Group Creativity and Collaborative Technologies: Understanding the Role of Visual Anonymity | 12 |
| <i>Traci Carte, Laku Chidambaram, Monica Garfield, Lindsey Hicks, Cassie Cole</i> | |
| InClass-RTD: Providing Support for Real-Time Threaded Discussions in the Classroom | 22 |
| <i>Alberto L. Morán, Carmen Perez, Marcela Rodríguez</i> | |
| Technical and Environmental Challenges of Collaboration Engineering in Distributed Environments | 38 |
| <i>Halbana Tarmizi, Matt Payne, Cherie Noteboom, Chi Zhang, Lucas Steinhäuser, Gert-Jan de Vreede, Ilze Zigurs</i> | |
| Monitoring and Analyzing Group Interactions in Asynchronous Discussions with the DIAS System | 54 |
| <i>Tharrenos Bratitsis, Angelique Dimitracopoulou</i> | |
| Analyzing Shared Workspaces Design with Human-Performance Models | 62 |
| <i>Pedro Antunes, Antonio Ferreira, José A. Pino</i> | |

Group Awareness

| | |
|--|----|
| Using Email-Based Network Analysis to Determine Awareness Foci | 78 |
| <i>Adriana S. Vivacqua, Jano Moreira de Souza</i> | |
| Cooperation Indexes to Support Workspace Awareness | 94 |
| <i>Benoît Otjacques, Monique Noirhomme, Xavier Gobert, Fernand Feltz</i> | |

Guidelines and Usability Principles to Design and Test
Shared-Knowledge Awareness for a CSCL Interface 102
María Paula González, César A. Collazos, Toni Granollers

Computer Supported Collaborative Learning

The Remote Control Approach - How to Apply Scaffolds to Existing
Collaborative Learning Environments 118
Andreas Harrer, Nils Malzahn, Benedikt Roth

Polyphonic Support for Collaborative Learning 132
Stefan Trausan-Matu, Gerry Stahl, Johann Sarmiento

On Supporting Users' Reflection During Small Groups Synchronous
Collaboration 140
Meletis Margaritis, Nikolaos Avouris, Georgios Kahrmanis

Interaction Analysis for the Detection and Support of Participatory
Roles in CSCL 155
*José Antonio Marcos, Alejandra Martínez, Yannis A. Dimitriadis,
Rocío Anguita*

Languages and Tools Supporting Collaboration

ORCHESTRA: Formalism to Express Mobile Cooperative
Applications 163
*Bertrand David, René Chalon, Olivier Delotte, Guillaume Masserey,
Matthieu Imbert*

A Decentralized and Flexible Tool Supporting Extreme Programming
Software Development 179
*Nelson Baloian, Francisco Claude, Roberto Konow,
Mitsuji Matsumoto*

The PoEML Proposal to Model Services in Educational Modeling
Languages 187
Manuel Caeiro-Rodríguez, Martín Llamas-Nistal, Luis Anido-Rifón

Groupware Development Frameworks and Toolkits

A Framework Designed for Synchronous Groupware Applications in
Heterogeneous Environments 203
Axel Guicking, Thomas Grasse

| | |
|--|-----|
| Implicit Plasticity Framework: A Client-Side Generic Framework for Collaborative Activities | 219 |
| <i>Montserrat Sendín, César A. Collazos</i> | |
| Supporting Mobile Collaboration with Service-Oriented Mobile Units | 228 |
| <i>Andrés Neyem, Sergio F. Ochoa, José A. Pino</i> | |
| SAGA: A Web Services Architecture for Groupware Applications | 246 |
| <i>Benjamim Fonseca, Eurico Carrapatoso</i> | |
| Towards a P2P-Based Active e-Learning Space | 262 |
| <i>Xianghua Xu, Jian Wan</i> | |
| Understanding the Trade-Offs of Blending Collaboration Services in Support of Contextual Collaboration | 270 |
| <i>Roberto S. Silva Filho, Werner Geyer, Beth Brownholtz, David F. Redmiles</i> | |

Collaborative Workspaces

| | |
|--|-----|
| Leveraging the Linda Coordination Model for a Groupware Architecture Implementation | 286 |
| <i>José Luis Garrido, Manuel Noguera, Miguel González, Miguel Gea, María V. Hurtado</i> | |
| Development of Groupware Based on the 3C Collaboration Model and Component Technology | 302 |
| <i>Marco Aurélio Gerosa, Mariano Pimentel, Hugo Fuks, Carlos José Pereira de Lucena</i> | |
| Ontoolcole: An Ontology for the Semantic Search of CSCL Services | 310 |
| <i>Guillermo Vega-Gorgojo, Miguel L. Bote-Lorenzo, Eduardo Gómez-Sánchez, Juan I. Asensio-Pérez, Yannis A. Dimitriadis, Iván M. Jorrín-Abellán</i> | |

Web-Based Cooperative Environments

| | |
|---|-----|
| CSCL, Anywhere and Anytime | 326 |
| <i>Stephan Lukosch, Matthias Hellweg, Martin Rasel</i> | |
| Web Management of Citizens' Complaints and Suggestions | 341 |
| <i>Victor M.R. Penichet, José A. Gallud, María Lozano, Manuel Tobarra</i> | |

Social Visualization Encouraging Participation in Online
Communities 349
Lingling Sun, Julita Vassileva

Mobile Collaborative Work

Analyzing the Roles of PDA in Meeting Scenarios 364
*Gustavo Zurita, Pedro Antunes, Luís Carricho, Felipe Baytelman,
Marco Sá, Nelson Baloian*

Supporting the Management of Multiple Activities in Mobile
Collaborative Working Environments 381
Jesus Camacho, Jesus Favela, Victor M. Gonzalez

Seamless Interaction Among Heterogeneous Devices in Support for
Co-located Collaboration 389
Antoine Markarian, Jesus Favela, Monica Tentori, Luis A. Castro

Collaborative Design

Predicting User Interest Region for Collaborative Graphics Design
Systems in Ubiquitous Environment 405
Jiajun Bu, Bo Jiang, Chun Chen, Jianxu Yang

A Conceptual and Methodological Framework for Modeling Interactive
Groupware Applications 413
Ana Isabel Molina, Miguel Ángel Redondo, Manuel Ortega

Collaborative Design and Tailoring of Web Based Learning
Environments in CURE 421
Mohamed Bourimi

Author Index 437

Task Analysis Based Methodology for the Design of Face to Face Computer Supported Collaborative Learning Activities

M.F. Capponi, M. Nussbaum, and M.E. Lagos

Department of Computer Science, P. Universidad Católica de Chile
Vicuna Mackena 4860, Santiago, Chile
mcapponi@ing.puc.cl, mn@ing.puc.cl, mrlagos@ing.puc.cl

Abstract. This paper shows how Task Analysis can be a powerful tool for the design of collaborative applications supported by wirelessly interconnected handhelds. We define a methodology for the design of such activities. It basically consists in performing a Task Analysis on an Interaction Model to obtain the set of all possible interactions between actors. Then a class of activities is defined by selecting a subset of tasks. These, applied to a specific topic, determine a set of specific tasks which constitute an instance of the class of activities. The specific tasks build the desired activity and define the possible face to face interactions that can happen during the activity execution. These specific tasks also allow us to define an observation guideline that assists the system validation. We show with an example how such a methodology is applied for a collaborative learning activity mediated by a teacher and wirelessly interconnected handhelds.

Keywords: Task Analysis, face to face Computer Supported Collaborative Learning.

1 Introduction

Collaborative Learning (CL) environments have shown benefits in the achievement of learning objectives, social aims, positive interdependence and motivation, acquiring student new skills, ideas and knowledge by working together [6]. CL, however has shown to have coordination, communication, organization and synchronization problems, which can be solved with face to face computer supported collaborative learning [5]. In this environments students work each with a handheld machine wirelessly interconnected [4].

Face to Face collaboration involves two dimensions: the human nature of communication [1] and the activities carried out in the shared workspace [2]. In the second dimension, we can distinguish the team work (making it happen collaboratively) and the task work (doing the actual job) [3]. Both the team and the task work have to be understood as interconnecting elements with the actors of the activity. Task Analysis allows us to decompose in basic building blocks the team and

the task work, which can later be used in the design of a face to face computer supported collaborative learning activity. This analysis helps us also to extract an observation guideline of the face to face interactions we can observe during the activity and provides a concrete representation of the actions taken towards user goals and the logical relationship between those actions [7]. So, obtaining from an abstract Interaction Model a set of tasks or building blocks, we can construct many different collaborative activities. This technique is well utilized to understand the usability of interactive systems and to understand users' work with a system in the abstract [8]. But there is no research about how this analysis can facilitate the whole design and development of face to face collaborative activities and how it can assist the later validation.

We propose a Methodology based on Task Analysis for the design of an effective human-human relation that involves the team and the task work for a collaborative classroom, supported by wirelessly interconnected handhelds. This methodology has to assure that the interactions selected for the activity are going to happen. Through the methodology steps we can design the activity by tasks, defining classes of tasks and sets of specific tasks. We can also extract an observation guideline from the same analysis, to indicate what to observe during the activity to validate the system, verifying that the observed interactions are the set of valid tasks defined in the design process. In this way for collaborative activities, we can study what sets of tasks we can use for the design of an activity to generate better social interactions, and recognize what interventions are necessary to change the social and communication behavior between actors to achieve a specific aim.

2 Task Analysis

To define how and between whom information flows in a classroom technologically mediated by mobile devices connected to a wireless network with students that work collaboratively, we use a generalized Interaction Model [9] for the main classroom components, showed in Fig. 1. Here we assume interaction to be the basic unit that occurs among various actors that work collaboratively.

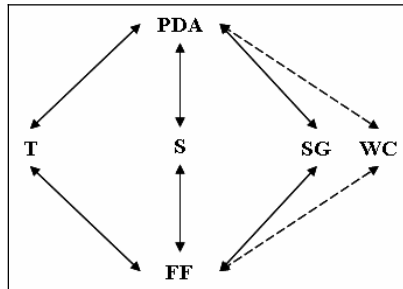


Fig. 1. Interaction Model

The components of the model shown in Fig. 1 are the following:

- *Actors*: Persons or groups of persons among whom information flows. They are information emitters and receptors, and include:
 - Student (S)
 - Whole Class (WC), comprising the N students in the class
 - Small Group (SG), a subset of A students in the class, which consists of N/A groups

Table 1. Set of All Tasks obtained from the Task Analysis with the Interaction Model for Collaborative Learning supported by wirelessly interconnected PDAs

| | | |
|----|-----------|--|
| 1 | T→PDA→S | Teacher sends information to a student |
| 2 | T→PDA→SG | Teacher sends information to a small group |
| 3 | T→PDA→WC | Teacher sends information to the whole class |
| 4 | T→PDA→T | Teacher sends information to another teacher |
| 5 | S→PDA→T | Student sends information to a teacher |
| 6 | S→PDA→SG | Student sends information to a small group |
| 7 | S→PDA→WC | Student sends information to the whole class |
| 8 | S→PDA→S | Student sends information to another student |
| 9 | SG→PDA→SG | Small group sends information to another small group |
| 10 | SG→PDA→S | Small group sends information to a student |
| 11 | SG→PDA→T | Small group sends information to a teacher |
| 12 | SG→PDA→WC | Small group sends information to the whole class |
| 13 | WC→PDA→S | Whole class sends information to a student |
| 14 | WC→PDA→T | Whole class sends information to a teacher |
| 15 | WC→PDA→SG | Whole class sends information to a small group |
| 16 | SG→FF→S | Small group discusses with a student |
| 17 | SG→FF→T | Small group speaks with a teacher |
| 18 | SG→FF→WC | Small group speaks with the whole class |
| 19 | SG→FF→SG | Small group speaks with another small group |
| 20 | T→FF→S | Teacher speaks with student |
| 21 | T→FF→SG | Teacher speaks with a specific small group |
| 22 | T→FF→WC | Teacher speaks with whole class |
| 23 | T→FF→T | Teacher speaks to another teacher |
| 24 | S→FF→T | Student speaks with a teacher |
| 25 | S→FF→SG | Student speaks with small group |
| 26 | S→FF→WC | Student speaks with whole class |
| 27 | S→FF→S | Student speaks with another student |
| 28 | WC→FF→S | Whole class speaks with a student |
| 29 | WC→FF→T | Whole class speaks with a teacher |
| 30 | WC→FF→SG | Whole class speaks with a small group |

- *Mediators*: The entities through which the information flows. Since the information may be transformed in the process, they are more than just communication channels:
 - Personal Digital Assistants (PDA). PDAs act as instruments that support and regulate relations between actors, and provide:

- organization and representation of the information
 - a negotiation space
 - coordination between activity states
- Face-to-face relationships (FF). The human medium in which information is exchanged, impacting on the student's commitment to their responses and their group as well as on the development of mutual understanding between the different actors.
- *Actor and mediator*: Teacher (T). In addition to sending and receiving information, the teacher is responsible for selecting the curriculum activities to be implemented and for guiding the students toward the achievement of the desired goals. The teacher is also in charge of delivering feedback to the students and filtering the information flowing between them and among the groups, in such a manner that the discussions take the desired course.

The interactions in the model, as shown in Fig. 1, are centered on the student, the use of PDAs and a face-to-face relationship (FF) as mediator components for the interactions with the other actors: the teacher (T), the small group he or she belongs to (SG) and his or her class (WC). The teacher also acts as mediator of the interaction and communication between the students (S), the small groups (SG) and the whole group, i.e., the class (WC).

From the model of Fig. 1, we can obtain all basic tasks, enumerated in Table 1. Every task is performed by two actors and a mediator through which information flows. These tasks are the key elements for understanding the social behavior. In Fig. 1, the arrows connecting the components indicate the direction of information flow. While the continuous line indicates that the flow is for both sides, the dashed line indicates that the flow is, in some cases, only for one side. In this model we have two interactions that are not possible, $WC \rightarrow FF \rightarrow WC$ and $WC \rightarrow PDA \rightarrow WC$, because we only have one class.

3 Task Analysis Based Methodology

In this section we show a methodology based on Task Analysis, graphically represented in Fig. 2, that supports the definition of an activity. This methodology consists in eight steps, described in Table 2. In Fig. 2, each step is represented as an arrow with the number of the corresponding step. Dashed arrows indicate the optional paths we have to follow when the class or instance of the activity is not well defined or implemented. Ellipses represent sets of tasks, the first rectangle represents the Interaction Model, and the rhombus represents the collaborative activity. In the seventh step, we compare the Observed Tasks with three subsets taken from the Specific Tasks, Class Tasks and All Tasks sets, selecting only the face to face interactions from each one. We named them Face to Face Specific Tasks (FFST), Face to Face Class Tasks (FFCT) and Face to Face All Tasks (FFAT) respectively. In the right bottom of the figure we draw three boxes representing the three possibilities for the observed tasks and in the left bottom, a fourth box indicating the valid set of tasks. Next we make a detailed analysis of the proposed methodology.

In the **first step** of the methodology we define a set of all possible tasks, performing a Task Analysis of the Interaction Model shown in Fig. 1. Table 1 shows

this set of tasks, or possible interaction between actors, for the design of a Collaborative Learning Activity supported by wirelessly interconnected PDAs. We named this set of tasks as All Tasks (AT).

In the **second step** we define a Class of Activities (CA) determining the kind of interactions we want to generate during the activity. In our example, the Class of Activities consists of an individual part, where each student solves a problem independently, and a collaborative part, where the whole class together finds out the correct answer.

For each Class of Activities we select a subset of tasks that allows us to design an activity with those interactions. We named this subset as Class Tasks (CT). For the example, the Class Tasks are shown in Table 3.

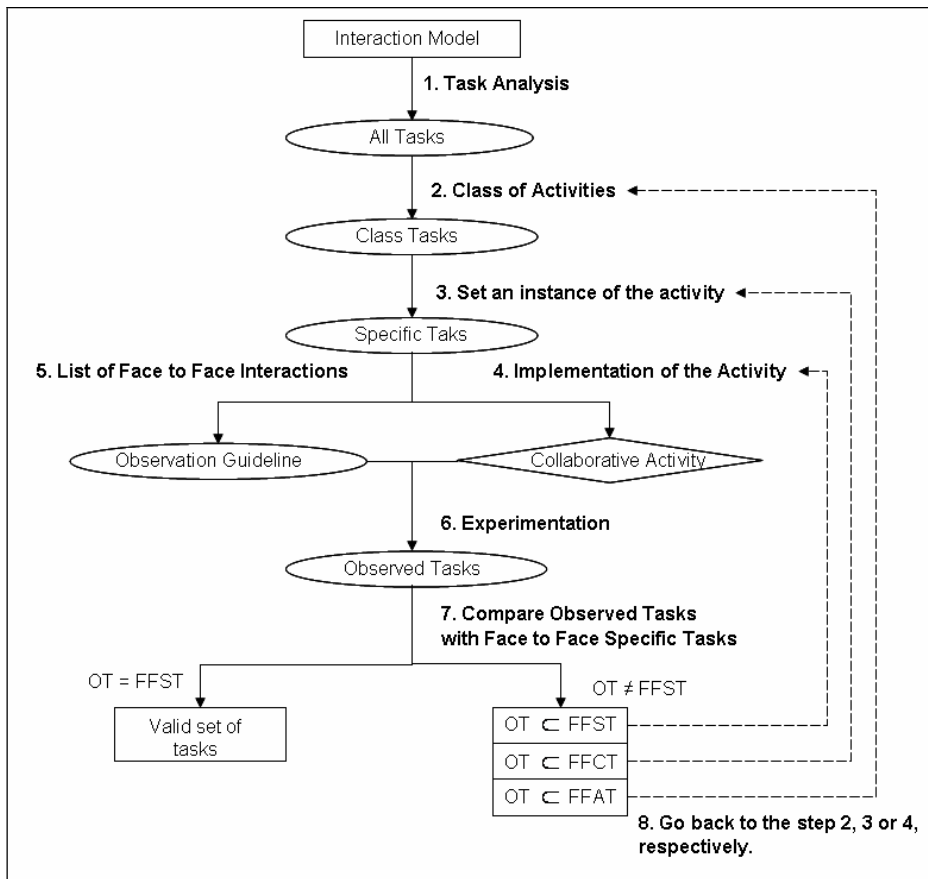


Fig. 2. Graphic representation of the Task Analysis based Methodology

The **third step** defines an instance of the selected Class of Activities. This determines the subset of Specific Tasks (ST) from the Class Tasks, which allows us to design the specific activity, as a sequence of these tasks. In the example, all the

students receive a different individual part ($T \rightarrow PDA \rightarrow WC$) they have to solve independently. They have to write an attribute they think is related with the received concept. In this way each student is responsible of his/her own work. Then the individual work response is sent from every student to the teacher ($S \rightarrow PDA \rightarrow T$). S/he sends the responses s/he thinks are relevant to every student to perform a collaborative work with the whole class, ($T \rightarrow PDA \rightarrow S$) to find together, by a voting mechanism ($S \rightarrow PDA \rightarrow T$) the right concept. Through the teachers' machine a student is randomly selected ($T \rightarrow PDA \rightarrow S$) to justify his/her answer in front of the whole class, so a collaborative class discussion is built ($S \rightarrow FF \rightarrow WC$, $T \rightarrow FF \rightarrow WC$). Taking into account the discussion, the teacher may eliminate the attribute, continuing the previous dynamics until the whole class finds the right solution. In Table 4 we show the set of Specific Tasks defined for our example, i.e., the set of valid tasks, from which we design an activity for high school physics, specifically for teaching sound waves. The activity aim is that students relate different attributes as transmission, vibration, frequency, etc, with the physical concept of sound waves.

Table 2. Methodology based on Task Analysis that supports the definition of an activity

| | |
|---|---|
| 1 | Perform a Task Analysis on the Interaction Model to define the set of all possible tasks for the design of a collaborative activity named All Tasks (AT) |
| 2 | Define a Class of Activities to obtain from AT the subset Class Tasks (CT) |
| 3 | Define a set of Specific Tasks (ST) setting an instance for the defined Class of Activities |
| 4 | Implement the sequence of Specific Tasks that builds the Collaborative Activity (CA), using the subset of Specific Tasks defined in the previous step |
| 5 | Design an Observation Guideline (OG) listing the face to face interactions that belongs to the three sets of tasks ST, CT and AT, obtaining three subsets: Face to Face Specific Tasks (FFST), Face to Face Class Tasks (FFCT) and Face to Face All Tasks (FFAT). |
| 6 | Obtain experimentally a set of interactions, or Observed Tasks (OT), from the execution of the activity |
| 7 | Compare the obtained set of Observed Tasks with the subset of Face to Face Specific Tasks to validate the system |
| 8 | If OT has not the same tasks as FFST, go back the corresponding number of steps, depending if the Observed Tasks elements belong to FFAT (step 2), FFCT (step 3) or FFST (step 4). |

In the **fourth step** we implement the sequence of tasks to build the Collaborative Activity (CA), using the subset of Specific Tasks (ST) defined in the previous step. We are going to show the design of the activity. It consists of eight parts; two of them belong to individual work and the others to collaborative work. First, we show the individual work and then, the collaborative one, each with the users PDAs' corresponding screens (Fig. 3 to Fig. 5).

Table 3. Class Tasks for the defined problem in our example

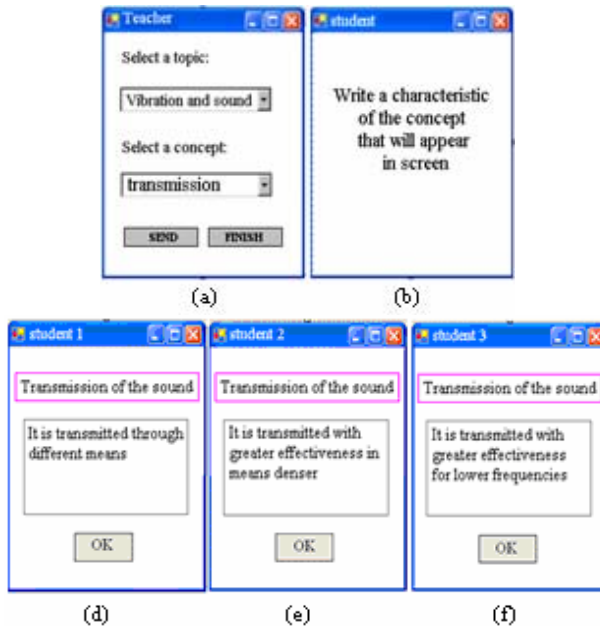
| | |
|---|------------------------------------|
| 1 | $T \rightarrow PDA \rightarrow S$ |
| 2 | $S \rightarrow PDA \rightarrow T$ |
| 3 | $T \rightarrow PDA \rightarrow WC$ |
| 4 | $T \rightarrow FF \rightarrow S$ |
| 5 | $S \rightarrow FF \rightarrow WC$ |
| 6 | $S \rightarrow FF \rightarrow S$ |
| 7 | $T \rightarrow FF \rightarrow WC$ |
| 8 | $S \rightarrow FF \rightarrow T$ |

Table 4. Specific Tasks for the specific problem in our example

| | |
|---|------------------------------------|
| 1 | $T \rightarrow PDA \rightarrow S$ |
| 2 | $S \rightarrow PDA \rightarrow T$ |
| 3 | $T \rightarrow PDA \rightarrow WC$ |
| 4 | $S \rightarrow FF \rightarrow WC$ |
| 5 | $T \rightarrow FF \rightarrow WC$ |

A.- Individual Work

- 1) $T \rightarrow PDA \rightarrow WC$: The teacher selects a topic and a related concept (Fig 3a). The student receives the corresponding instruction (Fig. 3b).
- 2) $S \rightarrow PDA \rightarrow T$: All the students receive the same concept and write an attribute they think is related with it and sends it to the teacher. Fig. 3c, d, e, shows the screens of three different students.

**Fig. 3.** Individual Work

B.- Collaborative Work

- 3) **T→PDA→S**: The teacher selects a subset of the received answers (Fig. 4a) and sends it to the students. In the student's screens appears the information sent by the teacher.

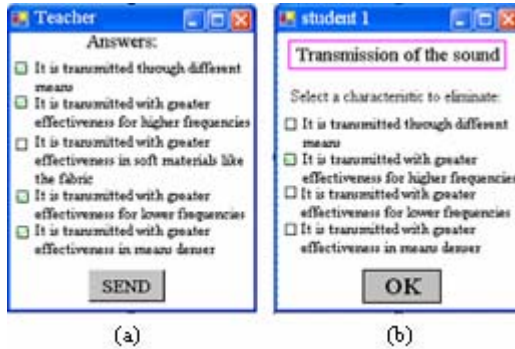


Fig. 4. Collaborative Work – Voting

- 4) **S→PDA→T**: The students vote to eliminate an attribute they think it does not correspond (Fig. 4b). Then, the students see on their screens the voting result (Fig. 5b, d).
- 5) **T→PDA→S**: The teacher asks the system to randomly select one of the students that chose the most voted alternative so that the corresponding student justifies it (Fig. 5a). The selected student is informed by an audible signal and by a colored background in his/her screen (Fig. 5c), while the other student's screen remains the same (Fig. 5b, d).
- 6) **S→FF→WC**: The selected student has to justify his/her vote in front of the class.
- 7) **T→FF→WC, S→FF→WC**: Begins an in class discussion mediated by the teacher.
- 8) **T→PDA→S**: If the class realizes that the most voted concept does not correspond, the teacher can eliminate it and repeat the cycle (4), (5), (6) and (7), without the already analyzed concept.

In the **fifth step** we design an Observation Guideline (OG) listing all possible face to face interactions to direct the viewer on what s/he has to watch during the activity.

Let FFST (Face to Face Specific Tasks) be the subset of all the face to face interactions of the Specific Tasks set, FFCT (Face to Face Class Tasks) be the subset of the face to face interactions of the Class Tasks set, and FFAT (Face to Face All Tasks) be the subset of the face to face interactions of the All Tasks set. To perform the Observation Guideline we first list the elements of FFST that constitutes the valid set of tasks. Then we list the elements of FFCT without repeating the already listed elements in FFST. Finally we list the elements of FFAT without repeating the already listed elements in FFCT. For the example, the obtained guideline is shown in Table 5.

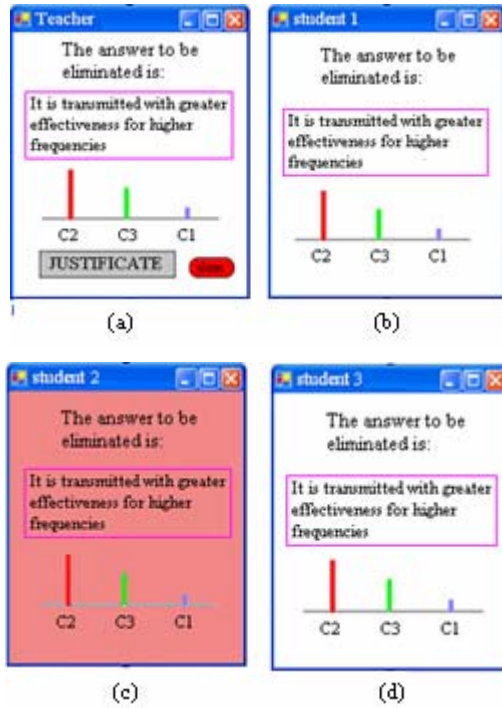


Fig. 5. Collaborative Work - Discussion

Table 5. Observation Guideline

| Tasks | Frequency | Observations |
|--|-----------|--------------|
| Face to face interactions from ST (FFST) taken from Table 4: | | |
| S→FF→WC | | |
| T→FF→WC | | |
| Face to Face interactions from CT (FFCT) taken from Table 3: | | |
| T→FF→S | | |
| S→FF→S | | |
| S→FF→T | | |
| Face to Face interactions from AT (FFAT) taken from Table 1: | | |
| SG→FF→S | | |
| SG→FF→T | | |
| T→FF→SG | | |
| S→FF→SG | | |

In the **sixth step** we write down all the observed face to face interactions generated during the execution of the Collaborative Activity in the classroom, their frequency of occurrence and any observation, oriented by the Observation Guideline. We obtain a set of tasks named Observed Tasks (OT).

In the **seventh step** we compare the obtained set of Observed Tasks with the face to face interactions listed in the Observation Guideline.

If the Observed Tasks are the Face to Face Specific Tasks, we can assure that the face to face interactions generated during the activity are those we wanted to happen when we designed it.

Otherwise, if OT has not the same tasks as FFST, we have to go to the **eighth step**. Here we describe what happens for the three different possible cases:

- If OT is a subset of FFAT, we experimentally observed at least one task that doesn't belong to FFCT, so we have to redefine the Class of Activities. We have to go back to the **second step**. In our example, we could have observed that the students discuss in small groups. This task doesn't belong to our FFCT. Therefore, to achieve the established activity aim we have to redefine the Class of Activities.
- If OT is a subset of FFCT, we experimentally observed at least one task that doesn't belong to FFST, so we have to redefine the instance of the activity and the sequence of tasks to perform the Collaborative Activity. We have to go back to the **third step**. In our example, we could have observed that a student discusses with another. This task doesn't belong to our FFST. Therefore, to achieve the established activity aim we have to redefine the Specific Tasks.
- If OT is a subset of FFST, we experimentally observed that at least one of the Face to Face Specific Tasks doesn't occur. So we have to rebuild the Collaborative Activity to obtain the desired interactions, going back to the **fourth step**. In our example, we could have observed that never occurs that a student discusses with the whole class. Therefore, to achieve the established activity aim we have to rebuild the activity to obtain it.

When the Class of Activities includes all the possible tasks, i.e., $AT=CT$, we have to ignore the first case. When we use all the tasks of The Class Tasks set to perform the instance of the activity, i.e. $CT=ST$, we have to ignore the second case. When $ST=CT=AT$ we have to consider only the third case.

4 Conclusions

In this work we have shown the use of Task Analysis for the design of collaborative learning activities supported by wirelessly interconnected handhelds that takes into account all possible interactions, between humans, and humans and computers. Through the use of the basic building blocks or tasks, resulting of the Interaction Model, we propose a methodology for the design of a face to face collaborative activity.

The proposed methodology, with the definition of classes of tasks and sets of specific tasks, facilitates the whole design, development and later validation of the system. It also allows us to make an observation guideline to direct the observer during the activity course, and to verify if the observed social interactions are the same subset of face to face specific tasks defined in the design of the activity. In this way, obtaining the same face to face interactions we select in the collaborative

activity design, we can study what interventions are necessary to achieve a determined social and communication behavior between the actors involved in the activity.

Acknowledgments

This work was partially funded by FONDECYT grant 1040605 and FONDECYT grant 1060712.

References

1. Fussell, S., Kraut, R., Siegel, J., Brennan, S. Workshop: Relationships among speech, vision, and action in collaborative physical tasks, CHI '02 extended abstracts on Human factors in computing system, April 2002.
2. Johnson, H., and Hyde, J. Towards Modeling Individual and Collaborative Construction of Jigsaws Using Task Knowledge Structures (TKS), ACM Transactions on Computer-Human Interaction, Vol. 10, No. 4, December 2003. pp 339-387.
3. Pinelle, D., Gutwin, C., Greenberg, S. Task analysis for groupware usability evaluation: Modeling shared-workspace tasks with the mechanics of collaboration. ACM Transactions on Computer-Human Interaction (TOCHI) Volume 10 , Issue 4 (December 2003) pp 281 – 311
4. Roschelle, J., Rosas, R., Nussbaum, M. Mobile Computer Supported Collaborative Learning (mCSCL): What adds “mobility to CSCL? Computer Supported Collaborative Learning Conference, Taiwan, July 2005.
5. Zurita, G., Nussbaum. MCSCL: Mobile Computer Supported Collaborative Learning. Computers & Education. Volume 42, Issue 3 , April 2004, Pages 289-314.
6. Zurita, G., Nussbaum, (2005). M. Dynamic Grouping in Collaborative Learning Supported by Wireless Handhelds, Educational Technology & Society, 8 (3), 149-161
7. Jon Stuart, Richard Penn, (2004). TaskArquitect: Taking the Work out of Task Analysis. ACM International Conference Proceeding Series; Vol. 86.
8. Youn-kyung Lim, (2004). Multiple Aspect Based Task Analysis (MABTA) for User Requirements Gathering in Highly-contextualized Interactive System Design. TAMODIA 2004, Prague, Czeck Republic.
9. Lagos, M., Nussbaum, M., Capponi M., (2005). A Mediation Model for Large Group Collaborative Teaching. 11th International Workshop, CRIWG 2005, Porto de Galinhas, Brazil, Vol 3706.

Group Creativity and Collaborative Technologies: Understanding the Role of Visual Anonymity

Traci Carte¹, Laku Chidambaram¹, Monica Garfield², Lindsey Hicks¹,
and Cassie Cole¹

¹ MIS Division, Michael F. Price College of Business, University of Oklahoma,
Norman, OK, 69121 USA

{tcarte, laku, Lindsey.B.Hicks-1, cass014}@ou.edu

² CIS Department, Bentley College

Waltham, MA 02452

mgarfield@bentley.edu

Abstract. This study expands on the current body of research examining technology-supported teams, individual creativity, and group diversity. By incorporating each of these elements into the experimental design, our objective was to determine how technology can best be leveraged to promote creativity in virtual teams. A lab experiment was conducted using 80 student teams by manipulating anonymity and capturing diversity characteristics. Preliminary results are presented which suggest that homogeneous teams generated more ideas; however, diverse teams were more satisfied with their output. Coding of the creativity of the ideas is on going.

Keywords: Creativity, Diversity, Lab Experiment.

1 Introduction

This study seeks to add to the study of creativity by integrating two areas of study: 1) the relationship between diverse team composition and creativity [1], and 2) the use of information technology (IT) to facilitate creativity [2, 3]. A prominent theme in research focused on workforce diversity is that companies should learn to manage diversity not only because of the on-going demographic trends but also because of diversity's potential as a source of competitive advantage [4]. This "value-in-diversity" theme rests on the hypothesis that diversity, when properly managed, produces tangible positive effects on organizational outcomes. One such positive effect is greater creativity. Diversity within a team refers to its composition in terms of the distribution of demographic traits and cognitive differences manifested as surface-level or deep-level attributes.

While much of the belief in the "value-in-diversity" hypothesis rests on anecdotal evidence, empirical evidence is emerging to substantiate this claim in general, and as it relates to creativity in particular. For example, McLeod, et al, [1] found ethnically diverse teams outperformed homogeneous teams by generating more creative ideas.

Further, Miura and Hida [5] found teams exhibiting deeper-levels of diversity (in terms of variety of perspectives) produced more creative ideas than teams that were similar in thinking.

Research on IT and creativity has focused on the productivity of electronic brainstorming (EBS) suggesting that technology provides a productivity boost to groups engaged in idea generation [3]. Typically, productivity of EBS has been operationalized as number of unique ideas generated and/or idea quality – both of which are elements of creativity. While not directly considering EBS, some previous work [6] describes how technology can help leverage the positive aspects of diversity while limiting its negative aspects. Essentially arguing that technology can reduce the immediate salience of surface-level diversity, the key source of process losses in diverse groups [7], this theory of accelerated technology deployment may provide additional insight into the use of technology to facilitate group productivity/creativity.

A factor that is likely to improve interaction processes and minimize process losses is the visual anonymity inherent in many collaborative technologies (CT). Visual anonymity prevents team members from seeing physical cues that identify their teammates as individuals [8] – in a virtual team context this means not putting a face with a (user)name. This capability may impede group members' perceptions of diversity within the team thereby limiting any process losses that otherwise might have occurred. In addition to visual anonymity, many CTs also provide a level playing field to all participants, and the combination of these capabilities has been shown to lower evaluation apprehension and increase participation [9] potentially leading to greater satisfaction.

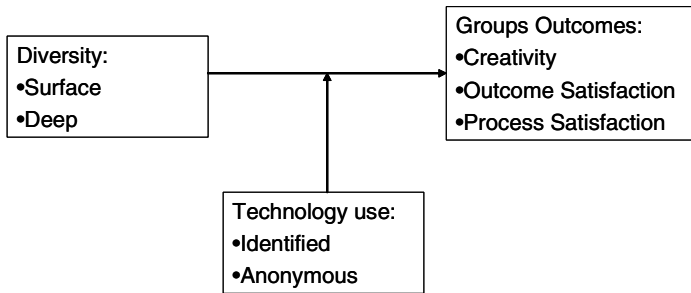


Fig. 1. Research Model

Integrating these research streams suggests that technology-supported diverse teams will outperform (i.e., be more creative than) homogeneous teams because technology is likely to improve interaction processes and facilitate more equal member participation. We propose that the cause of increased creativity is a specific capability of the technology, namely anonymity, which in combination with the

geographic dispersion inherent in virtual teams creates visual anonymity. Our research model is presented in Figure.1. Our primary research hypothesis is this:

H1: Visually anonymous diverse teams will produce more creative ideas than

a) anonymous or identified teams (diverse or homogeneous)

b) homogeneous teams (visually anonymous, anonymous or identified)

H2: Visually anonymous diverse teams will be more satisfied than identified diverse teams.

2 Empirical Design

Our study employed a 2x2 repeated measures design. Over 400 voluntary subjects were drawn from an introductory class and randomly assigned to 80 teams ranging in size from three to five students. The teams generated ideas for two tasks (using GroupSystems, a commercially available EBS environment). The tasks were: how can the university improve recruitment of diverse students (diversity was defined in terms of gender and ethnicity), and how can the university reduce risky student behavior (e.g., excessive drinking, drug use, sexual encounters). These tasks were created for two reasons: 1) to make salient surface and/or deep level differences in the teams and 2) to tap into the teams' inherent expertise. The recruitment task, due to its focus on recruiting ethnic minorities and women, was expected to make salient perceived differences in gender and ethnicity within the teams (i.e., surface-level diversity), while the risky behavior task (focus on reducing drinking, drug use, and sexual behavior) was expected to make salient differences in morals/values (a component of deep diversity). Further, these students having recently been the targets of the policies involved in these tasks may be seen as having reasonable expertise. The session procedures are detailed in Table 1.

The particular manipulation in which we were interested was identified versus non-identified/anonymous interaction. In our 2X2 design, participants were identified in one of two treatments. The repeated measures design created an opportunity to compare visual anonymity versus anonymity versus identified EBS. In the condition where anonymity happened first, the teams experienced visual anonymity because they did not know which people in the room were members of their team versus other teams. When anonymity happened second, the participants knew who was on their team; hence the treatment simulated anonymity but not visual anonymity.

In addition to creating a visual anonymity context, our design also attempted to simulate conditions ripe for creative outcomes. The factors that shape creative performance have been well studied; the creativity model applied in this study was proposed by Amabile [10] and suggests that there are three components of creative performance: task motivation, creativity skills, and expertise (see Figure 2). Drawing on the three-component model we used tasks for which our participants had expertise, questionnaire items to tap into creative self-efficacy (to capture the "creative skills" component), and a \$200 cash prize for the team that did the best job on the two tasks (to motivate the students).

Table 1. Session Procedures

| |
|--|
| <ol style="list-style-type: none">1. Participation of 15-30 students per session was solicited2. As participants arrived for their session, they were randomly assigned to a team (team seating was distributed throughout the lab so that team members were never seated next to each other);3. A warm-up task was completed to train the students in using the technology.4. The first task was completed. The process depended on treatment ordering:<ul style="list-style-type: none">○ If the first treatment was anonymous, the participants completed the first task (having no knowledge of who in the room was on their team)○ If the first treatment was identified, the participants met their teammates then completed the first task (with their names attached to each idea posted)5. Survey 1 was completed (this included demographic and perceived diversity questions).6. The second task was completed. This process also depended on treatment ordering:<ul style="list-style-type: none">○ If the first task was anonymous, the participants met their teammates before completing the second task – which would be identified (i.e., names attached to ideas).○ If the first task was identified, then this one was anonymous. Participants would have met their teammates before the first task, so had knowledge of who was on their team, but the messages for this task would not include identifying information.7. Survey 2 was completed (this asked about satisfaction with the team process and outcome, and creative self-efficacy). |
|--|

**Fig. 2.** Three-component model of creativity

Because not all students who signed up to participate actually showed up, our experiment design was not balanced. Ultimately 80 teams participated, and the number of teams in each cell is reported in Table 2.

Table 2. Number of teams by treatment

| | Risky task first | Diversity task first |
|------------------|------------------|----------------------|
| Anonymous first | 20 | 18 |
| Identified first | 23 | 19 |

3 Coding the Ideas

Two raters were recruited to code the ideas generated by the subjects in this experiment. This process is on-going. The ideas will be coded based on three theoretical constructs of interest: originality, meaningfulness and paradigm relatedness. Prior to coding, all data needed to be cleaned to remove non-ideas and redundant ideas (see appendix A for coding definitions).

To ensure our raters were adequately trained we used a set of practice data (data that was collected during pilot testing of our experimental design) for our coders to gain experience with the coding constructs as well as to refine the coding scales. The coders used three sets of practice data, after they coded each set we stopped and discussed any disagreements in how the data was coded and clarified any questions the coders had about the meaning of the constructs and coding scales. This process allowed our raters to develop shared mental models that reflected the underlying constructs (originality, paradigm relatedness and meaningfulness) used in this research. We determined training was complete when the Cronbach's alpha used in assessing the inter-rater reliability measures were .65 and over. Once training was completed, each coder was assigned to code all of one task and a third of the other task to enable us to calculate inter-rater reliability. This coding is expected to be completed by early May.

4 Preliminary Data Analysis

The data were first evaluated to verify that our manipulation worked. After completing the first task participants were asked: How different are the members of your group in their ethnic background? (1= not at all...4= neutral...7=very different). ANOVA results were significant ($F_{7, 67}=8.03$; $p=.000$), and in the direction anticipated. Participants in the unidentified treatments were largely neutral in their assessment of the ethnic similarity of their group, while participants in the identified treatments varied in their perception depending on the ethnic make-up of their teams.

While the coding is still on going for the creativity measures, we did run preliminary analysis on number of ideas generated per person and team satisfaction with the outcomes. There was a significant treatment effect for number of ideas ($F_{3, 62}=7.735$, $p=.000$) as show in Figure 3, but no significant differences in number of ideas generated by diverse versus homogeneous teams. Interestingly, our teams generated more ideas per person about increasing diversity than about reducing risky behavior regardless of treatment order. ANOVA results for satisfaction bordered on significant ($F_{3, 67}=2.44$, $p=.072$); see Figure 4. Teams who participated in the identified-first treatment were more satisfied, contrary to recent findings [11]. Diverse teams were more satisfied than homogeneous teams, but not significantly so. While

these results do not provide direct support for our hypothesis, they do not contradict it either. Taken together the lower number of ideas generated and the higher levels of satisfaction reported suggest that our diverse teams may have produced higher quality ideas.

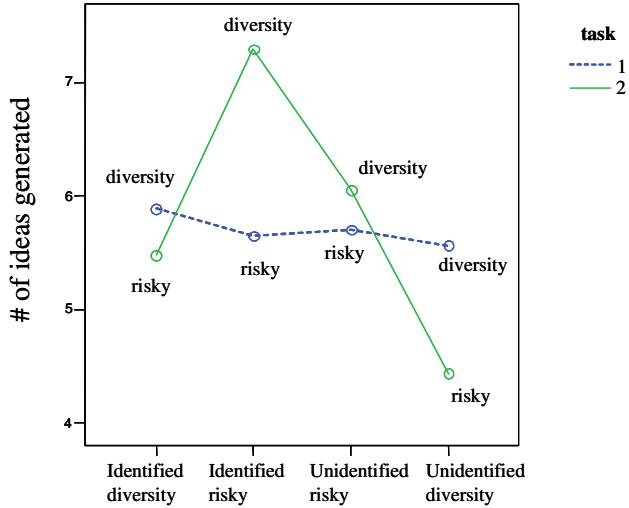


Fig. 3. Preliminary results for # of Ideas

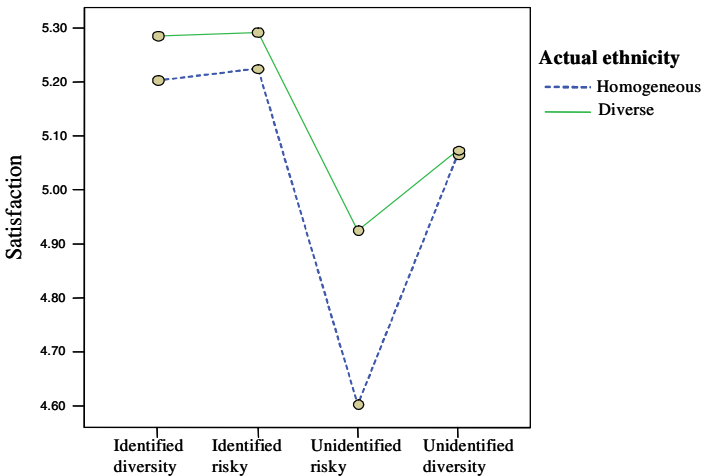


Fig. 4. Results for satisfaction

Two key findings emerge from our preliminary data:

1. Task salience is influenced by inherent attitudes, regardless of technology: A conservative student body, which the subjects in our study represent, is likely to be reluctant to discuss risky matters openly. We found evidence of such

behavior in our study. Under all conditions, the diversity task compared to the risky task resulted in more ideas. Moreover, the two “identified/risky” conditions had the lowest number of ideas compared to the other comparable conditions. Thus, member attitudes helped surface the salience of the task—regardless of their use of technology. It is likely that some issues, in conservative settings, are difficult to discuss openly. In addition to technology, greater interaction and longer timeframes may enable more open dialog in such cases.

2. A ‘reverse fatigue’ factor can enhance creativity: Conventional wisdom dictates that groups working on creative tasks become fatigued over time. Such fatigue may be exacerbated as groups switch from one task to another. However, in our study, we found that groups who did the risky task first and the diversity task next, contributed significantly more ideas the second time around. Thus, rather than declining creativity, they exhibited enhanced creativity. The relief from not having to deal with a risky task, translated into greater productivity despite the passage of time. In other words, when faced with the choice of delicate (or uncomfortable) tasks versus comfortable tasks, it appears that task order does matter. Hence, traveling the less traveled path first, and then turning to the more familiar path next might be an option that managers concerned with increasing the creativity of their teams may want to consider.

5 Expected Results

Ideally analysis of our final dataset will support our contention that visually anonymous diverse teams using EBS will be more creative than purely anonymous diverse teams and more creative than homogeneous teams. This analysis will be available for presentation prior to the September meeting date. Moreover, we hope to uncover new knowledge about best practices for eliciting creative output from teams. If our hypotheses are confirmed we will have a better understanding about using technology to improve the process and outcomes of diverse virtual teams. This knowledge is of growing importance as more and more organizations are relying on virtual teams that span geographic distances, time zones, and cultural and ethnic backgrounds.

References

1. McLeod, P., S. Lobel and T. Cox Jr. (1996) “Ethnic Diversity and Creativity in Small Groups,” *Small Group Research*, 27(2): 248-264.
2. Dewett, T. (2003), “Understanding the relationship between information technology and creativity in organizations,” *Creativity Research Journal*, 15 (2&3), 167-182.;
3. Dennis, A., Valacich, J., Carte, T., Garfield, M., Haley, B., and Aronson, J. (1997), “The effectiveness of multiple dialogs in electronic brainstorming,” *Information Systems Research*, 8(2), pp. 203-211.
4. Cox, T. and Blake, S., (1991), “Managing cultural diversity: implications for organizational competitiveness,” *Academy of Management Executive*, 5, 45-56.

5. Miura, A. and Hida, M. (2004), "Synergy between diversity and similarity in group idea generation," *Small Group Research*, 35(5), 540-564.
6. Carte, T.A. and Chidambaram, L., (2004), "A Capabilities-based Theory of Technology Deployment in Diverse Teams: Leapfrogging the Pitfalls of Diversity and Leveraging its Potential with Collaborative Technology," *Journal of the AIS*, 5(11-12), pp. 448-471.
7. Watson, W., K. Kumar and L. Michaelson (1993) "Cultural Diversity's Impact on Interaction Process and Performance: Comparing Homogenous and Diverse Task Groups," *Academy of Management Journal*, 36(3): 590-602.
8. Walther, J. B., Slovacek, C. L., and Tidwell, L.C., (2001) "Is a picture worth a thousand words? Photographic images in long-term and short-term computer-mediated communication," *Communication Research*, 28(1), 105-136.
9. Dennis, A., J. George, L. Jessup, J. Nunamaker Jr. and D. Vogel (1988) "Information Technology to Support Electronic Meetings," *MIS Quarterly*, 12(4): 591-525.
10. Amabile, T. M. (1997), "Motivating creativity in organizations: On doing what you love and loving what you do," *California Management Review*, 40(1), 39-58.
11. Pissarra, J. and Jesuino, J.C. (2005), "Idea generation through computer-mediated communication: the effects of anonymity," *Journal of Managerial Psychology*, 20(3/4), pp. 275-291.

Appendix 1: Steps in Coding the Data

1. *Idea identification* - Ensure each idea is only a single idea and is an "idea"

Idea definition: An idea is identified as a unique idea when it adds a new piece of information that pertains to the task domain beyond what the group has previously typed.

Idea or not?

- a) remove all agreement statement's or non-task oriented statements
- b) Is the idea too ambiguous? ... Can you see how it may be a solution or is it really a re-statement of the problem. i.e., There just aren't many minority students in Oklahoma or Other schools are a better fit for the minority students. – these are non ideas
- c) remove duplicate ideas

When to break an idea into 2 ideas

- a) You need to break an idea into 2 ideas if each idea is unique and adds something different to the solution space.
- b) Do not break the idea if there is a general idea followed by only one example. However, if there is a general idea followed by 3 examples you would make those three ideas.

Example

To attract more minorities we should advertise more like at sporting events, at college fairs, or at academic summer programs for high school kids.

This would be 3 ideas. Note: the general statement - advertise more - is not counted as an idea but the 3 specific places to advertise are.

2. *Code for originality*

Definition of Originality: An idea is most original if no one has expressed it before. Originality is judged from your perspective - that is according to the ideas you expected to see (it is not according to the actual ideas generated). Things to consider:

Is it really "out of the ordinary."

Does it provide an unconventional way to solve the problem

Could it be considered as revolutionary,

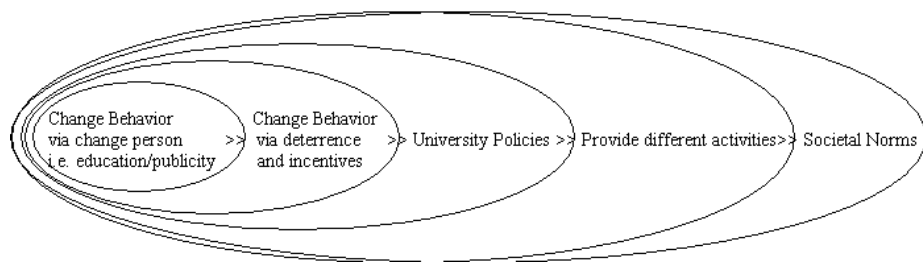
Would it make a radical difference

Is it interesting/stimulating.

Scale: 1 2 3 4 5
 Common/ordinary radical/revolutionary, unconventional

3. *Code for Paradigm Relatedness*

Paradigm relatedness: The degree to which an idea preserves or modifies a paradigm. Paradigm preserving (PP) ideas support or extend an existing paradigm; they are evolutionary in that they *adapt* elements of the existing paradigm. Paradigm modifying (PM) ideas are revolutionary in that they *redefine* the problem or its elements.



The first ellipse is the most paradigm preserving while the last ellipse is the most paradigm modifying.

Scale 1 2 3 4 5
 most paradigm preserving (PP) most paradigm modifying (PM)

4. *Code for meaningfulness*

Definition of meaningfulness: does it help find a solution to the problem in a useful and appropriate way?

Things to consider:

Is the solution relevant to the Universities needs and expectations?

Is the solution suitable and appropriate for the Universities needs and expectations?

Is the solution useful for University?

Scale: 1 2 3 4 5
 Unsuitable/not appropriate Very useful/suitable and appropriate

InClass-RTD: Providing Support for Real-Time Threaded Discussions in the Classroom

Alberto L. Morán¹, Carmen Perez², and Marcela Rodríguez³

¹ Facultad de Ciencias, UABC, Ensenada, B.C., México

² Instituto de Investigación y Desarrollo Educativo, UABC, Ensenada, B.C., México

³ Facultad de Ingeniería, UABC, Mexicali, B.C., México

{alberto_moran, cperez, marcerod}@uabc.mx

Abstract. Groupware applications that provide support for in-class collaborative knowledge construction are becoming an important research topic for the CSCW and CSCL communities. They provide support for lightweight interaction among participants, and for these participants with an increasing set of pervasive features of augmented classrooms, thus creating new opportunities to teach, learn and collaborate. Regarding in-class interaction, however, most current technologies focus on providing support for participation that is demanded by the lecturer (which we term Requested Participations), and that is limited to short interactions. Thus, support for Spontaneous Participation, or for longer interactions is limited or absent. In this paper, we present InClass-RTD, a system that provides support for spontaneous participation in the form of threaded discussions during the lecturer's presentation. We present scenarios that motivated its development, and its design and implementation as a service of an Augmented Classroom. Additionally, results from a 3-month preliminary trial showed that InClass-RTD allowed the participation of 80% of the group, with an average of 8 spontaneous participations per student from a group of 10. We also found that the system allowed an average of 54 participations per hour. Future work considers the development of additional visualization techniques, and performing in-depth analysis to evaluate its pedagogical in-class potential.

Keywords: InClass-RTD, Real-time threaded discussions, In-class requested and spontaneous interactions, CSCL.

1 Introduction

Providing support for learners that are distributed in time and space by means of technology has been the aim of applications developed in the CSCW, and most notably CSCL communities for at least one and a half decades. There are a myriad of applications that provide support for online education or e-Learning.

A more recent trend aims at providing technological support for processes that occur inside the physical classroom (i.e. synchronous, co-located support), either by augmenting the capabilities of the physical space with "virtual" capabilities, or by providing customized support for the pedagogical processes that occur inside the classroom [9]. Examples of these support include lecture/presentation capture,

annotation and replay systems [e.g. 1], digital ink note-taking and participation systems [2, 12], and teacher/student and student/teacher interaction tools [e.g. 7, 11]. The main features of these latter tools, most notably include support for voting, feedback and question-answer interactions.

However, we found that these tools are mainly oriented to provide support for participation that is solicited by the teacher/lecturer (which we term Requested Participations), or that is mostly limited to short interactions (e.g. single question-response interactions). Thus, the provision of support for unsolicited participations (which we term Spontaneous Participation) or for plural and longer interactions among the participants (e.g. actual dialogs or discussion about specific topics) is very limited or lacking.

To address some of these limitations, we propose a characterization of in-class participation, for instance, based on their being required by the teacher or their arising spontaneously from the students, and whether they are addressed at someone (Personal requests) or to no one in particular (Open requests).

Thus, an application that intends to provide support for the first type of interaction, should allow the teacher to be in control of the timing and mechanisms to initiate the exchange, while for the latter, the control should be flexible enough as to allow students to express (e.g. pose or annotate) their doubts and thoughts without being so intrusive as to disrupt the class, but noticeable enough to allow the teacher to become aware of them (e.g. opportunistically). Further, an application of this kind might introduce support to allow interactions not only between the teacher and the student or vice versa, but also to enable student-student interaction, so that they can participate more actively in the collective knowledge construction process.

To illustrate these needs, let us consider the following scenarios:

Scenario 1: *John, a student, is attending a lecture; when a doubt arises, he raises his hand to ask a question. However, there are other students with other questions. The teacher, due to timing constraints, decides to address only a couple of questions, not including John's question, and continues with the class. At the end, John decides not to ask his question again.*

Scenario 2: *Steve, a teacher, starts his daily lecture. There is a system in the classroom that allows students to submit questions and to vote on submitted ones. Steve has a special viewer of the system that aggregates and classifies questions depending on their rank. At a certain point in the lecture, Mary, a student, submits a question, but no one else votes for it, so that it rests far behind the most voted ones. A moment later, Steve, noticing the questions, decides to answer the three top ranked questions and continues with the lecture. Thus, Mary's question rests unanswered.*

Scenario 3: *Linda, a teacher, is giving her lecture. There is a system that allows her to send questions (prepared on the presentation slides) to the students, and that allow students to submit their answers back to her. On her presentation, Linda arrives to one of the "question" slides, and submits it to the students. Ron, a student, prepares and submits an answer, as many of his peers have done. Even though he is almost sure to have responded it correctly, he has a doubt concerning when to use procedure A rather than procedure B. Linda reviews promptly the responses and notices that her students have correctly solved the problem and continues with the*

class. As the class goes on, additional question slides are sent and answered, Ron forgets his doubt, and leaves the class without asking for a clarification.

These scenarios highlight the need for i) mechanisms that allow more flexible and concurrent means for teacher-student, student-teacher, and student-student interaction, as well as for ii) mechanisms that allow spontaneous documentation and discussion of multiple-actor multiple-turn and simultaneous participations. Until now, current systems partially fulfill this need, and more adequate solutions are required.

In this paper, we describe InClass-RTD, a system that provides support for spontaneous participation in the form of threaded discussions in the context of lecture presentations. We achieved this by combining features of two technologies: Asynchronous Threaded Discussions and Instant Messaging (IM) systems. From the former, our proposal inherits the structure that allows contributing to a discussion at any particular point, not only at the end of the discussion. From the latter, it inherits the gathering and presentation of information on awareness of presence, identity, and of the possibility of interaction, as well as the ability of participating in the discussion at nearly real-time.

The paper is organized as follows: Section 2 presents features of the kinds of contributions that we have identified in the classroom, and the model we proposed and use to organize discussions. In section 3, we present InClass-RTD as a service of an augmented classroom, providing support for real-time threaded discussion of a lecture. Also in this section, we present InClass-RTD design and implementation. An actual scenario of use and results of a preliminary trial are presented in section 4, while related work is presented in section 5. Finally, our closing statements and directions of future work are presented in section 6.

2 Lecture Presentations and Participation

Lecture presentations might take any of the usual formats, i) the lecture is prepared and presented by the teacher or by an invited guest, ii) the lecture is prepared and presented by a single student, iii) the lecture is prepared and presented by a group of students, or iv) a combination of them all.

Further, lecture presentations can be open or closed to students' participation. In Open-participation lectures, the presenter allows (even invites) the students to interrupt at any given time to participate (ask a question, give a response or comment). However, in this case, one problem could be that usually there is no record of the interactions between the participants and the presenter. An additional problem is that due to time constraints the question/answer interactions are not open to discussions or comments by other members of the group. Another problem is that each question or comment interrupts the presentation process, which may affect the rhythm of the lecture.

In Closed-participation lectures, the presenter, explicitly or implicitly, asks the group not to interrupt the presentation and to leave the questions or comments at the end. In this case, the students are required to write down, or recall their questions or doubts later, in order to participate. Unfortunately, it is highly common that students forget their comments or doubts, or simply restrain themselves from asking questions or making comments due to their shyness.

However, even in this later case, it is common to observe that students tend to interact among them in order to ask questions, respond them, and comment on any of the topics covered in the presentation with other students sitting near them. Nevertheless, as they have been asked not to interrupt, these behaviors usually occur in a “private” manner, with the most proximate neighbors, at a low voice, and usually for a very short time.

We found this to be problematic because the same questions might be asked by several students at the same or different time; questions could be left unanswered if none of the most proximal neighbors knows the response; there could be several “correct/adequate” answers to the same question and people could be limited to one or few of them due to their “local reach”; only very few benefit from the “local” discussion that occurs; students don’t know whether there are peers with the same doubt or question they are tempted to ask; among others. For these reasons, we propose to develop a lecture discussion system that addresses and solves some of these issues. In the next section, we examine some of the features of participation in the classroom aiming at identifying specific requirements for our proposal.

2.1 Interactions and Participation in the Classroom

In-class participation is one of the processes that allow students to take active part in the construction of their own knowledge and to better grasp and remember what has been presented during the lecture. For instance, Razmov and Anderson [8] pose that student participation allows for the creation of an atmosphere of engagement, the expression of diverse opinions, the provision of feedback from the student to the teacher and vice versa, and finally, for active and collaborative learning.

This way, participation in these collaborative environments results from the interaction between teacher and students, and among students. We propose to consider features of the interaction to characterize different types of participation, as an aid in the identification of the type of support required. Namely, the features of interactions that we consider are: who is the initiator, at whom it is addressed, the number of turns involved, the number of actors involved, degree of concurrency, whether the identity of participants is known, and finally, the degree of privacy.

Let us illustrate our typology with some examples of Public or Semi-public interaction:

Example 1: *The teacher asks a question to a particular student, and the designated student provides an answer.* This is an example of a Requested participation, Directed to a particular student, involving a Single-turn interaction between 2 actors. It is a Sequential interaction as the teacher has interrupted the lecture to verbally ask the question to a student.

Example 2: *The teacher asks questions to anyone in the group, and several students, at their turn, provide answers and pose further questions, generating a discussion on the topic.* This is an example of a Requested participation, Non-directed to any one in particular, involving Multiple turns and Multiple actors. It is a Sequential interaction as the teacher has interrupted the lecture to ask the question.

Example 3: *The teacher doesn’t ask for participation as he continues to present the lecture material; and someone asks a question, makes a comment, and discusses with others without requiring the participation of the teacher.* This example involves a

Spontaneous participation, Non-directed to any participant in particular, involving Multiple turns, and Multiple actors. We could also highlight the concurrent nature of the interaction as it occurs in parallel to the lecture presentation, among a sub group of the Identified students.

This way, the type of participations that we are interested in supporting is better illustrated by the last presented scenario: *Interactions that are spontaneous, not directed (but regarded) by the teacher, involving multiple turns and multiple actors, that occur in parallel with the lecture presentation, among identified students, and that happen in a public manner.*

2.2 Requirements for the Provision of Support for Participation in the Class

Based on the features of participation identified, and presented in the previous section, we identified that a system aiming at providing augmented support for in-class participation, must fulfill the following requirements:

- R1) Active student participation in collaborative knowledge construction. Students are not only allowed to ask questions, but also to answer them, and to make comments on questions and answers. Thus, everyone (including the teacher) is allowed, and has “equal” access to the mechanisms to participate, as well as to the actual participations submitted by others.
- R2) Student-centered participation, not only Teacher-guided participation. Students are allowed to participate spontaneously, that is, participation doesn’t have to be requested (or controlled) by the teacher.
- R3) Concurrent lecture and Opportunistic participation activities. Any participant (including the lecturer) must be able to participate at any moment he/she requires doing so (e.g. add questions, responses and comments). Thus, they are allowed to submit contributions during lecture presentations, while others (including the teacher) are also participating.

It should be highlighted that the considered requirements do not represent a definitive set of requirements for in-class interaction and participation support, but a starting point from where to start studying them.

2.3 Threaded Discussions as Support for In-Class Participation

From social learning theory we have learnt that the construction of knowledge is a social activity, and we have learnt that peers are a valuable resource for individual as well as collective learning [5]. For centuries, we have worked with the Socratic Method, where dialogs and guided conversations are central to teaching and learning, especially at the university level. However, by definition, lectures are addressed to large numbers of students, making it difficult for all involved to participate actively; to create the conditions for an environment conducive to collaborative/collective learning in the classroom becomes thus of paramount importance.

Asynchronous discussion forums are standard tools in online learning environments, and their pedagogical value is being studied from the perspective of both, teachers, and students. The level of interaction in discussion forums might be deep, and the students’ level of participation has been found to correlate with

academic achievement [3]. Although the cognitive processes generated to participate in online classrooms are very different from those of traditional classrooms, active involvement for appropriation of knowledge is a requirement for both environments. For students, writing their thoughts on the subject matter and the possibility of contrasting and enriching them with those of their peers, besides the teachers' explanations, helps their learning process. The very expression of their doubts or ideas in writing supports their construction of knowledge. The students' access to records of discussions of past lectures for study purposes is considered also a plus. These issues motivated us to develop a discussion system to be used in face to face classes. The discussion model in which we based the system is described below.

2.4 Discussion Model

The discussion model of InClass-RTD (see Fig. 1) is based on the argumentation model of the IBIS method [4]. It consists of discussions anchored in Topics derived from the lecture content. In this model, we consider that students construct their own knowledge based on their previous experiences and the situation they are faced with, and that this process is affected by those things that are clear on their mind, and by those that represent unanswered questions or doubts (c.f. the Issues). We also consider that the learning process is enriched with the active participation of the members of the group – the teacher, and other students (c.f. the Stakeholders), as all of them bring their own background and viewpoints to the process by posing questions, responding them, and commenting on their doubts, answers and thoughts (c.f. the Conversation).

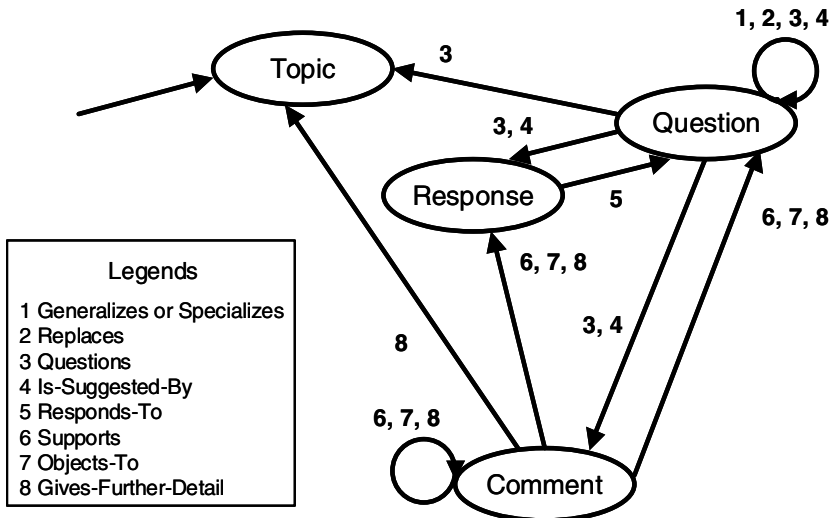


Fig. 1. IBIS-based discussion model used in InClass-RTD

Thus, our model focuses on the articulation of key Questions (c.f. Issues) that arise in, and are discussed as part of, the individual's knowledge construction process. Also, each question may have several different Responses (c.f. Positions). A response

is a statement that answers the question. There could be more than one “correct or adequate” response to each question.

In turn, each Question, or each question’s Response, may have one or more Comments (c.f. Arguments) associated. Comments support, object or provide further explanatory detail on Questions, Responses or other Comments. Therefore, each separate Question is the root of a thread, possibly empty, with the children of the Question being Responses or Comments, and the children of the Response being Comments. In addition, new Questions that are raised by the discussion may be inserted into the discussion in any node that most directly suggests them.

3 InClass-RTD: Support for Real-Time Threaded Discussions

InClass-RTD is an In-Class Threaded Discussion service that forms part of an Augmented Classroom. The main feature of InClass-RTD is that it allows having (near) real-time discussions on the contents of a lecture, while the latter is being presented, by organizing the participations in threads anchored on the topics and subtopics of the lecture.

These behaviors are inherited from two applications: Asynchronous Threaded Discussions (such as web-based Discussion Forums), and Instant Messengers (IM – such as Yahoo Messenger or MSN Messenger). This combination allows benefiting of the exclusive features of each technology. The former allows for having organized discussions, providing a space where the discussion of specific topics can be specialized, generalized, and even replaced as required by discussion on other topics. The latter, introduces features of awareness on elements such as presence and identity, as well as on the potential of participating in the discussion of one or more subtopics of a lecture in a lightweight and synchronous manner. Additionally, it allows moving seamlessly from attending the lecture and asking questions, responding them or commenting on the topics of the lecture.

An additional feature is that all clients of InClass-RTD are capable of presenting the same order for the discussion intra and inter threads, and updating it in (quasi) real-time. This was required as it is highly confusing trying to establish a common reference for the discussion, and looking at the colleague’s screen, and noticing that the presentation of each other is different, even though it refers to the “same” discussion or conversation. The case exacerbates when in addition one such a view is available to all the participants (e.g. through a public display). Thus, we needed to guarantee the same order and timely update of the discussions on all clients.

3.1 InClass-RTD Architecture

The architecture of InClass-RTD is based on a three-tier client-server model. The first tier, as illustrated in Fig. 2, refers to client applications, which provide the means to view and contribute to the class discussion, and to get the contributions of other participants. The second tier refers to services required for the client applications, such as User Authentication, Session Management, and Instant Messaging (reception, routing, and delivery). Finally, the third tier refers to storage services that are required by the previous services, namely, User and Session Storage services, as well as Participation Logging facilities.

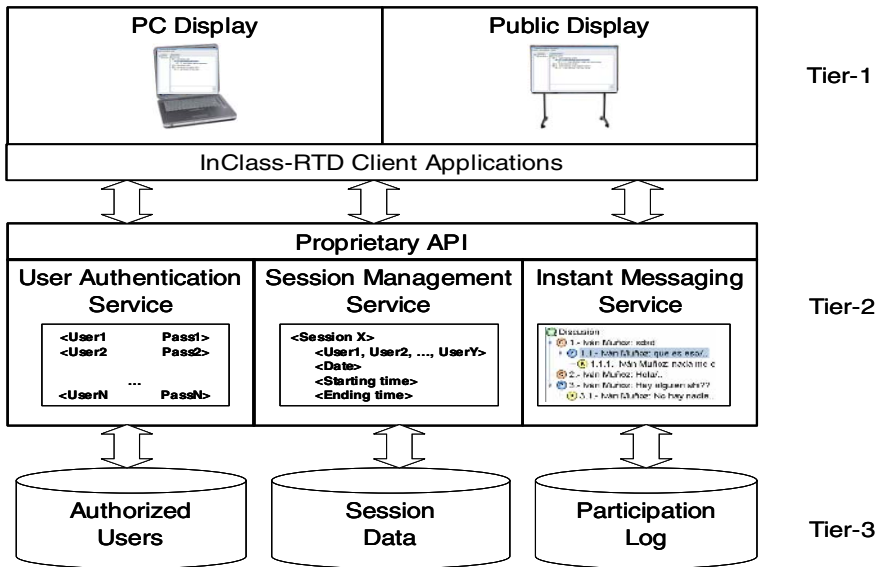


Fig. 2. InClass-RTD 3-tier client-server architecture

InClass-RTD Clients. Client applications come in two flavors, the “personal” or PC InClass-RTD client, and the Public Display InClass-RTD client. Currently, the main difference between them is the size of the font used in the client for the Public Display to facilitate reading at a distance. Both of them allow providing the information required for authentication, logging into a collaborative discussion session, selecting the particular point of a discussion on which to anchor the contribution, and actually sending, and visualizing contributions.

User Authentication service. It comprises both the server and the data storage. Currently it stores user’s information, and provides authentication based on specific information of the user (e.g. username and password).

Session Management service. Comprise both the server and the data storage for session information. Currently it stores information on the users present at the session, the date, and the session’s starting and ending time.

Messaging service. In a similar manner as the authentication and session services, this service comprises both the server and the storage management (log or history) of user activity (contributions) while at the discussion. This component is at the core of the whole system as it receives, processes, and delivers messages, as well as registers all participations of the discussion. This is the component in charge of guaranteeing /maintaining a unique order, and timely update, of the contributions in the discussions. This is achieved by maintaining a centralized master copy of the discussion tree. All other copies in the system (those of the clients), act only as references to this centralized copy under a master-slave relation. Thus, in order to modify the master copy (i.e. actually contributing something to the discussion), it is necessary to i) select an insertion point from the slave copy of the tree at the client-side, ii) enter the contribution, and iii) send it to the messaging service to try to execute this request. On the server-side, iv) the requests are received and serialized

into a unique order of execution (to guarantee the uniqueness of the master copy), and then v) applied to update the master copy of the tree in a mutually exclusive manner. Finally, vi) the changes resulting from the contributions are notified to the different clients of the system, so that they could update their copy of the tree to reflect the changes made to the master copy.

3.2 Implementation of the InClass-RTD System

The current implementation of InClass-RTD client applications was built using the Java language, version 1.5, and using the MySQL DBMS, version 5.0, as data storage. InClass-RTD clients provide the functionality described in the previous sections: 1) Allow users participating in a discussion, sending their own contributions, and receiving contributions performed by others; 2) Provide users with information to become aware of the presence and identity of other participants, and of the possibility of contributing to a discussion about a particular topic of the lecture; and 3) Allow users moving from being aware to actually contribute in a discussion.

At the server-side, InClass-RTD services are also built using Java version 1.5, along with the MySQL database for data storage. Specific modules were provided for each service: User authentication, Session management and Instant Messaging as specified by design. Specific tables containing the respective information were created in the database.

Communication between the client and middle-tier Java services were provided through a proprietary Java Object-based protocol, while the connectivity between the middle-tier Java services and the MySQL storage backend was achieved by means of the MySQL Connector/J Type IV JDBC driver version 3.1.

3.3 InClass-RTD's Graphical User Interface (GUI)

The GUI of the InClass-RTD client (see Fig. 3) heavily resembles the GUI's of the applications on which it is inspired. InClass-RTD GUI combines the tree-view message presentation of threaded discussion applications with the presence, conversation and submission panels of IM applications.

The Tree-view Message presentation (Fig. 3.a) permits the selection of the point in the discussion tree where the user wants to contribute. Each contribution (tree node) is decorated with iconic representations of the type of contribution (Q, R, C), with a text label with thread-related information, and a text label with information of the author of the contribution. Sub threads inside particular discussions can be shown or hidden by double-clicking on the parent node of the sub thread that we want to show or hide.

The Presence panel (Fig. 3.b) is located on the left-side of the application. It provides information on the presence and identity (nickname) of the participants with a combination of iconic and textual information. Nicknames can be changed to denote the "mood-of-the-day" or to provide some anonymity during the discussions.

The Conversation panel (Fig. 3.c) contains the tree-view message presentation, and provides vertical and horizontal scroll bar navigation to allow the visualization of discussion trees that grow bigger than the available visible space of the panel.

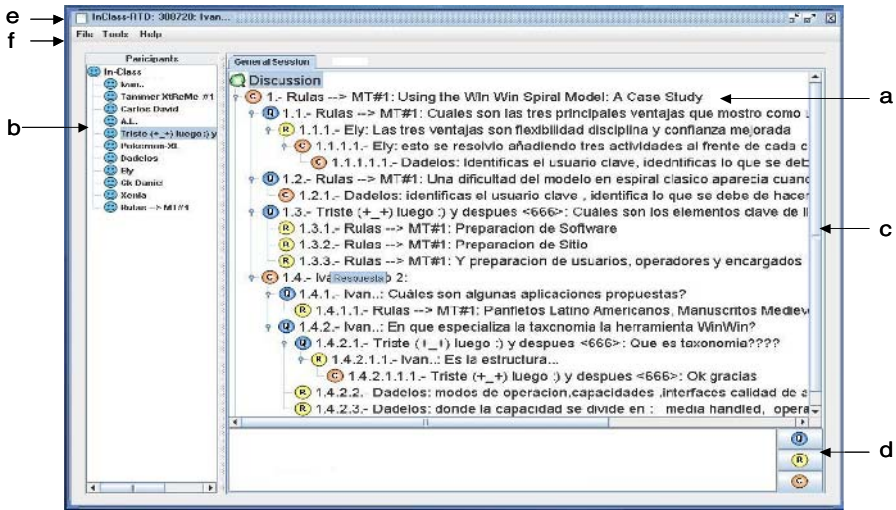


Fig. 3. InClass-RTD Graphical User Interface

The Submission panel (Fig. 3.d) contains the traditional text area where contributions are typed, and the buttons to send it. In this case, rather than providing the usual single button of IM applications, it provides tree buttons that allow sending the contribution as a (Q)uestion, (R)esponse, or (C)omment.

Additionally, there are an Application bar (Fig. 3.e) and a Menu bar (Fig. 3.f). The former provides awareness information on the identity of the current user, while the latter provides access to additional functionality such as i) logging or exiting from the system, ii) saving the threaded conversation to a file, iii) obtaining further information on the other participants (if available or allowed), iv) changing the user's nickname and password, and iv) obtaining help information on the application.

3.4 Content and Collaboration Public Displays

An additional key element of the InClass-RTD system is the configuration of the classroom (or lab) to support concurrent collaborative lecture presentations and discussions. We proposed the conceptual partition of the traditional Content Presentation Space (e.g. the whiteboard and/or LCD projector) where the content of the lecture is presented, to include a space to present information on the collaboration that occurs during the class (i.e. a Collaboration Information Presentation Space).

In our current setting (see Fig. 4), we achieve this with the introduction of a second LCD projector in addition to the one traditionally considered, in such a way that the first LCD projector is used as a Content Presentation Public Display, while the second LCD projector is used as a Collaboration Information Public Display.

Information that might be displayed in this latter space includes that on presence and identity, artifacts used or available for the work, and tasks performed by the participants, among others.

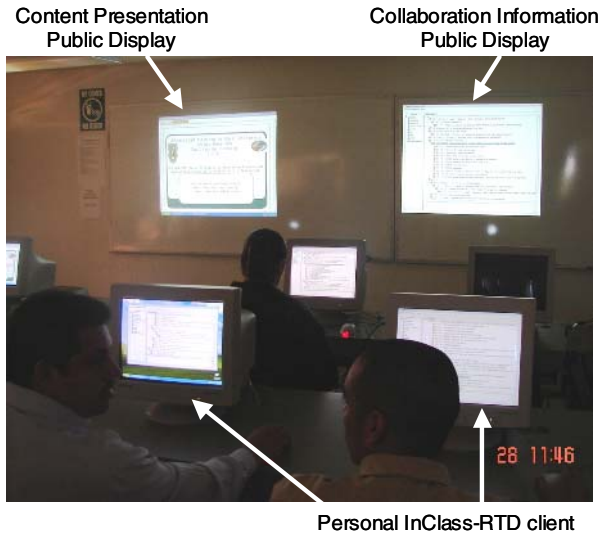


Fig. 4. Current InClass-RTD setting at a teaching laboratory

The Content Presentation Public Display is then associated to a service (or server) that allows any user to control the presentation (or production) of content from any computer connected to the service (currently limited to being present in the classroom or lab), while the Collaboration Information Public Display is associated to a service that allows customized applications to aggregate communication, coordination and production information on any of the tasks pertaining to the collaborative activity performed in the classroom or lab.

The novelty of our approach resides in that it makes possible to simultaneously follow the traditional lecture presentation, while being aware of, and capable of initiating and performing, additional collaborative knowledge construction processes (such as discussions on subtopics of the lecture), in a flexible manner. For instance, by means of InClass-RTD we provide participants with a potential collaboration space (the public display) and with an actual collaboration space (the personal client) where they could gather information to be aware of, and to decide when to seamlessly move from being aware of the possibilities of collaborative discussion, to actually participate in a discussion by either starting one, or contributing to an existing one [6], following the discussion model presented in section 2.

4 The System in Use

Let us illustrate InClass-RTD usage with the description of a typical scenario of the Software Engineering class where it is currently deployed.

4.1 Scenario of Use

It is 6:50 pm and the Software Engineering class is about to start. The teacher arrives, and the students start to get into the classroom. The InClass-RTD system (see Fig. 3)

is already up and running, and the students are starting to login, as presented by InClass-RTD in the Collaboration Information Public Display (rightmost projector in Fig. 4). A submission by one of them is already there as an anchor for the day's first Topic. Once all students are "logged in", the teacher starts the lecture using the Content Presentation Public Display (leftmost projector in Fig. 4). As usual, some of the students have changed their nicknames to denote their "mood of the day", as well as to acquire a more "comfortable" (quasi-anonymous) status during the discussion. As the topics of the lecture are presented, doubts start to arise and questions start to be posed through InClass-RTD. Questions start to be responded by students who know an answer, and without requiring interrupting the class. Further, the teacher opportunistically picks one of the threads of discussion to provide examples and counterexamples of what he is currently presenting. At the end, the whole group (teacher and students) verbally discusses what has been "annotated" in the InClass-RTD system, highlighting correct answers, clarifying misconceptions, expanding short or simplistic answers, and answering questions that remained unanswered. Students "document" these additional answers and comments in InClass-RTD as they are discussed.

4.2 Preliminary Results

At the time of the writing of this paper, InClass-RTD has been in use for nearly half a semester in a 10-student undergraduate Software Engineering class (3 female, 7 male, all of them between 21 and 25 years). Participants are advanced users of computers as well as of the tools on which InClass-RTD is inspired, such as web-based discussion and instant messenger tools. Our data corresponds to InClass-RTD use during 12 presentation-based sessions (February-April 2006), where readings on traditional and agile methods, as well as on CASE tools have been presented and discussed (4 presented by the teacher and 8 by the students). Table 1 presents a summary of the information gathered, and a discussion follows.

Regarding contributors, from the 10 potential users of the group (the lecturer does not contribute to the threaded discussion), an average of 8 persons regularly contributed by asking, responding or commenting on the lecture's topic, or by annotating other's verbal participations. Regarding the duration of the discussions, although the class is scheduled for 2 hours, the discussions, from the first contribution to the last, averaged 1 hour and 19 minutes. Thus, the longest session, number 7, lasted 1 hour and 34 minutes, while session 10, the shortest, lasted only 43 minutes.

Table 1. Contributions to the discussion using InClass-RTD per session

| Session | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | Mean | S.D. |
|------------------------|----|----|----|----|----|----|----|----|----|-----|----|----|-------|-------|
| Contributors | 10 | 7 | 7 | 10 | 10 | 8 | 9 | 9 | 7 | 9 | 4 | 10 | 8.33 | 1.82 |
| Duration (in minutes) | 92 | 90 | 79 | 80 | 61 | 88 | 94 | 73 | 59 | 43 | 78 | 86 | 79.91 | 15.52 |
| Total contributions | 91 | 44 | 69 | 46 | 67 | 73 | 97 | 96 | 46 | 78 | 17 | 78 | 66.83 | 24.32 |
| Contributions per hour | 59 | 29 | 52 | 34 | 65 | 49 | 61 | 78 | 46 | 108 | 13 | 54 | 54.59 | 24.48 |

Concerning participation, there was an average of 66 participations, which makes a rough 8 contributions by each student. The most participative session, number 7, obtained 97 contributions, while session 11, the least participative, obtained 17 contributions.

Additionally, regarding participation per hour, there was an average of 54 contributions per hour, which would represent nearly 1 contribution per minute in the class, if the participations had occurred sequentially.

Although a more complete analysis of the contributions is required in order to interpret the data, we consider that it is worth making some comments on the most and least participative sessions.

Session 10 was the shortest session of all (only 43 minutes). What we found intriguing was that this was the only session where the students worked by themselves on a designated topic, as the teacher was absent that day. So, they might have remained in the classroom, and in the discussion, just enough as to provide evidence of their attending the student’s lecture and discussing the topics – the teacher considered the log of the discussion as evidence of their participating in the class. However, this session is also the one with the most participation per hour (an estimate of 108 participations per hour). This may suggests that students are finding value in using the tool for discussion in the classroom, either on academic or social topics, regardless of the presence of the teacher, but certainly the analysis of the content of the discussion is required to confirm this. Also, it is important to review the role of the teacher in class discussions.

Session 11 was the session with the least participation, totaling only 17 contributions, with only 13 messages per hour. A possible explanation for this may be that the topic of the lecture (the presentation of a web-based Project Management tool) was not of interest to the members of the class, as they were assigned to roles such as architect, analyst, designer, and programmer, among others. Perhaps the most interested user in the topic (the project manager), was the one giving the lecture. Once again, further analysis on the content of the discussion is required to confirm this. Nonetheless, the handling of the topic in the presentation in order to make it interesting for the students and elicit discussion is worth reviewing.

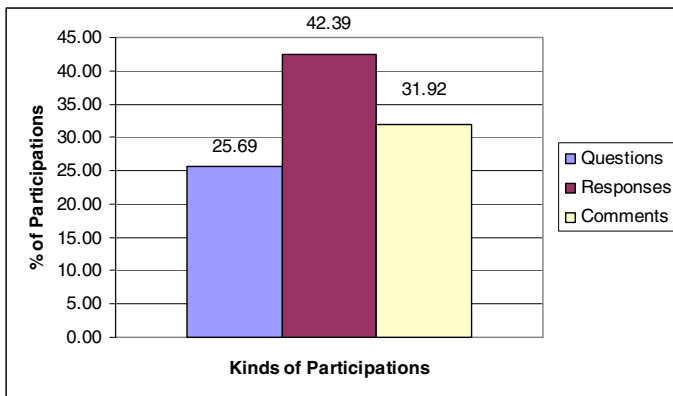


Fig. 5. Percentages of contributions in all classes by type

Finally, Fig. 5 shows the percentages of contributions in all class' discussions by type (questions, responses and comments). From a total of 802 participations in the 12 sessions, 25% of them (206) were questions, and 42.39% of them (340) were responses; an average of almost 1.7 responses for each question. As we can observe, about 2/3 of the total contributions were question/answer interactions, with the remaining 1/3 being comments on them (31.92% or 256). Considering the model proposed to form the discussions (see section 2), this may suggest that questions are mostly answered and commented, but once again in order to confirm this, content analysis of the contributions is required.

5 Related Work

Abowd [1], in the Classroom 2000 project, states that the capture of everyday experiences of the occupants of future computing environments, could provide support for both teaching and learning activities by creating a multimedia record of the activities that occurred on the standard synchronous university lecture. The proposed support consists in the automatic capture of the lecture presentation, augmented with audio or video recordings and notes from the teacher and the students during the class. It also includes the provision of the recorded multimedia information for replay through a web-based interface. InClass-RDT shares this vision of capturing in-class activity, but rather than documenting the lecture and note-taking processes as a whole, it focuses in offering and documenting an additional process, that of the discussion of the lecture occurring in a concurrent manner. Further, it could be considered that Classroom 2000 considers the whole lecture as a "global" interaction, while InClass-RTD not only provides support for the group discussion as a "global" interaction, but also it provides support for the "multiple" intra discussion interactions that make up the global discussion.

The ActiveClass project [7, 11] aims at providing support for increased student participation in large classes by allowing students to anonymously ask questions, answer polls and for giving feedback to the professor using personal mobile wireless devices (e.g. PDA's). Similarly, but focused on the provision of feedback to the teacher, is the Classroom Feedback System (CFS) [2]. The feedback is provided by selecting predetermined categorized feedback entries (More Explanation, Example, and Got It), which are presented at the lecturer's slides in real-time. Then the lecturer can respond or react to the feedback opportunistically. Although ActiveClass and CFS provide support for spontaneous participation involving multiple users, they do it by means of "discrete" or "single-turn" interactions (or no interaction at all in the case of CFS), thus lacking support for real-time, "multiple-turn" interactions as supported by InClass-RTD.

Classroom Presenter (CP) [2] and Ubiquitous Presenter (UP – an extension of CP) [12] are systems that support the sharing of digital ink written on electronic slides that integrate these slides with student devices (Tablet PC's, Laptops or Desktops), and which provide mechanisms for students submissions (students annotate and send back the slides to the instructor). However, although both CP and UP allow for teacher-student and student-teacher interaction, by means of single-turn, multiple-actor interactions, these interactions require to be requested or enabled by the instructor,

thus providing the means for Requested interactions, and a form of “Controlled” Spontaneous interactions. In contrast, InClass-RTD public discussion model, allows for Requested interactions, as well as for “Uncontrolled” Spontaneous interactions.

Threaded Chat [10], is a system designed to address some problems of current chat tools concerning support for task-based and decision-making discussions in the workplace. These problems include confusing history logs, lack of social history and rupture of turn sequences. As InClass-RTD, it combines features of asynchronous threaded discussions and synchronous chat tools to overcome these problems. It uses a turn-taking conversation model, a tree structure to represent the threaded discussion, selection of the insertion point (tree node) for each contribution (turn), and awareness information on the participants in the discussion. However, although Threaded Chat could be used to provide support for real-time “multiple turn” interactions, its main drawback resides in that it has not been designed to be used for in-class academic discussions. For instance, it lacks support for easily identifying specific threads for in-class verbal discussions; or for changing nicknames for (quasi) anonymous participation or to denote the “mood of the day” of participants. Finally, Threaded Chat’s amount of awareness information elements on the participants, and the mechanisms it uses to provide them (in a table format), are not well suited for in-class participation: they require a lot of screen real state and are a source of information overload not related to InClass-RTD’s central focus, the lecture topic discussion.

6 Conclusions and Future Work

In this work we have discussed InClass-RTD, a tool that allows discussing a lecture while it is being presented. The tool is based on a concept of concurrent “Spontaneous” contributions to discussions concerning subtopics of the lecture being presented. InClass-RTD features are based on a set of requirements, which are based in turn on identified features of the types of interactions and contributions that occur in the classroom.

Overall, preliminary results of a three-month trial period are encouraging. They suggest that InClass-RTD provides adequate support for discussing the lecture while it is being presented. We found that about 80% of the group usually contributed to the discussion and that there were in average 8 contributions per students at each class, with a contribution rate of about 54 participations per hour.

Finally, concerning directions of future work, they are directed to three specific areas: i) Continue with the development of the tool, mainly by introducing additional in- and post-discussion interaction and visualization techniques; ii) Perform in depth analysis of the data that it is being generated by InClass-RTD to determine its value as a learning tool; and, iii) Try out different pedagogical techniques for active learning and knowledge construction in the classroom with the use of InClass-RTD.

Acknowledgments. The authors would like to thank Carlos Iván Muñoz de la Cruz for implementing and deploying InClass-RTD. We would also like to thank the students of the CSCW class (2005-2) and the Research Exercise and Software Engineering classes (2006-1) for their participation in the development and experimentation of the tool. Finally, special thanks to Jesus Favela for his reviewing

and provision of insightful feedback on this paper. This work was supported by UABC, project 0188 of the Xma. Convocatoria Interna de Proyectos de Investigación.

References

1. Abowd, G. D.: Classroom 2000: An experiment with the instruction of a living educational environment. In *IBM Syst. J.* 38, 4, (1999), 508–530.
2. Anderson, R., Anderson, R., Chung, O., Davis, K.M., Davis, P., Prince, C., Razmov, V., Simon, B.: Classroom Presenter - A Classroom Interaction System for Active and Collaborative Learning. In *Proc. WIPTE 06*, (2006).
3. Cohen, M. & Ellis, T.: Predictors of success: A longitudinal study of threaded discussion forums. In *Proc. 33rd. ASEE / IEEE Frontiers in Education Conference*. Boulder, CO., November 5-8, (2003).
4. Conklin, J., Begeman, M.: gIBIS - A Hypertext Tool for Exploratory Policy Discussion. In *ACM Transactions on Office Information Systems*, vol. 6, (1988), 303-331.
5. Harasim, L., Hiltz, R., Teles, L., & Turoff, M.: *Learning networks: A field guide to teaching and learning*. Cambridge, Ma.: The MIT Press, (1995).
6. Morán, A. L., Favela, J., Martínez, A. M. and Decouchant, D.: On the Design of Potential Collaboration Spaces. In *International Journal of Computer Applications in Technology (IJCAT)*, Vol. 19, No. 3/4, (2004), 184-194.
7. Ratto, M., Shapiro, R.B., Truong, T.M., Griswold, W.G.: The ActiveClass Project: Experiments in Encouraging Classroom Participation. In *Proc. CSCL'03: Computer Support for Collaborative Learning 2003*, Kluwer, (2003), 477-486.
8. Razmov, V., Anderson, R.: Pedagogical Techniques Supported by the Use of Student Devices in Teaching Software Engineering. In *Proc. SIGCSE 2006*, March 2006, Houston, TX, US, (2006), 344-348.
9. Roschelle, J., Pea, R.: A walk on the WILD side: How wireless handhelds may change computer-supported collaborative learning. In *International Journal of Cognition and Technology*, 1(1), (2002), 145-168.
10. Smith, M., Cadiz, J.J., Burkhalter, B.: Conversation trees and threaded chats. In *Proc. CSCW'00*, Pennsylvania, USA, (2000) 97-105.
11. Truong, T.M., Griswold, W.G., Ratto, M., Star, S. L.: The ActiveClass Project: Experiments in Encouraging Classroom Participation. Technical Report CS2002-0715, Computer Science and Engineering, UC San Diego, (July 2002).
12. Wilkerson, M., Griswold, W.G., Simon, B.: Ubiquitous Presenter: Increasing Student Access and Control in a Digital Lecturing Environment. In *Proc. SIGCSE'05: 36th SIGCSE Technical Symposium on Computer Science Education*, (February 2005), 116-120.

Technical and Environmental Challenges of Collaboration Engineering in Distributed Environments

Halbana Tarmizi, Matt Payne, Cherie Noteboom, Chi Zhang, Lucas Steinhauser, Gert-Jan de Vreede, and Ilze Zigurs

College of Information Science and Technology
University of Nebraska at Omaha

{htarmizi, mpayne, cnoteboom, czhang, lsteinhauser, gdevreede, izezigurs}@mail.unomaha.edu

Abstract. Collaboration in distributed settings has become a reality in organizational life, yet we still have much to learn. One important area of study is the integration of Collaboration Engineering in distributed, or virtual, teams. Collaboration Engineering offers promising guidelines for process structure, but its application in distributed arenas is just beginning to be studied. We report on the design and development of a new collaboration environment for the incorporation of Collaboration Engineering principles, as well as the results of an initial study that examined leadership and process structure effects on the development of shared understanding. We discuss both technical and environmental challenges for research on Collaboration Engineering in distributed environments.

Keywords: Collaboration Engineering, Virtual teams, Shared understanding, Process structure, Leadership, thinkLets.

1 Introduction

The increasing use of virtual teams and distributed work has provided fertile ground for research. A growing body of knowledge has examined such topics as trust, communication, participation, coordination, and effectiveness [1], [2]. In addition, technological support for distributed teamwork has evolved considerably [3], [4]. However, one of the most significant challenges from traditional teams still remains an issue in distributed environments, namely the processes that team members can use to achieve maximum effectiveness in different tasks. We also need to learn much more about which types of support tools and structures can be provided for team members so they can select and carry out appropriate processes without having to depend on a facilitator.

Collaboration Engineering methods and techniques have been developed to address this important issue, via the capture and design of successful repeatable collaboration processes [5]. Collaboration Engineering began in the context of same-time same-place meetings, addressing design of recurring collaboration processes that team members could use to reach their goals. The methods and techniques of Collaboration Engineering have not yet been applied in distributed situations to any great extent.

Yet it is precisely in virtual teams that such process structure is likely to require even more attention.

The purpose of this paper is to advance the concepts and application of Collaboration Engineering in virtual environments via a specific implementation and example. The paper describes the design and development of a new environment for supporting process use in virtual teams. We also report the results of an initial study that was conducted in that environment. The study was designed to examine concepts that have proved both important and especially difficult in virtual teams, namely the achievement of shared understanding through process use and leadership.

The paper contributes in several ways. First, conducting research on virtual teams is challenging on several fronts, since researchers have to rely on effective communication among participants scattered in different places who are dependent on technology as the means of achieving shared understanding. Indeed, some have argued that attaining shared understanding is a necessary precondition to collaborating at all, since people cannot collaborate or agree if they do not understand what it is they are collaborating or agreeing on [6]. Achieving shared understanding among participants on a team, for instance, will require effective communication and coordination [7]. However, establishing communication and coordination among voluntary participants is challenging, and we provide an examination of how such challenges occur and might be mitigated. Second, we show how the successful method of Collaboration Engineering from face-to-face environments translates to virtual teams, in order to address important concepts that are particularly difficult in distributed work. Finally, the results of the initial study reveal both technical and environmental issues and how they might be addressed for more successful design and implementation of virtual environments and team processes.

2 Conceptual Background

Powell et al. [1] define virtual teams as “groups of geographically, organizationally and/or time dispersed workers brought together by information and telecommunication technologies to accomplish one or more organizational tasks” (p. 7). While virtual teams offer many benefits to organizations, such as flexibility, adaptability, and responsiveness, they also face many challenges in coordination and communication [1], [8]. Among other issues, virtual teams experience difficulty in establishing relational links and shared understanding. To overcome these challenges, we argue that virtual teams can rely on leadership and purposeful team processes, supported by appropriate technology.

2.1 Challenges in Virtual Teams

Virtual teams have a high degree of reliance on information and communication technology (ICT) [9]. The lack of face-to-face interaction may increase efforts to foster interaction, inclusion and participation [10], as well as to integrate channels for sharing social information that could lead to development of strong relational links that can take longer to develop in a virtual team [11]. McGrath’s Time-Interaction-Performance (TIP) theory [10] shows that the development of relational links in

groups, or teams, involves performing activities related to member support and group well-being functions. As groups interact in one of McGrath's four modes of inception, solution, resolution of conflict, and execution of performance, they can perform one or more of three functions: production, member-support, and group well-being. A team with no past history, as is typical in much of the virtual team research, that is working on a challenging problem surrounded by technological and environmental uncertainty will have to engage in all three functions and in all four modes to avoid detrimental effects on performance [12].

Media synchronicity theory [13] builds on media richness theory [14] to argue that "the key to effective use of media is to match media capabilities to the fundamental communication processes required to perform the task" (p. 9) [13]. For this reason, it is unlikely that a single medium will excel in performing all tasks that teams face. Having a set of alternative media that can be switched for different tasks to be performed would be more appropriate for teams. Thus, a technology for supporting teamwork should offer a set of alternative media that can be utilized by team members in performing their tasks.

Virtual teams go through several development phases, i.e., initiation, exploration, collaboration, culmination, and dissolution [15]. Collaboration among members does not typically occur at initial contact. Instead, efforts are needed to move virtual teams from initial or exploration phases to a collaboration phase, and some teams might simply fail to reach the collaboration phase. This developmental view of virtual teams creates a challenge for researchers who are interested in collaboration among virtual team members because it requires an investment of time and effort in observing virtual team activities. A short-lived virtual team, such as is often used in laboratory experiments, could lead to misleading conclusions regarding collaboration activities in virtual teams.

2.2 Purposive Processes Through Collaboration Engineering

Collaboration Engineering is a field that directly addresses the process challenges of collaborative work in a systematic way. The cornerstone of Collaboration Engineering is the design and development of a set of process objects called thinkLets, which taken together can be considered a Collaboration Engineering pattern language [16]. A thinkLet is "a named, packaged facilitation technique captured as a pattern that collaboration engineers can incorporate into process designs" (p. 1) [16]. Each thinkLet addresses a particular pattern of collaboration, that is, a generic activity that teams need to undertake in order to accomplish collaborative tasks. The instantiation of these patterns in virtual teams enables purposive process structures that can help teams execute collaboration processes and achieve predictable interactions among team members, and therefore assure better team performance.

Process structuring in virtual team collaboration often takes shape as *temporal coordination*. It has been argued that temporal coordination is an important success factor for virtual team collaboration. Temporal coordination involves the synchronization of team members' informational, decisional, and interpersonal behaviors [7]. It may also involve scheduling and the allocation of resources. In particular, temporal coordination can address the mechanisms through which a team communicates, as well as the sequencing of problem solving activities to perform the

team's task. Temporal coordination has been shown to enhance team performance [7], and it is interesting because it offers teams (and their leaders) the freedom to choose or sculpt their own communication processes while still prescribing an overall problem solving strategy. However, team performance also depends on the way in which communication processes are carried out.

Our research is a natural continuation of this line of investigation. We do not offer prescriptions on the level of an overall problem solving process; rather, we offer chunks of process support (thinkLets) that specify interaction patterns and leader instructions. In other words, we do not offer the 'what' but the 'how' and let the teams (or their leaders) decide for themselves about the 'what.'

2.3 Shared Understanding and Leadership

We have argued that development of shared understanding is essential and, furthermore, that coordination and communication are essential for the development of shared understanding. Two aspects that we argue could address the issues of coordination and communication among team members are team process and leadership. By having structured process and leadership, team members should be able to develop shared understanding through structured deliberations, discussions, information exchange, and guidance by a team leader. The critical role of leadership in a team has been recognized by most models of group and team effectiveness [17]. However, leadership is both especially difficult and important in virtual teams for several reasons. Leaders usually exercise influence face-to-face, and that capacity is lost in virtual environments [18]. Since team members are dispersed and disparate, leadership is especially important for bringing team members together. Furthermore, in the research on virtual teams, observing leadership roles and the use of different tools that supports leadership is difficult to do.

To successfully empower distributed teams to work together, group members must be able to establish shared understanding. Shared understanding is a multi-dimensional construct that refers to task, team well-being, and member support – the three dimensions of the TIP theory referred to earlier [10]. In general, shared understanding means convergence on a common set of reactions to stimuli. From a task perspective, this means that a team will have a common view of what needs to be done to achieve their goal. From a team well-being perspective, it means that a team will have a common set of norms, expectations, and values. From a member support perspective, it means that team members will see how they individually fit within the larger collective in terms of their roles, knowledge, skills, and abilities.

Shared understanding is an evolving state. To reach shared understanding, teams must rely on team development and performance management, which is enabled through effective leadership [17]. Some empirical work has been done on leadership in virtual teams, e.g., both participative and directive leadership styles have been shown to be positively related to performance [19]. The phenomenon of emergent leaders in virtual teams has also been examined [20], with emergent leaders using more task-oriented messages than non-leaders in their virtual teams. Moreover, emergent leaders were found not to support the socio-emotional side of group development more than other team members. We can speculate that socio-emotional

development is particularly difficult to do in virtual environments, yet it should be no less important.

Apart from the empirical research, a number of practical and conceptual papers argue for the importance of studying leadership within virtual teams. An expert review panel recommended specific behaviors and tasks for leaders in different phases of virtual projects [21]. Two essential tasks were the need to develop shared mental models within the team and the need to define roles. A comprehensive review of leadership and facilitation in GSS environments provides a number of future research directions, including studying interventions that promote the development of the human-machine system that a virtual team represents [18]. This work stresses the importance of the interaction of leadership characteristics or behaviors with technology and process.

2.4 Summary of Key Concepts

We have argued that virtual teams face unique challenges, one of which is achieving shared understanding. Effective use of process is one way to achieve shared understanding. Leadership in the team is also important and indeed might have a synergistic effect with process structure. The use of thinkLets that have been developed in Collaboration Engineering provides a unique opportunity to put all these concepts together. The next section describes an integrated technology and process environment that we designed and implemented in order to support research on these concepts.

3 Technology and Process Environment

The design of the technology environment followed from the requirements to support team process and the overall goals of the research, namely: (1) implement all six patterns of collaboration (diverge, clarify, reduce, organize, evaluate, and build consensus) [16]; (2) provide easy access for participants from anywhere; (3) be easy to use; (4) incur relatively low, or no cost; and (5) provide a valuable experience for student participants in particular, who could benefit from learning a new technology environment. Since the fourth criterion implied use or adaptation of existing software, we began by evaluating existing commercial technologies. Groove™ was chosen because it best met all five evaluation criteria and had not been previously used in this type of study, thus providing the opportunity to create a new environment with adaptability for future work.

We designed and implemented a workspace in Groove™ (www.groove.net) that allowed team members to communicate issues, make decisions, and create their deliverables. The Groove environment supports a set of alternative media that can be utilized by team members in performing their tasks, in accordance with Media Synchronicity theory [13]. Groove can support synchronous communication through chat or asynchronous communication through discussion board and instant messages.

We implemented the following six project objects, i.e., thinkLets, which cover the six patterns of collaboration that were referenced earlier (pattern name is in parentheses):

- LeafHopper: gather ideas on a number of topics simultaneously (Diverge)
- FocusBuilder: arrive at clearer descriptions of key ideas (Clarify)
- BroomWagon: select what are the key contributions from a larger set (Reduce)
- PopcornSort: organize a set of ideas into a set of categories (Organize)
- StrawPoll (3pt): take a vote on a set of proposals or options (Evaluate)
- CrowBar: explore reasons for differences of opinion (Build Consensus)

The thinkLets were chosen to be easy to execute and useful for a broad set of tasks, but particularly the task in the initial study. Each thinkLet was implemented as a separate tool in Groove, using Groove's Outliner tool. Participants also had access to Chat and Instant Message. We provided three types of thinkLet guidance. First, the task materials included a high-level description for the types of activities that could be supported by each thinkLet. Second, each thinkLet included a template with sample information that illustrated the results of using the tool. Finally, a separate tool in Groove was populated with more elaborate instructions for each thinkLet, which included guidance selection and a detailed step-by-step script that the team or leader had to follow.

To accelerate the development of relational links, i.e., closeness or intimacy among group members, participants were provided an opportunity to introduce themselves in a forum called "Meet and Greet." Through initiating interaction in this forum, team members could "break the ice" and get to know their teammates better. This forum would help them in performing not only the production function by focusing on the given task, but also the two other functions from McGrath's TIP theory, i.e., member-support function and group well-being function. A virtual helpdesk was also set up and staffed by the research team. Participants could send email to a single address and have their questions answered without delay.

4 Initial Study in the Distributed Virtual Environment

We conducted an initial case study of Collaboration Engineering in our distributed virtual environment to understand the best way to introduce Collaboration Engineering into virtual team work and what the barriers might be. Studies of Collaboration Engineering in virtual environments are relatively new. We found only one study [22] that tried to use collaboration engineering in a distributed setting, focused on designing and executing a process that could shorten the steps in a crisis situation. Our initial study was designed as a laboratory experiment with two treatment conditions related to leadership structure, i.e., assigned leadership and shared leadership. In addition, we were interested in how each team would utilize the collaboration tools provided to them for accomplishing their task.

Our initial study involved fourteen five-member virtual teams working on a predefined task related to emergency response. Each team consisted of students from three different universities acting in pre-defined roles: government official, utility infrastructure superintendent, police officer, aid organization representative, and information system developer. Students were randomly assigned to teams in two different treatment conditions. We expected different leadership styles to lead to different approaches in how teams utilize available collaboration tools in performing

their task. Those different approaches could also lead to different satisfaction levels and shared understanding among team members. Furthermore, as teams have a set of alternative media to choose from in performing their task, we expected each team to have different preferences in what media they were going to use. The choice of media for communication could be significant given the fact that teams were dealing with an urgent task that required rapid response. For example, a study about communication among clinicians in an emergency department found that synchronous channels of communication were used more frequently than asynchronous channels [23]. In our study, we expected that teams would utilize synchronous communication media such as chatting more than asynchronous media such as emails or a discussion board. To measure participants' perception of shared understanding and satisfaction, a questionnaire was administered at the end of the study.¹

4.1 Task

Teams were asked to come up with specifications that could be used to develop a website to assist people working in a disaster relief situation. The fictional scenario of the disaster was adapted from research on formation of creative solutions using electronic brainstorming [24]. Teams needed to deliver a specification that could be used by an information systems developer as guidance in developing this website. To introduce realistic constraints and the need for discussion and consensus, the teams were limited to a small number of features that could be incorporated into the website. Teams had one week to perform the task and turn in their deliverable.

4.2 Participants and Teams

Seventy students from three different universities in the U.S. agreed to participate in this study for an exchange of extra credit in the course in which the study was conducted, and 34 actually completed the study. Each team consisted of five people with each one playing a different role that was relevant to emergency relief. Collaboration was necessary because each role represented a different constituency or agency with different preferences for what should be included in the deliverable. Since the deliverable could have only a limited number of items and those items had to be prioritized, team members would have to collaborate to find the best way to integrate their incongruent goals, while still pleasing their constituencies or agencies. Upon joining their team, members were instructed to introduce themselves by posting information on the "Meet and Greet" forum.

As noted earlier, dispersed teams benefit from having a leader, who can help to build trust [25], foster role clarity, create clear structure, and enhance communication effectiveness [26]. Our study, therefore, tested the effectiveness of two different leadership structures. There are many ways to classify leadership, e.g., in terms of styles such as transformational vs. transactional [27] or technical, charismatic, caring-personal, and peer oriented [28]. In this study, we focused on what we call leadership structures rather than style, and examined the two structures of assigned versus shared leadership. In a virtual environment, observation of assigned vs. shared leadership

¹ Detailed descriptions of the thinkLets and the post-session questionnaire are available by request from the first author.

should be easier than observing style characteristics. But an even more important reason for this choice was that we believe it represents a base-line starting point. For this study the participant playing the role of government official was chosen as the leader of the team in the assigned leadership condition. To eliminate potential confounding from gender differences, the government official role in each team was played by a male participant.

4.3 Technology and Tools

The Groove technology that was chosen as the platform for this study is a peer-to-peer (P2P) based system that has several advantages compared to a client-server system. While a client-server system would store data in a central location, in a peer-to-peer system every peer or node acts as both client and server and provides part of the overall information available from the system [29]. A synchronization process becomes an important part of P2P technology to keep information for every team member up-to-date. We encountered two main problems related to the synchronization process: (1) the Groove Instant Message function was not synchronized correctly during this process, so that several members of the research team could not receive messages for certain periods of time; and (2) the synchronization process took a long time to finish.

Once the platform was chosen, collaboration tools or process objects were integrated into Groove, since no such tools exist in this technology. Although process objects have proved to be helpful in guiding face-to-face teams [30], their deployment in virtual environments was still rare. The newness of process objects to most of the participants could create confusion about the tools that in turn could cause them to skip the use of the tools. For these reasons, we provided teams with only a limited number of process objects. Six process objects, each supporting one pattern of collaboration, were integrated into Groove. The patterns of collaboration are: (1) Diverge - *Move from having fewer concepts to having more concepts*; (2) Clarify - *Move from less to more shared meaning for the concept(s) under consideration*; (3) Reduce - *Move from having many concepts to a focus on fewer concepts deemed worthy of further attention*; (4) Organize - *Move from less to more understanding of the relationships among concepts*; (5) Evaluate - *Move from less to more understanding of the utility or priority of concepts toward goal attainment*; (6) Build consensus - *Move from having more disagreement to having less disagreement on courses of action* [31]. Each of the patterns was represented by a process object with a unique name. To educate participants in how to use the process objects, a brief description of each of the process objects was provided. The descriptions included the purpose of the specific object in the form of "if ... then ..." sentences. A brief example of how to use each object was also provided.

4.4 Data Collected

Shared understanding and satisfaction were recorded using a post-session questionnaire that asked participants about their demographic information, perceived shared understanding, perceived satisfaction, performance of their teammates, and feedback on how to improve this study. Additionally, a transcript of each team's

communication via Chat on the Groove workspace was also recorded, for later use in analyzing how the leadership role was performed in each of the teams. Workspaces of each team were saved for analyzing how the team utilized the process objects provided. Message exchanges between team members and researchers were also recorded to analyze concerns raised by participants throughout the study. Team deliverables were saved for analyzing team performance in carrying out the task. Table 1 shows team descriptions by condition.

Table 1. Team descriptions by leadership condition

| | Assigned Leadership | Shared Leadership |
|-----------------------------------|----------------------------|--------------------------|
| # teams with deliverables | 5 | 3 |
| # teams using chat to communicate | 3 | 1 |
| # teams utilizing process objects | 4 | 2 |
| # teams utilizing meet and greet | 6 | 7 |

Even though six of the teams tried to utilize process objects as collaboration tools, we could argue whether they used the tools correctly or not. Only one process object was used by every team that used process objects, and that was LeafHopper, which supports the diverge collaboration pattern. Further investigation is still needed to determine the specific value of process objects for virtual teams. At this stage of the research, it is also unclear whether the leader role in each team was played correctly by either the appointed leader or the team members in the case of shared leadership. The fact that the majority of registered participants did not join their team at all raised questions about the challenges of performing distributed virtual team research. Therefore, next we discuss and identify some of the barriers in performing this type of research based on participants' feedback and the research team's own self-reflections during administration of the study.

5 Challenges in Distributed Collaboration Research

Based on qualitative and quantitative analysis of collected data and exchanged messages between researchers and participants, we saw several reasons why available tools were not used and why the participation rate in this study was below 50%. We categorize those reasons into technical and environmental challenges. Technical issues relate to a technological problem or problem in using the technology, while environmental issues relate to participants' situation at the time of the study.

5.1 Participant Concerns

Qualitative analysis of participant comments and messages revealed some causes of low participation. An analysis of exchanged messages shows that those messages can be categorized into three categories: (1) task; (2) team; and (3) technical. The task issues refer to participants' messages discussing the task, including but not limited to

deadlines and meeting appointments, while the technical issues refer to messages about problems in using the technology, e.g., inability to see other team members. The team issues refer to messages complaining about team work or other team members, e.g., no other member joined the team. Table 2 shows a content analysis of exchanged messages.

Table 2. Categorization of messages based on their content

| Issue addressed | Message received | Percent of total |
|-----------------|------------------|------------------|
| Task | 19 | 40.4% |
| Team | 15 | 31.9% |
| Technical | 13 | 27.6% |

A high percentage of task-related messages is a good sign, since it can indicate that teams were working on the task as required. However, a high percentage of team-related and technical-related issues could indicate that teams might have had problems in performing their task. A detailed look at these messages revealed that most of the time team members were concerned with their teammates not joining the team. In such situations, members would ask for guidance from the researchers in how to proceed with incomplete teams. Although a complete team would be an ideal condition for the researchers, team members also needed to know how to proceed in such case. We observed that not knowing how to proceed frustrated members who had already joined the team. This frustration could be one of the reasons why participants only showed a mediocre satisfaction level at the end of the study. Table 3 shows the means by treatment condition of satisfaction with the process and satisfaction with the outcome, where a score of 1 means less satisfied and a score of 7 means more satisfied. Participants seem to prefer clear instructions on how to proceed, which is associated with the assigned leadership condition.

Table 3. Means of post-session satisfaction scales by leadership condition

| Construct | Assigned leadership | Shared leadership |
|---------------------------|---------------------|-------------------|
| Satisfaction with Process | 3.08 | 2.60 |
| Satisfaction with Outcome | 3.43 | 2.40 |

5.2 Technical Barriers

The teams in our study had a high degree of reliance on technology, since members were geographically dispersed. Technical issues were identified by a number of participants as a source of problems in collaboration, arising from a technology malfunction or from participants' inability to use or navigate the technology itself. The main problem was the difficulty that participants experienced in setting up the workspace itself. Even after being provided with instructions on how to join and

having email exchanges with the help desk, several participants simply were not able to download and join their workspaces.

It is difficult to get the exact numbers of people who chose not to participate due to technical issues, since only those who contacted the researchers and complained about their problems could be identified as having a technical problem. Furthermore, it was also difficult to verify the complaints of those who did contact the researchers. However, we did provide them with a hotline where they could report a problem and ask for advice from the research team. We set up a virtual helpdesk that was staffed between 7:00 am – 11:00 pm throughout the study period to respond immediately to participants' concerns. All concerns or questions were recorded into a helpdesk log that could be used as sources for answering similar questions from other participants. This concept of a virtual helpdesk proved to be useful in conducting this type of research.

5.3 Environmental Barriers

Working in distributed environments has become a challenge, especially for those who have not had experience working in such settings. Some participants would have preferred a face-to-face meeting instead, supporting Sarker and Sahay's [15] assertion that initial contact in a virtual team will not automatically result in collaboration. Some of the teams simply failed to reach the collaboration phase. The probability to fail for a virtual team seems to be higher if the leadership role is not filled. Especially in the case of assigned leadership, a team leader should guide team members in developing processes leading to expected outcomes. A team leader should act as a collaboration engineer who designs processes for the team and chooses appropriate thinkLets to be used. In the case of shared leadership, each member should take responsibility to move their team forward with collaboration. Some team members should take the lead in designing collaboration process for their team and they should make themselves familiar with available thinkLets. Distributed environments, however, have increased the challenges for doing so. The challenge of jump starting a distributed collaboration should be taken into consideration when designing tools for virtual teams. Although it has been shown that a designed process can be executed successfully in a distributed setting [22], our study indicated that designing the process itself is a challenge for a virtual team.

Our approach of not providing training to participants about designing collaboration and utilizing collaborative tools sheds light on the level of awareness among participants about collaboration and process objects. The three types of thinkLet guidance provided to participants were insufficient in helping them understand the purpose and the usefulness of the tools. The following message reflected a participant's confusion on how to use existing tools:

“How do we actually use the tools such as Leaf Hopper? Do we need to start a new workspace or do we use the Project Relief Team 6 workspace? I don't understand how to open a new tool, or if I need to use the example shown in the Project Relief Team 6.”

Clearly, there is need for some level of training in introducing collaboration and process objects to virtual teams. One aspect of Collaboration Engineering principles is

that thinkLets should be easy to use by anyone, without the intervention of a professional facilitator. Although this may be true for the process that is described within the thinkLet, there still is a challenge in seamlessly presenting the process objects through technology.

Another barrier that can easily be overlooked by researchers is the timing of conducting a virtual team study. Several participants commented on the difficulty of completing the virtual project at the end of the semester when other workload was heavy. The difficulty with the timing also led to several participants dropping out, even though they expressed interest in the project itself. Timing plays a significant role in the participation rate of distributed virtual collaboration teams. Although the research team had estimated that on average participants would only need to spend three to four hours on the project, participants still perceived the project as taking too much of their precious time. This non-participation issue can be explained by the rational choice model [32], which states that an economically rational actor tries to maximize benefits from any activity, while minimizing costs. Therefore, if participants perceived that the cost for participating in this project was higher than the perceived benefits, then they would be more likely to drop out of the project. To overcome this problem, researchers need to think about how to increase the perceived benefits of this type of project, while at the same time try to lower the perceived costs of involvement. The following table shows several ways to increase perceived benefits and to lower perceived costs of participations in distributed virtual team collaboration. These guidelines are stated in the context of a research project in a student setting, but they translate reasonably well to field settings.

Table 4. Ways to increase perceived benefits and lower perceived costs

| Increasing perceived benefits | Lowering perceived costs |
|--|---|
| Make the project a required part of the course so that it is integrated with the regular activities of the course. | Time the project in balance with other activities occurring in the course. Manage expectations. |
| Emphasize potential achievement for participants. | Provide time for participants to learn about the technology but also about the interplay of the technology with the process. |
| Offer a reward for excellent work, within the guidelines and requirements of Institutional Review Boards. | Provide clear task instructions but also clear process instructions, in terms of how to proceed under different circumstances, e.g., lack of participation by some members. |

Another problem was that team members joined their team in the last stage of the study. Since there was no penalty or cost for coming late to the teams, some of the participants maximized their benefits by participating only in the late stage of the project. This situation created a problem for researchers to make sense of shared understanding data collected at the end of the study, since there was no meaningful development of shared understanding in such teams. To address this issue, a strict rule is needed such as awarding credit only to those who fully participate from the beginning to the end of the project. Additionally, asking teams for temporary

deliverables in the middle of the project would help in boosting team works and preventing last minutes only participations.

6 Conclusion

We have presented a discussion of the technical and environmental challenges of conducting research on Collaboration Engineering in distributed teams. Prior research on Collaboration Engineering has been done mostly using GroupSystems technology [33], [34], [35]. Our study is new in that we used Groove as a platform for collaboration. We have shown that process objects or thinkLets can be implemented in this technology, which suggests its adaptability and usefulness for collaboration. While research in distributed virtual collaboration is important, our initial study indicated that it is also challenging. High commitment from participants is required and unpredictable technological problems can lead to non-participation. We saw a high degree of probability that a team will not reach the collaboration phase at all and there can be a dependency on the team leader or members' initiative to design the collaboration process. Furthermore, the newness of collaboration tools or process objects can hamper participants from using those tools correctly. As our study indicated, some level of training for team leaders and/or members might be needed to make them familiar with collaboration tools in terms of process objects. At the same time, appropriate translation of process objects for their use in virtual environments is still needed in order to ensure that each process object is applied correctly and brings the intended results. Our effort to provide guidance for thinkLet use failed to increase awareness among most of the participants regarding the usefulness of thinkLets. Therefore, future studies should help in addressing the issue of effective guidance in virtual environments.

The limitations of the current study include a small numbers of participants and the short period of the study. However, we expect through this initial study to be able to record and disseminate best practices in how to conduct this type of study. Based on the current study, we offer the following recommendations for future research in Collaboration Engineering for distributed teams:

1. Provide some level of training for designing collaboration processes and using process objects for participants in virtual environment.
2. Provide enough time for the team to move from initial contact to collaboration, since teams need to go through several development phases.
3. Ask for interim deliverables in order to study how shared understanding is emerging.
4. Find the best way to integrate collaboration technology with process objects, in order to lower the perceived barriers among team members to use those tools.
5. Know the technology platform and its potential problems, so that technical assistance can be provided to participants immediately.
6. Provide warm-up time for participants to learn more about the technology used, existing tools, and their teammates, before starting with data collection.
7. Schedule the project with an eye to balancing other commitments, while managing expectations for engagement with the virtual experience.

8. Set a strategy to increase perceived benefits of involvement in the project among participants, while at the same time lowering perceived costs of involvement.
9. Be clear on rewards for participation in the study.

Several of the recommendations reinforce what we know from the domain of traditional teams and projects. The key issue in this study has been the question of how well the principles and practices of Collaboration Engineering translate to distributed environments, including the ease with which we can develop and implement a flexible infrastructure for the use of a broad range of process objects. This study shows that it is possible to use a peer-to-peer application as the foundation for creating a collaboration environment that can then be used in different ways by different teams. However, the study has also reinforced that distributed teams need more work than we expected in order to get up to speed with being able to use and adapt processes for themselves. This initial test of Collaboration Engineering, in this one specific environment, shows that there is interesting work yet to be done on the best ways to integrate collaboration tools with processes and leadership, in order to help virtual teams perform rapidly and effectively across a wide variety of tasks.

References

1. Powell, A., Piccoli, G., & Ives, B: Virtual teams: A review of current literature and directions for future research, *The DATA BASE for Advances in Information Systems*, 35(1) (2004) 6-36
2. Pinsonneault, A. and Caya, O.: Virtual teams: What we know, what we don't know. *International Journal of e-Collaboration*, 1(3) (2005) 1-16
3. Khazanchi, D. & Ziguers, I.: Patterns for effective management of virtual projects: Theory and evidence, *International Journal of e-Collaboration*, 2(3) (July-Sept 2006) 25-48
4. Munkvold, B. E. & Ziguers, I.: Integration of e-collaboration technologies: Research opportunities and challenges, *International Journal of e-Collaboration*, 1(2), (April-June 2005) 1-24
5. Vreede, G.J. de and Briggs, R.O.: Collaboration engineering: Designing Repeatable processes for high-value collaborative tasks, *Proceedings of the 38th Hawaiian International Conference on System Sciences*, Los Alamitos: IEEE Computer Society Press. (2005)
6. Dillenbourg, P., Baker, M., Blaye, A. & O'Malley, C.: The evolution of research on collaborative learning. In: Spada, E. & Reiman, P. (eds): *Learning in Humans and Machine: Towards an interdisciplinary learning science* (1996) 189-211
7. Massey, A.P., Montoya-Weiss, M.M. and Hung, Y-T.: Because time matters: Temporal coordination in global virtual project teams, *Journal of Management Information Systems*, 19(4) (2003) 129-155
8. Qureshi, S. and Ziguers, I.: Paradoxes and prerogatives in global virtual collaboration, *Communication of the ACM SPECIAL ISSUE: Global Applications of Collaborative Technology*, 44(12) (2001) 85-88
9. Dubé, L., and Paré, G.: The multi-faceted nature of virtual teams. In: Pauleen, D.J. (ed): *Virtual Teams: Projects, Protocols, and Processes*. Idea Group Publishing (2004)
10. McGrath, J.: Time interaction and performance (TIP): A theory of groups, *Small Group Research*, 22 (1991) 147-174

11. Chidambaram, L.: Relational Development in Computer-supported Groups. *MIS Quarterly*, 20(2), (1996) 143-165
12. Warkentin, M. and Beranek, P. M.: Training to improve virtual team communication. *Information Systems Journal*, 9(4) (1999) 271-289
13. Dennis, A. R. and Valacich, J. S.: Rethinking Media Richness: Towards a theory of Media Synchronicity. In proceedings of the 32nd Hawaii International Conference on System Sciences, (1999) 1-10
14. Daft, R. L. and Lengel, R. H.: Organizational information requirements, media richness and structural design. *Management Science*, 32(5), (1986) 554-571
15. Sarker, S. and Sahay, S.: Understanding virtual team development: an interpretive study. *Journal of the Association for Information Systems*, 3 (2002) 247-285
16. Vreede, G.J. de, Kolfshoten, G.L. and Briggs, R.O.: ThinkLets: A Collaboration engineering pattern language, *International Journal of Computer Applications and Technology*, 25(2/3) (2006) 140-154
17. Bell, B.S. and Kozlowski, S.W.J.: A typology of virtual teams: Implications for effective leadership, *Group & Organization Management*, 27(1) (2002) 14-49
18. Avolio, B.J., Kahai, S. and Dodge, G.E.: E-leadership: Implications for theory, research, and practice, *Leadership Quarterly*, 11(4) (2001) 615-668
19. Kahai, S.S., Sosik, J.J. and Avolio, B.J.: Effects of participative and directive leadership in electronic groups, *Group & Organization Management*, 29(1) (2004) 67-105
20. Yoo, Y. and Alavi, M.: Emergent leadership in virtual teams: what do emergent leaders do? *Information and Organization*, 14(1) (2004) 27-58
21. Beranek, P.M., Broder, J., Reinig, B.A., Romano, Jr., N.C. and Sump, S.: Management of virtual project teams: Guidelines for team leaders, *Communications of the Association for Information Systems*, 16 (2005) 247-259
22. Appelman, J. H. and van Driel, J.: Crisis-response in the Port of Rotterdam: can we do without a facilitator in distributed settings? *Proceedings of 38th Hawaii International Conference on System Sciences* (2005) 1-9.
23. Spencer, R., Loga, P. and Coiera, E.: Supporting communication in the emergency department. Center for Health Informatics, University of New South Wales, Australia. (2002) Available online: <http://www.ehealthnurses.org.uk/pdf/A&EAustralia.pdf>
24. Santanen, E.L.: Directed Brainstorming and the Cognitive Network Model of Creativity: An Empirical Investigation of Cognitive Factors Related to the Formation of Creative Solutions Using an Electronic Brainstorming Environment, Unpublished doctoral dissertation, University of Arizona, Tucson, AZ (2001)
25. Jarvenpaa, S., Knoll, K. and Leidner, D.: Is Anybody Out There? Antecedents of Trust in Global Virtual Teams. *Journal of Management Information Systems*, 14(4) (1998) 29-64
26. Kayworth, T.R. and Leidner, D.E.: Leadership Effectiveness in Global Virtual Teams. *Journal of Management Information Systems*, 18(3), (2001-2002) 7-41
27. Burns, J. M.: *Leadership*. New York: Harper & Row. (1978)
28. Oliverson, L. R.: Identification of dimensions of leadership and leader behavior and cohesion in encounter groups. *Dissertation Abstract International*, 37 (1976) 136-137
29. Aberer, K., Puceva, M., Hauswirth, M. and Schmidt, R.: Improving data access in P2P systems. *Internet Computing* 6(1) (2002) 58-67
30. Vreede, G.J. de, Fruhling, A., Chakrapani, A.: A Repeatable Collaboration Process for Usability Testing. *Proceedings of the 38th HICSS, IEEE Computer Society Press.* (2005)
31. Hengst, M.d., Dean, D. L., Kolfshoten, G. and Chakrapani, A.: Assessing the Quality of Collaborative Processes. *Proceedings of the 39th Annual Hawaii International Conference on System Sciences* (2006)

32. Downs, A.: *An Economic Theory of Democracy*. New York: HarperCollins (1957)
33. Briggs, R. O., Vreede, G.-J. de and Nunamaker, J. F.: Collaboration Engineering with ThinkLets to Pursue Sustained Success with Group Support Systems. *Journal of Management Information Systems*, 19(4) (2003) 31-64.
34. Harder, R. J., Keeter, J. M., Woodcock, B. W., Ferguson, J. W. and Wills, F. W.: Insights in Implementing Collaboration Engineering. *Proceedings of the 38th Annual Hawaii International Conference on System Sciences* (2005) 1-10.
35. Hengst, M. d., van de Kar, E. and Appelman, J.: Designing mobile information services: user requirements elicitation with GSS design and application of a repeatable process. *Proceedings of the 37th Annual Hawaii International Conference on System Sciences* (2004) 1-10.

Monitoring and Analyzing Group Interactions in Asynchronous Discussions with the DIAS System

Tharrenos Bratitsis and Angelique Dimitracopoulou

LTEE Laboratory, University of the Aegean, 1, Democratias Str,
85100 Rhodes, Greece
bratitsis@aegean.gr, adimitr@rhodes.aegean.gr

Abstract. DIAS is an Asynchronous Discussion Forum Software, mainly developed in order to offer extended monitoring and interaction analysis support, by providing a wide range of indicators jointly used in various situations, to all discussion forae users (individual user/students, groups, moderators/teachers or even researchers/observers), appropriate for their various roles in different activities. In this paper we describe some of the integrated Interaction Analysis (IA) features and we provide information concerning case studies, some of which are in progress.

Keywords: Interaction Analysis, Asynchronous discussions, CSCL, CSCW.

1 Introduction

Computer Mediated Communication (CMC) tools and in particular asynchronous discussion forae are widely used in formal or informal educational contexts, applying principles of constructivism, emphasizing in social interaction during learning activities [3],[7]. The latest quinquennium research is focusing towards finding methods for the evolvment and support of critical thinking through interactions, taking place within asynchronous discussions, in order to achieve high quality learning [13]. Such a goal requires tools, frameworks and methods for the facilitation of monitoring, and/or self-reflection and therefore selfregulation that could be supported by the automated analysis of the complex interactions that occur. Computer based Interaction Analysis (IA) is an emerging field of research within the academic community, focusing in analyzing interactions among users, borrowing elements from the CSCW, CSCL and Artificial Intelligence research fields.

2 Interaction Analysis

Computer based IA provides mainly information directly to technology based activities' participants, in order to self assess their activity [5]. The IA results are presented to the participants in an appropriate format (graphical, numerical, literal), interpretable by them. The corresponding information provide an insight of their own current or previous activity allowing them to reflect on a cognitive or metacognitive level, and thus act in order to self-regulate their activities. Computer based IA

provides also information to the activity observers, in order to analyse the complex cognitive and social phenomena that may occur.

This approach can produce flexible IA tools, which in an educational context, support directly the learning activities' participants (e.g. students, teachers, moderators) or even the observers (e.g. teachers, administrators, researchers) of these activities. The need for such tools derives from the complexity of interactions occurring within computer based learning environments (as described in many CSCL approaches). It would be legitimate to say that the IA research field has partially emerged from the application of methods, frameworks and techniques developed originally within the CSCW field and especially awareness (workspace awareness in particular) information provision, in combination with corresponding elements from the AIED (Artificial Intelligence in Education) field.

Regardless of the origin, the IA research field aims at providing methods and tools that support the participants of learning activities in three major levels: awareness, metacognition and evaluation level [8]. The expected outcome is the optimization of the activity through: a) better activity design, regulation, coordination and evaluation by the forum moderator, and b) refined participation and learning outcome for the students through reflection, self-assessment and self-regulation.

The IA process consists in recording, filtering and processing data regarding system usage and user activity variables, in order to produce the analysis indicators. These indicators may concern: a) the mode or the process or the 'quality' of the considered 'cognitive system' learning activity; b) the features or the quality of the interaction product; or c) the mode, the process or the quality of the collaboration, when acting in the frame of a social context forming via the technology based learning environment [5].

Our main concern in this paper is IA tools concerning asynchronous discussions.

3 Related Work

While examining Forum and Forum type software, we find that commercial or open source products, such as WebCT, WebWiz and PhpBB provide minimum analysis information. Most of them present simple usage indicators, such as activity information (number of messages posted and read), a few statistical indicators (most and least busy day, etc), online users, number of messages per day, number of unread messages, etc. We consider this minimal information, which supports forum usage only as a subsidiary tool of a Learning System [2].

Several new and promising approaches that implement graphical representations of asynchronous discussions' features and parameters can be found while reviewing recent literature. For example, the i-Bee system is a software that visualizes relationships between users and keywords in online messages, in real time. It also provides snapshots of past discussions and animations. Keywords appear as flowers and users as bees. The distance between flowers and bees, their status (e.g. flying/sleeping bee, blossomed/closed flower) and their orientation depend on discussion parameters, such as keyword usage frequency and recent user activity [10].

Another example of the use of powerful visualisations via metaphors is the i-Tree system that visualises the discussion status on mobile phones using a tree representation. The tree corresponds to a single user, whose activities designate the

tree's appearance. Thereby the tree's log and branches are relevant to the number of messages, the leaves' range and colour are relevant to message reading, the fruits are relevant to the answers the user has received and the appearance of the sky is designated by the whole discussion status [11].

Mailgroup is a Forum Type tool with integrated analysis tools emerging from the Social Network Analysis field. It implements an alternative method of representing the message sequence in an asynchronous discussion, taking into account both chronological and logical constituents [12].

Other approaches also exist, integrating Fuzzy Logic techniques in order to assess and evaluate the collaboration level in a discussion based on several parameters (Degree system) [1] or providing a variety of visualised statistical information (add-on for the AulaNet platform) in order to help the teacher coordinate discussions and obviate undesirable situations or progress of the discussion activity [6].

The aforementioned approaches constitute a representative specimen of asynchronous discussion software, used for learning purposes. All of them provide tools and functionalities for supporting and facilitating user activity in various levels. Nevertheless a closer examination leads us to the conclusion that they can only be used under specific usage settings. Some disadvantages are described in Table 1.

Table 1. Discussion Forum software characteristics

| Software | Functionalities | Disadvantages |
|----------------------|--|---|
| WebCT, phpBB, WebWiz | Simple statistical awareness information | No real IA indicators |
| i-Bee | Visualized representation of user – keyword relation | No empirical research about learning utilization of this feature |
| i-Tree | Visualized representation of user activity on mobile phones | Considers few activity characteristics. Seems to encourage message reading only |
| MailGroup | SNA indicators | Indicators are addressed only to the moderator. Adequate number of messages is required to produce meaningful results |
| Degree | Various collaboration quality indicators & advising mechanisms | Closed system, not easy to customize, with non-transparent indicator calculation. |
| AulaNet add-on | Visualized statistical information drawn from log files | Various diagrams, addressed only to the moderator |

4 The DIAS System

The DIAS system (Discussion Interaction Analysis System) has been developed by the LTEE laboratory of the University of the Aegean. It is a fully functional discussion forum platform, with an underlying database management system for data recording and several implemented functionalities in order to facilitate user participation as well as the moderators' alternative discussion strategy planning. Additionally about sixty five (65) visualized indicators (including all possible variations) are produced, varying from simple statistical awareness information to complex cognitive and metacognitive indicators. Different sets are addressed to the teacher or moderator and the students - users, along with the corresponding interpretation schema for various discussion strategies or usage scenarios.

Our main goal is to offer direct assistance to users, supporting them in the level of awareness of their actions, as well as their collaborators, in order to activate their metacognitive processes, thus allowing them to self-regulate their activities. In parallel, we aim in supporting the discussion moderators (eg teachers) in order to 'identify' problematic situations and difficulties that require regulative interventions. The design of the system is based on three core design principles [2]:

1. Take into account the totality of the users involved in a 'learning activity', as well as the cognitive systems they may form, students as individuals (in various roles), as members of one or more groups or even communities, teachers in different roles according the category of learning activity, etc.
2. Provide a rich range of IA indicators for the various user profiles and points of view of the activity process, its quality, as well as its product.
3. Create an independent, flexible, customizable and interoperable system. Forae are tools that can be used in a variety of contexts and activity categories. Furthermore forum participants take various roles and have different needs according to their discussion subjects, the available time etc. Thus, customization and flexibility are crucial characteristics.

This lead us to the selection of open source web based technology, making it easy to share with the academic community. More information about the system's architecture and functionality can be reviewed in [2].

5 DIAS Interaction Analysis Indicators

By combining some of the indicators produced by the DIAS system and applying the appropriate interpretation schema (guidelines for interpreting and combining information), interesting conclusions can be drawn. Let's examine a set of indicators addressed to the teacher, which may help him/her evaluate the quality of a student's participation (from now called User X). These indicators are: Classification Indicator, SNA Answers, SNA Reads, User - Tree Structure and several statistical Bar Charts.

User Classification Indicator (Fig 1a): It is a XY scattered chart with the X-Axis representing the amount of contribution (messages written as a percentage of the total number of messages) and the Y-Axis representing the amount of Interaction (messages read as a percentage of the available number of messages) by a user. Both Axes are scaled from Low to High. By inspecting this indicator, the moderator may see how active (writing and reading messages) User X is, in comparison with the other users and the mean values of activity (represented by the two Axes' position). The first conclusion is whether User X has extreme or balanced behaviour (Arrogant: writes many messages but doesn't read other users' messages. Passive: reads many messages, but doesn't write enough). The second conclusion is whether User X's performance is far ahead from the mean values in any of the two activity constituents.

SNA Answers Indicator (Fig 1c): The system can produce social matrices according to Ucinet DL format and Agna matrix format for further processing. For N users, the Answers social matrix is a NxN matrix. The number placed in the cell designated by line A and column B is equal to the number of messages written by user A as answers to messages of user B. By quickly inspecting the SNA diagram

deriving from the social matrix, the moderator can see whether User X is isolated or holds a central position within the discussion. Furthermore, if User X seems active in message writing (conclusion drawn from the Classification Indicator), this diagram can show if he/she is exchanging information with other users or not, by posting answers to them. Additionally the number of other users who have posted answers to User X can be detected, revealing interesting information. For example a very active user (Classification Indicator) may be isolated in this diagram, thus not contributing to the quality of the discussion and the overall collaboration (no one is posting answers to him/her). This could indicate low argumentative value of this user’s messages, off topic writing, arrogant behaviour or lack of knowledge regarding the topic. In any of these cases, the moderator may diagnose a problematic situation and act accordingly.

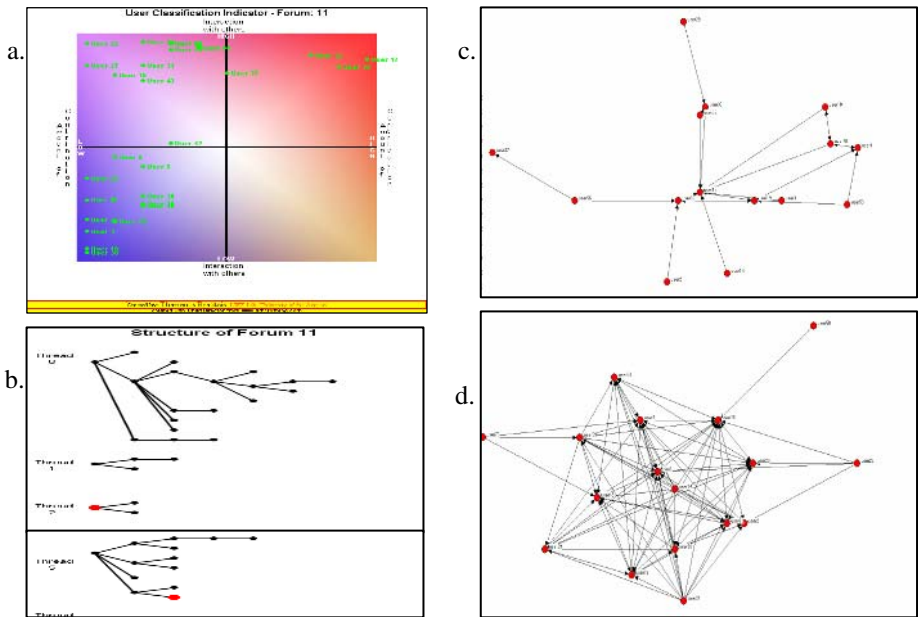


Fig. 1. Interaction Analysis Indicators by the DIAS system

SNA Reads Indicator (Fig 1d): This diagram is similar to the previous. In this social matrix, the numeric value in a cell designates the number of messages written by user B, that user A has read. This diagram indicates the amount of other students whose messages User X reads and consequently his/her involvement in the collaborative discussion activity. While the Classification Indicator shows the amount of messages read, this diagram additionally shows the dissemination of these messages to the according amount of authors. In combination with the Answers SNA diagrams, the moderator can see whether User X is participating in a closed user group, interacting heavily inter se and lightly with the rest of the users. This may designate undesired behaviour regarding the collaborative activity.

Furthermore, this diagram reveals the amount of users who have read messages posted by User X. If he/she holds a relatively central position within this diagram but

appears to be isolated or obscure in the Answers SNA diagram, then he/she writes messages which are read by many other users, but not answered to. Consequently User X could be a discussion coordinator or possibly face a participation problem that needs further attention by the discussion moderator.

User – Tree Structure Indicator (Fig 1b): This is a visualised representation of the discussion evolution in a threaded tree-like format. Messages appear as black dots, where the ones posted by User X are marked red. Line segments connecting two dots designate responsive relation (the message on the right is an answer to the one on the left). By quickly inspecting this diagram the moderator can see whether User X is active mostly in earlier or later phases of the discussion activity. In combination with the previous indicators, interesting conclusions may arise. For example an active user (Classification Indicator) who writes many but receives few answers (SNA Answers) and appears to write messages in later phases of the discussion may possibly have low performance in the activity. This could be the case of a user who simply agrees or disagrees with other users' arguments but doesn't contribute with new information and ideas, which may be confirmed by further inspecting his/her messages.

Bar Chart Indicators: Besides the aforementioned indicators, many simple awareness, statistical ones regarding User X can be produced in a Bar Chart format. These indicate for example the number of various types of messages (questions, answers, arguments, etc) per day. By further examining such information, the teacher may acquire a more concrete reflection of the quality of User X's activity.

This indicator set constitutes an example of indicator information utilization. Many combinations may be formed with various indicators, forming various interpretative schemas. Furthermore the information that can be extracted from a single indicator may have different meaning for different kind of user roles or interpretative schemas.

6 Case Studies

Our main goal is to assess the indicator's usage, while using asynchronous discussions within learning activities. In particular our aim is to:

- Assess the correctness and clarity of the produced indicators and the proposed interpretative schemas (or even construct new schemas).
- Detect the effect of the indicators in the users' self-regulation processes.
- Evaluate the indicators' contribution to the facilitation of the moderator's, coordinator's and observer's work. We intend to provide ways of evaluation, coordination and assessment, bypassing the need of thoroughly reading all the messages or using time-consuming methods, such as content analysis.
- Assess the potentiality of a qualitative evaluation of discussion without applying content analysis methods.
- Designate the appropriate set of indicators for each role and phase of a discussion learning activity, defining interpretative schemas.

Several case studies have been designed for that matter, one of which is complete.

Completed Case Study: In the first case study, forty (40) postgraduate students were involved in a non-restrained discussion activity, for six (6) weeks. Their first

contact with the system occurred via a three hour seminar. The discussion topics were relevant to the course syllabus and the assignments they had to prepare for the end of the semester. A total of 553 messages were posted, while trying to exchange ideas, information and arguments. Our intension was to study the effect of the indicators in the students' activity behaviour. The results revealed that the indicators increased the student's motivation for involvement in the activity (70% increase of messages). They showed increased interest in observing the indicators, especially the ones providing comparative information with the rest of the students (for example the Classification Indicator). They were curious to examine the impression and reception of their messages by other students through indicators containing information about reading of messages and posting of answers. Some additional results could be extracted, such as the fact that the students' criterion for the acceptance of their messages by others was initially the number of answers they received and gradually altered to the number of users reading their messages. Significant part of this alteration was due to the information presented in some of the more complex indicators.

In progress Case Studies: Three ongoing case studies involve students (postgraduate and undergraduate) divided into two equivalent groups. Only the first group may examine the indicators. They participate in restrained discussion activities for seven (7) weeks, where the teachers have more active roles in coordinating the discussion evolution, according to certain usage scenarios (different in each case study). Our intension is to compare the behaviour of the two groups. Additionally we want to examine the facilitation provided by the system to the teacher, as he may examine the indicators only regarding the first group of students.

In all the case studies, data are recorded, while semi-structured interviews with all DIAS' users take place after the conclusion of the activities.

7 Conclusions – Future Work

During the first testing of the DIAS system in real settings we wanted to examine how the IA indicators influence the discussion activity evolvment, focusing on students' (users) behaviour. The main conclusion is that the indicators act as an additional motive for user's activity. Thus this discussion platform can be considered as an additional tool in any distance learning setting, providing means for increased interaction between the students. This effect of the visualized representation of interaction information seems to comply with the results presented by other researchers [2],[9],[10],[11]. Of course it relies upon the teacher to manage this tool to his/her benefit, by proper interpretation of the presented information, as well as by providing an appropriate set of indicators to forum participants so as to selfregulate their own activity. In the first case study attempts were made by some students to 'manipulate' the system and improve their position in the produced diagrams, without significantly contributing to the discussion activity. In one case a student wrote more than 1/3 of her messages the last two days of the activity, in order to appear as one of the most active users. Such behaviour can be revealed to the moderator by combining indicator information, as described in the aforementioned example. Thus the moderator can designate such possible abnormalities easily, without thorough examination of the messages' content. Currently we have constructed several similar interpretative schemas combining indicators, but each moderator may design

activities where he/she decides which sets of indicators are appropriate for the specific activity and the participating users.

Our future plans include completion of the case studies in progress and evaluation of the results. More case studies are under consideration for the near future, mostly addressing questions regarding the moderator's facilitation. Furthermore, we explore the needs of moderators, in asynchronous discussion fora other than for learning purposes (e.g. in the CSCW spectrum: such as open-audience discussions fora within corporative networks, scientific networks, etc). A complementary, overall goal is to associate activities and identifiable user action patterns, easily inspected through the visualized IA indicators.

References

1. Barros, B., Verdejo, F.: Analysing student interaction processes in order to improve collaboration. The DEGREE approach. *IJ AIED*, 11, (2000) 221-241
2. Bratitsis, T., Dimitracopoulou, A.: Data Recording and Usage Interaction Analysis in Asynchronous Discussions: The D.I.A.S. System. Workshop on Usage Analysis in Learning Systems, 12th Int. Conference AIED, Amsterdam. (2005).
3. Collins, M., Berge, Z.: Resources for moderators and facilitators of online discussion. (2001). Available online at: <http://www.emoderators.com/moderators.html>
4. Corich S., Kinshuk, Hunt L.: Assessing Discussion Forum Participation: In Search of Quality. *International Journal of Instructional Technology and Distance Learning*. TEIR Center, Duquesne University, Pittsburgh. (2004).
5. Dimitracopoulou, A et al.: State of the art of interaction analysis for Metacognitive Support & Diagnosis. IA JEIRP Deliverable D.31.1.1. Kaleidoscope NoE, December 2005. Available online at: www.noe-kaleidoscope.org
6. Gerosa, M.A. et al.: No need to read messages right now: helping mediators to steer educational forums using statistical and visual information. In: Koschmann, T., Chan, T., Suthers, D. (eds): *Proceedings of CSCL 2005*, LEA, USA (2005) 160-169
7. Gunawardena, C., Lowe, C. & Anderson, T.: Analysis of global online debate and development of interaction analysis model for examining social construction of knowledge in computer conferencing. *Educational Computing Research*, 17(4), (1997) 397-431
8. Jerman P., Soller A. & Muhlenbrock M.: From Mirroring to Guiding: A Review of State of the Art Technology for Supporting Collaborative Learning. In: P. Dillenbourg, A. Eurelings, & K. Hakkarainen (eds): *Proceedings of EuroCSCL*, Maastricht, (2001) 324-331
9. Mazza, R., Milani, C.: Exploring Usage Analysis in Learning Systems: Gaining Insights from Visualizations. Workshop on Usage Analysis in Learning Systems, The 12th International Conference on Artificial Intelligence in Education AIED, Amsterdam. (2005).
10. Mochizuki, T., et al: Promotion of Self-Assessment for Learners in Online Discussion Using the Visualization Software. In: Koschmann, T., Chan, T., Suthers, D. (eds): *Proceedings of CSCL 2005*, LEA editions, USA (2005) 440-449
11. Nakahara, J., et al: iTree: Does the mobile phone encourage learners to be more involved in collaborative learning? In: Koschmann, T., Chan, T., Suthers, D. (eds): *Proceedings of CSCL 2005*, LEA editions, USA (2005) 470-478
12. Reyes, P., & Tchounikine, P.: Mining learning groups' activities in Forum-type tools. In: Koschmann, T., Chan, T., Suthers, D. (eds): *CSCL 2005*, LEA, USA (2005) 509-513
13. Stahl, G.: *Group Cognition: Computer Support for Building Collaborative Knowledge*. Acting with Technology Series, MIT Press. (2006).

Analyzing Shared Workspaces Design with Human-Performance Models

Pedro Antunes¹, Antonio Ferreira¹, and Jose A. Pino²

¹ Department of Informatics, University of Lisbon, Portugal
paa@di.fc.ul.pt, asfe@di.fc.ul.pt

² Department of Computer Science, Universidad de Chile, Chile
jpino@dcc.uchile.cl

Abstract. We propose an analytic method to evaluate synchronous shared workspaces design. The method uses human-performance models, developed in the Human-Computer Interaction field, to make time predictions about collaborative actions performed in selected critical scenarios. We apply this method to two case studies: the design of a collaborative game and the redesign of a collaborative tool for software engineering requirements negotiation. The benefits and limitations of the method are discussed, as well as some implications for design.

1 Introduction

The design and evaluation of groupware usability is a challenging endeavor for practitioners and researchers because existing methods have considerable trade-offs and impose significant constraints:

- On the one hand, the required evaluation resources (time, users, experts, apparatus) may be hard to find or simply unavailable, a condition that worsens due to the iterative nature of formative usability evaluation. This applies especially, but not exclusively, to controlled laboratory experiments [1];
- On the other hand, several evaluation methods are either descriptive or prescriptive and therefore provide little support for comparing design options and predicting usability results. This applies to methods such as Groupware Task Analysis [2], Collaboration Usability Analysis [3], Groupware Walkthrough [4], and Groupware Heuristic Evaluation [5].

We argue that groupware designers should complement existing practice and knowledge with the ability to make quick measurements and calculations about key characteristics of computer-mediated collaboration. Our motivation is based on the century-old need to measure before improving as well as on the evidence that fast evaluation enables several design iterations. We introduce a method that can be applied without users or functional prototypes to quantitatively *predict* and *compare* the usability of synchronous shared workspaces (here referred to as shared workspaces).

Shared workspaces present an interesting challenge to usability evaluation because collaboration among group members features strong interdependencies

wherein individual actions affect the choices and outcomes of the other users. Furthermore, the impact of small, low-level, design decisions requiring perceptual or motor activity is much higher than in other contexts, where the emphasis may be on more cognitive tasks, such as decision making. These characteristics of shared workspaces are usually not captured by existing methods and tend to be overlooked. Instead, these methods focus on generic, high-level, communication and coordination mechanisms that the groupware should provide to support collaboration (e.g. the mechanics of collaboration [3]). We approach these two aspects of shared workspaces—action interdependencies and attention to detail—by focusing on the analysis of *critical scenarios* and by applying existing human-performance models from the Human-Computer Interaction (HCI) field:

- The analysis of critical scenarios raises the designer’s consciousness about collaborative actions that have a potentially important effect on individual and group performance;
- The human-performance models address the fine-grained details of the interaction with the shared workspace and provide performance estimates without the participation of users or the development of prototypes.

Human-performance models, such as the Keystroke-Level Model (KLM) [6], are based on a cognitive architecture that approximates single-user interaction at a low level of detail (e.g. perceptual, motor, and cognitive processors). We discuss the contextualization of this cognitive architecture to the specific characteristics of groupware in Sect. 3, and introduce some basic concepts necessary to model awareness and control of information about the users’ collaborative actions.

Section 4 describes the proposed method for evaluating group performance of users working together in a shared workspace. Two case studies are presented in Sect. 5 involving shared workspaces design to demonstrate the value of this method. We conclude the paper in Sects. 6 and 7 with a discussion of the benefits and limitations of the method, as well as with some implications for design.

2 Related Work

The application of human-performance models to the groupware context is very rare in the literature and virtually inexistent for workspace collaboration. We start this section with an overview of Distributed GOMS (DGOMS) [7] and a recent study involving a complex group task [8]. In both cases the same family of techniques, called GOMS (Goals, Operators, Methods, and Selection rules) [9], is used to provide quantitative estimates of human performance.

DGOMS [7] applies hierarchical task analysis and human-performance models to represent group activity and to predict execution time, distribution of workload, and other performance variables. This method successively decomposes group work in group tasks until individual subtasks can be identified. At this level of detail the subtasks are defined in terms of perceptual, cognitive, and motor operators, as well as with a new communication operator that is used to coordinate individual tasks executed in parallel. The problem, however, is that

such a coordination mechanism is more appropriate to groups where users react to predefined events, and not sufficiently rich to describe the type of interdependency established by users working through shared workspaces [10].

Another application of human-performance models to groupware considers “teams of models” to analyze a complex task executed by a group of users [8]. The task involved several users with individual roles monitoring a display and executing actions in a coordinated way, via a shared radio communication channel. While this approach assumes that several individual models are necessary to explain collaborative work, the study does not address workspace collaboration and focuses instead on coordinated work.

We now review some usability evaluation methods specifically developed for groupware. Groupware Task Analysis [2] is a method that combines high-level hierarchical task analysis and field observations for addressing all stages of groupware design. It is based on a conceptual framework including agents, group work, and situation, in a similar manner to the work models defined by the Contextual Design approach [11], well known in the HCI field.

The next three methods of groupware usability evaluation are based on a common descriptive framework called “mechanics of collaboration” [3], whereas each method applies a different evaluation perspective. The mechanics are formalizations of high-level group work primitives (e.g. communicating and coordinating) that helps the designer focus on how the shared workspace supports the required collaboration. Starting with Collaboration Usability Analysis [3], this method couples field observations and a version of hierarchical task analysis that allows variation, iteration, and parallel work, for representing group work. The Groupware Walkthrough [4] method uses step-by-step written narratives or task diagrams corresponding to collaborative scenarios, and it aims at gathering the opinions of expert inspectors while using the shared workspace. Finally, Groupware Heuristic Evaluation [5] is based on a number of experts evaluating the compliance of a shared workspace with a list of heuristics.

In summary, existing methods for groupware usability evaluation are of two types: the first type is targeted at predicting performance in coordinated work scenarios where users react to predefined events (not even requiring group awareness); the second type can be applied to shared workspaces but have a descriptive or prescriptive nature that allows for high-level task analysis or depends on inspections performed by multiple usability experts. Our proposed method complements these two types of methods by addressing detailed interdependencies in critical scenarios of collaboration using existing human-performance models.

3 Theoretical Background

In general, human-performance models have been associated with the Model Human Processor (MHP) [9], that represents human information processing capabilities using perceptual, motor, and cognitive processors. Nevertheless, several architectural differences are identified when considering individual models: for instance, KLM uses a serial-stage architecture, while CPM-GOMS addresses

multi-modal and parallel human activities (e.g. recognizing an object on the display while moving the hand to the keyboard) [12]. In spite of these differences, a common characteristic of existing human-performance models is that they are singleware, that is, they assume that just one user interacts with a physical interface. Figure 1 depicts this singleware architecture based on the MHP. We also illustrate that there is a conventional information flow in this architecture, from the cognitive to the motor processors, from the input to the output devices, and from the perceptual to the cognitive processors.

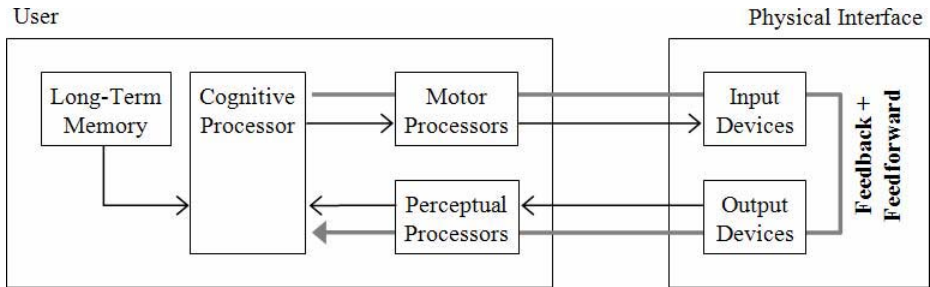


Fig. 1. Singleware architecture

According to some authors [8], the architecture depicted in Fig. 1 applies to groupware: to model a group of users, one can have individual models of the interaction between each user and the physical interface; one can also assume that the physical interface is shared by multiple users, and that the users will deploy procedures and strategies to communicate and coordinate individual actions. Thus, according to this view, groupware usage is reflected in some conventional information flows, spanning multiple users, which still may be described using the conventional production rules and representations.

The problem, however, is that this approach does not consider two fundamental groupware features: (1) the conventional information flows are considerably changed to reflect collaborative actions, mutual awareness, and interdependence; and (2) the focus and granularity should not remain on the interactions between the user and the physical interface but should significantly change to reflect the interactions between users, mediated by the physical interface. We address these two issues in the next section.

3.1 Groupware Conventional Information Flows

Let us start with the singleware architecture. In this context, we may characterize the conventional information flows in two categories: feedback and feedforward. The first category corresponds to information initiated by the user, for which the physical interface conveys *feedback* information to make the user aware of the executed operations [13,14]. The second category concerns the delivery of *feedforward* information, initiated by the physical interface, to make the user aware of the afforded action possibilities [14]. Now, when we regard groupware, some

additional categories have to be considered. In this paper we analyze explicit communication, feedthrough, and back-channel feedback.

Explicit communication addresses information produced by one user and explicitly intended to be received by other users [3]. For example, a user may express a request for an object to another user. This situation can be modeled as a physical interface capable of multiplexing information from input devices to several output devices. The immediate impact on the model in Fig. 1 is that we now have to explicitly consider additional users connected to the physical interface, as shown in Fig. 2.

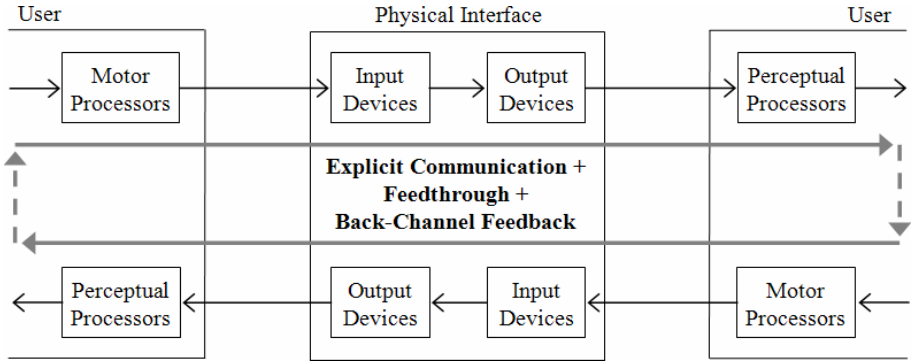


Fig. 2. Groupware architecture

Feedthrough concerns implicit information delivered to several users reporting actions executed by one user [15]. Feedthrough is essential to provide group awareness and to construct meaningful contexts for collaboration. For example, the shared workspace may show currently selected menus for each user that is manipulating objects. This information is automatically generated by the physical interface as a consequence of the user's inputs, and it is directed towards the other users. A very simple way to generate feedthrough consists of multiplexing feedback information to several users. Sophisticated schemes may consider delivering less information by manipulating the granularity and timing associated to the operations executed by the groupware [16].

Finally, *back-channel feedback* concerns unintentional information flows initiated by one user and directed towards another user to facilitate communication [17]. No significant content is delivered through back-channel feedback, because it does not reflect cogitation from the user. Back-channel feedback may be automatically captured and produced by the physical interface based on the users' body gestures and vocal activities.

3.2 Groupware Specializations of Physical Interface Devices

All information flows in the groupware architecture are naturally processed by the user's cognitive, perceptual, and motor processors, and the corresponding

physical input and output devices. However, we regard the separate processing of explicit communication, feedthrough, and back-channel feedback in specialized input and output devices to show the distinction between collaborative and non-collaborative interactions. We define the *awareness input/output devices* as devices specialized in processing sensory information about who, what, when, how, and where are the other users operating in the shared workspace.

Another specific feature of the awareness input/output devices is that they not only afford users to construct a perceptual image of the collaborative context, but they also allow users to perceive the role and limitations of the physical interface as a mediator. This is particularly relevant when the Internet is being used to convey feedthrough information, where feedthrough delays are significantly longer and less predictable than feedback delays [18] and the available bandwidth and network availability may be limiting factors [19].

A further reason for proposing the awareness input/output devices is related to another particular characteristic of groupware: it lets users loose the link between executed operations and group awareness, a situation called “loosely coupled” [20]. Two types of coupling control may be considered: at the origin and at the destination. Users may control coupling at the origin to specify what and when private information should become public. But coupling can also be controlled at the destination: getting awareness information on a per-object demand basis, e.g. by specifying filters that restrict received awareness to some selected objects and types of events. In all cases this situation requires some cognitive activities from the user to discriminate and control awareness information delivery, and we model this situation with the *coupling input device*.

We illustrate the resulting groupware physical interface in Fig. 3. In summary, our interpretation of the MHP architecture, taking the groupware context in consideration, essentially emphasizes the cognitive activities related to the awareness and coupling features supported by the groupware physical interface.

4 Method to Evaluate Group Performance

Step 1: Defining the physical interface. The method starts by defining the physical interface of the groupware under analysis. We propose that the physical interface may be decomposed into several shared workspaces. Such decomposition simplifies the analysis of complex groupware tools, that often organize collaborative activities in multiple intertwined spaces, usually human recognizable, supporting various purposes, objects, and functionality.

Using the groupware physical interface in Fig. 3 as reference, we define a shared workspace as a distinctive combination of awareness and coupling devices. We exclude from the analysis any workspaces not having, at least, one awareness or coupling device, since they would not involve collaboration.

The outcome of this step is then: (1) a list of shared workspaces; (2) definition of supported explicit communication, feedthrough, and back-channel feedback information; and (3) characterization of supported coupling mechanisms. In this step alternative design scenarios may also be defined, considering

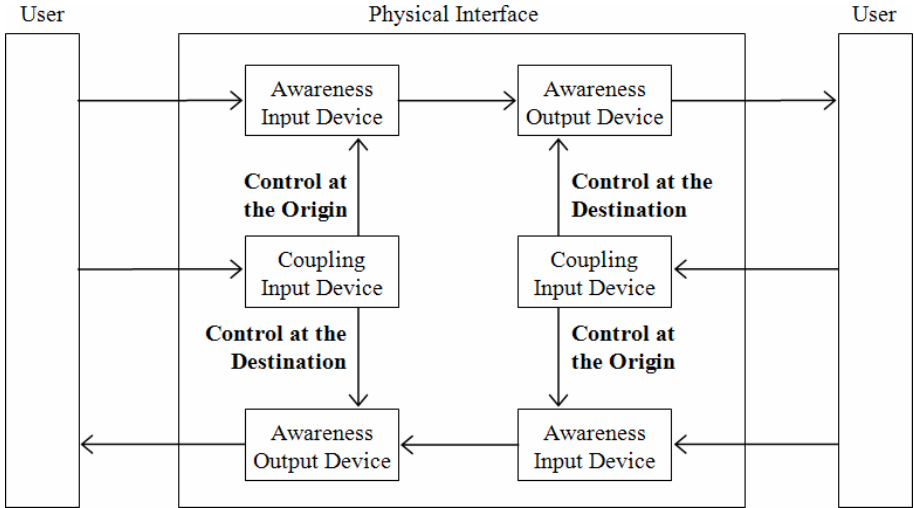


Fig. 3. Groupware physical interface

different combinations of shared workspaces, awareness information, and coupling mechanisms.

Step 2: Breakdown definition of critical scenarios. The second step describes the functionality associated with the various shared workspaces with respect to critical scenarios, i.e. with a special focus on collaborative actions that have a potentially important effect on individual and group performance. This functionality is successively decomposed from the more general to the more detailed, using a top-down strategy, typical of hierarchical task analysis. Alternative design scenarios may be defined, considering several combinations of users' actions.

Step 3: Comparing group performance in critical scenarios. The final step is dedicated to compare the alternative design scenarios defined in the previous steps. These comparisons require a common criteria, for which we selected the *predicted execution time* in critical collaborative scenarios.

We utilize the Keystroke-Level Model (KLM) [6] to predict execution times because it is relatively simple to use and has been successfully applied to evaluate single-user designs [12]. In KLM, each user action is converted into a sequence of mental and motor operators, whose individual execution times have been empirically established and validated in psychological experiments: M is for mental preparation and takes 1.2 seconds; P is for pointing with the mouse to a target on a display, requiring 1.1 seconds; and K is for pressing or releasing a mouse button, taking 0.1 seconds [6,9]. Therefore, the designer may find out which sequence of operators minimizes the execution time of a particular user action.

Naturally, the application of KLM must be adapted to groupware, considering that the execution time we want to evaluate encompasses several users who work in parallel. Our approach consists of focusing the analyses on critical scenarios

involving selected sequences of operations from more than one user. For instance, suppose we want to analyze several design alternatives for managing the access to shared workspace objects. A critical scenario occurs when a user accesses the object, immediately followed by another one trying to access the object but finding it locked. We may use KLM to estimate the execution times of these combined operations for each design option, and thus finding out which one minimizes the overall execution time. We discuss in detail the application of this method to groupware design in the next section.

5 Using the Method

5.1 Collaborative Game

We apply the method to a collaborative game in this case. The game explores a collaborative scenario where players have specific roles and act opportunistically according to the current state of the shared workspace. In particular, players can make either vertical or horizontal connections between points in a board. The objective of the game is to connect all points in the board as quickly as possible (Fig. 4). The points are connected in pairs, but this is only allowed if at least one of the to-be-connected points is already linked to a third point via a perpendicular connection. Initially, the board contains a single connection line.

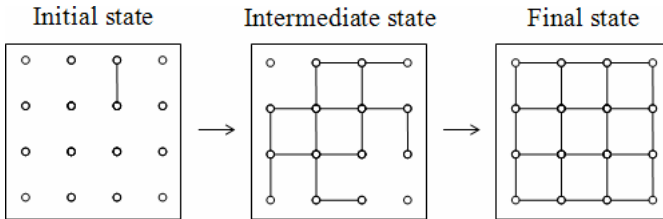


Fig. 4. Players act opportunistically to make connections between all points

Step 1: Defining the physical interface. The game provides a shared workspace displaying a public updated view of the board (Fig. 4). There exist several private workspaces also, one for each player, allowing them to actually connect the points with horizontal or vertical lines, depending on their specific role. However, the analysis of these private workspaces is out of scope, since we are only interested in collaborative actions. The player's moves are restricted to be done with a mouse having a single button.

Step 2: Breakdown definition of critical scenarios. The board operates in the following way. In order to connect two points, a player must first reserve them on the board. Multiple players may not simultaneously reserve the same points, but as this can happen and have a considerable performance penalty, we consider this a critical scenario. Reservation is done by selecting two adjacent points with

the mouse and dragging them out of the board (to a private workspace). The connection is made public when the points are dragged back to the board, an action that also automatically releases the points. This is our design scenario A.

We also analyze an alternative scenario B, where, in order to increase awareness and minimize inadvertent selections of reserved points, the board displays a letter identifying the current owner next to the reserved points. This design provides awareness information only at the end of the reserve or release actions.

An additional alternative design scenario C features extra awareness while the points are being selected on the board. The main justification for this refinement is the production of more fine-grained and up-to-date awareness information.

We will next proceed with a detailed specification of the collaborative actions for the selected critical scenario. For now, we observe there are only two collaborative actions in this game, which we designate **RESERVE** and **RELEASE**.

Step 3: Comparing group performance in critical scenarios. We now focus on the fine-grained details of the **RESERVE** and **RELEASE** collaborative actions, to the point where they can be described with KLM operators. Starting with the **RESERVE** action, we assume the player begins by searching the shared workspace for a point that satisfies three conditions: (1) it must not be reserved; (2) it must allow a new connection; and (3) it must have a perpendicular connection to another point. This is converted into a single **M** operator because the verification of the three conditions is highly repetitive and players are trained in the game.

Once a point is located, the player moves the mouse pointer near it, **P**, presses the mouse button, **K**, and moves the pointer to an adjacent point, **P** (the connection between these two points will be drawn afterwards in the private workspace). The player then releases the mouse button, **K**, to complete the selection.

The last part of a reservation is done by dragging the selected points out of the board: the player adjusts the mouse pointer so that it rests on top of the adjacent point, **P**, presses the mouse button, **K**, drags the selected points out of the board, **P** (no **M** operator is required because the workspaces are always in the same place), and releases the mouse button, **K**. The complete sequence of KLM operators for the **RESERVE** collaborative action is **MPKPKPKPK**, which has a predicted execution time of 6 seconds. The **RELEASE** collaborative action is very similar to **RESERVE** in two ways: the predicted execution time is also 6 seconds, and the sequence of KLM operators is again **MPKPKPKPK**.

Now, having determined the sequences of operators for managing the board, we focus on the comparison of group performance in the critical scenario—when two players have the intention of reserving the same points—for the design alternatives A and B. We assume the first player will always succeed in order to simplify the analysis, and also that having more than 2 players reserving the same points is a rare event that does not deserve further attention.

Considering the design scenario A, the best case happens when two players start the reservation for the same points at the same time. In this case, after the 6 seconds needed for a complete reservation, the second player notices an error indication on the board (an **M** operator) and starts again with other points, which takes additional 6 seconds. The best execution time is then 13.2 seconds.

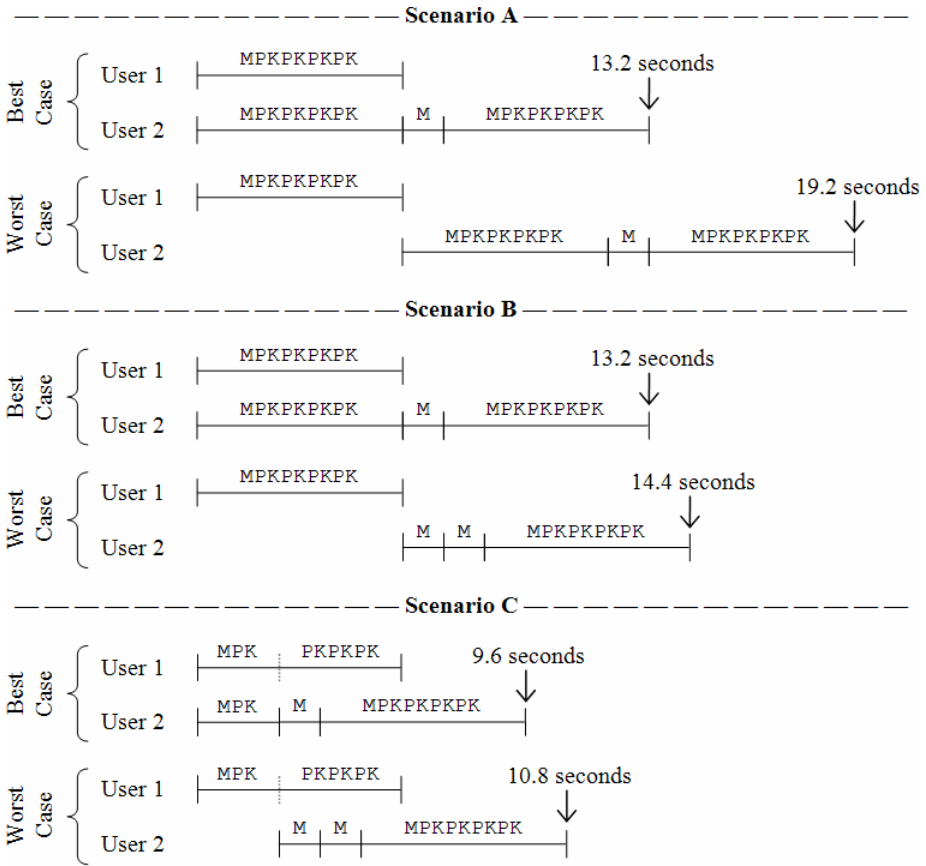


Fig. 5. Best and worst execution times for handling the critical scenario

The worst case happens when the second player begins just after the first player finishes a reservation; since no awareness information is provided, the total execution time increases to 19.2seconds (see scenario A in Fig. 5).

For the scenario B, the best case is identical to that of scenario A. However, the execution time for the worst case is significantly reduced because the second player can interrupt an ongoing reservation as soon as the owner letter is displayed on the board. We represent this situation with two M operators: the first corresponds to the initial M of any reservation, while the second M is for interpreting the critical situation. The total execution time for the worst case is now 14.4seconds (see scenario B in Fig. 5).

The optimization considered in scenario C provides awareness information upon the selection of the first point, i.e. right after a sequence of MPK (instead of the full MPKPKPKPK). In these circumstances both the best and worst cases benefit from reduced execution times (see scenario C in Fig. 5). If the two players start the reservation at the same time, then at about 2.4seconds they both see

their simultaneous selections on the board. Then, the second player (by our assumption) decides to stop the current selection and starts another one, an M followed by a new reservation, which takes 9.6seconds. The worst case takes 10.8seconds; its explanation is analogous to the worst case for scenario B, except the awareness supplied by the owner letter upon a full reservation is substituted by the awareness provided by the selection of the first point.

In summary, the method brought quantitative insights about the role of feedthrough information in group work support, predicting that for the selected critical scenario the design option C is faster than B by 3.6seconds, and that B is faster than A by about 4.8seconds, but only in the worst case scenario.

5.2 Software Requirements Negotiation Tool

We now demonstrate the application of the analytic method to an existing groupware tool that supports collaborative software quality assessment using the Software Quality Function Deployment (SQFD) [21] methodology. The objective of this tool is to facilitate the SQFD negotiation process by providing mechanisms in a same-time, different-place mode. Our starting point in this case was a previous experiment with the tool that gathered data via questionnaires, and that reported some usability problems, namely that it was considered difficult to use. Further details about this tool and about the previous usability evaluation can be found in [22].

Step 1: Defining the physical interface. The tool has two shared workspaces: SQFD matrix and “Current Situation.” The SQFD matrix allows users to inspect a matrix of correlations between product specifications and customer requirements, as well as observing which correlations are under negotiation. Limited awareness information is provided by the matrix, but there is a coupling mechanism allowing users to analyze a cell in more detail. This coupling mechanism leads users to the “Current Situation,” where they can observe the negotiation state in detail, including the proposed correlation, positions in favor or against, and supporting arguments. We briefly characterize the two shared workspaces in terms of input, output, and coupling devices in Figs. 6 and 7.

Step 2: Breakdown definition of critical scenarios. We focus our discussion on the “Current Situation” shared workspace to illustrate the method application. The user arrives to this space with the purpose of analyzing the negotiation state in detail. As currently implemented by the tool, the status information is hierarchically organized, showing: (1) the product specifications and customer requirements under negotiation; (2) the currently proposed correlation; (3) positions in favor, followed by positions against the currently proposed correlation; (4) arguments supporting positions in favor or against. We designate this as design scenario A.

An alternative design scenario B considers a variation in the way information is shown to the user. We assume that users may give more importance to the aggregate information about the number of positions against/in favor, neglecting

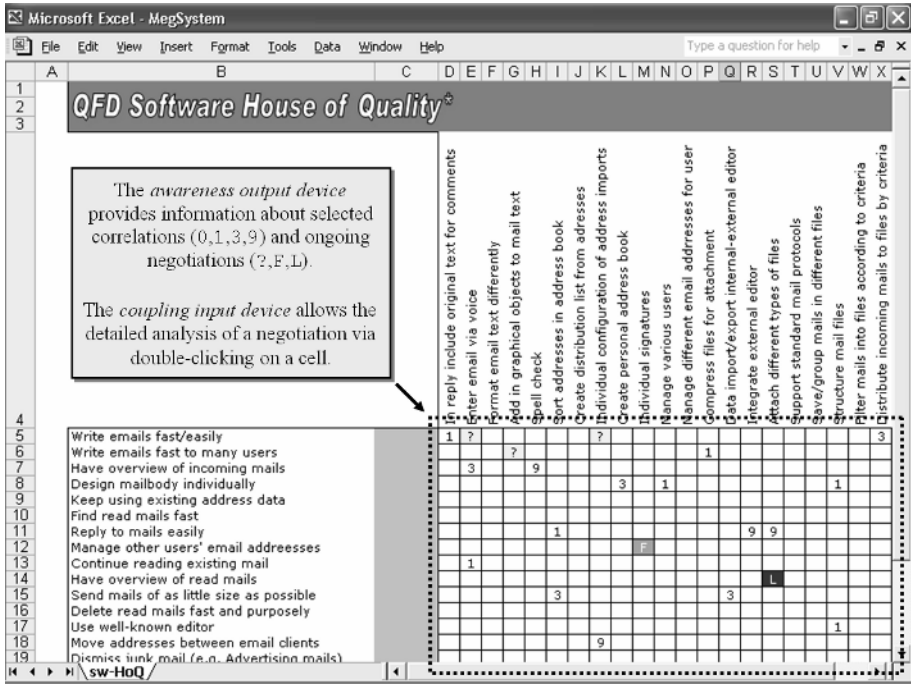


Fig. 6. The SQFD matrix

positions when there is a clear push towards one side or the other, and analyzing arguments in detail only when positions are balanced.

The selected critical scenario considers the proposal of an alternative correlation value in “Current Situation” after analyzing the negotiation status. We also consider a variation in the number of users involved in the negotiation process. The “Current Situation” may display the positions and arguments for up to 3 users (see Fig. 7). Beyond that number, a user has to scroll down the window to completely analyze the situation. Thus, we consider 3 and 6 users involved in the critical scenario. We assume that having more than 6 users negotiating the same cell is a rare event, which does not deserve further analysis.

Step 3: Comparing group performance in critical scenarios. For the design scenario A and 3 users, we have: the interpretation of the negotiation status, M, followed by a decision, M, which is expressed via the selection of a check box, PKK, and a press in the “ok” button, PKK. This gives MMPKKPKK, which has a total execution time of 5.0seconds. With 6 users, the execution time increases to 8.6seconds, corresponding to MPKPK MMPKKPKK, in which the MPKPK operators are related to scrolling.

Considering the design option B, we have two situations: either the positions are balanced (a tie or simple majority) or unbalanced (i.e. absolute majority).

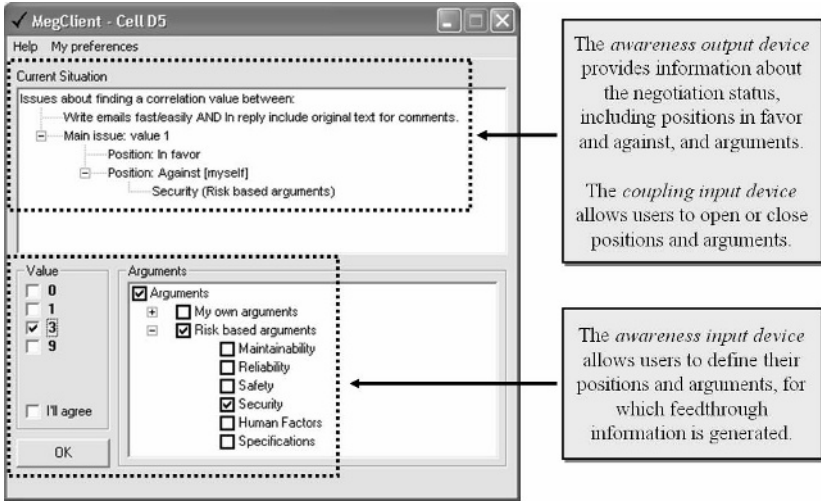


Fig. 7. The “Current Situation” shared workspace

In the unbalanced case, we assume the user will neglect arguments and thus we have MMPKKPKK (5.0seconds to execute), similar to the previous scenario with 3 users. In the balanced case the user will analyze the positions in detail via the interpretation of the negotiation status, M, followed by the opening of the list of favorable arguments, PKK, and corresponding analysis, M, upon which the list is closed, PKK, to give room for the opening and interpretation of the against arguments, PKK M, so that, finally, the decision is made, MPKKPKK. The total execution time for the balanced case, MPKKMPKKPKKMPKKPKK, is then 11.3seconds. Note that these measures apply to the scenarios with 3 and 6 users. We also assume that the probability of having unbalanced positions is 25%¹. Hence, in these circumstances, the average execution time for scenario B is about $0.75 \times 11.3 + 0.25 \times 5.0 \approx 9.8$ seconds, which is higher than scenario A for both 3 and 6 users. In other words, scenario B may be better or equal than scenario A, but there is a 75% probability that it is worse than scenario A, which severely penalizes the overall appreciation of the design in scenario B.

6 Discussion and Implications for Design

Both the collaborative game and the requirements negotiation tool analyzed in this paper heavily depend on shared workspaces to orchestrate multiple users accomplishing a collaborative task. The design of these workspaces is thus critical to the overall task performance. Since we use a quantitative common criterion

¹ This is the probability of having an absolute majority with 3 or 6 voters, assuming a uniform distribution. For 3 voters, the absolute majority requires having all in favor or against, i.e. 2 out of 8 combinations, or 25%.

to evaluate group performance—the execution time predictions of collaborative actions in critical scenarios—we may benchmark various design solutions to estimate which shared workspace functionality offers the best performance.

It is important to note the two cases studied in this paper are quite distinct. The collaborative game is a fictitious tool intended to test preliminary design ideas in environments where players act opportunistically, while the requirements negotiation tool is a completely functional tool, that could nevertheless benefit from further optimizations. We analyzed several design solutions with the collaborative game related to the way users structure their actions according to awareness on other people. The requirements negotiation tool helped us to analyze how a coupling mechanism could be designed to conserve individual cognitive effort. We defined a critical scenario to evaluate the collaborative game highlighting coordination problems. By contrast, the critical scenario used to evaluate the requirements negotiation tool shows escalating problems with the number of users engaged in collaborative actions. Taken as a whole, the method always contributed to formative evaluation, offering clear indications about the potential performance of users working with shared workspaces.

The proposed method has two important limitations that we would like to discuss. First, it assumes a narrow-band view about collaboration, restricted to shared workspaces and their mediation roles. This contrasts with the other available groupware usability evaluation methods offering a wide-band view about collaboration, encompassing, for example, various communication channels, coordination policies, and broader issues such as group decision making or learning. However, the tradeoff to ponder is that the proposed method restricts the view in order to increase the detail about the mediating role of shared workspaces. This restricted view has ample justification in contexts where shared workspaces are heavily used, even when users perform intellectual tasks (such as in the requirements negotiation case, where users apply their expertise to evaluate software quality, but are still requested to repetitively operate the tool).

Second, the method is somewhat limited by the selection of critical scenarios. As designers and evaluators, we have to ponder whether the selected critical scenarios are representative and have sufficient impact on the overall collaborative task to deserve detailed analysis. In our first case, the collaborative game, this question is delicate because the game was conceived to illustrate the method application with that critical scenario. However, the situation was quite different in the second case, because we started our analytic evaluation with a preliminary evaluation study indicating that the tool had usability problems [22]. Thus, some prior evaluation results allowed us to determine the critical scenarios for the subsequent evaluation task. We conjecture that this cyclic approach may reduce the bias introduced by critical scenarios. Furthermore, critical scenarios are commonly used as a sampling strategy in qualitative inquiry, allowing generalization [23]. The proposed method combines qualitative and quantitative approaches with the same purpose.

7 Conclusions

Confronting the obtained results with the driving forces mentioned in Sect. 1, we may conclude from this research that the proposed method can be used to quantitatively predict and compare the usability of shared workspaces, without requiring users or the development of functional prototypes. More specifically, available knowledge about human-performance models can be applied to predict execution times in critical scenarios involving intricate collaborative actions that have a potentially important effect on individual and group performance.

Research described in this paper is a preliminary step in the direction of exploring human-performance models to evaluate shared workspaces design. Our performance estimates were based on experimental measures of time spent by humans executing single user operations. Experimental research with groupware will be accomplished in the future, in an attempt to provide estimates for typical groupware interactions in critical scenarios.

Acknowledgments

This work was partially supported by the Portuguese Foundation for Science and Technology, Project POSI/EIA/57038/2004 and Fondecyt (Chile) Project No. 1040952.

References

1. Fjermestad, J., Hiltz, S.: An assessment of group support systems experimental research: Methodology and results. *Journal of Management Information Systems* **15**(3) (1999) 7–149
2. van der Veer, G., van Welie, M.: Task based groupware design: Putting theory into practice. In: *DIS'00: Proceedings of the conference on Designing interactive systems*, New York City, New York, United States (2000) 326–337
3. Pinelle, D., Gutwin, C., Greenberg, S.: Task analysis for groupware usability evaluation: Modeling shared-workspace tasks with the mechanics of collaboration. *ACM Transactions on Computer-Human Interaction* **10**(4) (2003) 281–311
4. Pinelle, D., Gutwin, C.: Groupware walkthrough: Adding context to groupware usability evaluation. In: *CHI'02: Proceedings of the SIGCHI conference on Human factors in computing systems*, Minneapolis, Minnesota, USA (2002) 455–462
5. Baker, K., Greenberg, S., Gutwin, C.: Empirical development of a heuristic evaluation methodology for shared workspace groupware. In: *CSCW'02: Proceedings of the 2002 ACM conference on Computer supported cooperative work*, New Orleans, Louisiana, USA (2002) 96–105
6. Card, S.K., Moran, T.P., Newell, A.: The keystroke-level model for user performance time with interactive systems. *Communications of the ACM* **23**(7) (1980) 396–410
7. Min, D., Koo, S., Chung, Y.H., Kim, B.: Distributed GOMS: An extension of GOMS to group task. In: *SMC'99: Proceedings of the IEEE international conference on Systems, man, and cybernetics*, Tokyo, Japan (1999) 720–725

8. Kieras, D.E., Santoro, T.P.: Computational GOMS modeling of a complex team task: Lessons learned. In: CHI'04: Proceedings of the SIGCHI conference on Human factors in computing systems, Vienna, Austria (2004) 97–104
9. Card, S.K., Newell, A., Moran, T.P.: The psychology of human-computer interaction. Lawrence Erlbaum Associates, Mahwah, NJ, USA (1983)
10. Malone, T.W., Crowston, K.: The interdisciplinary study of coordination. *ACM Computing Surveys* **26**(1) (1994) 87–119
11. Beyer, H., Holtzblatt, K.: Contextual design: Defining customer-centered systems. Morgan Kaufmann Publishers, San Francisco, CA, USA (1998)
12. John, B.E., Kieras, D.E.: Using GOMS for user interface design and evaluation: Which technique? *ACM Transactions on Computer-Human Interaction* **3**(4) (1996) 287–319
13. Douglas, S.A., Kirkpatrick, A.E.: Model and representation: The effect of visual feedback on human performance in a color picker interface. *ACM Transactions on Graphics* **18**(2) (1999) 96–127
14. Wensveen, S.A.G., Djajadiningrat, J.P., Overbeeke, C.J.: Interaction frogger: A design framework to couple action and function through feedback and feedforward. In: DIS'04: Proceedings of the 2004 conference on Designing interactive systems, Cambridge, MA, USA (2004) 177–184
15. Hill, J., Gutwin, C.: Awareness support in a groupware widget toolkit. In: GROUP'03: Proceedings of the 2003 international ACM SIGGROUP conference on Supporting group work, Sanibel Island, Florida, USA (2003) 258–267
16. Gutwin, C., Greenberg, S.: The effects of workspace awareness support on the usability of real-time distributed groupware. *ACM Transactions on Computer-Human Interaction* **6**(3) (1999) 243–281
17. Rajan, S., Craig, S.D., Gholson, B., Person, N.K., Graesser, A.C.: AutoTutor: Incorporating back-channel feedback and other human-like conversational behaviors into an intelligent tutoring system. *International Journal of Speech Technology* **4**(2) (2001) 117–126
18. Gutwin, C., Benford, S., Dyck, J., Fraser, M., Vaghi, I., Greenhalgh, C.: Revealing delay in collaborative environments. In: CHI'04: Proceedings of the SIGCHI conference on Human factors in computing systems, Vienna, Austria (2004) 503–510
19. Cosquer, F.J.N., Antunes, P., Verissimo, P.: Enhancing dependability of cooperative applications in partitionable environments. *Lecture Notes in Computer Science* **1150** (1996) 335–352
20. Dewan, P., Choudhary, R.: Coupling the user interfaces of a multiuser program. *ACM Transactions on Computer-Human Interaction* **2**(1) (1995) 1–39
21. Haag, S., Raja, M.K., Schkade, L.L.: Quality function deployment usage in software development. *Communications of the ACM* **39**(1) (1996) 41–49
22. Antunes, P., Ramires, J., Respicio, A.: Addressing the conflicting dimension of groupware: a case study in software requirements validation (2006) To appear in *Computing and Informatics*.
23. Miles, M.B., Huberman, M.: *Qualitative data analysis: An expanded sourcebook*. Sage Publications, Thousand Oaks, CA, USA (1994)

Using Email-Based Network Analysis to Determine Awareness Foci

Adriana S. Vivacqua¹ and Jano Moreira de Souza^{1,2}

¹ COPPE/UF RJ, Graduate School of Computer Science,

² DCC/IM, Institute of Mathematics

Federal University of Rio de Janeiro,

Rio de Janeiro, Brazil

{avivacqua, jano}@cos.ufrj.br

Abstract. A number of studies have indicated that awareness of others' activities plays an important part in collaboration. Consequently, awareness has been a frequent theme in cooperative work research. Researchers have acknowledged that proximity has a strong effect on collaboration, and that maintaining awareness of peers becomes harder in distributed environments. Many awareness systems require configuration by the user and work only in predefined shared environments. In this paper, we present an investigation into the determination of awareness targets, through email-based user interaction analysis. The final goal is to be able to draw inferences as to who and what a user would be interested in maintaining awareness of, enabling a system to automatically determine awareness foci and adjust itself according to its user.

1 Introduction

The dissemination of network technology and adoption of distributed work teams by companies has led to a move towards remote work: individuals that used to be collocated might now be spread throughout the world. Studies have shown increased adoption of virtual work teams, in which members are geographically dispersed and communicate and coordinate mainly via electronic communication tools [16]. People participate in several projects at the same time, dividing their time and attention accordingly [21]. Individuals must therefore organize themselves and their work to accomplish different tasks, very often with different collaborators. Participation in different groups usually means that, depending on the situation, a person might have different roles and obligations, perform different activities and work towards different goals, all of which must be managed so they do not conflict with each other.

This leads to the notion of supporting individual work and tying it to the group as appropriate [25]. We work with looser collaborative environments, in which individuals need tools that enable them to quickly switch into closer interaction when necessary, and to easily relate their work to that of others.

In collocated environments, individuals are capable of observing others' actions, thereby gathering awareness information [14]. With increased distribution and implementation of virtual teams, opportunities for collaboration, interaction and information

exchange may be compromised: in these environments, casual interactions seldom happen and observation of others becomes harder.

A looser structure and distance sometimes lead to fragmentation: members may not communicate very often or be kept up-to-date of the latest evolution in others' work, resulting in rework, delays or confusion. The focus of this research is on improving awareness of the work environment in order to facilitate the group's work. This paper describes a method to automatically distribute task awareness information among group members and an initial reflection upon some of the assumptions underlying this method. Our system has been conceived as a means of integrating individual work with the shared group context, with the final goal of improving cohesion and reducing fragmentation. We expect such a system will promote informal interaction and facilitate opportunistic collaboration when deployed.

The remainder of this paper is organized as follows: in the next section we present a brief literature review of the area, followed by the envisioned system in section 3. Section 4 contains a preliminary analysis, followed by a discussion in section 5.

2 Background Literature

2.1 Self Governing Groups

In self-governing groups, actors have control over job allocation, day-to-day production planning and control [2]. These groups emerge out of a need to handle unpredictable events or contingencies (and usually dissolve when they are no longer necessary), and enable an organization to quickly adapt to new demands generated by the environment, sometimes deviating from pre-established norms and rules. In many cases, groups are composed of peers, where there is no formal hierarchical structure. This means that many of the decisions are the result of arrangements between peers, as is the work that finally gets done [1].

Due to the underlying interdependence between tasks, workers have to articulate (i.e., divide, allocate, coordinate, schedule, interrelate, etc.) their activities [28]. When individuals collaborate, they often shift back and forth between individual and shared work, and between loosely and tightly coupled collaboration [14]. This is especially true when there is low interdependence between them [15]. A reasonable approach in these cases is to provide individual work support and add collaboration support to the individual work tools, enabling collaboration when necessary. Awareness of current and past efforts becomes necessary, since one individual might work on a shared artifact for a while and another may pick it up later [5]. This looser structure and distance may lead to a decrease in involvement and interaction. As a consequence, individuals miss opportunities for collaboration, and sometimes end up working individually because they are unaware of each other's activities, performing overlapping tasks or duplicating work.

2.2 Awareness

Situation awareness involves perception and interpretation of relevant elements of the environment. The basic set of elements that compose workspace awareness information are those that address the "who, what, where, when and how" questions: who are

we working with, what are they doing, where are they working, when and how certain events happen.

Awareness is knowledge about the environment that must be maintained as it changes. It is maintained through perceptual information gathered from the environment; and it is generally secondary to other goals. Staying aware of others is taken for granted in everyday interactions, but becomes hard in distributed systems, where communication and interaction resources are poor [14]. This information facilitates collaboration by simplifying communication and coordination, allowing better management of coupling and determination of the need to collaborate: prior research has established that awareness of others is important in integrating a group [23], creating and maintaining shared context [13], and establishing contact [11].

2.2.1 Focus and Nimbus Theory

The Focus and Nimbus model of awareness for shared applications is based on spatial models of interaction [26]. It considers a set of objects in space, which interact based on their levels of awareness. Awareness, in turn, is manipulated via focus and nimbus, subspaces within which an object directs its presence or attention. Awareness is the overlap between nimbus and focus, where:

- Nimbus is the information given out by each object in the space, which can be perceived by others, and
- Focus describes the objects at which a user directs his or her attention.

In a collocated environment, individuals give out a large amount of information while working, which can be picked up by anyone paying attention to it. In computational settings, users give out information via the applications they interact with and the operating system, which is normally not relayed to others. We believe some of this information might be of use to help the group coordinate and conduct its work: other users should be able to pick up part of the information generated, depending on their focus. In our approach, we determine a user's focus through an analysis of his or her ongoing interactions. We are also working on a privacy scheme to automatically determine a user's nimbus.

2.3 Social Worlds and the Locales Framework

The Locales Framework [8] provides a set of abstractions to support the design and analysis of collaborative work. It is based primarily on the notion of continually evolving action and of *Social Worlds*. A Social World is a group of people who share a commitment to collective action, and it forms the prime structuring mechanism for interaction (as defined by Strauss, cited in [9]). Individuals are usually involved in multiple social worlds at a time, which means that different social worlds are interconnected and that actions in one social world may reflect in another. Each individual typically engages in multiple activities that span more than one social world.

In this framework, a *Locale* is an abstract concept that arises from the use of space and resources by a group. It maps the relationship between a Social World (and its interaction needs) and the *sites* and *means* its members use to meet those needs. Sites are the spaces (e.g. shared file systems) and means are objects contained in these spaces (e.g. the files and documents stored in this file system) [7].

Following these lines of thought, we have been working on collaboration support systems that take into account the emergent and situated nature of work, and the fact that individuals constantly reorganize to perform their tasks. We are working on systems to help the individual connect his work to others who relate to them.

2.4 Social Network and Interaction Analysis

Social network analysis is used widely in the social and behavioral sciences, as well as economics. It concerns the study of social entities and their relationships: communication among individuals, trade between businesses or treaties between nations. It considers structures such as the sociogram, a graph that represents individuals and the relations between them [31]. These relations can be of diverse nature (communication, party attendance, etc.), and are usually expressed as graphs and matrixes; upon which network analysis can be performed [29]. Social network analysts look at the world in terms of patterns or regularities in relationships between actors.

Sociocentric analysis looks at relationship structures from a global perspective (e.g., a graph of the communication between all members of a department or group). Egocentric network analysis, on the other hand, focuses on the individual (*ego*), and analyzes his or her interactions with a set of others (*alters*). This type of network has been used to study the social environment surrounding individuals or families, and social support structures [31].

Electronic interactions usually leave traces, such as email, fora or messenger logs. These interactions display certain rhythms that correspond to the individuals work patterns [24], and can be used to study the evolution collaborative endeavors. For instance, intense message exchange usually accompanies cooperative work. Additionally, individual patterns of email exchange can also indicate hierarchy and positioning in a group [6]. We construct an egocentric network based on the email records of electronic communication, and search this network to discover ongoing collaboration and need for awareness information.

3 An Approach for Awareness Information Distribution

To bridge the gap between individual and joint work, we have designed a distributed, peer to peer system to provide awareness information. In this system, agents check each user's current activities and ongoing interactions and exchange information with other peers to keep its user informed of their activities. Each agent's goal is to *maintain awareness between peers by displaying information about the activities of its user's acquaintances*. To reach this goal, the agent:

1. collects information generated by the user while working on his or her computer;
2. exchanges information with other users' agents; and
3. provides information to the user about his or her alters' activities.

This means that the agent must filter the information down to that which might be of interest to its user. In this paper, we present a method to determine awareness foci.

An egocentric network is built based on the set of user acquaintances. This network can be viewed as a tree with ego (the user) at the root and his or her alters (the

acquaintances) at the first level, to which the information generated by each user (the list of tasks he or she is currently performing) is added as a second level. Selecting the appropriate information thus becomes a problem of determining which of the leaves in this tree are of interest to the user and pruning the answer space accordingly. The determination of interest foci is divided into two stages, discussed in more detail in the following subsections:

1. discovering which peers the user might be interested in (selecting nodes at the first level); and
2. deciding which of their activities the user would want to know about (selecting leaves).

In this section, we describe the reasoning used to select from the universe of available information (everything generated by other users and provided to the assistant agent) that which is relevant to the user, which we call the *focus of interest*.

3.1 Information Processing and Organization

To reason about its user's needs, the agent gathers information about ongoing interactions from email logs. This information is organized to represent ongoing relationships and interest foci. The following concepts are used: a *tie* is a relationship between two users. It is composed of *interactions* between these parties. These interactions in turn are composed of *message exchanges*, which are groups of *email messages* (raw data). A series of email messages is grouped into an interaction, and several interactions define a tie. These concepts are illustrated in Figure 1.

To construct the user's network, we take the values of the *From*, *To*, *CC* and *BCC* fields and build the user's list of acquaintances. In this network, *Ego* is the user (normally determined by looking at the *From* field of outgoing emails), and his or her *Alters* are the many peers with whom ego exchanges email. The system groups email messages and their replies (determined via *Subject*, *Message-ID* and *Reference-To* tags) into interactions (conversations in Gmail), qualifying each reply by the time it took the user to respond (extracted from the *Date* field). An interaction contains several messages, qualified by length (number of emails) and duration (time from first to last message). A tie is characterized by the frequency of interaction between alters, i.e. how often they exchange mail.

For each user, average frequency of interaction and average response time are also calculated generically (how quickly does ego respond to email or how often he or she sends/receives email) and per alter, which we believe will turn out to be more significant (how often does ego send email to alter A and how quickly does ego reply to messages from alter B).

To reduce the search space, this network is pruned before adding the set of activities each alter is performing. It is easy to see how this search space can become quite large, which is why the system attempts to infer the need for awareness information. As an illustration, picture a user with 100 contacts in his or her address book, each of which performing 3 or 4 tasks simultaneously – if we were to provide the user with this raw data, he or she would have to keep track of 100 different people performing 300-400 different tasks to determine which ones are interesting, which would most

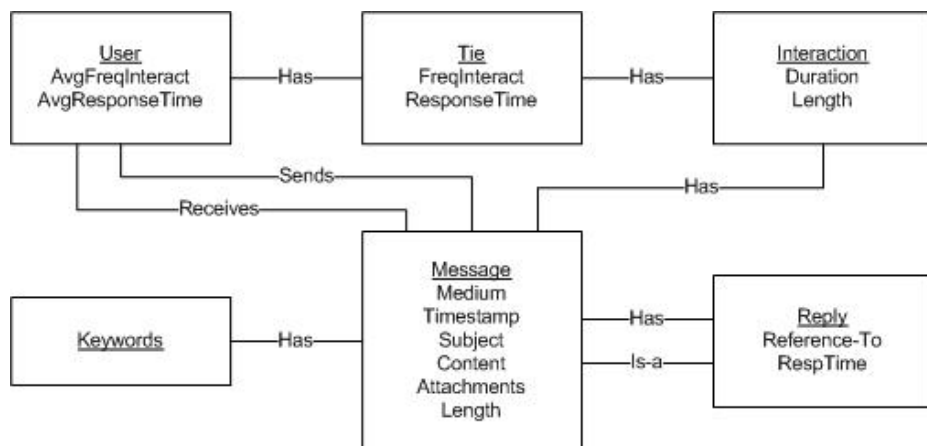


Fig. 1. Conceptual model of messages and interactions

likely result on serious information overload. We estimate only about 10-15 peers will be of interest at any given moment, depending on the groups and activities a user is engaged in.

In addition to the structural processing, the system performs content analysis on the messages, clustering them according to their topics as well as sender-recipient groups. Message bodies are processed for keywords and keyword vectors are built for each message. Interactions and ties are also classified according to the keywords found in the messages. This enables us to determine the themes of the interactions and defines the shared context for ego and each of his or her alters, which is later used for matching the group context to the individual tasks. We are also considering the use of concepts and activity ontologies to enrich the classification and matching.

The agent also keeps track of its user's activities, periodically extracting ongoing activity lists from the operating system, with application and file names. Textual files (pdfs, word documents) are processed for keywords in the same manner as messages, and compared to ongoing interactions. This ties work in progress to ongoing interactions, and should hopefully yield a relation between individual tasks and the social world a user is inserted in.

3.2 Determining Who: First Level Prune

The first level prune tries to answer the following questions: given the universe of user acquaintances, which ones would the user be interested in keeping track of? We focus on ongoing collaboration, as shared work often benefits from awareness.

Taking the values of the *From*, *To*, *CC* and *BCC* fields, a full list of acquaintances is built. Senders and recipients determine the working groups that form a user's focus of interest: individuals co-occurring in messages (e.g. multiple recipients) form the social worlds a user is part of. There are certain rhythms to work, and activity within a social world ranges according to the need. Thus, a social world may be very active for a certain period of time and slow down after a certain point (e.g., project completion or reaching a milestone). Therefore, the system must check for the formation of new

social worlds or change in activity patterns. We look for discrepancies between current behavior and “normal” behavior. Variables that currently characterize email exchanges are the number of messages exchanged (*Message Quantity*) and response time (*Response Time*). For each alter, we compare the current behavior to the normal behavior (the previously calculated average).

Given that patterns of email exchange usually emerge over a length of time, we are currently experimenting with different combinations of variables to determine the appearance of new working relationships. Intense message exchange usually accompanies ongoing collaboration, so Message Quantity is one of our qualifiers. We check if there are series of replies in a period of time shorter than the average, or whether there is an intensification of the exchanges (i.e., more messages are being exchanged than usual). Response Time should also be considered, as lower response time might mean a higher priority subject. We consider that social worlds in which the user is very active will be of more interest, with activity providing an indication of the focus of attention. It is important to note that social worlds are not defined only by a group of individuals, but also by the shared context that bring them together. This means that content analysis is needed to disambiguate interactions, defining the social worlds as a set of individuals with a shared theme, goal or project.

3.3 Second Level Prune: Determining What

After determining which social worlds are of interest to the user, other peers are queried for information about alters’ ongoing activities. The determination of which activities are of interest to the user will be done using keyword matching, comparing the contents of the interactions with the contents of the documents relating to the ongoing tasks, so that only activities related to ongoing interactions are shown to the user. Hopefully, the first level prune will significantly reduce the list of acquaintances and, consequently, the number of peers that need to be contacted and the amount of information that will be exchanged and processed in this stage.

Each agent periodically queries the operating system to elicit its user’s task list. It then analyzes the text relating to the tasks at hand to build keyword vectors to represent these. Our first approach is to build these using the TFiDF algorithm [27], which generates weighed keyword vectors given textual documents, and match these using the vector space model, where documents are matched using the cosine measure of proximity. Given that most of the activities under consideration are information processing tasks that involve a large amount of textual information (word processing, website surfing and searching, chat, etc.), this is a feasible approach, which should elicit activities that are related to previous conversations. Being established methods for information retrieval and matching, TFiDF and cosine measures have been extensively applied and tested, with good results. However, other text matching methods that may yield better results exist, and we will be experimenting with these.

In [3], a method for eliciting speech acts from email is presented and tested. It is based on a previously constructed taxonomy of speech acts applied to email (email-acts) describing verbs and nouns, with promising results. We hope to explore this approach as well, since it would provide better descriptions of activity information.

For now, we are keeping granularity coarse, picking only high level tasks. Thus, a user sees that an alter is editing a file they have exchanged, but not what paragraph or

text has been changed. We are working on more fine grained analysis and display to enable the user to “drill down” into the peers’ tasks to obtain more information.

4 Preliminary Analysis

As we construct the system, we chose to build intermediary versions that would allow us to work with some of the assumptions and get user feedback to adjust our algorithms and approach. The current implementation performs structural email parsing, extracting senders and recipients, building graphs (sociograms) and keeping count of messages exchanged between individuals. No content analysis is performed. We built an interface to display the corresponding sociograms, with which we can explore temporal boundaries, data sources and cutoff points. With this we can interview users regarding the social worlds and how they relate to ongoing work and awareness needs. In this fashion, we were able to perform a few preliminary analyses and get user feedback before proceeding with system implementation.

4.1 Working Assumptions

For these initial verifications, we were interested in working with four assumptions that underlie the system under construction. The first one is that social worlds are reflected in email. Thus, our first question was whether it was possible to identify social worlds through email based structural network analysis. What this means in practical terms is that cliques found in email-based social networks correspond to the different social worlds a user takes part in. If this is true, a system can infer working groups by identifying cliques in a graph (cliques are subgraphs where every node is connected to all others.) Our first assumption thus reads: *a clique represents a social world. For every clique in a sociogram, there will be a corresponding social world.*

Our second assumption is that activity patterns and social worlds change with time. These temporal patterns reflect the rhythms of group activity, from inception to project completion. By slicing the data into different timeslots, different social worlds should become apparent. This characterizes the changing patterns of collaboration a user typically engages in. The social worlds become stronger or fade away depending on the project dynamics. If this is true, by keeping track of these patterns, a system should be capable of adjusting the awareness needs of its user. We question whether *given different timeslots, different social worlds will become active; and if given a social world, it is possible to identify a pattern of intensification and decay in message exchange that corresponds to the activity in that social world.*

In [30], it is suggested that contents of the Outbox are more important than the contents of the Inbox in this type of analysis, since they reflect interactions the user has actually decided to engage in. Should this be true, the number of email messages to be processed and the resulting network would become considerably smaller, significantly speeding up processing time. However, Inbox contents cannot be completely discarded, since they contain valuable interaction information. Our third assumption is that a series of email messages is relevant only if a user has sent messages as well as received. It should be possible to construct a sufficiently elaborate social network to represent collaboration based on the interactions found in the user’s outbox, discarding

messages from the user’s inbox which have not been replied to. We assume that *if a message belongs to an interaction in which the user has not taken part, then it is of little importance to the user (it can be discarded)*.

We wanted to reflect on two additional points: the first is whether short timeslots are significant for the identification of collaboration. In [24], a 15 year email log was analyzed, with data aggregated into 1-year slices. To be useful for the distribution of awareness information, this time frame needs to be significantly reduced (to days or weeks), since we are interested in collaboration at the moment it is happening. Thus our visualization tool allows us to slice time arbitrarily and check what sorts of patterns become visible, and if shorter periods (e.g. a month or two out of a 4-year email log) will display the same patterns as the ones found in the one year time slices. The second point we want to reflect on is the identification of thresholds. That is, how different does behavior have to be to be considered relevant to awareness needs? How many messages should be exchanged and how low a response time must be observed to characterize collaboration? This would help us determine how to detect ongoing collaboration on the fly. We would also like to verify how useful *Message Quantity* is as a qualifier and how well it ties into the determination of awareness needs.

To verify our assumptions, we built an interface that allows us to visualize data, slicing it into different timeslots and sources, shown in Figure 2. It implements a spring-embedded graph layout, using the Fruchterman-Rheingold force model [10],

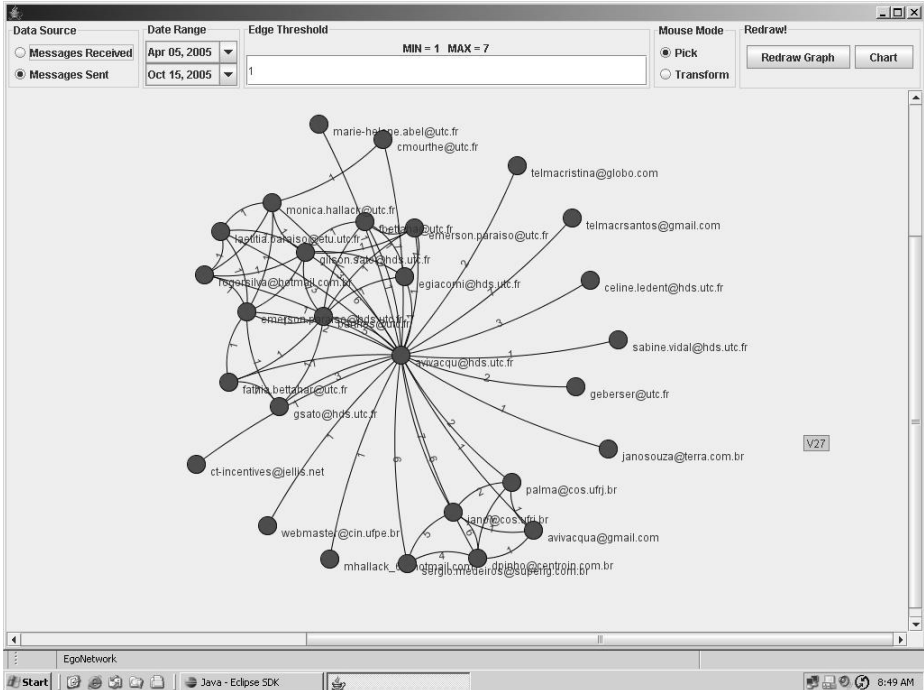


Fig. 2. Visualization screenshot, where several cliques are visible

which treats a graph a set of nodes that repel each other connected by springs which attract them. The resulting graph reflects node proximity while minimizing line crossings. The visualization was built using the Java language and the JUNG library for graph construction and display. While all the aforementioned points have not been fully addressed yet, we are keeping them in mind: our current version allows us to slice time as desired, but new visualizations are needed to help analyze the data.

We processed 4 users' email histories and asked them questions regarding the resulting sociograms. We chose our users based on the fact that they were all heavy email users (with several thousand email messages in their mailboxes), but had different profiles, and we expected the data to follow different patterns. Our users came from different backgrounds, and the data they brought with them reflected as much: 2 were full time students, with many short projects and collaborations and a few longer collaborations with other students; 1 was a professor with several short and long term collaborations, some requiring close control, some not and 1 was a navy officer, with long and medium term projects requiring control and coordination. We explored the sociograms with our users, slicing the data in different ways. We took the opportunity to ask whether they believed that some sort of additional awareness information might have been beneficial to the work in progress. We asked users if:

1. the cliques they identified in their sociograms were related to projects or other collaborations going on at that time;
2. different cliques became active when the temporal range was changed;
3. patterns of message exchange reflected projects;
4. the social worlds in which the user had not participated (other than as an "observer") were of interest as far as peer task awareness.

4.2 Verification and Analysis

When asked, our users were capable of relating social worlds to the cliques that showed up in their sociograms. However, not all of these social worlds were related to ongoing work. There were situations where a clique represented a group that shared some sort of context (e.g., students in the same department), but were not in direct collaboration. There were large amounts of group emails exchanged for information only, but no actual collaboration going on. Thus, we confirm that it is possible to identify social worlds from cliques, but it doesn't follow that all these represent joint projects. Further investigation is necessary to determine how to differentiate between work and non-work messaging.

With changes in time slots, different groups became active, showing up on the visualization. It must be noted that, since the data was historical and cumulative, the social worlds don't actually disappear, they become more or less active (and a user became more or less active within them) according to the situation. While the full view was somewhat cluttered, slicing it to shorter periods considerably reduced the number of messages, making it easier to identify different subgroups. This confirms that social worlds come and go, which is reflected on email. Inspecting the temporal graph seen in Figure 3 (where time is sliced into daily email exchanges), we could easily see changes in interaction pattern. A dormant relationship suddenly springs to life, with emails being exchanged daily (sometimes several messages a day,

depending on the urgency), and then dies out as abruptly when deadlines are reached. This confirms our second assumption, and is particularly interesting since we were able to explore considerably shorter periods than those presented in [24] and still detect collaboration. However, changes were often quite abrupt, going from no interaction to 4 messages a day overnight. While we expected this to happen, we also hoped to see softer patterns, where interactions would gradually increase with time.

When asked, users said that social worlds in which they did not actively participate were not of much interest to them (as far as task awareness). Users wanted to be aware of their closer, more immediate collaborators, where there was a lot of coordination to be done. They had no desire to be aware of everybody's work, although in some cases they would like to remain superficially aware of what was going on. This indicates that, for task awareness purposes, we can leave out all incoming threads in which the user has not participated. In computational terms, this significantly reduces graph size, and, consequently, memory needed and computation time.

Within the emails, there were several instances of project-related social worlds, usually qualified by intense interaction in a shorter period of time (weeks or a few months). This suggests a way of more effectively picking activity-related groups. However, when inspecting the data, it became apparent that structure alone was not sufficient to tease these apart, especially when there were overlapping social worlds. These needed to be qualified according to the activities or themes of the interactions, so that they could be effectively set apart. There were quite a few overlapping social worlds (including temporal overlaps, where a group works together on more than one project at the same time). Within our data sets, there were also several social worlds embedded in other social worlds. Large groups who perhaps work in a same building and smaller subgroups who work closely together. While a user will probably not be interested in keeping close track of the activities of members of the larger group, he or she may want to have periodic summaries or reports on how work has been progressing. This leads us to think of awareness as a continuum, with awareness needs tied to a user's participation in a group. The user might desire to have more or less information (depth and frequency) about others, depending on his or her level of involvement with the group. We are considering the use of artificial intelligence techniques such as fuzzy sets to better represent a user's focus, and how much information he or she would want to have.



Fig. 3. Time based interaction graph

Message Quantity was a reasonable qualifier when looking at a user's outbox, but not at all when inspecting an inbox. Some alters sent 200+ messages over a 6-month period and were neither collaborators nor of any interest to the user. A user's outgoing

messages, however, seemed to reflect the social worlds a user engaged in more accurately. Participation in conversations involves an investment of time and effort that indicates a certain level of interest and commitment to the group. Accordingly, we are changing our message processing algorithms to process outgoing messages first, building social worlds and then trying to fit incoming messages into these.

Users often engaged in animated discussions which were not work related. While this does denote a certain level of interest in the subject (our users were actually interested in what was going on), it doesn't mean the user would want to keep track of others' work. For instance, one of our users had quite a few discussions with a group of friends regarding TV Series, politics and movies. While the content of these interactions would not match any ongoing tasks, it might still be a costly false positive, which increases the search space. We are refining our algorithms to disregard these threads from the start. One possibility would be to perform an initial match with the user's own work to see if the user was actually working on the subject.

5 Discussion and Future Work

Lack of space precludes a lengthy discussion of related systems. Messengers in general have been widely adopted and have become a frequent means of communication. Most provide ways for a user to express whether he or she is available, busy or "out for lunch", passing that information on to their peers. A few more complex task awareness approaches exist: for instance, MultiVNC [15] displays miniatures of peers' desktops in order to improve awareness in a working group and increase collaboration. It doesn't filter or verify what is actually of interest to the user, and interface is quite busy. Community Bar [20] allows users to specify what peers they want to be aware of, organizing them into social worlds. Their focus is on media items (webcam shot, calendar, post-it, chat software, etc.), not content, so users tell the system what media they want keep track of and the user is left to sift through the information contained therein and decide which are valuable. Doc2U [22] is a shared editing environment where information about who is editing which parts of a shared document is distributed among peers. It requires logging in to a shared environment and is only applicable to one activity. A number of agent-based systems have been created to provide information that the user might otherwise not have had. Many also deal with the problem of the overwhelmingly large amount of information available at any given moment by sorting out what is useful to the user at the time [17].

A number of systems to classify emails into activities have recently been developed [4], and activity modeling has been growing as a research area. Unified Activity Modeling, for instance, proposes a generic model of activity and a framework to integrate individual, informal, work with more strict organizational workflows [21]. This research seeks to help users organize and contextualize emails within activities, and might be useful in our context as well: through an accurate classification of emails into tasks, it becomes easier to determine the activities within a social world, which should then lead us to appropriate information dissemination.

A method to construct networks of people and keywords and for discovery of people with similar interests is presented in [19]. It mines email data and constructs networks of people, which can later be used to determine who has knowledge on what

topics, displaying the networks for user inspection. Another approach for social network use is presented in [12], where the author uses networks to locate individuals with a certain expertise and availability through an analysis of their activities and tasks. In the aforementioned approaches, the emphasis is on finding experts, and navigating the social network to create an awareness of who knows what. Our interest, on the other hand, is quite distinct: we aim to identify working groups and to provide activity awareness information only of these individual, as it happens. It is not meant as a system for group formation or expertise location, but as tool to assist coordination and collective action. Our intent is to use Social Networks as an active way for making inferences, monitoring and influencing collaboration, as suggested in [18]. Our emphasis is not in the display and visualization of the networks, but in what patterns can be found and what calculations and inferences can be made. In [6], a series of patterns and work rhythms are presented, we plan on building on this work to determine what these mean in terms of information needs and distribution.

Our system is currently under implementation, and at this point, this approach seems promising: it provides a way to explore awareness needs of individuals in relation to their ongoing collaborations. Email-based analysis can elicit interaction patterns that denote role attribution or the organization of a team. We expect these will have different information needs (e.g., core vs. periphery members differ in terms of nature, quantity and depth of the information needed), and further research is needed.

The system is being built using the Java language, with several specific open source libraries: so far, JUNG has been used for graph construction and display and JFreeChart for the time charts. For the following phases, we plan on using Java and JNI to monitor users' ongoing tasks (this information can be obtained directly from the Windows operating systems APIs) and JACOB, a Java library to interface with COM automation (present in all office applications and many others), to communicate with Windows-based applications. Emails are stored in an Access database.

When managing a few thousand emails, the system becomes a bit slow especially when drawing, something we are trying to work around. It currently reads Eudora mailboxes, but we are already checking on other possibilities, such as reading directly from the server. Another difficulty was dealing with raw data: in general, our users' email files were fairly disorganized and sometimes contained duplicate messages. Additionally, several individuals had more than one email address, which means they must be organized into personas so that the data makes more sense.

We will continue to explore the interplay between interaction and awareness needs. Even though our preliminary analysis was small, with only a few subjects, it indicates some directions for further research: to develop a more complete mapping between interaction levels and awareness needs, other variables need to be taken into account, such as response time and content. New experiments need to be designed, with more users and different emphasis, so that other information can be gleaned from the data. One of our next activities will be a controlled experiment to check on the effects of different types of information at different moments.

5.1 Privacy Issues

Whenever information is automatically collected or distributed, privacy becomes an issue. The automatic management of a user's nimbus is an open issue at this point,

although we are experimenting with network based calculations for that as well. For the time being, we leave the choice of what to make available to the user. We are adopting a three-tiered privacy scheme, where a user can define whether a task is public (all can see), protected (some can see) or private (none can see). The user will be able to determine alters, keywords, or resources that fall within each of the tiers, and who has access to what in the protected level. When a task is found that should be propagated to other peers, it is checked against the specified restrictions to see if it falls within a specific privacy tier and whether it can be sent to the requesting agent.

We are currently fitting users' activities into one of the following categories: manipulation of shared objects, manipulation of non-shared objects and chat between members. We are working with the assumption that all shared objects and interactions within a social world should be made public to members of that social world. For instance, editing or forwarding a file that has been sent around as an attachment, or chat related to the project between members of the social world. Manipulation of non-shared objects is a more complex case. For our initial prototype, we prefer to err on the side of caution and block all non-shared material. These simple heuristics should help us decide on whether to send information around until a better privacy scheme is in place. Upon reflection, this transparency might compromise the capability of political articulation within a group, so we expect some reaction from users.

When we look beyond organizational structures, protocols and hierarchies, modern organizations are composed of networks of interacting actors [1]. More often than not, knowledge is exchanged and work is undertaken through these informal relations between workers, in networks that cut across departmental, functional and organizational boundaries. Thus, modern organizations require coordination and integration of activities across these boundaries, and information systems should provide support for distributed coordination and decision-making.

In this paper we have presented an approach to the determination of awareness foci based on egocentric email-based social network analysis. We believe this is a promising line of research that holds many possibilities for further work. Many studies have applied social network analysis to uncover relations between people and patterns of interaction, but few have used these patterns as a basis for a system to actively assist the user, choosing only to display this information.

Acknowledgements

This work was partially supported by CAPES and CNPq.

References

1. Bernoux, P. *La Sociologie des Entreprises*. Éditions du Seuil, Paris (1999)
2. Carstensen, P., Schmidt, K. *Self Governing Production Groups: Towards Requirements for IT Support*. In 5th IFIP Int. Conf. on Information Technology in Manufacturing and Services (BASYS'02), Cancun, Mexico, Kluwer Academic Publishers (2002) 49-60
3. Cohen, W.W., Carvalho, V.R., Mitchell, T.M. *Learning to Classify Email into Speech Acts*. In Proc. Conf. on Empirical Methods in Natural Language Processing (2004)

4. Dredze, M. Lau, T., Kushmerick, N. Automatically Classifying Email into Activities. In Proceedings of Intelligence User Interfaces (IUI 06), Sydney, Australia (2006)
5. Edwards, K., Mynatt, E. Timewarp: Techniques for Autonomous Collaboration. In Proc. CHI 1997, Atlanta, GA (1997)
6. Fisher, D., Dourish, P. Social and Temporal Structures in Everyday Collaboration. In Proc. CHI 2004, ACM Press (2004) 551-558
7. Fitzpatrick, G., Kaplan, S., Mansfield, T. Applying the Locales Framework to Understanding and Designing. In Proc. OzCHI 98. Australia, IEEE (1998) 122-129
8. Fitzpatrick, G. The Locales Framework: Understanding and Designing for Cooperative Work. PhD Thesis, University of Queensland (1998)
9. Fitzpatrick, G., Tolone, W., Kaplan, S. Work, Locales and Distributed Social Worlds. In Proc ECSCW 95. Stockholm, Sweden, Kluwer Academic Publishers (1995) 1-16.
10. Fruchterman, T., Reingold, E. Graph drawing by force-directed placement. *Software: Practice and Experience*, 21(11), John Wiley and Sons (1991) 1129-1164
11. Greenberg, S., Johnson, B. Studying Awareness in Contact Facilitation. In CHI 97 Workshop on Awareness and Collaborative Systems. Atlanta Georgia (1997)
12. Groth, K. Using Social Networks for Knowledge Management. In Proc ECSCW 03 Workshop on Moving From Analysis to Design: Social Networks in the CSCW Context. Helsinki, Finland (2003)
13. Gutwin, C. Greenberg, S. The Importance of Awareness for Team Cognition in Distributed Collaboration. In *Team Cognition: Understanding the Factors that Drive Process and Performance*. Washington, APA Press (2004) 177-201
14. Gutwin, C., Greenberg, S. A Descriptive Framework of Workspace Awareness for Real-Time Groupware. In *Computer Supported Cooperative Work 11*. Kluwer Academic Publishers, Netherlands (2002) 411-446
15. Gutwin, C., Greenberg, S., Blum, R. Dyck, J. Supporting Informal Collaboration in Shared-Workspace Groupware. The Interaction Lab Technical Report HCI-TR-2005-01, University of Saskatchewan, Canada (2005)
16. Hertel, G., Geister, S., Konradt, U. Managing Virtual Teams: A review of current empirical research. In *Human Resource Management Review* 15. Elsevier (2005), 69-95
17. Maes, P. Agents that Reduce Work and Information Overload. In *Communications of the ACM Vol. 37, No. 7*. ACM Press (1994) 31-40
18. Martinez, A., Dimitriadis, Y. Tardajos, J., Velloso, O., Villacorta, M. Integration of SNA in a Mixed Evaluation Approach for the Study of Participatory Aspects of Collaboration. In Proc ECSCW 03 Workshop on Moving From Analysis to Design: Social Networks in the CSCW Context. Helsinki, Finland (2003)
19. McArthur, R. Bruza, P. Discovery of Social Networks and Knowledge in Social Networks by Analysis of Email Utterances. In Proc ECSCW 03 Workshop on Moving From Analysis to Design: Social Networks in the CSCW Context. Helsinki, Finland (2003)
20. McEwan, G., Greenberg, S. Community Bar: Designing for Awareness and Interaction. In ACM CHI Workshop on Awareness Systems: Known Results, Theory, Concepts and Future Challenges (2005)
21. Moran, T.P. Unified Activity Management: Explicitly Representing Activity in Work Support Systems. In Proc. ECSCW 05 Workshop on Activity: From a Theoretical to a Computational Construct. Paris (2005)
22. Morán, A. L., Favela, J., Martínez-Enríquez, A. M., Decouchant, D. 2002. Before Getting There: Potential and Actual Collaboration. In Proc. CRIWG 02 Springer-Verlag (2002)
23. Narine, T. Leganchuk, A., Mantei, M., Buxton, W. Collaboration Awareness and its use to consolidate a Disperse Group. Proc. of Interact 97 Sydney Australia (1997)

24. Perer, A., Shneiderman, B., Oard, D.W. Using Rhythms of Relationships to Understand Email Archives. In *Email Archives Visualization Workshop*
25. Pinelle, D., Gutwin, C. A Groupware Design Framework for Loosely Coupled Groups. In *Proc. ECSCW 05, Paris, France (2005)*
26. Rodden, T. Populating the Application: A Model of Awareness for Cooperative Applications. In *Proc. CSCW 96, ACM Press (1996) 87-96*
27. Salton, G. *Automatic Text Processing: the Transformation, Analysis and Retrieval of Information by Computer*. Addison-Wesley Publishing (1988)
28. Schmidt, K., Bannon, L. Taking CSCW Seriously: Supporting Articulation Work. In *Computer Supported Cooperative Work 1, vol. 1. Kluwer Academic Publishers, Netherlands (1992) 7-40*
29. Scott, J. *Social Network Analysis: A Handbook*. Sage Publication, London (1991)
30. Tyler, J., Tang, J. When can I expect an Email Response? A Study of Rhythms in Email Usage. In *Proc. ECSCW 03 (2003)*
31. Wasserman, S., Faust, K. *Social Network Analysis: Methods and Applications*. Cambridge University Press, Cambridge US (1994)

Cooperation Indexes to Support Workspace Awareness

Benoît Otjacques¹, Monique Noirhomme², Xavier Gobert¹, and Fernand Feltz¹

¹ Public Research Center – Gabriel Lippmann
Department ISC – Informatics, Systems and Collaboration
41, Rue du Brill
L-4422 Belvaux, Luxembourg
otjacque@lippmann.lu, gobert@lippmann.lu, feltz@lippmann.lu
² University of Namur
Computer Science Institute
21, Rue Grangagnage
B-5000 Namur, Belgium
monique.noirhomme@info.fundp.ac.be

Abstract. Awareness is now acknowledged in the CSCW domain as an important element to take into account. This paper explores and refines the concept of workspace awareness. A categorization is proposed based on two criteria: the granularity and the ability to identify the cooperating entities. Two concepts are defined on this basis: Workspace Individual Awareness (WIA) and Workspace Global Awareness (WGA). New kinds of metrics are then described to support WGA and a prototype implementing WGA is shortly discussed.

1 Introduction

Prior research in Computer-Supported Cooperative Work (CSCW) has highlighted the importance for cooperating people to be kept aware of the others presence and activity. Nevertheless, some research is still needed concerning what kind of data to communicate to the users as well as when and how to notify them. This paper aims to contribute to this field of research by exploring the concept of workspace awareness.

2 Prior Works on Awareness

Prior work in the field of CSCW stresses the importance of awareness. Dourish and Belloti [7] introduced the concept of awareness that they defined as ‘an understanding of the activities of others, which provides a context for your own activity’. Gutwin and Greenberg [9] explain that ‘it is becoming more and more apparent that being able to stay aware of others plays an important role in the fluidity and naturalness of collaboration’. Awareness can take several forms: informal awareness, social awareness, group-structural awareness and workspace awareness [8]. In this paper, we focus on workspace awareness, which has been defined [10] as ‘the up-to-the-moment understanding of another person’s interaction with the shared workspace. [...] It includes awareness of people, how they interact with the workspace, and the events happening within the workspace’.

Considering the importance of awareness, it is worth exploring which techniques can be used to support it. Several solutions have been developed like update notifications via email or displays of the number of people reading the same webpage [15]. Other methods make use of graphics such as: icons indicating who is online, avatars [11], representing users as abstract shapes [6], visualisations on mobile phones [1], or representing the other users' actions in virtual environment [12]. We can also observe that the awareness cues can be embedded within user interface of the CSCW application (e.g. [14]) or directly displayed in the operating system interface (e.g. [5]).

3 Notification and Awareness: Exploring the Concepts

3.1 Model of Cooperation

From a general point of view, our reflection relies on modeling the cooperation as a set of *interactions* among *entities*. In this context, an entity can be a person or a resource (e.g. document, web page in a collaborative platform, group agenda, shared object in virtual design environment...) and an interaction is defined as '*any exchange of information among entities*' (e.g. phone call, electronic message, download of a document from a web site, posting news on a web page...). The creation of an interaction by an entity is called '*interaction instantiation*' and an interaction can *target* one or many entities. This definition of '*interaction*' encompasses all kinds of interactions (e.g. electronic vs. physical, synchronous vs. asynchronous, co-located vs. distant...). In order to restrict the concept, we introduce the notion of '*computed-mediated interaction*' (CMI), defined as '*any electronic exchange of information among entities*'.

3.2 Workspace Awareness: Some Refinements

Workspace awareness refers to the notification of the workspace interactions to the persons taking part to the collaboration. In this paper, the user that is kept aware of workplace events is called 'reference user' (U_R). The techniques to communicate workplace awareness to U_R can take various forms. We have categorized them according to two axes: identification and granularity.

Identification specifies whether the awareness information provided to user U_R allows identifying the related entities. Two cases are possible: *identified* and *anonymous*. Identified workspace awareness information relies on monitoring some specific entities and notifying to U_R the interactions concerning these entities so that U_R can identify the involved entities. The notified interactions include both those initiated by a monitored entity (e.g. informing U_R when a user U_A connects to a chat system) and those targeting a monitored entity (e.g. keeping U_R aware of the modifications of a shared document D_i). Anonymous workspace awareness information gives the user U_R some feedback on what happens in the collaborative environment but does not explicitly indicate which entities are involved. For instance, a specific icon is displayed when the user receives a message but this icon does not indicate who is the sender. Similarly, some graphics change their colour when some news has been posted on the collaborative platform but nothing is said about the author of the news.

Granularity is the second categorization axis. It specifies whether the notified information concerns specific *individual* entities within the workspace (e.g. notification

via email of an update of document D_i , icon associated to the presence status of a user U_A) or is *aggregated* (e.g. number of users connected to the workspace displayed but not their names, the number of new documents uploaded since U_R 's last visit shown without providing any details about those documents).

For privacy, security or information overflow reasons, only a subset of all the interactions occurring in the workspace can be notified to U_R . Moreover, the subset of notified interactions differs according to the type of awareness. Identified / individual information is more sensible than anonymous / aggregated data. Therefore, the subset of notified anonymous aggregated information can be very large and can potentially include all interactions in the workspace. Contrariwise, the set of identified individual data that can be notified to U_R is much more limited.

As table 1 shows, four cases are theoretically possible. Nevertheless, it appears that not all cases are as useful in practice. Communicating identified individual and anonymous aggregated information to support workspace awareness is common, the two others cases are quite unusual.

Table 1. Categorization of workspace awareness information

| Granularity | Identification | |
|-------------|--|--|
| | identified | anonymous |
| individual | <ul style="list-style-type: none"> - user U_A is online - document D_i has been modified | <ul style="list-style-type: none"> - unlabelled graph |
| aggregated | <ul style="list-style-type: none"> - some of these users have sent e-mails to U_R | <ul style="list-style-type: none"> - some users are connected to this web page - some new documents have been uploaded |

Anonymous individual information refers to the notification to U_R that a specific entity E_A has been involved in interactions without specifying which entity it is. Usually, this kind of information has little value. Displaying the interactions occurring among some users as a node-link graph without any labels telling whom the nodes refer to is an example of this case.

Providing aggregated identified awareness information can also be questioned. Indeed, this would mean notifying U_R that some identified entities are involved in some interactions without specifying which entity is concerned by which interaction (e.g. informing U_R that some of the users U_A , U_B and U_C have posted news and some of them have sent e-mails without specifying what each of them has done). In general, this kind of information is either too detailed or too simplified.

To sum up, it appears that identified individual and anonymous aggregated information are the most adequate to support workspace awareness. Therefore, we refine the workplace awareness concept in two sub-concepts referring to the two cases found to be the most useful. *Workspace Individual Awareness (WIA)* is associated to the communication to U_R of identified individual workspace awareness information. *Workspace Global Awareness (WGA)* aims to convey global information about the collaborative environment via anonymous aggregated information. Usually CSCW tools offer some means to switch between features supporting the different types of

workspace awareness. For instance, the graphics rendering WGA (e.g. single icon showing that three users are connected) can be used as entry point to access WIA representations (e.g. three pictures of the connected users).

In this paper, we have chosen to focus on Workspace Global Awareness, especially because it has been less explored than WIA (cf. [3], [6], [11], [12] for examples of tools supporting WIA). In fact, maintaining WGA requires notifying U_R some global information about the activities within the workspace. We have thus studied how to design global metrics able to provide a general feeling on the cooperation.

4 Metrics of Cooperation

4.1 Prior Works

Evaluating the degree of cooperation within a group is a complex task due to issues associated to (among others) the design of meaningful metrics, the collection of data required to compute these metrics and the interpretation of the resulting values. Nevertheless, some researchers have proposed such indicators.

The simplest methods are founded on calculating the number of occurrences of some types of interactions in the group. For instance, Prinz et al. [14] report that the stream of emails in a project can be interpreted as indicating the level of activity. The ‘*Participameter*’ widget [4] is another approach based on the relative level of participation of each member within a group discussion. Displaying the number of online, unavailable and offline users has also been included in some applications to support awareness [3]. The *Community Toolbar* software [13] is a relatively advanced solution that allows U_R choosing which awareness data he wants to be kept informed of, like the total number of users currently present in all communities of which U_R is member, the number of users currently visiting a given community or the number of users visiting the same web page.

Some researchers have tried to design richer metrics based on more complex processing of the rough data. For instance, the Social Network Analysis (SNA) theory, relying on the similarity of the graph theory and the structure of the relationships within a group, explores the social meaning of mathematical properties of the graph of relations (e.g. centrality). Some SNA concepts have been used to measure and visualize how active students are in a class [16]. Barros and Verdejo [2] adopt another perspective. They use a chained inference process to derive high-level metrics (e.g. level of cooperation) from quantitative but also qualitative description of the contributions of the members of a group in a collaborative learning context.

4.2 Proposal of New Metrics

At this stage of the reflection, we face the challenging issue of designing a metric representative of the global level of electronic cooperation within a group. We will call it *Glocoopex* (*Global electronic Cooperation Composite Index*). In addition, we want to design an index that can be automatically computed in order to avoid the time and cost issues caused by manual data collection methodologies. The basic idea consists in collecting quantitative data that can be enriched by qualitative indicators automatically derived from generic rules.

We consider that the level of cooperation within a group is composed of two basic elements: the number of interactions and the cooperative nature of these interactions. Consequently, *Glocoopex* is built by combining two simpler indexes: *Coopadex* (*electronic Cooperation Activity Composite Index*) which is representative of the number of interactions and *Coopidex* (*electronic Cooperation Interest Composite Index*) which render the cooperative nature of the interactions.

The *Coopadex* index is intended to be representative of the mean level of use of electronic tools supporting the cooperation. In this context, we consider the mean number of computer-mediated interactions by person by period of time as a reliable indicator of the volume of cooperation. *Coopadex* is therefore defined by the formula:

$$Coopadex = \frac{\sum_j (I(G_j))}{n \Delta T}$$

where G is a group of n persons in a situation of cooperation, G_j is the j^{th} member of the group G , $I(G_j)$ is the number of CMI in which G_j is involved, and ΔT is the duration of the period under examination.

Note that n is the total number of G members and not the number of G members that are involved in some CMI during the period. Moreover, the formula shows that an interaction initiated by G_j and targeting G_k and G_m counts as three interactions in the *Coopadex* value. This approach has been adopted to give an increasing weight to the interactions as the number of persons they involve raises.

Coopadex reflects the mean rate of use of the electronic cooperation tools but it does not render the level of interest or motivation of the persons towards the cooperation, which is the role of *Coopidex*. It is founded on the observation that some interactions express a higher motivation to cooperate than other ones. For instance, automatically generated notification e-mails, explicit invitations in a shared electronic calendar and active use of a co-authoring system can reasonably be associated to increasing levels of motivation for IT-supported cooperation. At this point, we face then the challenge of evaluating the cooperative nature of CMI. *Coopidex* tackles this issue by combining two properties: whether CMI are optional or imposed on one hand, and active or passive on the other hand.

The optional vs. imposed nature of the interactions is taken into account by introducing two coefficients: α_o (optional) and α_i (imposed). These coefficients take a value in the range $[0,1]$. In addition, we state that optional interactions reflects in average a higher motivation to cooperate than imposed ones because they imply a deliberate choice of the involved users. This statement imposes that $\alpha_o > \alpha_i$.

The active or passive nature of CMI is the second property considered in *Coopidex*. An active interaction is defined as an interaction that occurs due to an explicit action of the concerned entity (e.g. sending an e-mail). In contrast, a passive interaction does not require such an explicit action (e.g. receiving an invitation to a meeting). We introduce two additional coefficients: β_a (active interaction) and β_p (passive interaction). These coefficients also take a value in the range $[0,1]$. We state that active interactions are a sign of higher interest in cooperation than passive ones due to the higher work load that they imply. This condition imposes that $\beta_a > \beta_p$.

Any interaction now falls in one of the four following categories: active-optional, passive-optional, active-imposed and passive-imposed. We can then sum up the number

of CMIs in each category: $I_{a,o}$: number of active optional CMIs; $I_{p,o}$: number of passive optional CMIs; $I_{a,i}$: number of active imposed CMIs; $I_{p,i}$: number of passive imposed CMIs. The *Coopindex* can then be computed by the expression:

$$Coopindex = \frac{\alpha_o \beta_a I_{a,o} + \alpha_o \beta_p I_{p,o} + \alpha_i \beta_a I_{a,i} + \alpha_i \beta_p I_{p,i}}{I_{a,o} + I_{p,o} + I_{a,i} + I_{p,i}}$$

The *Coopindex* value varies between the low limit ($\alpha_i \beta_p$) and the high limit ($\alpha_o \beta_a$). However, this value may be difficult to interpret as it demands to know the value of the weights: α_i , α_o , β_a et β_p . In order to make it easier to understand, we normalize *Coopindex* with a projection on the interval [0,1]. This operation defines a new index, called *N-Coopindex* (*Normalized electronic Cooperation Implication Composite Index*) computed by the expression:

$$N-Coopindex = \frac{(Coopindex - \alpha_i \beta_p)}{(\alpha_o \beta_a - \alpha_i \beta_p)}$$

N-Coopindex varies between the value '0' (when all interactions are passive and imposed) and the value '1' (when all interactions are active and optional). The higher the *N-Coopindex* value, the more the members of the group take the initiative to instantiate some unforced CMIs and the higher the interest for IT-supported cooperation within the group. For each interaction, the parameters values (α_o , α_i , β_a , β_p) are set according to general rules (R_i) defined by the researcher. These rules specify how the categorization of the interactions depends on factors like the communication medium (e.g. *e-mail, shared calendar, web page with news...*), the communication direction (e.g. *send-receive, upload-download...*) and potentially other elements (e.g. *the hierarchical position of the interaction initiator vs. the targeted entities...*). Designing the set of rules R_i demands a good preliminary knowledge of the cooperative situation but once they are specified the indexes may be automatically computed.

Glocoopex index is defined as the product of *Coopindex* and *N-Coopindex*:

$$Glocoopex = \frac{[(\alpha_o \beta_a - \alpha_i \beta_p) I_{a,o} + \beta_p (\alpha_o - \alpha_i) I_{p,o} + \alpha_i (\beta_a - \beta_p) I_{a,i}]}{n \Delta T (\alpha_o \beta_a - \alpha_i \beta_p)}$$

N-Coopindex is a factor that renders the average degree to which the interactions are close to the best ones in terms of motivation to cooperate electronically. Multiplying *N-Coopindex* by *Coopindex* provides the mean number of interactions having the average level of motivation for IT-based cooperation. In order to convey this idea, we introduce the concept of *Equivalent High Quality Interactions* (EHQI). It renders the idea that the number of interactions can be expressed in terms of theoretically best ones. The units of *Glocoopex* are then EHQI by person by time unit.

We have also calculated the *Glocoopex* sensitivity to the different variables: n , ΔT , $I_{p,i}$, $I_{a,o}$, $I_{p,o}$, $I_{a,i}$. The most important results are discussed hereafter.

Glocoopex does not depend on the variable $I_{p,i}$. This result is not really a problem as it means that the involvement of the group members in passive imposed CMIs is not considered as an indicator of a significant level of cooperative activity.

Considering the constraints set on the weights α_o , α_i , β_a , β_p , *Glocoopex* is always less sensitive to a variation of the number of passive optional CMIs ($I_{p,o}$) than to a variation of the number of active optional CMIs ($I_{a,o}$). Similarly, it can be shown that *Glocoopex* is always less sensitive to a variation of the number of active imposed CMIs ($I_{a,i}$) than to a variation of the number of active optional CMIs ($I_{a,o}$).

Concerning the relative sensitivity of *Glocoopex* to some variations of $I_{p,o}$ and $I_{a,i}$, we hypothesize that the most important indicator of the interest in IT-based cooperation is the choice of the persons to participate. From our perspective, this means that the optional/imposed factor must influence more the value of *Glocoopex* than the active/passive one. This statement implies that passive optional CMIs are a sign of greater motivation than the active imposed ones: $\alpha_o \beta_p > \alpha_i \beta_a$.

5 Prototype

The theoretical constructs discussed in the previous section have been implemented in a prototype that aims to support WGA for the identified users of a web-based collaborative platform. Conceptually, the prototype regroups three main components. First, a specific data model based on the concepts of entities and interactions has been designed. This database is supplied by triggers and stored procedures that collect rough data from the collaborative platform and transform it according to our model. Second, some modules calculate and store the indexes values. Third, the indexes values are transformed in visual representations communicated to U_R . As peripheral awareness seems adequate to support WGA, we have chosen an easy-to-understand metaphor based on flags of which size and color are associated to the indexes values. Flags are displayed directly on the desktop, using the Active Desktop feature of MS Windows.

6 Conclusion

This paper explores the workspace awareness concept and refines it by introducing two new notions: Workspace Individual Awareness and Workspace Global Awareness. The paper focuses on the latter and proposes three indexes to convey global information about a cooperative situation. They are quite original as they combine quantitative and qualitative data about the cooperative activities, which is not so common. Nevertheless, further works are still needed from different perspectives. Comprehensive comparative studies of WGA and WIA implementations could enhance the understanding of workspace awareness. New features supporting WGA in electronic cooperation could also be designed and assessed in different contexts. The relevance to include other properties of the interactions in the indexes to better reflect the reality of the cooperation could also be investigated.

References

1. Bardram, J.E. and Hansen T.R. The AWARE Architecture: Supporting Context-Mediated Social Awareness in Mobile Cooperation. In *Proceeding of the ACM Conference on Computer-Supported Cooperative Work (CSCW'04)* (Chicago, Illinois, November 6-10, 2004). 192-201.

2. Barros, B. and Verdejo, M.F. An Approach to analyze collaboration when shared structured workspaces are used for carrying out group learning processes. In *Proceedings of the 9th International Conference on Artificial Intelligence in Education (AIED'99)* (Le Mans, France, July, 19-23, 1999). 449-456.
3. Begole, J., Tang, J., and Hill R. Rhythm Modeling, Visualizations and Applications. In *Proceedings of the 16th annual ACM symposium on User interface software and technology (UIST'03)* (Vancouver, Canada, November 2-5, 2003) 11-20.
4. Borges, M.R.S. and Pino, J.A. Awareness Mechanisms for Coordination in Asynchronous CSCW. In *Proceedings of the 9th Workshop on Information Technologies and Systems (WITS'99)* (Charlotte, North Carolina, December 11-12, 1999). 69-74.
5. Cadiz, J.J., Venolia, G., Jancke, G., and Gupta, A. Designing and Deploying an Information Awareness Interface. In *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW'02)* (New Orleans, Louisiana, November 16-20, 2002). 314-323.
6. Donath, J., Karahalios, K., and Viegas, F., Visualizing Conversation. In *Proceedings of 32th Hawai'i International Conference on System Sciences (HICSS-32)*, (Hawaii, January 5-8, 1999).
7. Dourish, P. and Bellotti, V. Awareness and coordination in shared workspaces. In *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW'92)* (Toronto, Canada, October 31 - November 4, 1992). 107-114.
8. Greenberg, S., Gutwin, C., and Cockburn, A. Awareness Through Fisheye Views in Relaxed-WYSIWIS Groupware. In *Proceedings of the Graphics Interface 1996 Conference* (Toronto, Canada, May 22-24, 1996).
9. Gutwin, C. and Greenberg, S. The effects of Workspace Awareness Support on the Usability of Real-Time Distributed Groupware. *ACM Transactions on Computer-Human Interaction*, Vol. 6, N° 3, September 1999, 243-281.
10. Gutwin, C. and Greenberg, S. The Importance of Awareness for Team Cognition in Distributed Collaboration. In E. Salas and S.M. Fiore (eds.) *Team Cognition: Understanding the Factors that Drive Process and Performance*, APA Press, Washington, 2004. 177-201.
11. *The Palace User Software Guide for Windows*, version 3.5, <http://www.palacechat.us/documentation.php>, accessed 24 March 2005.
12. Pinho, M.S., Bowman, D.A., and Freitas C.M.D.S. Cooperative Object Manipulation in Immersive Virtual Environments: Framework and Techniques. In *Proceedings of the ACM symposium on Virtual Reality Software and Technology (VRST'02)*. (Hong Kong, China, November 11-13, 2002). 171-178.
13. Prinz, W., Kolvenbach, S., and Klöckner, K. Situative Cooperation Support for Communities. In *SIGGROUP Bulletin*, December 2002, Vol. 23(3), 24-28.
14. Prinz, W., Pankoke-Babatz, U., Gräther, W., Gross, T., Kolvenbach, S. and Schäfer, L. Presenting Activity in an Inhabited Information Space, in Snowdon D. N., Churchill, E.F., and Frécon E. (eds): *Inhabited Information Spaces*, London, Springer, 2004, 181-208.
15. Robinson, M., Pekkola, S., Korhonen, J., Hujula, S., Toivonen, T. and Saarinen M.-J. O. Extending the Limits of Collaborative Virtual Environments, in Churchill, E.F., Snowdon D. N. and Munro A. J. (eds): *Collaborative Virtual Environments*, London, Springer, 2001.
16. Saltz, J.S., Hiltz, S.R., Turoff, M. Student Social Graphs: Visualizing a Student's Online Social Network. In *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW'04)* (Chicago, Illinois, November 6-10, 2004). 596-599.

Guidelines and Usability Principles to Design and Test Shared-Knowledge Awareness for a CSCL Interface

María Paula González¹, César A. Collazos^{1,2}, and Toni Granollers¹

¹Group GRIHO – Escola Politècnica Superior
University of Lleida, c/Jaume II, 69
25001 Lleida, Spain

{mpg, tonig}@diei.udl.es

²Department of Systems
Universidad del Cauca
FIET-Sector Tulcan, Popayán, Colombia
ccollazo@unicauca.edu.co

Abstract. Shared Knowledge Awareness is defined as the consciousness on the shared knowledge that a particular student has when carrying out a collaborative learning activity in a CSCL environment. In fact, an adequate level of Shared Knowledge Awareness can be promoted by including in the interface of this environment some specific features that improve the student perception related to such knowledge. This paper proposes some basic design guidelines that should be taken into account when designing a CSCL interface in order to promote an adequate user's behavior with respect to his/her Shared Knowledge Awareness. Besides, a set of usability principles is identified and linked to every suggested guideline to evaluate its quality (in terms of "easiness to use and learnability") in an actual interface. Two different experiments are included as real-life examples that are analyzed within the proposed approach.

1 Introduction

In Computer Supported Collaborative Learning (CSCL), awareness can be used for enhancing collaborative opportunities reducing the meta-communicative efforts needed to collaborate across physical distances and in computer-mediated environments [1]. In fact, collaborative oriented activities such as negotiation of meaning, creation of joint understanding, and division of labor and responsibility require meta-communicative actions for maintaining certain cognitive and collective effects when the scenario is a distributed collaborative learning environment [2]. Thus, awareness mechanisms usually allow learners to maintain in an implicit way information about the others' interactions with common problem areas and corresponding tasks.

In the above context, Collazos et al. have proposed a particular kind of awareness called Shared Knowledge Awareness (SKA) whose purpose is to increase the perception about the shared knowledge students have in a collaborative learning scenario

[3]. The SKA not only tries to improve and maintain the shared knowledge of a student group but also concerns the understanding that this group has about it [3]. Although the definition of SKA includes a series of questions that should be considered to reach it, it is difficult to ascertain how to provide mechanisms in the interface of a real system in order to measure the occurrence of SKA in a CSCL scenario. This problem could be coped by formulating a set of general design guidelines (DGs) destined to assure a minimum coverage of the different questions included in the definition of SKA. In addition, it would be desirable to outline some mechanisms to test the quality of every DG in the set, as quality is crucial to guarantee the effectiveness and efficiency of every SKA-related element in a real CSCL interface. As we will see in the next Section, usability is an adequate alternative to perform this measuring.

This paper proposes a set of minimum DGs that should be considered by a CSCL developer in order to ensure an acceptable level of SKA in a computer-mediated and distributed collaborative learning interface. General mechanisms and elements are added to every SKA-related question. Furthermore, classical usability principles are proposed to measure quality in use of every design guideline, as usability is a concept that provides a ratio of how easy an interface is to understand and use [4]. Consequently, every original question related to the SKA definition is enriched with two minimum sets of DGs and predominant usability principles (UPs). Our goal is to help CSCL designers and evaluators to design and test an appropriate Shared-Knowledge Awareness for a CSCL interface.

The rest of this paper is structured as follows. First, in Section 2 we give an overview of the most relevant features in SKA. Next, Section 3 defines the concept of usability and describes most relevant UPs. Then, Section 4 describes our proposal for extending every SKA-related question by adding minimal sets of DGs and UPs in order to assure and evaluate an appropriate level of SKA in a CSCL interface. Section 5 presents some experimental results which demonstrate the relevance of our extended approach and Section 6 discusses some related work. Finally, Section 7 concludes by summarizing the main results that have been obtained and discussing further work.

2 Knowledge Awareness in CSCL Scenarios

In CSCL scenarios, collaborative learning is effective if people succeed in building and maintaining a shared understanding of the problem [4]. For this reason, the shared understanding should be represented and promoted. A way to do this is by capturing this shared understanding into an awareness mechanism. Also, the shared understanding could be promoted only if people can know its current state during the collaborative activity.

In the above context, Shared Knowledge Awareness (SKA) can be defined as the consciousness on the shared knowledge of the students that carry out a collaborative learning activity when working in groups. This shared knowledge is more than the

shared understanding of the problem and it is composed of the understanding of several aspects of the collaborative work, including coordination, strategy communications, monitoring, and shared comprehension of the problem [3]. For constructing this shared knowledge it is necessary to wonder how one may become aware of one’s own knowledge and, how the actions people do affect the knowledge of the other members within the group, self-control and self-monitoring of the learning process. The questions in Table 1 are examples of what students consider during a typical collaborative activity in order to be aware of the shared knowledge they have.

Table 1. SKA-related questions [3]

| <i>Awareness</i> | <i>Questions</i> | <i>SKA Id</i> |
|--|---|---------------|
| <i>Knowledge construction (individual)</i> | Is what I am doing helping to solve the task? | A |
| | Do I need more time/resources? | B |
| | What else do I need to find out about this topic? | C |
| | How much time is available? What is our score? | D |
| | Is what I did helping to solve the task? | E |
| | What and how did I learn from the others members of the group? | F |
| | Did I finish the work? | G |
| | What am I learning from the group work? | H |
| | What I need to know about the topic? | I |
| <i>Shared Knowledge construction (group)</i> | What are the other members of the group doing to complete the task? | J |
| | Is what the others are doing helping to solve the task? | K |
| | What do others members know about the topic? What do others members need to know about the topic? | L |
| | How can I help other students to complete the task? | M |
| | What did other members of the group learn from me? | N |
| | Where are the other members of the group? | O |

Although the SKA definition provides some general questions, it is not easy to decide how to provide mechanisms in the interface of a system that could be used to measure the appearance of SKA in a CSCL scenario. This aspect will be discussed in the Section 4. In addition, it would be desirable to describe some issues to test the quality of every design guideline trying to guarantee the effectiveness and efficiency of every SKA-related element in a CSCL interface. As we will see in the next Section, usability is an adequate alternative to perform this type of measuring.

3 Usability and Most Relevant Usability Principles

Usability is a quality of the user interface formally defined as “the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use” [6, 7]. Usability plays a major role during the evaluation stages of a software development under HCI where actual features in the interface of the system that is being developed are contrasted against ideal usability values and premises [8].

Usability is a complex concept that has been divided in a series of principles (also denoted usability attributes) in order to be understood in a better way. Thus, usability principles (UPs) are mainly intended to stress some part of the usability definition, and HCI researchers are aware that in a real interactive system these principles might overlap. Diverse authors have proposed alternative sets of UPs according to the most relevant features in the definition of usability they want to emphasize. Also diverse classifications are proposed when linking them with the formal definition. In spite of this situation, nowadays some common UPs presented in [4,8,9,10,11,12,13,14] have been agreed within the major part of the HCI community, as well as the classification shown in [15]. Consequently, we also have considered these UPs and this classification as the most relevant for the CSCCL scope. For clarity reasons, we have compiled the most relevant features of the above consensus in Table 2, including some typical questions that must be answered when testing UPs in a real interface. It must be stressed that some concepts like visibility or feedback are not always considered as basic usability principles but as part of them (visibility) or as consequences of following the usability principles in a correct way (feedback).

Table 2. Most relevant usability principles and its related questions

| <i>General Usability Principle</i> | <i>General Questions</i> | <i>Specific Usability Principle</i> | <i>Specific Questions</i> |
|------------------------------------|---|-------------------------------------|--|
| Effectiveness | How do users define success? Is success the same for all stake-holders? What are the goals; what are the tasks? Are there hidden goals? | Completeness | Was the task fully completed? |
| | | | Were the user's goals met? |
| | | Accuracy | Was the task completed successfully? |
| | | | Did the user get the right or correct result? |
| Efficiency | In how many ways the user and system exchange information? | Flexibility | Had the user, in dialogue situations, freedom from system imposed constraints on input dialogue? |
| | | | Did the system offer to support user interaction for more than one task at a time? |
| | | | Equivalent values of input and output could be substituted for each other? |
| | How long do users expect a task to take? Is the task completed in a single session? What styles of interaction do users prefer? What would make the interface feel efficient? | Speed | Was the user able to complete the task quickly? |
| | | Effort | Was the user able to complete the task without undue cognitive effort? |
| Engaging | What kind of work (or play) is this? What are the expectations for style and tone? How often? How long? When, where, how | Pleasant | Did the user have a pleasant experience when working on the task? |
| | | Satisfying | Was the user satisfied by the way |

Table 2. (continued)

| | | | |
|-----------------|---|----------------|---|
| | and why? | | in which the application supported their work? |
| Error Tolerance | How familiar is the domain? The terminology? What will users find difficult? What kinds of errors are likely? How serious are their consequence? Will the users understand the problem, or they will need an explanation? | Prevention | Did the user interface help user avoid making mistakes? |
| | | | Were mistakes minor rather than major? |
| | | Recovery | If the user made an error, did the interface assist him/her in making a successful recovery? |
| Easy to Learn | Will users expect to have to learn to use it? Are they learning something new? How complex is the task? How often will it be used? How important is it to get it right? | Predictability | Was the user able to work with some certainty because the user interface built on him/her previous knowledge? |
| | | Consistency | Was the interface consistent, so that once a user learnt how to use part of the application, he was able to easily learn how to use another part? |
| | | Affordance | Shape? Visible? Coherent action? |

As we will see in Section 4, UPs are necessary when evaluating the quality of DGs for assessing an appropriate level of SKA in a CSCL interface. Note that the classification shown in Table 4 still remains valid independently of any particular usability testing methodology.

4 The Proposed Model

The third column in Table 3 describes the basic design guidelines (DGs) that should be taken into account when designing a CSCL interface in order to promote an adequate user’s behaviour with respect to his/her SKA. Each DG is identified by a DG Identifier (second column in Table 3) and linked to its correspondent SKA-related question in Table 1 by means of a SKA Identifier (first column in Table 3). In the same way, every design guideline is associated with a set of UPs that should be highlighted when evaluating the interface of a CSCL system (either the final version or the prototype). The UPs are selected by generalizing common criteria used to test similar elements as those listed in Table 3 (third column) during the evaluation of a considerable amount of different interfaces [16] and 69 websites [17]. In order to improve this selection, UPs presented in practical examples in [12] and [13] which are handled to test interface features that materialized similar DGs as those shown in Table 3 are considered.

In that way, while the proposed DGs can be applied in one part of the CSCL interface development, the related UPs can help us to asses the quality (in terms of “easiness to use and learnability”) of the interface elements which have materialized these guidelines in an actual interface or prototype. In the next section we will show how the proposed DGs are instantiated in two different CSCL real-life interfaces. It must be stressed that the design guideline #1 (communication mechanism) is a general

design guideline inherent to the definition of CSCL environments, so that some of the other DGs described in Table 3 are just a particularization of it (e.g. design guidelines #7, #8, #16, #17 or #31).

Table 3. SKA-related design and evaluation guidelines for a CSCL interface

| <i>SKA Id</i> | <i>Design Guideline Id</i> | <i>Design Guideline Description</i> | <i>Usability Principle</i> |
|---------------|----------------------------|--|---|
| A | 1 | Communication mechanism | Effectiveness (Accuracy) |
| | 2 | Explicit sensor of task advance | |
| | 3 | Explicit sensor of self-collaboration performance | |
| B | 4 | Alert mechanisms | Effectiveness (Completeness) Efficiency (Effort) Easy to Learn (Consistency) |
| | 5 | Explicit sensor of task advance | |
| C | 6 | Explicit sensor of task advance | Efficiency (Flexibility) |
| | 7 | Sent information representation | |
| | 8 | Received information representation | |
| | 9 | Mechanism for classifying received information | |
| D | 10 | Alert mechanisms | Efficiency (Effort) |
| E | 11 | Explicit sensor of self-collaboration performance | Effectiveness (Accuracy) |
| F | 12 | Explicit sensor of others' collaboration performance | Effectiveness (Accuracy) Efficiency (Flexibility) |
| G | 13 | Alert mechanisms | Effectiveness (Completeness) Error Tolerance (Prevention) Easy to Learn (Consistency) |
| H | 14 | Explicit sensor of others' collaboration performance | Efficiency (Effort) |
| I | 15 | Explicit sensor of self collaboration performance | Efficiency (Effort and Flexibility) |
| | 16 | Received information representation | |
| | 17 | Mechanism for classifying received information | |
| | 18 | Explicit sensor of task advance | |
| J | 19 | Explicit sensor of task advance | Efficiency (Effort) |
| | 20 | Explicit sensor of others' collaboration performance | |
| K | 21 | Explicit sensor of others' collaboration performance | Efficiency (Effort) |
| L | 22 | Explicit sensor of task advance | Efficiency (Effort and Flexibility) Easy to Learn (Consistency and Affordance) |
| | 23 | Received information representation | |
| | 24 | Mechanism for classifying received information | |
| | 25 | Others' user profiles (only if the profile defines part of the topic) | |
| M | 26 | Received information representation | Effectiveness (Accuracy) Easy to Learn (Consistency and Predictability) |
| | 27 | Mechanism for classifying received information | |
| | 28 | Others' user profiles (only if the profile defines part of the topic or includes user's expertise) | |
| N | 29 | Explicit sensor of self-collaboration performance | Efficiency (Effort) Easy to Learn (Consistency, Predictability and Affordance) |
| | 30 | Received information representation | |
| | 31 | Mechanism for classifying sent information | |
| O | 32 | Mechanism to highlight others' last contribution | Efficiency (Effort) Easy to Learn (Predictability) |
| | 33 | Mechanism to restart actual state of the task (only asynchronous CSCL interfaces) | |

5 Evaluated Systems

In order to illustrate the model presented in Section 4 two different classes of CSCL interfaces have been analyzed: a distributed and synchronous game called “Case the Cheese” (see Figure 1) and a asynchronous system to support collaborative knowledge building in schools classrooms called “Synergeia”¹. In what follows, we will detail the materialization of every design guideline proposed in Section 4 for these two sample cases. For the sake of clarity, every SKA-related question in Table 1 will be referred to with the associated SKA Identifier (a letter from “A” to “O”). Also every design guideline presented in Table 2 will be denoted by using its DG Identifier (#1 to #33). In the case of the game “Chase the Cheese” the column corresponding to each design element description will include a reference to the corresponding screen area in Figure 1 (from sa#1 to sa#12).

Concerning the set of UPs associated with design guidelines in Table 3, it must be remarked that these principles were used to assess the general quality of the screen elements that materialized the above guidelines in both CSCL interfaces. In general, for these elements an acceptable range of the ideal usability score has been reached. However, these results should not be considered as properly justified, as a more deep and complete usability study must be still carried out. In spite of this, the selected UPs have proven to be the appropriate ones when performing this first approximation to the final usability evaluation.

5.1 The Game *Chase the Cheese*

Chase the cheese [18] is a game played by four persons, each with a computer physically distant (the only communication allowed is computer-mediated). Players are given very few details about the game, and the rest of the rules must be discovered while playing, forcing participants to develop a joint strategies to succeed.

A complete description of the Chase the Cheese game can be found in [18]. The Figure 1 describes the game interface for one player distinguished with the yellow color. The board is one but is replicated in four quadrants, each having a coordinator – one of the players– permitted to move the mouse with the arrows. The other participants are called collaborators and can only help the coordinator sending their messages. The aim is to lead the mouse to the cheese with a high total score (400 points maximum). Note that if the mouse is not in the last quadrant it has to move to a traffic light (which indicates starting position for each coordinator). The mouse has to go around obstacles (general obstacles or grids visibles to everyone and colored obstacles visible only to each player). Indeed, when starting to move the mouse, the coordinator has an individual score (11) of 100 points. Whenever the mouse hits an obstacle, this score is decreased in 10 points. Therefore, collaborators have to develop a shared strategy to communicate obstacle locations to the coordinator of the current quadrant. When the coordinator finishes with his/her quadrant, his individual score is added to the total score of the group. If any of the individual scores reaches a value below or equal to 0, the group loses the game.

¹ See Synergeia homepage at <http://bscl.fit.fraunhofer.de/en/about.html>

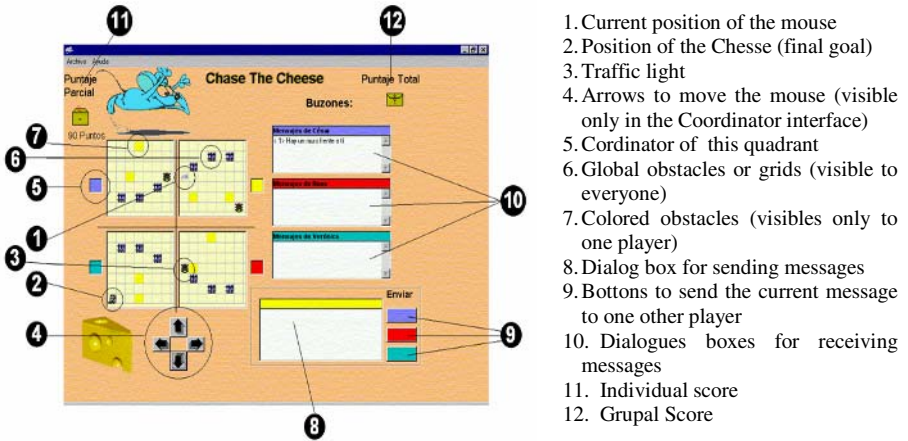


Fig. 1. Interface of the “Chase the cheese” for yellow player (current Coordinator) [18]

Table 4 summarizes the SKA-related design elements observed in the interface of the game. In the comment column some questions are included for clarification. The first column links every instantiated element with the corresponding DG Identifier (used to index Table 3). Note that some DGs do not exist as particular features in the interface, but they can be inferred by considering some elements simultaneously. Actually, there is no single element to GDs #2 and #5, but they can be derived by combining the elements *C3* and *C4* shown in Table 4. In the same way, the GD #4 can be inferred as a merging of the elements *C4* and *C5* in Table 4. Whith respect to the DG #11, it can be discerned by looking at the elements *C10* and *C11* in Table 4.

In addition, the DGs #21 and #25 have no specific feature which could be considered as their specification in the interface of “Chase the Cheese”. However, both DGs #21 and #25 can be determined if some flexibility is adopted. In the case of the DG #21, others’ collaboration can only be understood as changes in the global score (sa#12 in Figure 1) after each other coordinator had finished its quadrant. Also the quality of the recieved messages (sa#10) can help to assess others’ performance and consequently *C2* should be considered a specification of DG #21. Concerning the DG #25, it could be deduced by transitivity: the current position of the mouse determine the active quadrant (item *C3* in Table 4), and the colour of these quadrant (sa#5) determine who is the coordinator in this moment of the play.

It must be stressed that even though the interface of the game provides an explicit sensor of grupal score (sa#12 in Figure 1), this sensor should not be confused with the item that represents DGs #12 or #20, as part of the grupal score can include the self performance (if my turn has already ended). Consequently, other features in the interface of “Chase the Cheese” must be considered to materialize DGs #12 and #20. Indeed, while the DG #12 can be induced by observing the quality of the received messages (screen element *C2*), the DG #20 is reflected as a combination of the elements *C20* and *C21* shown in Table 4.

Respecting the DG #29, even though the item *C4* in Table 4 makes this DG explicit in the interface of “Chase the Cheese”, we also considered that the item *C3* shown in

Table 4. SKA-related design elements in the interface of the game Chase the Cheese

| <i>Design Guide line Id</i> | <i>Element Id</i> | <i>Design Element Description</i> | <i>Comment</i> |
|-----------------------------|-------------------|---|--|
| 1 | C1 | Dialog box for sending messages (sa#8) + Color Buttons (sa#9) | Coordinator: Am I asking clearly about colored obstacles? / Collaborator: Am I sending my colored obstacles to the coordinator? |
| | C2 | Dialogues boxes to receive messages (sa#10) | Coordinator: recognizing colored obstacles helps to solve the task / Collaborator: reading carefully questions from the coordinator helps to solve the task |
| 2, 5, 22 | C3 | Mouse position in the game board | Guidelines 2 and 5: Coordinator: Is my movement bringing the mouse closer to the cheese? |
| 2, 3, 4, 5 | C4 | Individual score (sa#11) | Coordinator: After moving the mouse, does my individual score increase or decrease? |
| 4 | C5 | Idem C2 | Coordinator: collaborators' comments can be used as alert mechanism |
| 6 | C6 | Idem C3 | Coordinator: if the mouse is close either to the cheese or to the next semaphore, then the self task is ending |
| | C7 | Idem C4 | Coordinator: if the coordinator's score is decreasing then more information about others' colored obstacles is needed |
| 8 | C8 | Idem C2 | Coordinator: Do I have information about others' colored obstacles in the board positions adjacent to the mouse? / Everybody: Am I understanding the game strategy (what do I have to do)? |
| 9, 17, 24, 27 | C9 | Idem C2 | Separate dialogue boxes for each player (each participant identified with a colour) |
| 10, 11 22 | C10 | Groupal score (sa#12) | Guideline 11: Coordinator: after ending my turn, does the group score increment or decrement? |
| 11 | C11 | Idem C3 | Coordinator: after ending my turn, is the mouse closer to the cheese or we have lost the game? / Collaborator: after sending my colored obstacles to the coordinator, did the mouse skip them? |
| 12, 14 | C12 | Idem C2 | Coordinator: he/she has learnt others colored obstacles / Everybody: he/she could lerant the game strategy if related information has been recieved |
| 13 | C13 | Game over message (no shown in Figure 1) | |
| | C14 | Arrows (sa#4) | If arrows are not available then the turn changes |
| | C15 | Idem C3 | The mouse goes to other quadrant in the board |
| 15 | C16 | Idem C4 | |
| 16, 23 | C17 | Idem C2 | Coordinator: needs information about others' colorated obstacles adjacent to the current mouse position / Collaborator: needs to know coordinator questions / Everybody: the game strategy can be understood by analysing recieved information |
| 18 | C18 | Idem C10 | Grupal score shows if the task is advancing properly or not |
| | C19 | Idem C3 | |

Table 4. (continued)

| | | | |
|--------|-----|----------|--|
| 19,20 | C20 | Idem C3 | Collaborator: is the coordinator moving the mouse by skipping my colored obstacles? |
| 20, 21 | C21 | Idem C2 | Coordinator: are the other members of the team sending to me their colorated obstacles? Collaborator: is the coordinator asking me something? / Everyone: Are the others' messages helping me to understand the game strategy? |
| | C22 | Idem C10 | Collaborator: after ending another player's turn, is the grupal score better or not? |
| 25, 28 | C23 | Idem C3 | Every quadrant in the board is associated with the colour of the participant that coordinates it. Therefore the mouse position defines the current profile of the other players (Coordinator or Collaborator) / Guideline 28: Collaborator: knowing who is the current coordinator helps to focalize answering his/her questions |
| 26 | C24 | Idem C2 | Coordinator: he/she has to read others' colored obstacles to skip them / Collaborator: he/she has to read coordinator's questions to help him/her |
| 29 | C25 | Idem C3 | Collaborator: After sending a message to the coordinator with his/her coloured obstacles adjacent to the mouse, the coordinator skipped them |
| | C26 | Idem C4 | Everyone: my self performance is reflected in the individual score |
| 30 | C27 | Idem C2 | Everyone: observing others' questions and comments can help him/her to perceive his/her performance |
| 32 | C28 | Idem C2 | Everyone: others' messages are listed chronologically, therefore it is possible to see the last contributions of the other players |
| | C29 | Idem C10 | Collaborator: after seeing the mouse changing from one quadrant to another, the last contributions of the coordinator are reflected in the increasing or decreasing of the grupal score |

Table 4 could help improve the awareness related to DG #29 (question N in Table 1). Finally it must be stressed that the DGs #7, #31 and #33 are not included in Table 4 since they could not be identified as part of the “Chase the Cheese” interface. Indeed, the interface of this game does not provide a representation of the sent information beyond its edition. Additionally, the DG #31 seems to be omitted to simplify the interface. Respecting the DG #33, it must be remarked that this DG has no sense in a system like “Chase the Cheese”, as all the actions done by the participants are performed during the current execution of the game.

5.2 The Educational System Synergeia

Synergeia is designed to support collaborative knowledge building in school classrooms. It provides a shared, structured, web-based work space in which collaborative learning can take place, documents and ideas can be shared, discussions can be stored

and knowledge artifacts (portfolios) can be developed and presented [19]. As pointed out in [20], the Synergeia system strives to support the synergistic construction of knowledge at the group level that is quite distinct from what any of the students could produce on their own. Consequently, the SKA-related elements present in the Synergeia interface are crucial and should be easily recognized as a particularization of the DG presented in Section 4.

A complete description of the whole Synergeia system can be found in [19]. As SKA usually emerges from a group perspective, we will focalize in the part of the Synergeia interface used by the students, specifically we will take into account only the workspaces shared for the group. These workspaces are mainly formed by a Group Learning Place or GLP, a group Shared Knowledge Building Area or GKBA, a whiteboard called MapTool (MT) to draw concept maps and schemas, an Instant Messaging tool, several kinds of Menus, a Calendar and a Negotiation Environment or NE that provides mechanisms for storing and voting knowledge artifacts.

The information in Table 5 summarizes the results obtained when materializing general DGs of Table 3 in the interface of the above groupal workspace. As in Table 4, numbers in the first column link every instantiated element with the corresponding DG Identifier (used to index Table 3). Note also that some DGs are not included since they have not been clearly perceived or they have no meaning in the Synergeia scenario. Particularly, DG #25 and #28 in Table 3 could not be specified in the observed workspaces. Note that guideline #25 is intended to encourage SKA related to other students' knowledge about the current topic (see SKA-related question L in Table 1), and the student profile is not relevant as part of this knowledge. In the same way, design guideline #28 is associated with the consciousness about the self possibility of helping other students to complete the task (see SKA-related question M in Table 1) and involves the self perception of the other students' profile only if this profile defines part of the topic or includes user expertise. Note that in Synergeia student profiles are not part of the current topic and each student expertise is unknown to the rest of the classmates.

Although the Synergeia system does not provide an explicit sensor of the task advance, DG #2, #5, #6, #18,#19 and #22 in Table 3 could be materialized in the workspaces that have been analysed. With respect to the design guideline #5, it could be observed that a combination of the *S5* and *S6* elements in Table 5 can help to improve its related SKA (see SKA-related question B in Table 1). Similarly, our analysis have described that the design guidelines #2 and #6 can be respectively deduced from the *S2* and the *S7* elements shown in Table 5. The experimentation also revealed that the design guideline #18 can be inferred as a combination of the elements *S26* and *S27* in Table 5. In the case of the design guideline #19, a merger of the elements *S17*, *S28* and *S29* give enough hints to improve the user perception about the other students collaboration to solve the current task (i.e., the SKA-related question J in Table 1, which is the SKA associated with design guideline #19). Finally, also design guideline #22 could be partially deduced by observing the elements *S26* and *S27* in Table 5.

Additionally, the Synergeia system does not provide any specific design elements to depict self-performance or others-performance when solving the current task. Consequently, as in the previous analysis, DGs #3, #11, #12, #14, #15, #20, #21 and #29 could not be directly materialized in the inspected workspaces. However, as

mentioned before different features in the Synergeia workspaces can be used (individually or combined) to amend the corresponding SKA apprehension (distinguished by its identification letter in the row corresponding to every design guideline in Table 3). The design elements that could be used to cope with these design guideline visualization in Synergeia can be checked in Table 5. Finally it must be stressed that the design element #32 is not part of the Synergeia interface because it is mainly intended for synchronous systems.

Table 5. SKA-related design elements in the interface of Synergeia

| <i>Design Guide-line Id</i> | <i>Element Id</i> | <i>Design Element Description</i> | <i>Comment</i> |
|-----------------------------|-------------------|---|--|
| 1 | S1 | One's current sent messages available in the GKBA | Thinking type of the messages is "starting", "working", "deepening" or "reflection" |
| 2 | S2 | Personal contributions to the concept map and diagrams in MT | Contributions must be done during the current session |
| 3 | S3 | Personal contribution in the proposed portfolio before submitting it (NE) | Contributions must be doing during the current session |
| 4 | S4 | Calendar | |
| 5 | S5 | No agreement in portfolio negotiation (Icons to vote in the NE) | Voting during the current session |
| | S6 | Idem S1 | Thinking type of the messages is "help" or "problem" |
| 6 | S7 | Current messages after no agreement in the NE | View of the messages |
| 7 | S8 | Idem S1 | Thinking type of the messages is "starting", "reflection", "help" or "problem" |
| 7 and 8 | S9 | Concept map and diagrams view in the MT | Guideline 7: One's contribution Guideline 8: Other's contribution |
| 8,9 | S10 | Other's messages available in the current GKBA | Guideline 9: Messages can be classified by selecting different Thinking Types |
| 10 | S11 | Calendar | Awareness about remaining time |
| 11 | S12 | One's contribution in the History and Info menus | These menus are available by clicking on icons in the event column of a folder display (main screen) |
| | S13 | Idem S1 | Thinking type of the messages is "starting", "working", "deepening" or "reflection". Messages must be sent during a past session. |
| | S14 | Idem S2 | Contributions must be done before the current session |
| 12 | S15 | Others contribution in the History and Info menus | Idem S11 |
| 12 and 16 | S16 | Other's messages queued to answer my own messages (GKBA) | Guideline 12: Highlight the analysis of other's messages when the thinking type of my own message is "starting", "help" or "reflection" Guideline 16: Thinking type of my own messages is "help", "problem" or "reflection" |

Table 5. (continued)

| | | | |
|-------------------------------|-----|--|---|
| 12,16 , 19 and 20 | S17 | Other's contributions to the concept map and diagrams in the MT | |
| 13 | S18 | Current portfolio submitted to the CLP | The name of the portfolio is part of the CLP list of topics |
| 14 | S19 | Other's contribution in the proposed portfolio before submitting it (NE) | |
| | S20 | Idem S16 | |
| | S21 | Idem S17 | |
| 15 | S22 | Idem S1 | Thinking type of the messages is "help", "problem" or "reflection" |
| | S23 | Idem S2 | |
| | S24 | Idem S3 | |
| 17 | S25 | Idem S10 | Messages can be classify using different Thinking Types |
| 18 and 22 | S26 | Message displayed in the top area of the GKBA | The message is displayed above the thread of queued messages |
| | S27 | Elements of the proposed portfolio after submitting it (NE) | |
| 19 | S28 | Idem S10 | Thinking type of the messages is "starting", "working", "deepening" or "reflection". |
| | S29 | Idem S22 | |
| 20 and 21 | S30 | Idem S10 | Messages can be listed by authors. Then Thinking type of the messages should be "working", "deeping" or "reflection" |
| 23 | S31 | Idem S9 | Depict what do other members know about the topic |
| 23 and 24 | S32 | Idem S10 | What do other members know about the topic: thinking type of the messages is "starting", "reflection", "working" or "deeping"/ What do other members need to know about the topic: Thinking type of the messages is "starting", "problem" or "help" |
| | S33 | Idem S19 | Depict what do other members know about the topic |
| 26 | S34 | Idem S9 and S2 | Personal concepts relevant for the task that are not depicted in the concept map |
| | S35 | Idem S27 | Information known by me and relevant for the task that are not part of the current portfolio |
| | S36 | State of negotiation process in the NE | Casting my vote about the current portfolio is necessary to complete the task |
| 26 and 27 | S37 | Idem S10 | Reply messages which thinking type is "starting", "problem" or "help" can help to solve the task |
| 29 | S38 | Idem S12 | Idem comment in S12 |
| | S39 | Personal contribution in the portfolios submitted to the CLP | |
| 30 and 31 | S40 | One's messages queued to answer other messages (GKBA) | Thinking type of the other messages is "starting", "help", "problem" or "reflection" |
| 32 | S41 | Idem S10 | Others' messages can be listed chronologically |

Table 5. (continued)

| | | | |
|----|------------|--|---|
| | <i>S42</i> | Idem S7 | Observing messages in the NE it is possible to know the last others' contribution to the negotiation |
| | <i>S43</i> | Idem S27 | Elements in the proposed portfolio have the date of submission as one of their attributes. Consequently it is possibly to observe others' last contribution |
| 33 | <i>S44</i> | Recovering of the last interface state after entering to a new session | |

6 Related Work

As awareness is considered a relevant feature in CSCL, several efforts have been devoted to cope with its definition and evaluation. In particular, different authors have proposed some activities that comprise the mechanics of collaboration using a conceptual framework for developing discount usability evaluation techniques that can be applied to shared-workspace groupware [21]. Although the goals pursued in that proposal are also related with awareness design and evaluation, the work has been oriented towards groupware awareness. Other contributions related to groupware usability evaluation can be found in [22, 23].

Many authors have proposed mechanism to design CSCL scenarios. However, there is a lack of information about usability aspects that need to be considered in order to have CSCL usable systems. Georgiakakis et al. have designed Asynchronous Network-Supported Collaborative Learning (ANSCL) systems, providing some guidance for build usable CSCL scenarios. They propose a set of relevant patterns for designing usable ANSCL systems [24].

On the other hand, there are some works related with the use of awareness in order to improve collaboration on CSCL scenarios. For example, Fjuk & Krange [25] have provided insights about how various forms of awareness information should be supported by a computer to enable collaboration in distributed environments. Based on an understanding of learning as mediated by social interaction and artifacts, they argue that the effects of task and workspaces awareness are highly situated with respect to collaborative knowledge construction. However, to the best of our knowledge there are no similar works to provide and test mechanisms in the interface of a real system in order to measure the occurrence of SKA in a CSCL scenario as it is presented in this paper.

7 Conclusions and Further Work

In Computer Supported Collaborative Learning (CSCL) awareness is a central concept related to the individual and the groupal perception, helping to decrease cognitive effort associated with communication, thus enhancing the quality of the collaborative

learning processes. In this context Shared Knowledge Awareness (SKA) [3] provides powerful mechanisms to ensure the adequate perception that the students should have about their shared knowledge in such processes. However, even though the definition of SKA includes a series of questions that should be considered to reach it, it is difficult to ascertain how to provide mechanisms in the interface of a real system in order to measure the occurrence of SKA in a CSCL scenario.

In this paper we have presented a model to design and test SKA-related features in a real CSCL interface. To do this, some general design guidelines (DGs) that should be included in a CSCL interface are proposed in order to guarantee an adequate user's behaviour with respect to his/her SKA. Particularly, each different question included in the definition of SKA is linked to a minimal set of DGs, hence assuring its minimum coverage. The proposed model is used to analyse the occurrence of SKA in two different CSCL interfaces (one synchronous and one asynchronous). This way, two experiments were carried out to illustrate the novel model.

Besides, a set of usability principles (UPs) is associated with each general DG in order to test its quality in terms of "easiness to use and learnability", as usability is defined as the extent to which a product can be used by specified users to achieve a given goal with effectiveness, efficiency and satisfaction in a particular context of use [6, 7]. In fact, every proposed DG is connected with a set of UPs that should be highlighted when evaluating the interface of a CSCL system (either the final version or the prototype). These UPs are selected by generalizing common criteria used to test elements similar to the proposed DGs during the evaluation of a considerable amount of different interfaces [16], 69 websites [17] and some practical examples shown in [12] and [13] where UPs were used to test interface features that materialized similar DGs as those presented here.

Although the selected UPs have proven to be the appropriate ones when performing this first approximation to the final usability evaluation, we think that it would be convenient to carry out a more complete usability study. Consequently, part of the future work is focused on the performing a deep usability evaluation of different CSCL interfaces. In this respect, an exhaustive inspection of the usage ideal usability values and premises in the context of the CSCL environment is currently being pursued on the basis of the proposed UPs. Besides, classical methodologies used to test usability of general systems should be reformulated to make them more efficient for assessing the occurrence of SKA in CSCL scenarios. We also believe that SKA and its related DGs and Ups can be expanded beyond a CSCL scenario. In that respect an extension to the ambit of the organizational groups must be considered, as SKA could be crucial in order to achieve specific organizational goals. It seems plausible that SKA model may have an indirect effect on performance mediated by team coordination.

Acknowledgments. This work was partially supported by Colciencias (Colombia) Project No. 4128-14-18008, Cicyt TEN2004-08000-C03-03, Colciencias (Colombia) Project No. 030-2005, Cicyt Project ADACO (TIN2004-08000-C03-03), and Project SGR-00881 (Generalitat de Catalunya, Spain). We also would like to acknowledge Red de Parques Software-Colombia.

References

1. Palfreyman K. A. and Rodden T., A protocol for user awareness on the World Wide Web. In *Procs. of CSCW'96*, Boston, MA, USA, Nov. 1996, pp. 130 - 139, ACM Press (1996).
2. Fjuk, A., & Dirckinck-Holmfeld, L, Articulation of actions in distributed collaborative learning. *Scandinavian Journal of Information Systems*, 9(2), 3-24 (1997).
3. Collazos, C., Guerrero, L., Pino, J., and Ochoa, S., Introducing Knowledge-Shared Awareness. *Procs. of IASTED'02*, USA, pp.13-18 (2002).
4. Preece, J: *Human-computer interaction*. Addison-Wesley, Reading, MA. (1994).
5. Dillenbourg, P.: Some technical implications of the distributed cognition approach on the design of interactive learning environments. *Journal of Artificial Intelligence in Education*, Vol. 7, No.2, pp.161-180 (1996).
6. Usability Net: International standards for HCI and usability. Available on http://www.usabilitynet.org/tools/r_international.htm (Retrieved March 2006).
7. ISO Standards No. 9241-11: Guidance on usability. (). Geneva, Switzerland: ISO. (1998).
8. Nielsen, J., *Usability Engineering*. Academic Press Professional, Boston, MA. (1993).
9. Bevan, N., Kirakowsky, J. Maissel, J., What is Usability. *Proc. of 4th Intl. HCI* (Sep. 1991).
10. Brock, S.A., Steven, L.E., *Handbook of Usability Principles*. Center for Learning, Instruction, & Performance Technologies. San Diego State University. (1997).
11. Norman, D.A., *Cognitive engineering*. In D.A. Norman y S.W. Draper (Eds.), *User Centred System Design*. Hillsdale, N.J., Erlbaum. (1986).
12. Redish, J., Are we really entering a post-usability era?. *ACM SIGDOC Asterisk Journal of Computer Documentation*, vol. 19 (1), págs. 18-24. (1995)
13. Shneiderman, B., *Designing the user interface: Strategies for effective human-computer interaction*. 3rd ed., Reading, MA: Addison-Wesley (1998).
14. Constantine, L.L., Lockwood L.A.D., *Software for Use: A Practical guide to the Models and Methods of Usage-Centered Design*. Addison-Wesley. (1999)
15. Quesenbery, W., Using the 5Es to Understand Users. Available in WQUsability website in <http://www.wqusability.com/articles/getting-started.html> (2006).
16. Granollers, T., *MPLu+a*. PhD Thesis. Available in <http://griho.udl.es/catala/equip/invest/granollers.html> (2004).
17. Lorés, J., González, M.P., Pascual, A., *Primera Etapa de la Iniciativa UsabAIPO: usabilidad de páginas de inicio de universidades españolas*. VI *Interacción'05*, pp 217-221 (2005).
18. Collazos, C., Guerrero, L., Pino, J., and Ochoa, S., A Method for Evaluating Computer-Supported Collaborative Learning Processes. *International Journal of Computer Applications in Technology*, Vol. 19, Nos. 3/4, pp.151-161 (2004).
19. Appelt, W., Ruland, R., Gómez Skarmeta, A., Stahl, G.: *Synergeia Version 2 User Manual*. Available in <http://bscl.fit.fraunhofer.de/download/SynergeiaManual.pdf> (2002).
20. Stahl, G., *Groupware Goes to School*. 8th *Workshop on Groupware*. Chile, pp 1-4 (2002).
21. Gutwin, C., and Greeberg, S., The mechanics of collaboration: Developing low cost usability evaluation methods for shared workspaces. *IEEE 9th International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET-ICE'00)*, 2000.
22. Pinelle, D., and Gutwin, C., "A Review of Groupware Evaluations", *Proceedings of 9th IEEE WETICE 2000*, Gaithersburg, Maryland (2000).
23. Pinelle, D., Gutwin C., *Groupware walkthrough: adding context to groupware usability evaluation*. In *Procs of the SIGCHI'02*, pp 455-462. ACM Press (2002).
24. Georgiakakis, P., Retalis, S., *Demystifying the asynchronous network supported collaborative learning*. *Journal of Comp. App. in Tech.*, Vol. 25. No.2/3, pp. 116-127 (2006)
25. Fjuk, A., & Smordal, O., *The computers incorporated role in work*. In Buch, N. J. Et all (Eds.), *IS Research in Collaboration with Industry* (pp.207-221). Aalborg University (1997)

The Remote Control Approach - How to Apply Scaffolds to Existing Collaborative Learning Environments

Andreas Harrer, Nils Malzahn, and Benedikt Roth

COLLIDE research group, University of Duisburg-Essen
Department of Computer Science and Interactive Systems
Lotharstr. 63/65, 47057 Duisburg, Germany
{harrer, malzahn, roth}@collide.info.de

Abstract. In this paper we present an architecture for the integration of tutoring and process scaffolds into existing collaborative applications. The architecture allows to combine existing research results concerning collaborative processes and their formalization, and existing and tested collaborative learning environments. The architecture allows to control the learning environments either by a human or a pedagogic agent. Both types of tutors are using the same set of primitives - either via an intuitive user interface or a slim Java interface. To prove the soundness of the architecture an example is given using IMS LD collaboration scripts with Coppercore as a workflow engine controlling the Cool Modes environment. A description of the possible applications of the architecture in intelligent tutoring systems gives an insight into the opportunities opened by such a flexible approach. The paper closes with an outlook concerning the use of the architecture with more and different learning systems and process control engines.

1 Introduction – Structuring and Scaffolding Collaboration

Collaboration has become an important factor in learning activities, especially in disciplines that require substantial phases of working in teams, such as computer science, communication sciences etc. This can be seen in the emergence of the research field Computer-Supported Collaborative Learning (CSCL) in the last decade. Yet, just reducing the computer-based support to providing the suitable technological means to communicate, which is often called Computer-Mediated Communication, is most often not sufficient to promote the collaborative learning activity: Studies, like Weinberger [1] showed, that collaboration does not happen effectively in every situation just by initiating the collaborative situation.

Scaffolds [2] or collaboration scripts [3] are means to structure the learning activity and support the learners in organizing their activities or acquiring the skills to collaborate effectively. Thus their use in computer-based learning support environments (LSE) is a major topic of recent research in the CSCL

community ([4]; [5]). At the moment the term "script" is used in a highly ambiguous way: The pedagogical rationale of a collaborative learning activity, such as introducing collaboration by splitting the task such that interaction happens at the split (cf. [6]), that shapes the general context and sequence of a whole learning activity is called a CSCL script, as well as the fine-grained prescription how argumentation should happen in complete argumentative sequences (cf. [7]).

Interestingly a parallel discussion occurs also in a field of computer-supported learning that has evolved independently of CSCL, the discipline of Intelligent Tutoring Systems (ITS): The support of the learners by the system to promote them in the learning process is often called tutoring or interventions. The components called "intelligent tutors" or "pedagogical agents" are used comparably ambiguous as the term "script" in CSCL with respect to the granularity and competencies the tutor/agent should provide.

It is obvious that the expertise and experiences of these two fields should be combined in collaborative computer-supported learning activities. One of the grand challenges for the shared interest between the communities will be the representation and implementation of scaffolds respectively tutoring processes for collaborative scenarios. The definition of formal models for collaboration support results in the explication of the pedagogical and psychological rationale for the scientific exchange between researchers and practitioners, but also in the practical application of the models in computer-based learning environments. This article will present our approach of combining aspects from CSCL, pedagogical design, and ITS in an integrated architecture for supporting collaborative learning activities.

2 Formal Models of Learning Processes and Collaborative Applications

Up to now complex learning support environments and explicit scaffolding/tutoring models are largely unrelated and co-exist, but do not co-operate. On the one hand LSEs, such as WISE [8], CoLab [9] or Belvedere [10], either have a specific ("hard-wired") process model embedded or do not have an explicit learning process model at all. On the other hand environments that use explicit process models for supporting the learning process, typically fall short in at least one of these two criteria: Re-usability of the process model in other contexts: most systems using a formal model for structuring the interaction between the learner(s) and the system define their own proprietary model for the learning process which is not understandable and thus re-usable by other applications: this may happen because of proprietary formats that cannot be mapped to other formal approaches, a lack of explicitness of the operational semantics of the model, or a lack of explicitness of the model itself, which is often deeply intertwined with the graphical user interface. Among the explicit models for defining the learning process are production rule systems [11], automata-based models [12], and flow-oriented models [13]. Expressiveness for complex learning processes: systems that have explicit mechanisms for structuring activities usually tend to

have a very narrow focus, such as sequencing the presentation of learning material in web-based hypertext systems or intervening on the first deviation from an "ideal" learning path [14]. Especially more coarse-grained learning activities, such as experimentation, model construction, and argumentation are usually not scaffolded in these systems. There are very few approaches, that explicitly try to scaffold collaboration with adaptive approaches: coming from the ITS area, the "Collaborative Tutor" approach in [15] attempts to support the fine grained level of user actions in relatively small problem-oriented tasks, such as object-oriented modelling. The GridCOLE approach [16], that is rooted in the CSCL field, combines the explicit description of coarse-grained learning activities with the launching of services suitable for the specific activity. IMS Learning Design [17] can be considered as a formal approach with explicit representation of both the models and the operational semantics, even though both aspects could be discussed even more precisely as in [18]. Surprisingly up to now learning design documents as process scaffolds or "scripts" are usually oriented towards delivery of web-content and some simple services, such as conference tools. Yet, making the learning processes explicit in a formal specification, such as IMS LD, offers also the possibility to re-use the pedagogical rationale that is reflected within the specification and define more complex learning activities than just sequenced content delivery.

We also assume that the formal character of IMS LD can be utilized to scaffold and apply tutoring support for pre-existing LSEs. The availability of learning design engines (LDE), such as CopperCore¹, can provide explicit process support without having to implement a process model from scratch for each individual environment, if we can meet the challenge of integrating pre-existing LSEs and LDEs in a flexible, interoperable architecture.

In the next section we will present our approach to achieve this synergy between both lines of computer-based learning and an architecture supporting this approach. In the subsequent section we will discuss both the re-use of pre-existing learning support environments and the re-use of artefacts within a learning process in an implementation utilizing the IMS-LD standard for learning processes and an IMS-LD engine for the execution of complex learning processes in external learning support environments.

3 A Flexible Architecture for Tutoring in Collaborative Settings

We propose an approach that aims at a clear separation of the learning design engine together with the specification and implementation of the learning flow (as LD documents) and the collaborative learning environments. In this proposal we assume that the learners interact exclusively with the LSE without having to know anything about being "scripted" or "scaffolded" by the LDE

¹ CopperCore — The IMS Learning Design Engine, <http://coppercore.sourceforge.net>

respectively the LD document. According to Vogten, Koper, Martens, and Tattersall [19] learning design engines can be considered as a collection of finite state machines that react to changes of properties with state transitions by sending events of a specific output alphabet. In the loosely-coupled connection of an engine with a learning support environment presented in figure 1, the engine controls the learning environment with output events (such as "start a new phase", event 1.), defined as a vocabulary for a set of environments, that are mapped by the environment to its existing functionality (such as "create new workspace", the configuration of the LSE through event 1.1). Since the LDE interacts closely with the LSE, the LSE is more than an IMS LD service which is not monitored and controlled by the LDE during the activity. The learners interacting with the learning support environment create events (user action 2.), such as "phase is completed" (either directly or monitored by the LSE), that map to the input alphabet of the engine's state machines and are propagated to the LDE (message 2.1). The triggered state transition (message 2.2) causes the learning process to advance and will again trigger control messages (event 3.) to be accepted by the LSE. In that way we get the regulation cycle of figure 1 with the LDE and the LSE influencing each other's state. Using a generic vocabulary of communication primitives between the LDE and LSE has the advantage, that the LD document can be used with a variety of different LSEs without any changes to the document, given that the LSE can make use of primitives of the vocabulary.

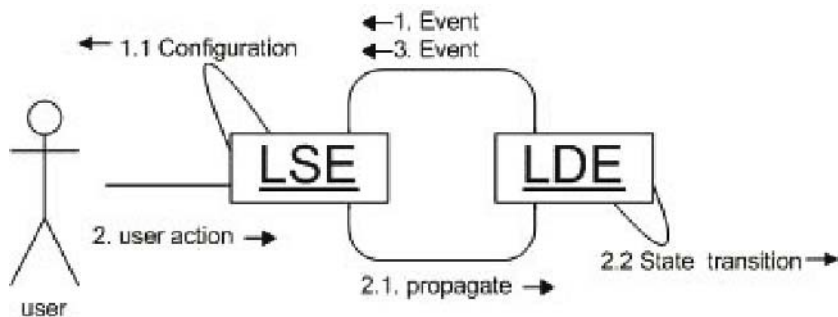


Fig. 1. UML communication diagram for interaction schema between LDE and LSE

For the concrete realization of our approach we defined an architecture that brings together LSEs and LDEs without having to make substantial changes in either of the two components: the schematic overview of the architecture can be found in figure 2 and the components introduced have the following function:

Engine Extension (CopperCore Extension): this component extends the event propagation mechanism of the learning design engine, so that on state transitions within the engine, events are sent to the LSE to remotely control the learning process according to the LD document's description. This event is sent indirectly to the LSE via the Remote Control Component.

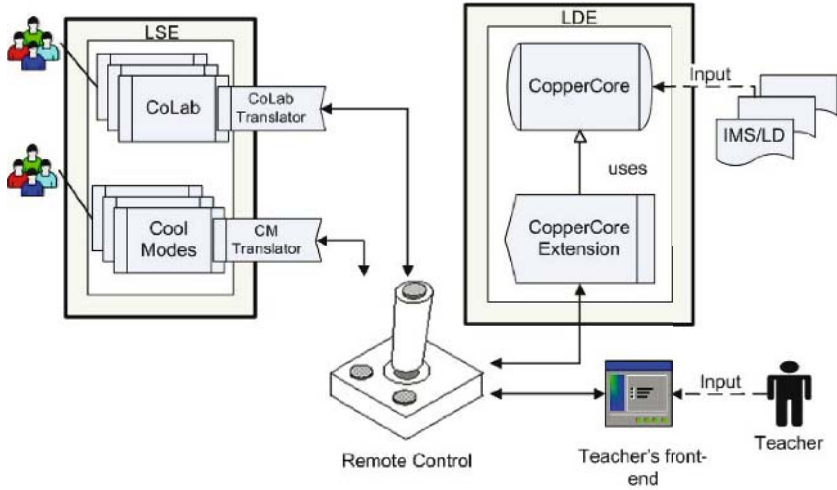


Fig. 2. Remote Control Architecture for interaction between LDE and LSE

Remote Control Component: this component is the mediator between LDE and LSE; it maps events coming from the LDE to one or more communication primitives, that build the vocabulary for remotely controlling learning support environments, such as CoLab or Cool Modes [20]. These "commands" are then sent to the "remote API" of the specific LSE.

LSE Remote API (Translator): this interface accepts communication primitives that have been defined for a variety of different LSEs and maps these primitives to the specific functionality available in the concrete LSE. For example the communication primitive "Show workspace for voting phase" could be mapped to calling the functionality "Make visible a workspace with title 'Decision on Solution' and add a Voting Plugin" in the Cool Modes environment (see figure 4). The primitive that has been sent out from the Remote Control Component to the subscribers of this primitive (all LSEs that understand the primitive) is then translated to a call of the respective functionality of the LSE; thus this can be considered a remote call of the LSE functionality by the Remote Control Component. To integrate an LSE into the proposed architecture a translator mapping the command primitives to the LSE in question has to be implemented.

An interesting feature of this architecture is, that besides our main purpose, i.e. the realization of collaboration scaffolds in pre-existing learning support environments, the remote control can be used by a variety of different actors (in the socio-technical sense of actors being both humans and technical systems):

- A virtual agent/tutor, that has some model for scaffolding/tutoring the learning, such as in [21], if it uses messages that can be mapped by the remote control to the communication primitives of LSEs. The LDE component can be considered our standardized type of such a virtual agent using

the IMS LD dialect, but it could be replaced by a full-fledged intelligent tutoring agent regulating the learning process adaptively using the remote control vocabulary.

- A human teacher, who can react at runtime to the learning situation and give hints, additional tools, and/or instructions, as she thinks are appropriate. This intended use-case of the remote control can be seen in the lower right of figure 2, where the teacher has a dedicated user-interface.
- A human administrator, who can use the remote control with a similar user interface to the teacher’s to setup collaborative sessions, assign rights and roles to the students. Such an interface can substantially reduce the administrative effort in setting up experiments and practical use of learning support environments, as was prototypically shown in an early version of our architecture in [22]. In the next section we will present some details of our concrete prototypical implementation of the Remote Control approach and architecture.

Of course the remote control can be used by several actors at the same time (e.g. teacher and virtual agent), but this might produce inconsistency in the regulation of the process by the agent. On the other hand this provides a convenient way for the teacher to react to unforeseen situations in the learning activity, such as breakdowns of groups or tools.

4 Prototypical Implementation

As a proof of our concept we chose to combine the Cool Modes application with the CopperCore Engine, currently the most advanced IMS LD engine and the Reload player², a graphical interface for run time configuration of learning designs. The prototypical implementation of our architecture will be described in three steps: At first we will describe the extensions of the CopperCore engine, second we will sketch the implementation of the Remote Control plus an extension of the Reload IMS LD Player as a teacher’s frontend and afterwards the extension of the Cool Modes translator (cf. figure 2) will be explained.

4.1 CopperCore

Since the CopperCore engine shall be used as an agent that uses the Remote Control, it is essential to be able to receive events from the engine. This is done by adding an *EventPublisher* to the CopperCore engine that notifies the Remote Control when changes of CopperCore’s inner state machine occur. Another possible approach would have been to poll the CopperCore’s inner state continuously. Although this approach may have avoided changes to the CopperCore engine completely, we decided to implement the first option to avoid the resource consuming polling requests of Remote Control Component. CopperCore already provides an

² RELOAD Project – Learning Design Player <http://www.reload.ac.uk/ldplayer.html>

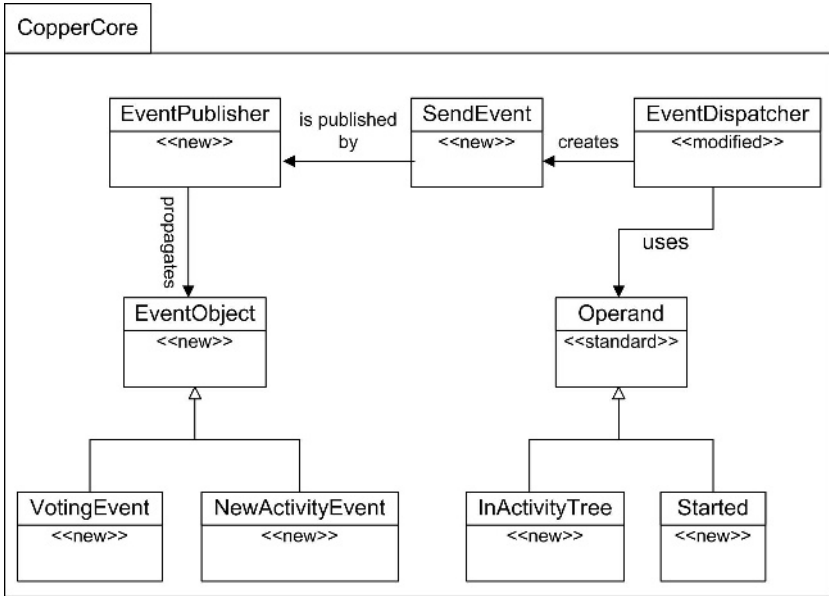


Fig. 3. UML-like diagram for extensions of the CopperCore engine

event propagation mechanism managed by an *EventDispatcher* which is used internally to calculate the consequences of advancements in the Learning Design script (e.g. if a learning-activity is completed, a role-part is completed). The given set of expressions and actions of the original CopperCore engine were extended by introducing the conditions *Started* and *InActivityTree* as well as a class called *SendEvent* that notifies the Remote Control Component via the *EventPublisher*. These two new conditions help us to synchronize the Remote Control with the CopperCore engine and to distinguish which collaborative application has to be contacted because of the particular change in the activity tree.

The parser of CopperCore has been extended to fire the *SendEvent* whenever an activity ends. This was necessary to ensure that the *SendEvent* is executed if the state changes are of interest (e.g. if an act has been made visible). Whenever a *SendEvent* occurs, an appropriate *EventObject* (e.g. a *NewActivityEvent*) is created and sent to the Remote Control (see figure 3). The communication between the CopperCore engine and the Remote Control component, as well as the communication to and from the LSE, is realised via Java Message Service³ (JMS), because it was convenient for the presented combination of LSE and LDE. Nevertheless it is possible or even necessary, depending on the controlled LSEs, to exchange the communication channel with other techniques in the future. For the communication format we chose XML, since IMS Learning Design is specified in XML and CopperCore makes already extensive use of it. If an *EventObject* is created all relevant data is collected from the CopperCore

³ Java Message Service(JMS) <http://java.sun.com/products/jms>

engine and an XML string is constructed. This string is sent by the EventPublisher via a JMS TextMessage to the Remote Control. If a message arrives at the Remote Control the EventObject is re-constructed by parsing the received XML string. EventObject is used here as a placeholder for a family of concrete events like the above mentioned NewActivityEvent. These events are subclasses of the EventObject as shown in figure 3.

4.2 Remote Control

The Remote Control is the intermediate device between the learning process management and the collaborative applications. It should be able to run either without human interaction, monitored and operated by a pedagogic (computer) agent or operated by a human. Both types of actors shall be enabled to interact with the learners through the connected LSEs. Concerning the idea of using IMS LD documents to scaffold the learning process we wanted to provide an intuitive and easy-to-use (teacher) interface to configure the LDE (i.e. manage learning scripts, create users, assign them as appropriate etc.). Since the Reload LD Player already has a quite intuitive UI and supports to start the CopperCore server we used this application as a starting point to add some functionality. First of all the user management has to combine the internal users of the CopperCore engine and the users

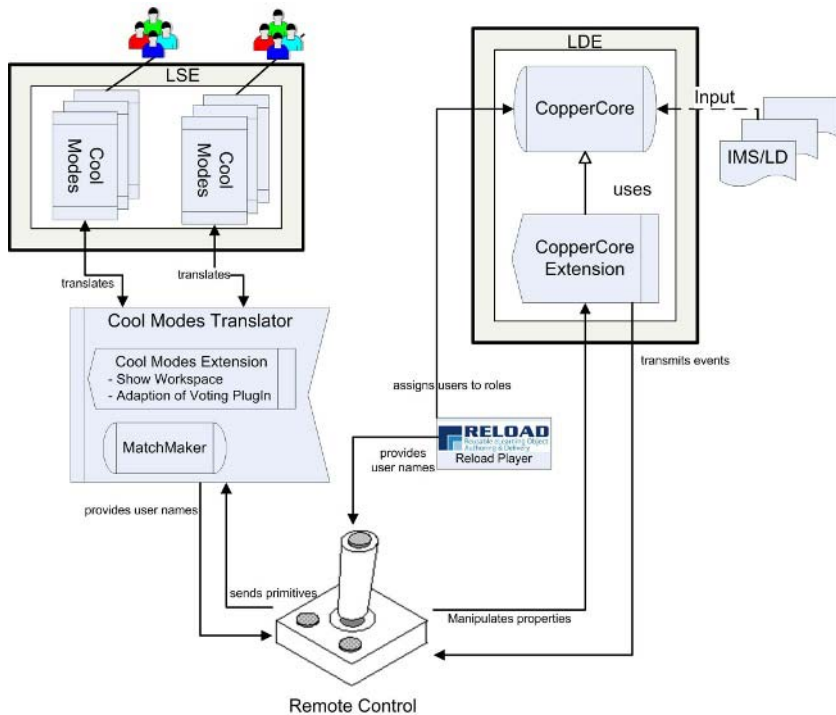


Fig. 4. Overview of the Remote Control implementation

sitting in front of the LSEs. The Remote Control maps the Copper Core’s participants to the users of the learning environments. In our example the information about the LSE users is provided by the MatchMaker [23] server that is used as a collaboration server for the Cool Modes and FreeStyler environment.

The assignment of users to units of learning can either be done by humans like it was already possible in the original Reload LD Player or by an agent which notices new users and assigns them automatically to a script and starts it afterwards. The human actor on the one hand has the option to create users and assign them from the beginning (e.g. while preparing a lecture) and on the other hand he can wait for the users to start their LSEs, so they will automatically show up in the Remote Control and assign them to an specific script on the fly.

4.3 Application Specific Extensions

As stated above our goal is to support a wide range of existing applications, so we specified Translators which shall be loosely attached to each LSE. Those classes that ”translate” the command primitives sent by the remote control to applications specific events or method calls implement the specified Translator interface. These classes are also responsible to ”translate” the application specific results to primitives the remote control can handle. Depending on the specific LSE this can be done with more or less effort. In general we think that applications that support collaborative learning should be easily adaptable to be remotely controlled, because in most cases there already are events to be distributed among other clients of the LSE. In our example the Translator can act as an additional client in the collaborative environment. If it is possible to implement the translator as such a way, the LSEs do not have to know anything about the whole learning process and just react to the instructions of the Remote Control, which are again based on the Learning Design engine. So the Translator acts like a tutoring agent.

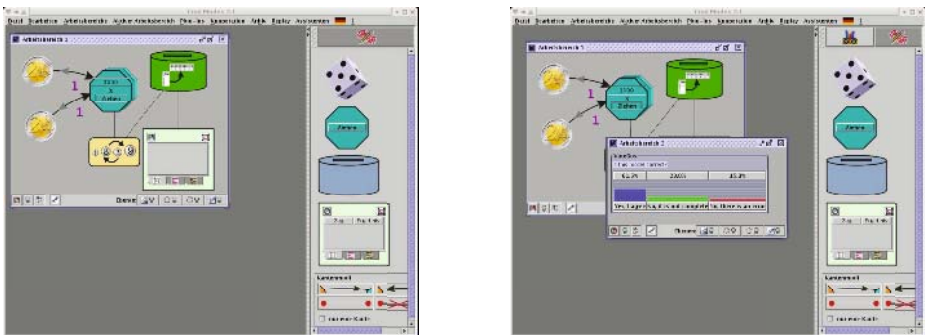


Fig. 5. The Cool Modes learning environment before (left) and after (right) transmission of the communication primitive ”ShowWorkspace for VotingPhase” from Copper-Core Engine. In the right part the voting plugin was added (small icon in top right corner) and an additional window appeared to conduct the voting.

For example during a collaboration session with the goal to model a stochastic experiment (see figure 5) the participants indicate that they do not want to change the model anymore. The users' indication is sent to the CopperCore engine via the Cool Modes translator and the Remote Control. The learning design script changes its states and tells the Remote Control to start a voting activity. In turn the Remote Control distributes a *Show workspace for voting phase*

```

<imsld:play identifier="PLAY-1" isvisible="true"> ...
  <imsld:act identifier="Act-1"> <imsld:title>Model a Coin Experiment</imsld:title>
    <imsld:role-part identifier="RP-1-1">...
      <imsld:activity-structure-ref ref="LA-Modelling"/>
    </imsld:role-part>

    <imsld:role-part identifier="RP-1-2">...
      <imsld:learning-activity-ref ref="LA-Further-Modelling"/>
    </imsld:role-part>

    <imsld:role-part identifier="RP-1-3"> ...
      <imsld:learning-activity-ref ref="LA-Presentation"/>
    </imsld:role-part>

    <imsld:complete-act>
      <imsld:when-role-part-completed ref="RP-1-3" />
    </imsld:complete-act>
  </imsld:act>

  <imsld:complete-play>
    <imsld:when-last-act-completed />
  </imsld:complete-play>
</imsld:play> <imsld:conditions>
  <imsld:if>
    <imsld:is>
      <imsld:property-ref ref="P-Voting"/>
      <imsld:property-value>true</imsld:property-value>
    </imsld:is>
  </imsld:if>

  <imsld:then>
    <imsld:hide>
      <imsld:learning-activity-ref ref="LA-Further-Modelling"/>
    </imsld:hide>
    <imsld:show>
      <imsld:learning-activity-ref ref="LA-Presentation"/>
    </imsld:show>
  </imsld:then>

  <imsld:else>
    <imsld:show>
      <imsld:learning-activity-ref ref="LA-Further-Modelling"/>
    </imsld:show>
    <imsld:hide>
      <imsld:learning-activity-ref ref="LA-Presentation"/>
    </imsld:hide>
  </imsld:else>
  ...
</imsld:conditions>
</imsld:method>

```

Fig. 6. IMS LD Play used in the example script - Note that 3 Role Parts are needed since IMS LD does not allow to switch forth and back between acts

command primitive. The Translator translates this primitive application specific commands for the creation of a new window containing a voting opportunity.

While the availability of a voting option is quite wide spread and therefore reasonable to be taken as a command primitive the concrete implementation of the voting is application specific. So every LSE has to use its own means to enable the students to give their vote. This so called VotingService enables the user to choose between different options which in turn affect the learn flow of the IMS LD play. Technically the voting results returned by the Translators are stored in IMS properties. This enables their use as variables that determine the further steps in Copper Core's learning script. This technique can easily be adopted for other activities like evaluation of tests etc.

To conclude this section we describe the IMS LD script, which was used to control the above mentioned process. The IMS LD document consists of a one acted play structured in three Role-Parts, whereas one Role-Part (RP-1-2) can be skipped by the users based on their decision in a voting (cf. figure 6).

Each voting has a title containing the question and at least two options to select from. The options again have a title, refer to an IMS LD property and indicate to what value this property should be set, if this option is the result. Furthermore the attribute voting-type indicates whether this service is scripted or unscripted. A scripted voting service results in getting configured and placed for the user ready to use, in contrast to an unscripted voting service which just indicates the user has the optional ability to configure his own voting. In Cool Modes the latter will result in the Voting Plugin getting loaded and in case of a scripted voting event a voting node provided by this plugin will get placed on a shared workspace.

In this case, the environment shown in figure 7 consisting of a voting service is referenced in a learning activity. Thus the learners have the ability to vote if their model needs further work or whether it's already correct and the next modelling phase should be skipped. This works by using IMS LD Level B conditions and showing or hiding the relevant activity based on the value of the property P-Voting.

```
<imsld:environment identifier="LD-Environment1">
  <imsld:title>Modelling Environment</imsld:title>
  <imsld:service identifier="LD-Service1">
    <imsld-ext:voting identifier="LD-Voting1" voting-type="scripted">
      <imsld:title>Is this model correct?</imsld:title>
      <imsld-ext:choice identifier="LD-Choice1" property-ref="P-Voting">
        <imsld:title>Yes</imsld:title>
        <imsld:property-value>1</imsld:property-value> </imsld-ext:choice>
      <imsld-ext:choice identifier="LD-Choice2" property-ref="P-Voting">
        <imsld:title>No incomplete</imsld:title>
        <imsld:property-value>2</imsld:property-value> </imsld-ext:choice>
      <imsld-ext:choice identifier="LD-Choice3" property-ref="P-Voting">
        <imsld:title>No incorrect</imsld:title>
        <imsld:property-value>3</imsld:property-value> </imsld-ext:choice>
    </imsld-ext:voting>
  </imsld:service>
</imsld:environment>
```

Fig. 7. Environment with a voting service used in the example script

5 Conclusion

We have presented a flexible architecture to combine Learning Support Environments with Computer Supported Collaborative Scripting approaches. This was done because we think that the ideas of scripts should be transferred to well elaborated learning environments which have been developed in recent years. The idea is to combine the flexibility of learning scripts, which can be adapted to different learning groups and tasks, with the often task-oriented and domain specific ITS systems. Since the already present systems shall not be rewritten we decided to use a loosely coupled approach that allows to be adjusted for different learning support environments on the one hand and on the other hand different scaffolding agents to be applied. We proved our conceptual ideas by presenting the current prototypical implementation of the proposed architecture using the CopperCore engine and the Cool Modes learning environment. For the specification of the learning processes the Learning Design Standard is used with no limitations on the tools used to create the LD documents. Currently we are working on means for graphical group formation to ease the effort for teachers. For this we plan to use the SessionManager [22]. This solves the matter for human agents. Another open question is the definition of an automatic mechanism for group formation in IMS LD scripts. Currently groups are not explicitly contained in the specification, yet there are some workarounds discussed in the literature such as in [24] where groups are represented by specific roles. So either substitutes have to be found or new constructs have to be introduced. Following our principle of being as little intrusive as possible, we prefer to extend existing constructs in a way that they conform to the syntax specification of standard IMS LD and support the semantics needed for automatic group formation.

The second line of research we plan to pursue is the transfer of the approach to other collaborative applications with the final goal to create scripted applications which enable the learning designers to specify the interoperability between application rather than programming it. Given this it will be possible to use one learning flow for more than one learning environment at the same time. That means the script (agent, tutor) can be used for other collaborative learning environments, enabling students using different learning environments to collaborate with each other.

6 Outlook

With respect to the practical use of our architecture we are currently working on the implementation of more complex learning processes, such as scientific inquiry learning [25]. These processes consist of several phases, such as hypothesis generation, experimentation/simulation, evaluation, and potentially several cycles through these phases. Resources and artefacts that have been available resp. produced in earlier phases, should be available for the learners at later stages to reflect on it and improve their hypotheses in the next cycle. Here it becomes obvious that resources and objects need references through which they can be addressed to be used in multiple phases and cycles of the learning process.

Because the artefacts within the learning process, such as a simulation model created by the student, can change over time, they have to be available both for the LDE and the LSE: the LDE has to initiate the re-appearance of the object in the LSE according to the description of the learning process, while the LSE potentially has to change the content of the object, when the student modifies it in the learning process.

To achieve this flexible use of learning objects in different phases and cycles we chose to represent each re-usable object as a *global personal property* in the LD description. The property is globally defined, because while the initial state of the property can be set in the LD document, the external LSE can manipulate the content during the learning process via the *URI* the property is associated with. A personal property is required here, because every student involved in the learning process might possess an individual version of the artefact, e.g. the simulation model that should be produced in the inquiry process.

References

1. Armin Weinberger. *Scripts for Computer-Supported Collaborative Learning Effects of social and epistemic cooperation scripts on collaborative knowledge construction*. PhD thesis, University of Munich, 2003.
2. C. Quintana, Joseph Krajcik, and Elliot Soloway. A case study to distill structural scaffolding guidelines for scaffolded software. Technical report, CHI, 2002.
3. A. M. O'Donnell and D.F. Dansereau. Scripted cooperation in student dyads: A method for analyzing and enhancing academic learning and performance. In R. Hertz-Lazarowitz and N. Miller, editors, *Interaction in cooperative groups: The theoretical anatomy of group learning*, pages 120–141. Cambridge University Press, 1992.
4. A. Weinberger, F. Fischer, P. Häkkinen, P. Dillenbourg, and A. Harrer. Computer-supported scripting of interaction in collaborative learning environments. Workshop at the Conference on Computer Supported Collaborative Learning, 2005.
5. A. Harrer, D. Hernandez Leo, and Y. Dimitriadis. Languages for modeling of collaborative learning processes: Formalization, practical uses and tools. Workshop at the Conference on Computer Supported Collaborative Learning, 2005.
6. P. Dillenbourg. Split where interaction should happen. In F. Fischer, H. Mandl, J. Haake, and I. Kollar, editors, *Scripting Computer-supported Collaborative Learning - Cognitive, computational, and educational perspectives*. to appear, 2006.
7. S. Leita. The potential of argument in knowledge building. *Human Development*, 43:332–360, 2000.
8. The web-based inquiry science environment. <http://wise.berkeley.edu> as of 3rd May 2005.
9. W.R. van Joolingen, A.W. Lazonder, T. de Jong, E.R. Savelsbergh, and S. Manlove. Co-lab: Research and development of an online learning environment for collaborative scientific discovery learning. *Computers in Human Behavior*, (21):671–688, 2004.
10. D. Suthers, A. Weiner, J. Connelly, and M. Paolucci. Belvedere: Engaging students in critical discussion of science and public policy issues. In J. Greer, editor, *Proceedings of AI-ED 1995*, pages 266–273, 1995.

11. V. Aleven, B. McLaren, I. Roll, and K. Koedinger. Toward tutoring help seeking: Applying cognitive modeling to meta-cognitive skills. In *Proceedings of the Seventh International Conference on Intelligent Tutoring Systems (ITS-2004)*, pages 227–239, 2004.
12. A. Martens. Case-based training with intelligent tutoring systems. In *Proceedings of the International Conference on Advanced Learning Technologies*, pages 191–195, 2004.
13. P. Dawabi and M. Wessner. Modellierung von blended learning szenarien. In *Fachtagung "e-Learning" der Gesellschaft für Informatik*, pages 115–126, 2004.
14. J.R. Anderson, A.T. Corbett, K. Koedinger, and R. Pelletier. Cognitive tutors: Lessons learned. *Journal of the Learning Sciences*, 4:167–207, 1995.
15. B. McLaren, L. Bollen, E. Walker, A. Harrer, and J. Sewall. Cognitive tutoring of collaboration: Developmental and empirical steps towards realization. In *Proceedings of the Conference on Computer Supported Collaborative Learning Conference (CSCL-05)*, pages 418–422, 2005.
16. L. Bote, L.M. Vaquero-Gonzalez, G. Vega-Gorgojo, Y. Dimitriadis, J. Asensio-Perez, E. Gomez-Sanchez, and D. Hernandez-Leo. A tailorable collaborative learning system that combines ogsa grid services and ims-ld scripting. In *Proceedings of the X International Workshop on Groupware, CRIWG 2004*, pages 305–321. Springer, 2004.
17. R. Koper and C. Tattersall, editors. *Learning Design: A Handbook on Modelling and Delivering Networked Education and Training*. Springer, 2005.
18. R. Amorim, M. Lama, E. Sanchez, and X. Vila. A learning design ontology based on the ims specification. In R. Koper, C. Tattersall, and D. Burgos, editors, *Current State on IMS Learning Design. Proceedings of the UNFOLD/Prolearn joint workshop*, pages 203–225, Valkenburg, 2005.
19. H. Vogten, R. Koper, H. Martens, and C. Tattersall. An architecture for learning design engines. In R. Koper and C. Tattersall, editors, *Learning Design*, pages 75–90. Springer, 2005.
20. N. Pinkwart. *Collaborative Modelling in Graph Based Environments*. PhD thesis, University of Duisburg-Essen, 2005.
21. A. Martens and A.M. Uhrmacher. A formal tutoring process model for intelligent tutoring systems. In *Proceedings of the 16th European Conference on Artificial Intelligence, ECAI 2004*, pages 124–128, 2004.
22. M. Kuhn, M. Jansen, A. Harrer, and H.U. Hoppe. A lightweight approach for flexible group management in the classroom. In *Proceedings of the International Conference on Computer Supported Collaborative Learning (CSCL2005)*, pages 353–357, 2005.
23. M. Jansen. Matchmaker tng - a framework to support collaborative java applications. In H.U. Hoppe, F. Verdejo, and J. Kay, editors, *Proceedings of Artificial Intelligence in Education*, pages 529–530. IOS Press, 2003.
24. D. Hernandez Leo, J. I. Asensio Perez, Y. Dimitriadis, M. L. Bote Lorenzo, I. M. Jorin Abellan, and E. D. Villasclaras Fernandez. Reusing ims-ld formalized best practices in collaborative learning structuring. *Advanced Technology for Learning (extended version of the WBE05 paper)*, 2(4):223–232, October 2005.
25. Ton de Jong and Wouter van Joolingen. Discovery learning with computer simulations of conceptual domains. *Review of Educational Research*, 68:179–201, 1998.

Polyphonic Support for Collaborative Learning

Stefan Trausan-Matu¹, Gerry Stahl², and Johann Sarmiento²

¹“Politehnica” University of Bucharest,
313, Splaiul Independentei,
and

Romanian Academy Research Institute for Artificial Intelligence,
13, Calea 13 Septembrie
Bucharest, Romania
trausan@cs.pub.ro

²Math Forum and the I-School at Drexel University,
3210 Cherry Street, Philadelphia, PA 19104, USA
gerry.stahl@cis.drexel.edu, jsarmi@drexel.edu

Abstract. This paper argues that one reason for the success of collaborative problem solving where individual attempts failed is the polyphonic character of work in small groups. Polyphony, a concept taken from music, may occur in chats for problem solving, transforming dialog into a “thinking device”: Different voices jointly construct a melody (story, or solution) and other voices adopt differential positions, identifying dissonances (unsound, rickety stories or solutions). This polyphonic interplay may eventually make clear the correct (“sound”) construction. The paper illustrates the polyphonic character of collaborative problem solving using chats. It also proposes prototyped software tools for facilitating polyphony in chats.

1 Introduction

This paper is considering the role of polyphonic inter-animation of multiple voices in collaborative learning. Inspired by the work of Mikhail Bakhtin, this idea shed new light on the dialogic nature of discourse in human language. It would also have consequences for the design of collaborative learning environments.

In polyphony, several voices jointly construct a melody (or a story, or a potential solution in the textual-chat case) while other voices situate themselves on a differential position, identifying dissonances (unsound, rickety stories or solutions). This polyphonic game may eventually make clear the correct, sound solution.

The ideas are exemplified with chat excerpts for collaborative learning of mathematics problem solving, investigated in the Virtual Math Teams (VMT) project at Math Forum @ Drexel University. Inter-animation patterns in two dimensions were discovered: longitudinal (chronologically sequential) and vertical, towards two opposite trends: unity vs. difference. We consider that even individual thinking is also an implicit collaborative (dialogic) process that involves multiple voices. However, actual collaborations, in small groups of different personalities empower the dialogic process.

An environment for collaborative learning (that may be seen also as a groupware) based on the polyphonic inter-animation principles is introduced. Several modules are already implemented while others are in a final stage.

The paper continues by introducing discourse, the dialogic theory of Mikhail Bakhtin and polyphony. The next section of the paper introduces Computer-Supported Collaborative Learning (CSCL) and analyses the polyphonic welding of longitudinal-vertical unity-difference dimensions. Software tools that support the polyphonic inter-animation are presented in the fourth section. The paper ends with conclusions and references.

2 Discourse, Dialogic and Polyphony

Learning may be seen as directly related to discourse building, as Sfard remarked: “rather than speaking about ‘acquisition of knowledge,’ many people prefer to view learning as *becoming a participant in a certain discourse*” [11]. Koschmann [5] emphasized the social dimension of learning and discourse, quoting Deborah Hicks [4]: “Learning occurs as the co-construction (or reconstruction) of social meanings from within the parameters of emergent, socially negotiated, and discursive activity” (p. 136).

The above ideas follow the socio-cultural learning paradigm initiated by Vygotsky. He has a permanently increasing influence on learning theories, stating that learning is a social process, mediated by specific tools, in which symbols and especially human language plays a central role [15]. However, he did not investigate in more detail how the language and discourse are actually used in collaborative activities. It is the merit of Mikhail Bakhtin to propose a sound theory of how meaning is socially constructed.

Mikhail Mikhailovici Bakhtin extended Vygotsky’s ideas in the direction of considering the role of language and discourse, with emphasis on speech and dialog. Bakhtin raises the idea of dialogism to a fundamental philosophical category, *dialogistics*. For example, Voloshinov (a member of Bakhtin’s circle who, according to many opinions, signed a book written by his more famous friend because the former has an interdiction to publish during Stalin regime) said: “... *Any true understanding is dialogic in nature*. Understanding is to utterance as one line of dialogue is to the next” [14]. This is in consonance with Lotman’s conception of text as a „thinking device” [17], determining that: “The semantic structure of an internally persuasive discourse is not *finite*, it is *open*; in each of the new contexts that dialogize it, this discourse is able to reveal ever new *ways to mean*” [1].

Any discourse may be seen as an intertwining of at least two threads belonging to dialoguing voices. Even if we consider an essay, a novel or even a scientific paper, discourse should be considered implying not only the voice of the author. The potential listener has an, at least, as important role. The author makes a thread of ideas, a narrative. Meanwhile, in parallel to it, he must take into account the potential flaws of his discourse; he must see it as an utterance that can be argued by the listener. In this idea, discourse is similar to dialog and to music polyphony (in fact, it should not be a surprise that different art genres like music, literature and conversation have similar features), where different voices interanimate.

Discursive voices weave sometimes in a polyphonic texture, feature which Mikhail Bakhtin admired so much in Dostoyevsky's novels. They are characterized by Bakhtin as "a plurality of independent and unmerged voices and consciousnesses" [2]. However, polyphony is not only a randomly overlay of voices. It has also musicality; it is in fact one of the most complex types of musical compositions, exemplified by the complex contrapuntal fugues of Johann Sebastian Bach. "When there is *more than one independent melodic line happening at the same time* in a piece of music, we say that the music is contrapuntal. The independent melodic lines are called counterpoint. The music that is made up of counterpoint can also be called polyphony, or one can say that the music is polyphonic or speak of the polyphonic texture of the music." [7].

In polyphonic music, the melodic, linear dimension is not disturbing the differential, vertical harmony. Moreover, for example, in Bach's fugues, the voices inter-animate each other. The main theme is introduced by a voice, reformulated by the others, even contradicted sometimes (e.g. inverted) but all the voices keep a vertical harmony in their diversity.

Starting from Bakhtin's ideas, we extend these ideas to collaborative learning. Therefore, we will further describe how polyphony may arise in collaborative learning and we will propose ways of supporting it in learning environments.

3 The Polyphony of Problem Solving Chats

3.1 Collaborative Learning in Virtual Math Teams

Computer and communication technologies offer now new possibilities for collaboration, by virtualizing classroom group interaction. New types of artifacts like hypertext, the World Wide Web, chats or forums of discussions, are changing the classical learning scenarios. In addition to classical sheets of paper or blackboards for drawing diagrams and writing formulas and sequences of problem solving steps, computer animations, simulations or even virtual participants in the dialog (artificial agents) may be used now for collaboration. It is extremely important to analyze the particularities of discourse in this new context. A good example is the fact that in chats we can much more easily use a multiple threaded discourse, similar to contrapuntus in classical music than in face-to-face conversations.

The (VMT) research program investigates the innovative use of online collaborative environments to support effective K-12 mathematics learning as part of the research and development activities of the Math Forum (mathforum.org) at Drexel University. VMT extends the Math Forum's "Problem of the Week (PoW)" service by bringing together groups of 3 to 5 students in grades 6th to 11th to collaborate online in discussing and solving non-routine mathematical problems. Currently, participants interact using a computer-supported collaborative learning environment, which combines quasi-synchronous text-based communication (e.g. chat) and a shared whiteboard among other interaction tools.

At the core of VMT research is the premise that primarily, group knowledge arises in discourse and is preserved in linguistic artefacts whose meaning is co-constructed within group processes [10]. Key issues addressed by the VMT include the design challenge of structuring the online collaborative experience in a meaningful and

engaging way, and the methodological challenge of finding appropriate methodological approaches to study the forms of collaboration and reasoning that take place.

3.2 Polyphonic Inter-animation in Chats

Let us consider the following problem:

Three years ago, men made up two out of every three internet users in America. Today the ratio of male to female users is about 1 to 1. In that time the number of American females using the internet has grown by 30,000,000, while the number of males who use the internet has grown by 100%. By how much has the total internet-user population increased in America in the past three years? (A) 50,000,000 (B) 60,000,000 (C) 80,000,000 (D) 100,000,000 (E) 200,000,000

This problem was one of an eleven problems set that were used for an experiment in which a group of students had to solve first individually and after that collaboratively, using chat. It was one of the two that were not solved individually by any students but it was solved collaboratively.

Let us now consider a chat excerpt that includes the main utterances that contributed to the finding of the solution (see figure 1).

| | |
|--|--|
| 350 Mic how do we do this.. | 376 Cosi ok, so what's 11 – just guess on 10 |
| 351 Mic Without knowing the total number | |
| 352 Mic of internet users? | 386 Mic lets get back to 5 |
| | 387 Cosi i think it's more than 60,00000 |
| 357 Dan it all comes from the 30000000 | 388 Mic way to complicate things |
| 358 Mic did u get something for 10? | 389 Cosi haha sorry |
| 359 Dan we already know | 390 Mic life was good until you said that |
| 360 Mic 30000000 is the number of increase in | 391 Mic :(|
| american females | 392 Cosi they cant get higher equally and |
| 361 Mic and since the ratio of male to female | even out to a 1 to 1 ratio |
| 362 Mic is 1 to 1 | 393 Cosi oh, no wait, less than that |
| 363 Mic thats all i got to give. Someone finish it | 394 Cosi 30000000 |
| 364 Mic Haha | 395 Cosi yeah, it's that |
| 365 Cosi haha you jackass | 396 Cosi im pretty sure |
| 366 Mic Haha | 397 Mic Haha |
| 367 Dan Hahaha | 398 Mic how? |
| 368 Mic u all thought i was gonna figure it out | 399 Cosi because the women pop had to grow |
| didn't | more than the men in order to even out |
| 369 Mic U | 400 Cosi so the men cant be equal (30) |
| 370 Mic huh? | 401 Mic oh wow... |
| 371 Hal it would be 60,000,000 | 402 Mic i totally skipped the first sentencw e |
| 372 Mic Hal | 403 Cosi therefore, the 50,000,000 is the only |
| 373 Mic its all u | workable answer |
| 374 Mic See | 404 Dan very smart |
| 375 Mic i helped | 405 Cosi Damn im good |

Fig. 1. An excerpt illustrating the collaborative solution construction

Discourse begins with Dan's idea of starting from the 30000000 number specified in the problem statement (line 357). It continues with Mic's problem solving buffoonery (lines 360-364, 366 and 368-370), remarked by Cosi (line 365) and Dan (line 367): Mic seems to start writing a reasoning but he only fakes, writing fragments

of the problem statement linked by a typical phrase "... and since ...". However, this fake discourse fragment seems to belong to a mathematics speech genre and, even being a pastiche, is continued by Hal which extrapolates the 1:1 ratio from the present (as stated in problem) to the whole 3 years and advances 60000000 as a solution (line 371).

Mic continues the buffoonery (lines 372-375). After about one minute, Cosi's (incorrect) utterance "i think it's more than 60,00000" appears as a critique or as an intuition of something wrong, of some kind of an "unsuccessful story". Nevertheless, after less than another minute, she realizes that her own supposition is wrong because the ratio cannot be 1:1 or bigger.

The collaborative discourse enabled Cosi to solve the problem. She didn't solve it in the first phase, when they had to solve it individually. However, when she listened to the discourse proposing a solution (correct in the case of Dan's beginning proposal, fake at Mic and wrong at Hal), she felt the need to put herself on a different position. Therefore, the discourse acted as a tool, as an artifact that enabled Cosi to find the correct answer.

Discourse in chat collaborative problem solving has an obvious sequential, longitudinal, time-driven structure in which the listeners are permanently situated and in which they emit their utterances in a threaded manner. In parallel with this linear threading dimension, the participants situate themselves meanwhile also on a critical, transversal (or differential) position. For example, in the excerpt considered in this section, Dan's theme was continued by Mic's buffoonery, continued itself by Hal and then contradicted by a first theme of Cosi that was eventually totally changed, in its opposite. We could say that the critique of Cosi appeared as a need to bring the harmony of a correct solution.

In this longitudinal-transversal space, voices behave in an unity-difference manner. This phenomenon is not specific solely to chats. It appears also to polyphonic music: "The deconstructivist attack (...) – according to which only the difference between difference and unity *as an emphatic difference* (and not as a return to unity) can act as the basis of a differential theory (which dialectic merely claims to be) – is the methodical point of departure for the distinction between polyphony and non-polyphony." [6].

The unity and difference trends take different shapes in chat problem solving. We can include in the unity category cumulative talk [8] or collaborative utterances [9], repetitions [12], socialization or jokes. For example, many times participants in chats feel the need to joke, probably in the need to establish a closer relation with other participants, in order to establish a group flow state [3]. In fact, in all the chats we examined there is a preliminary socialization phase, inter-animation appearing not immediately after the beginning of chats.

4 Groupware for Polyphonic Inter-animation

Difference making has a crucial role in chats for collaborative learning, role which may be best understood from a polyphonic, musical perspective. The possibility of contemplating (listening), from a critical position, the ideas (melodies) of other peoples and entering into an argumentation (polyphony of voices), enhance problem

solving and enables learning through a trial-error process. Such processes appear also in individual problem solving (we can say that thinking is also including multiple inner voices) but the presence of multiple participants enhance both the possibility of developing multiple threads and, meanwhile, of differences identification. The inter-animation of the multiple perspectives of the participants, the opposition as result of contemplation and the presence of a third opinion in case of conflict, and sometimes the synthesis it brings are a better asset to success than a multi-voiced discourse performed by an individual (as inner thinking), that is inherently much less critique.

Evidence that participants permanently keep a differential position is also provided by the statistics of personal pronouns usage in chat sessions. For example, in a corpus of chats recorded in May 2005, “I” was used 727 times, much more than the usage of “we”, with 472 occurrences. First person “me” was used 84 times comparing to “us”, used only 34 times. However, the second person addressing is very well represented by 947 uses of “you”.

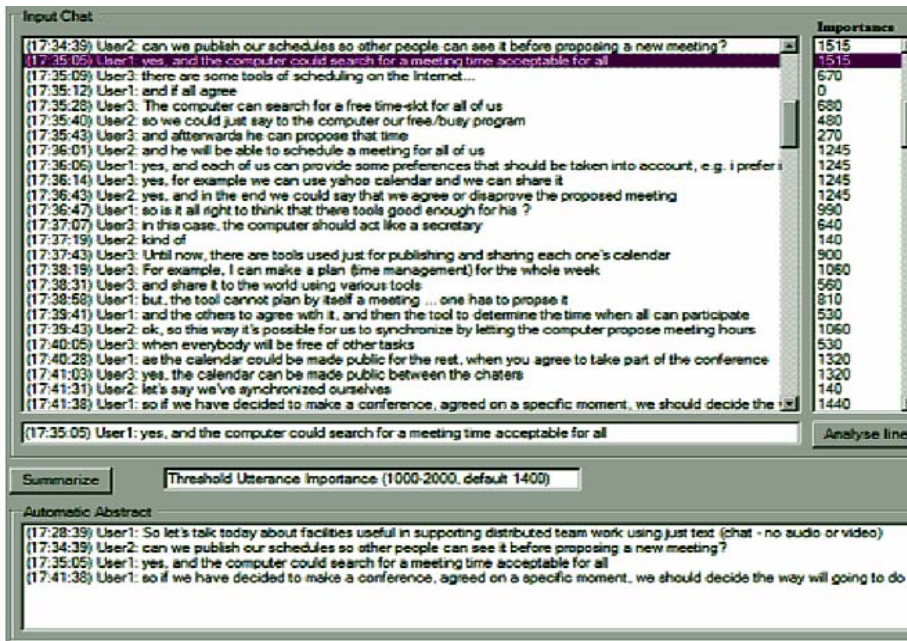


Fig. 2. A summarization module that offers an abstraction of the flow of main ideas

A natural consequence of the theoretical considerations discussed above is the need for a software support for small groups that facilitates polyphonic development. Such a groupware, named “POLYPHONY”, is now under development. The system is built around a chat system, which has some additional modules, not present in usual instant messaging. These modules offer abstractions of the ongoing chat, in the idea of making clear the flow of ideas and the other “voices” (the melody) and, the most important, to induce polyphonic, differential ideas .

In figure 2, a snapshot of one of the first implemented modules, the summarizer, is illustrated. This module builds a summary using natural language processing and heuristics. It automatically assigns an importance score to each utterance, and selects the most important utterances. Summarization is important in chats because knowing what came before, starting from clear summaries would help people to respond, to carry on the “melody” and to contribute to the polyphony with a personal, differential voice.

In addition to the summarization module, other facilities for chats, based on natural language processing are developed in POLYPHONY. They abstract and display facts about each participant, for example, the emotional state, the degree of relevance of the utterances of each participant. A module for speech acts identification has been already implemented [13]. The goals aimed by these modules are to induce self-reflection and images about the others, to facilitate inter-animation, and finally to encourage multiple voices to enter into a polyphonic framework.

5 Conclusions

Discourse in chats implies an inter-animation of multiple voices along two dimensions, the sequential, utterance threading and the transversal, differential one. These two dimensions correspond to a unity-difference (or centrifugal-centripetal, [1]) basic feature of polyphony. The unity directed dimension is achieved at diverse discourse levels by repetitions, collaborative utterances, socializing and negotiation discourse segments.

The second, differential dimension could be better understood if we consider discourse as an artifact that, taking into account that every participant in collaborative activities has a distinct personality, is a source of a critical, differential attitude. Even if individual, inner discourse may be multi-voiced, difference and critique are empowered in collaborative contexts, in a community of different personalities.

A consequence of the sequential-differential perspective for the design of CSCL environments is that they must facilitate inter-animation not only on the longitudinal dimension, through threading but also the transversal, differential, critical dimension. Tools that may enter in this category should be able to provide abstractions or summarizations of previous discourse, in order to facilitate differential position taking. They should also allow the participants to emphasize the different proposed themes and to relate them in threads, polyphonically.

Wegerif also advocates the use of a dialogic framework for teaching thinking skills by inter-animation: “meaning-making requires the inter-animation of more than one perspective“ [16]. He proposes also that questions like “what do you think?” and ‘why do you think that ?’ in the right place can have a profound effect on learning” [16]. However, he did not remark the polyphonic feature of inter-animation.

Acknowledgements

The authors wish to express their appreciation to the members of the Virtual Math Teams research project at Drexel University, whose voices are present in different

ways in the paper. They also want to thank to the anonymous reviewers for their very useful remarks. The research presented here has been partially performed under a Fulbright Scholar post-doc grant (awarded to Stefan Trausan-Matu) and was also supported by the NSF grants REC 0325447 and DUE 0333493. Any opinions, findings, or recommendations expressed are those of the authors and do not necessarily reflect the views of the sponsors.

References

1. Bakhtin, M.M. (1981). *The Dialogic Imagination: Four Essays*, University of Texas Press.
2. Bakhtin, M.M. (1984). *Problems of Dostoevsky's Poetics*, Theory and History of Literature Series, vol. 8, Minneapolis.
3. Csikszentmihalyi, M. (1990). *Flow: The Psychology of Optimal Experience*, Harper Collins.
4. Hicks, D. (1996). Contextual inquiries: A discourse-oriented study of classroom learning. In D. Hicks (Ed.), *Discourse, Learning, and Schooling* (pp. 104-141), Cambridge University Press.
5. Koschmann, T. (1999). Toward a Dialogic Theory of Learning: Bakhtin's Contribution to Understanding Learning in Settings of Collaboration, in C.Hoadley and J. Roschelle (eds.), *Proceedings of the Computer Support for Collaborative Learning 1999 Conference*, Stanford, Laurence Erlbaum Associates.
6. Mahnkopf, C.S. (2002). Theory of Polyphony, in Mahnkopf CS, Cox F & Schurig W (eds). *Polyphony and Complexity*, Hofheim, Germany: Wolke Verlags GmbH.
7. Polyphony (2005). <http://cnx.rice.edu/content/m11634/latest/>, retrieved on 4th May, 2005
8. Mercer, N. (2000). *Words and Minds. How we use language to think together*, Routledge.
9. Sacks, H. (1992). *Lectures on conversation*. Oxford, UK: Blackwell.
10. Schegloff, E. A. (1997). "Narrative Analysis" Thirty Years Later. *Journal of Narrative and Life History*, 7(1-4), 97-106.
11. Sfard, A. (2000). On reform movement and the limits of mathematical discourse, *Mathematical Thinking and Learning*, 2(3), 157-189.
12. Tannen, D. (1989). *Talking Voices: Repetition, Dialogue, and Imagery in Conversational Discourse*, Cambridge University Press.
13. Stefan Trausan-Matu, Costin Chiru, Radu Bogdan, (2004). Identificarea actelor de vorbire în dialogurile purtate pe chat, in Stefan Trausan-Matu, Costin Pribeanu (Eds.), *Interactiune Om-Calculator 2004*, Editura Printech, Bucuresti, pp. 206-214.
14. Voloshinov (1973). *Marxism and the Philosophy of Language*, New York Seminar Press.
15. Vygotsky, L. (1978). *Mind in society*. Cambridge, MA: Harvard University Press.
16. Wegerif, R. (2005). A dialogical understanding of the relationship between CSCL and teaching thinking skills. In T. Koschman, D. Suthers, & T.W. Chan (Eds.). *Computer Supported Collaborative Learning 2005: The Next 10 Years!* Mahwah, NJ, pp. 707-716.
17. Wertsch, J.V. (1991), *Voices of the Mind*, Harvard University Press.

On Supporting Users' Reflection During Small Groups Synchronous Collaboration

Meletis Margaritis, Nikolaos Avouris, and Georgios Kahrmanis

Human-Computer Interaction Group, University of Patras, GR-26500 Rio Patras, Greece
{margaritis, kahrmanis}@ee.upatras.gr, avouris@upatras.gr

Abstract. During computer-mediated synchronous collaboration there is need for supporting reflection of the partners involved. In this paper we study techniques for determining the state of an evolving collaborative process, while the activity is in progress, making the users aware of this state. For this reason, a State of Collaboration (SoC) indicator has been defined, which is calculated using a combination of machine-learning and statistical techniques. Subsequently a study was performed during which SoC was presented to a number of groups of collaborating partners engaged in problem-solving activities. It was found that this group awareness mechanism influenced in a significant way the behavior of the groups in which it was used. This study has wider implications to the design of groupware and in particular towards gaining an insight into the effect of group awareness mechanisms on computer-mediated collaborative learning.

Keywords: collaborative problem solving, small group interaction, synchronous collaboration, computer supported collaborative learning, interaction analysis.

1 Introduction

Socially inspired theories, supported by the growing development of network and collaborative technology and increased connectivity, have advanced interest on computer-based collaborative problem solving environments. These theories usually influence our considerations on effectiveness of the collaborative problem solving process, as well as the design of the collaboration-support tools involved. While most research and development of collaboration support technology has been directed towards asynchronous collaboration settings, in which usually large numbers of partners are engaged, there is a growing interest in supporting synchronous interaction in which usually small groups of actors are involved (e.g. 2 to 5 partners). In a recent outlook of the Computer-Supported Collaborative Learning field, Stahl [1] suggests that collaborative learning should be primarily studied at the small group unit of analysis where contributions coming from individual interpretive perspectives are interwoven into group cognition. There seem to be some benefits in this kind of group activity when it is computer-mediated. In cases of problem solving in rich and critical conceptual domains it appears that computer supported collaboration could be significantly effective: For activities aiming at conceptual development, communication in written

forms combined with communication through graphical representations, seems to be more effective than face to face interaction alone because it requires a more extensive thinking process [4]. The need to externalize one's own thoughts, in a written or a graphical way, could have significant effects, especially when the learning activity implies rich conceptual knowledge that is under development.

In many of these environments, when actors interact in a synchronous collaborative mode, they work in a shared workspace while they communicate using written dialogue often in combination with gestures in the shared workspace that can take the form of sticky notes and tele-pointer operations. Additional affordances of these environments contribute further towards enriched collaborative experience. For instance the substantiation of communication and interaction, which takes the form of a history log, can be used for supporting supervisor' tasks and actors' reflection and self-awareness. In Computer Supported Collaborative Learning activities, the state of evolving knowledge must be continually displayed by the collaboration participants with each other [6], thus history logs provide a treasury of information directly related to knowledge building.

The paper first presents the Synergo environment, and then describes run-time support features for building awareness at group level, illustrating their usage with the example of some validation studies.

2 The Synergo Environment

This section presents the Synergo environment from two standpoints. First, in section 2.1 the diagram building tool for supporting collaboration among students is discussed, followed by an introduction to analysis tools that aid students' reflection.

2.1 Synergo's Diagram Building Tool

Synergo (www.synergo.gr) supports synchronous collaborative building of diagrammatic representations by small groups of students.

The environment has been used in Secondary and Higher education settings for teaching computer science and other subjects. The typical client view of Synergo is shown in Fig. 1, which includes a snapshot of a concept mapping activity. Synergo supports building of different kinds of diagrams. It contains libraries for building flowcharts, entity-relationship diagrams, concept maps, data flow diagrams etc. On the left-hand side column of Fig. 1, libraries of primitive objects are shown. The activity is monitored and logfiles are generated and made available for inspection by the users or supervisors. On the right hand side the group coordination panel and the chat window is shown. Different color codes are used to represent the group members in the chat window, while various attempts have been made to represent the state of the peers during interaction. In the following sections some of these group awareness mechanisms are described.

Examples of use of Synergo in authentic educational conditions include collaborative building of algorithm flowcharts by large number of students in a distance learning course of the Hellenic Open University [7], class activities in the frame of an

introductory to computing course in a High school [8], collaborative problem solving of distant groups across two Universities [9].

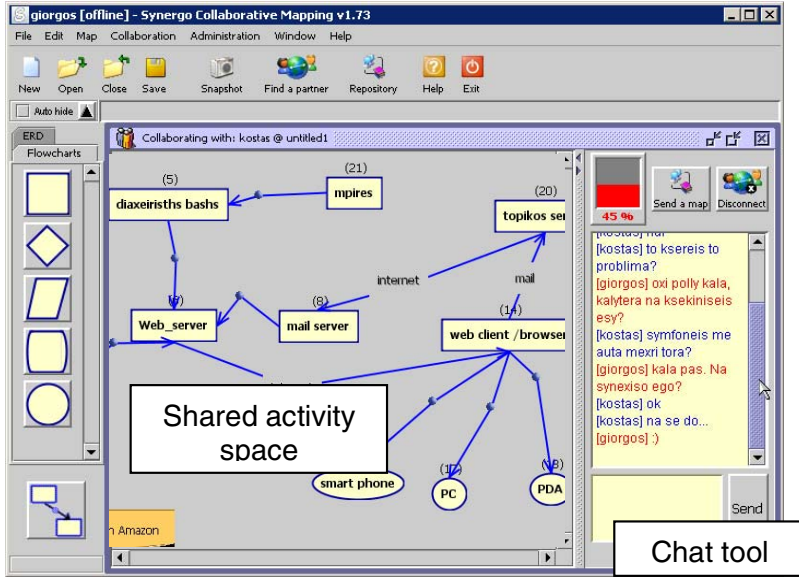


Fig. 1. The Syngo environment: client user interface

2.2 Syngo Analysis Toolkit

An additional feature of Syngo is its inherent support for analysis and supervision of the activity. So a set of analysis and supervision tools is included in the environment, typically in enabled client nodes, called *Teacher nodes*. These are mainly used by the teachers and researchers, while limited versions of the tools may be used in some cases by students as meta-cognitive aids. For instance, the student version of these tools permit playback of the so far activity while problem solving is in progress.

The main functionality of the Analysis tool is the presentation and processing of logfiles which have been produced during group activities. These logfiles contain actions and exchanged messages of group members, in sequential order. An extract of a logfile is shown in Fig. 2. The logfile is based on the same format of the exchanged control and chat messages and is stored in XML form. This file can be viewed, commended and annotated by a researcher using an adequate analysis framework, as discussed by Avouris in [10]. A related functionality of the analysis tool is its capability of post reproduction of the modeling activity, using the logfile, in a step-by-step or continuous way using the playback tool. Further annotation of activity logs through this playback tool can also be done, as discussed in more detail in [11].

The annotated or original history logfiles contain references to the objects involved in the developed activity, by their unique identifier GUID. So if an entity X is used by a logfile L and is not available in the local libraries, the analyst needs to search and download the related entities in order to be able to playback the model and reproduce

accurately the activity. In case of missing entities the environment will reproduce them by a default entity with no behavior or iconic representation associated. This decision to disentangle, the logfiles from the often heavy structures associated with model entities is made in order to keep the history logfiles small in size and facilitate their easy exchange and storing. The logfiles can be stored and exchanged in various formats including XML and the tools are based on a database of logfiles, which serve for studies of modeling activities. The format of logfile data is compliant to a proposed model for interoperability of CSCL-related log data described in [2].

| | | |
|-----------------|-------|---|
| 1) 00 : 48 : 55 | User1 | Request Key |
| 2) 00 : 49 : 05 | User2 | Accept To Give The Key |
| 3) 00 : 49 : 12 | User1 | Chat " <i>I asked for the key</i> " |
| 4) 00 : 49 : 20 | User1 | Chat " <i>ok I got it</i> " |
| 5) 00 : 49 : 26 | User1 | Rename Object Ellipse 1 from END USER to END USER #2 (A2412) |
| 6) 00 : 51 : 06 | User1 | Chat " <i>Get the key and change all relations with those connected to LANS</i> " |
| 7) 00 : 52 : 05 | User2 | Chat " <i>OK</i> " |
| 8) 00 : 52 : 08 | User2 | Request Key |
| 9) 00 : 52 : 13 | User1 | Accept To Give The Key |

Fig. 2. Extract of a history logfile from collaborative problem solving

3 Run Time Support at Group Level: Building Awareness Mechanisms

One key feature of the presented environment is the support provided at run time to the collaborating partners through a view of the state of evolution of the collaborative activity. Based on the fact that the activity is logged at both the client and the server nodes, some abstract representations of the activity have been defined with the objective to feed them back to the group members in order to increase group awareness and motivate meta-cognitive processes for self-regulation. In this section a mathematical model of collaboration is presented, reflecting the symmetry of participation in dialogue and solution building of the group members. In the following section 4 a new approach, based on data mining of historical data is proposed.

3.1 Modeling Collaborative Activity

In this section the key parameters are described through which collaborative problem solving activity can be modeled in Synergo. In typical problem solving scenarios, dialogue and action are interleaved supporting each-other. So the activity is based on both direct communication acts (e.g. chat messages) and indirect communication through operations in the shared workspace.

This activity can be modeled according to the following four dimensions:

- The time dimension t : (when the action is taking place)
- The actors' dimension: $A = \{A_1, A_2, \dots, A_k\}$ (who is acting)
- The objects' dimension: $O = \{O_1, O_2, \dots, O_l\}$ (the object of action in shared space):
- The typology of events dimension: Ty (what is the type of action).

This latter dimension leads to interpretation of the activity that takes place. It is assumed that there is an existing analytical framework, which defines this typology Ty . If r is the finite number of expected event types, then we define a set $Ty = \{T_1, T_2, \dots, T_r\}$ as the analytical framework of the study. Ty can be defined by the framework user.

Using the above four dimensions we can describe any given activity as a set of discrete non-trivial events produced by the actors, contained in the logfile. These define an ordered set of m events $E = \{E_1, E_2, \dots, E_m\}$. Each one of these events is related to meaningful actions of the actors who interact with objects of set O incrementally contributing to the problem solving activity. Each event is defined as a tuple $E_{i_{actor}} = (t_i, A_i, [O_i], [T_i])_i$, where $i \in [1, m]$, t the event timestamp, A the actor who performed the action of the specific event, O an optional parameter referring to the object of the specific action and T an optional parameter which interprets the event according to the analysis framework Ty .

This is a useful general model for logging collaborative activities. Every time an event is produced by the actors, this is recorded and a history of such events, i.e. an ordered list of E s can be produced, as a result of such an activity. This record of the activity can be further annotated by including mental or cognitive operators, as interpretations of the recorded activity. This model permits further off-line analysis and interpretation of the activity, while quantitative indices of the activity can be easily produced at run time, given that some of the Ty annotations can be produced automatically, by the software itself (e.g. actions of insert, delete, chat, etc.) As a result visualizations of the progress of problem solving can be generated [14], as discussed in the next section.

Synergo permits definition of a typology of generated events Ty , and automation of the task of categorization of observed events (e.g. insertion, modification, deletion of primitive objects in the workspace and exchange of text messages). The Synergo environment facilitates the Ty definition process, by allowing association of kinds of low level software generated events, to event types. So for instance, all the low level events of type “Change of textual description of concepts” in a concept-mapping tool are associated to the “Modification” type of action, as shown in figure 3. Every time an action is recorded, this is automatically categorized according to the analytical typology defined by the user. Various formal models like OCAF [15] suggest interpretation of exchanged messages (written dialogues during collaboration by distance), or recorded oral utterances (during face to face collaboration), in relation to operations towards “objects” of the activity space, using a language for action approach [16], defining a unifying framework for analysis of dialogue and action. However interpretation of dialogue events at run time is not possible, unless the users themselves classify their exchanged messages through a dialogue annotation scheme. However this approach has not been used in our case, as we considered that it imposes a meta-cognitive load to the users and lacks reliability. Instead in the next section quantitative measures of collaboration are described, using just the automatically classified events.

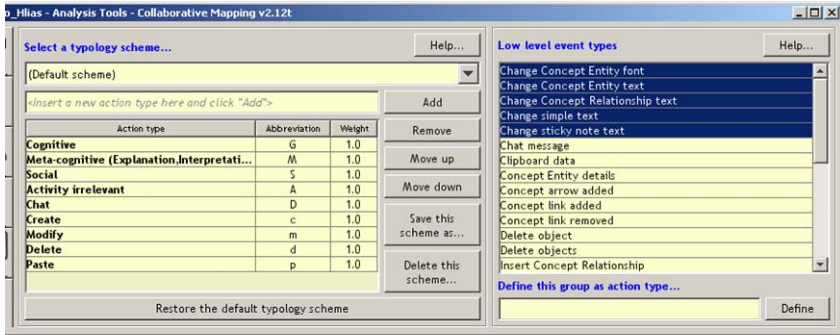


Fig. 3. Definition of an Event Typology scheme Ty: The low level recorded events, generated by the software (right) are related to event types (left)

3.2 Quantitative Indices of Collaboration

Using the model of activity described above, a number of indices, characterizing the state of group activity, have been defined. The objective was to calculate them in order to present them to the group members in a visual form. Some of these indices relate to the density of occurrence of some specific types of event per time interval t_q , e.g. number of exchanged text messages per t_q , number of new objects in the shared space per t_q , etc. These can be calculated at the group level or at the individual partner level.

One other kind of index is related to the degree of symmetry of activity of the group members. This index describes the relative contribution of the group members in a specific type of events.

An example of an empirical general index, called Collaboration Factor is described here. This reflects the symmetry of contribution of actors in the solution, taking into account the relative weights of actors, objects and types of actions.

If we assume that N events of Actor A concern object O , then the contribution of Actor A to object O is measured as:

$$AC_{AO} = W(A) \cdot \sum_{i=1}^N W(T_i) \text{ where } W(A) \text{ is the relative weight of actor } A \text{ and } W(T_i) \text{ is the}$$

weight of type T_i of event i , that contributed to O history.

The history factor HF of object O , is defined as $HF_O = 1 - \frac{stddev(AC)}{M \sqrt{k}}$, where

$HF \in [0,1]$ and M is the mean value of all actors contributions AC for object O . HF takes value close to 1 when there is symmetrical contribution of all actors in the history of object O and close to 0 when the object has been discussed and used by small part of the group.

The collaboration factor of object O is defined subsequently, as

$$CF_O = HF_O \cdot W_o \cdot \frac{L(OE_o)}{m}, \quad CF_O \in [0,1]$$

Where W_o the relative weight of object O in the model, $L(OE_o)$ is the length of action events of object O and m the total number of action events in E .

Finally the collaboration factor of the activity CF is defined as the mean value of all components' collaboration factors, including the abstract objects, or objects that were introduced in the solution and later rejected: $CF = \frac{\sum_{i=1}^{\ell} CF_{O_i}}{\ell}$, $CF \in [0,1]$

In the formulas of CF defined here, a number of weight factors have been introduced: W(A) is the weight of actor A, W_o is the relative weight of object O and W(T) is the relative weight of type T of event. These factors are defined a priori for a certain kind of activity and reflect the relative importance of the corresponding entities. So for instance in a problem solving activity the learners' contributions are considered more important than those of the tutor, some objects of the problem representation are more important than others (e.g. an entity is more important than an attribute in an Entity-Relation Diagram), while some types of events (e.g. insert a new entity) are more important than others (e.g. modify the description of an existing entity). It should be observed that all dialogue messages were classified as of T=T_{dialogue} without refining further their typology, as already discussed in the previous section.

The Collaboration Factor CF, in addition to the other indices introduced here, like the density of activity of specific type of action events per time unit, can be presented in visual form to the group members in order to support understanding of the collaboration dynamics. An example of use of these indices is included in the following section.

3.3 A Case Study of Calculation and Visualization of Indices of Collaboration

In this section we describe an example of visualization of collaborative activity in the frame of the Synergo tool from a case study. The activity involved building of a *concept map* of an Internet service (an electronic bookshop was the service to be model by the participants in this case) by small groups of students of an undergraduate University course, in the frame of one lab session (45'). We focus on one of these groups made of 4 students in this section. The logfile of the activity of this specific group was studied using Synergo. More details of this study can be found in [11]. First the relative weights of the activity types and the actors were defined. In our case events related to creation and modification of sticky notes are assigned lower weight (0.3), as they are used for administration purposes and were not related to problem solving. The actors were all considered of the same weight W(A)=1, while the objects used (concepts and relations) were also considered of similar importance.

A number of representations were produced using the described model. One possibility was to show the current value of CF at the side of the workspace. Also the users can choose to playback the activity and produce in numeric and visual form the evolution of their contribution to the solution and the evolution of the Collaboration Factor CF. This is shown in figure 4(a), and 4(b). In 4(a) the solution is shown with associated history of contribution of various actors to the objects. In 4(b) the evolution of CF with time is shown. This graph provides an indication of the degree of collaboration of the group of the four students as they are building the e-shop concept map. From this graph it seems that while for the first period of the activity the degree of collaboration was high, subsequently the partners became more

individualistic, working on parts of the solution, as also shown in the annotated concept map of fig 4(a). Later on towards the end of the session, there is more interaction, the final value was $CF=0.073$.

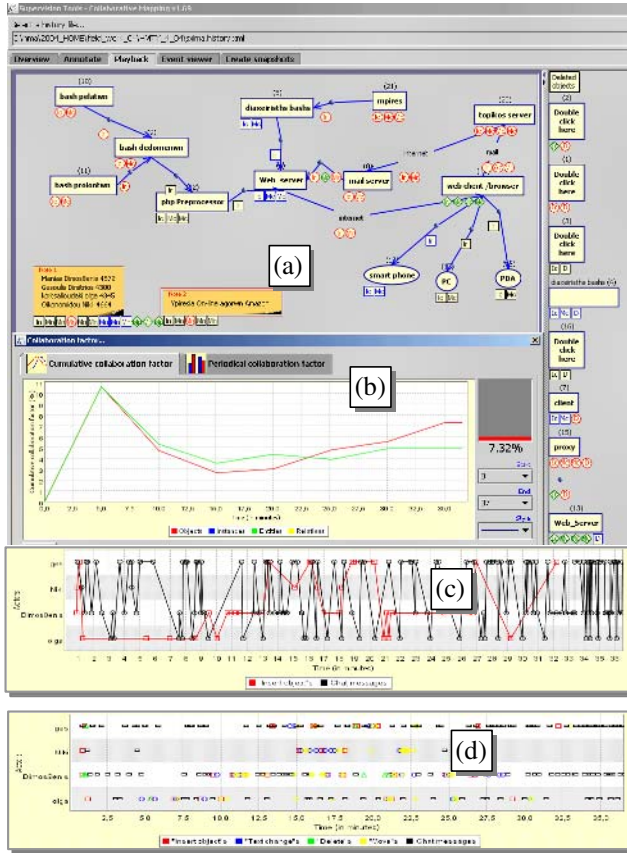


Fig. 4. Visualization of (a) annotated solution (b) Evolution of Collaboration Factor, (c-d) Evolution of Actor activity

Other indices can also be presented, like the density of actors' activity of various types. Also the contribution of each actor in the activity can be visualized. In figure 4(c) and 4(d) the actor contribution of "insert object" events and chat messages is shown. Each line of these diagrams represents one of the four group members. From this picture, it is deduced that the second actor shows relatively low activity.

4 Support for Group Awareness: A Machine Learning Approach

In addition to the method for calculating and visualizing at run time indices of collaboration, described in the previous section, a new approach that is based on data

mining of rich sets of historical logfiles is proposed in this section. The premise of this approach has been that given a rich set of examples of collaborative situations that have been evaluated in terms of the collaborative value of the activity, a module can be trained to be able to classify them accurately enough to be used in the future for classifying other unknown situations. Through this approach it is expected that discourse-related characteristics of the activity that were ignored in the previous case, can be taken into account and a more accurate interpretation of the collaborative progress is made at run time.

First a number of attributes characterizing given segments of previously recorded collaborative activities have to be defined. Then adequate data sets should be selected and effective machine learning algorithms should be used for training classification algorithms, the performance of which subsequently need to be evaluated and to be tested in a typical field study. The described process is a typical data mining approach that has been used often in problems with rich data sets, not solved by analytical or algorithmic approaches [12]. Our problem appears to have these characteristics. So a first attempt to use data sets from previous recorded collaborative problem solving activities in order to train a classifier of collaborative value was made.

We made an assumption that we need to fragment the logfile L of a given activity in consecutive segments $L=\{S1, S2, \dots Sk\}$, each one of which containing enough activity in order to be able to establish for the specific segment the quality of collaboration. The fragmentation criterion was established first as a constant time slice t . However we soon discovered that the activity often does not evolve with uniform density with respect to time, so in certain time segments there was enough activity to establish the quality of collaboration factor while in other segments the activity was insufficient. So a second fragmentation criterion was used subsequently: this was the Number of Events (NE) recorded in the events set E . It was decided to set $NE=60$ which produced a number of segments k for a given logfile. It is obvious that there is a tradeoff between the value of NE and the number of segments k that can be produced from a given logfile of activity, i.e. the higher NE the less number of segments are deduced. In the following, a sensitive analysis was performed in order to establish the effect of the value of NE on the performance of collaboration classifiers. Given a certain segment S_i in which NE events have been included, we need to identify the attributes that would be related to the quality of collaboration. These attributes should be measurable characteristics of the monitored activity, without human intervention, as otherwise deduction of the attribute values would be a tedious process for large data sets.

A set of such attributes was defined and subsequently their predictive power, in terms of the quality of collaboration was tested. The original set of attributes of a given segment of collaborative activity is the following:

- Total number of exchanged dialogue messages (integer)
- Degree of symmetry in participation in dialogue [0..1]
- Number of alternations of speaker in dialogue (integer)
- Average number of words per dialogue message (integer)
- Number of questions in the dialogue - as identified by question mark character (integer)

- Total number of activity events in the shared workspace (integer)
- Degree of symmetry in participation in workspace activity [0..1]
- Number of alternations of actor in workspace activity (integer)
- Degree of symmetry in object modifications in the workspace [0..1]

From these nine (9) attributes, the first five (5) are related to dialogue while the other four (4) are related to activity in the shared workspace. A key attribute is the symmetry of participation of the partners in certain kind of activity, like the dialogue or the modification of the objects of the workspace. A fully symmetrical activity (measured as Symmetry=1) is that in which all partners contribute equally in the activity, while asymmetrical activity (Symmetry=0) is that in which a single partner dominates the activity and collaboration is doubtful.

The effectiveness of this model was tested using logfiles from a number of distinct recorded collaborative activities. The main source of data has been the logfiles of problem solving activities of small groups of students (made of 2 to 3 students) of the Hellenic Open University and of the University of Patras, engaged in building concept maps and flow chart diagrams to given problems, using Synergo. Data from 23 such groups were used. Different segmentation factors NE have been used in these files. The different values of NE and the corresponding different numbers of segments that were created in this study are shown in Table 1.

Table 1. Different fragmentation criteria for segment creation

| # of events (NE) per segment | # of segments of activity |
|------------------------------|---------------------------|
| 60 | 306 |
| 80 | 234 |
| 100 | 188 |
| 200 | 99 |

For each one of the segments, manual characterization of the quality of collaboration was qualitatively performed by human evaluators. This was defined using three quality measures: low collaboration (1), medium collaboration (2) and high collaboration (3). Subsequently using these data sets, an attribute selection process was performed in order to establish which of the originally proposed attributes contributed more effectively towards prediction of the quality of collaboration. For attribute selection the Correlation based Feature Selection (CFS) technique was used [13].

As with the most of the feature selection techniques, CFS makes use of a heuristic algorithm along with a gain function to validate the effectiveness of feature subsets. This heuristic rule takes into account the usefulness of the independent features to predict the class feature(s) as well as the level of their correlation. Using this technique in the four data sets defined according to different values of NE, shown in Table 1, we established the most effective predictors, shown in Table 2.

Table 2 shows that the attributes that appear in all data sets are: the number of dialogue messages (2), the number of alternations of speaker in dialogue (4), the

average message size (5) and the number of actions in the shared workspace (7). From these four attributes the first three are related to the dialogue and just the fourth one is related to the activity in the shared activity space.

Table 2. Attribute selection using CFS

| NE=60 | NE=80 | NE=100 | NE=200 |
|------------------|-----------------|------------------|-----------------|
| (2) num_chat | (2) num_chat | (2) num_chat | (2) num_chat |
| (3)symmetry_chat | | (3)symmetry_chat | |
| (4) altern_chat | (4) altern_chat | (4) altern_chat | (4) altern_chat |
| (5) avg_words | (5) avg_words | (5) avg_words | (5) avg_words |
| (6) num_quest | (6) num_quest | (6) num_quest | |
| (7) num_draw | (7) num_draw | (7) num_draw | (7) num_draw |

A number of alternative classification algorithms were used for building the classifier of the quality of collaboration (Naïve Bayesian Network, Logistic Regression, Bagging, Decision Trees, Nearest Neighbor). Using the open source data mining environment WEKA [12] we trained a number of these classifiers that belong to different categories and use distinct techniques. It is important to note that Synergo facilitates the export of log file data in the form of tab separated document files that are easy to handle by tools such as WEKA.

Evaluation of the performance of the produced classifiers was performed using a 10-fold cross validation technique, separating our data set in training and testing data. In figure 10 the performance of a set of six classifiers in terms of percentage of correctly classified segments is shown for different values of NE. From this figure it is deduced that the best performance was achieved in the case of fragmentation factor NE=60. For this data set all classifiers achieved success rate of over 85%, with best performance by the Logistic Regression classifier who achieved a performance of 87%. As NE increases, the performance of the classifiers deteriorates with the case of NE=200, as worse case in which the average performance of the six classifiers was just over 80%. If we take in consideration the fact that an additional disadvantage of high values of NE is that it inflicts long waiting times at run time, as a large number of events should be accumulated before a new value of the factor is calculated, the conclusion of this part of the study is that the most effective values of the fragmentation factor NE should be around the lowest value NE=60, while experimentation with even lower values of NE made the task more difficult and increased the number of indecisive segments since the number of events was too low for a clear verdict on collaboration by the human expert.

As a conclusion of this phase of experimentation with building a mechanism for evaluating the quality of collaboration in the frame of our framework at run time, in order to use it as a group awareness mechanism, we discovered that this machine learning approach was effective since the trained classifiers were capable, with accuracy close to 90%, to classify segments of activity in a qualitative way. It should however be observed that this second approach produced a qualitative index of group

collaboration (high, medium, low) contrary to the statistical approach that produced a more accurate numerical value.

A final attempt was made to use a combination of the two approaches discussed here, and in section 3. So we built a hybrid collaboration awareness mechanism as a linear combination of normalized values of the collaboration factor (CF) discussed in section 3 and the quality of collaboration factor discussed in section 4. The result was a measure of the state of collaboration (SoC) which was implemented and used in the frame of a case study discussed in the final section of the paper.

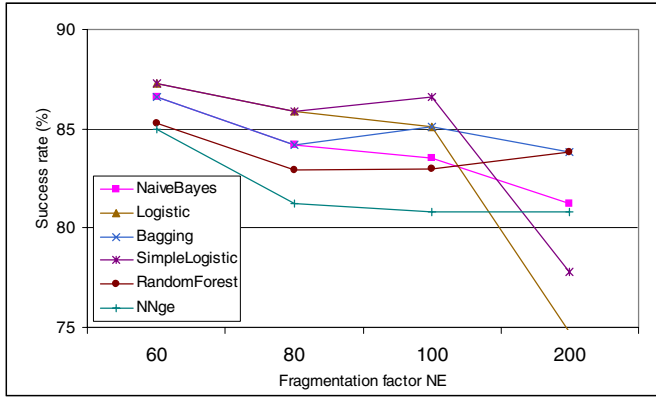


Fig. 5. Performance for different values of the fragmentation factor

5 Evaluation Study

An evaluation study of the developed group-awareness mechanism was performed next. The objective of the study was to establish the effect of this mechanism to small groups of users of Synergo. In the study thirty-three (33) students of the Electrical & Computer Engineering Department of our University participated. In the context of a laboratory session of the Human-Computer Interaction course they were asked to evaluate collaboratively in small groups the usability of a web-based accommodation booking service of a major international conference. Subsequently they were asked to build a state transition diagram of the typical user interaction with the system, in which to associate usability-related comments. The group members interacted exclusively through the Synergo chat tool and the Synergo shared activity space in which they built the requested diagram. The students were assigned to 11 groups made of 3 students each. Six (6) of these groups were provided with the group awareness collaboration mechanism. The other five (5) groups did not have that facility.

A comparative qualitative and quantitative evaluation of group interaction of these two sets of groups was performed. We measured how symmetrical the interaction of the group members were in the two sets. The overall measure combined the degree of symmetry of dialogue events and actions in the shared activity space. This measure took the values shown in Table 3 for the groups of the study.

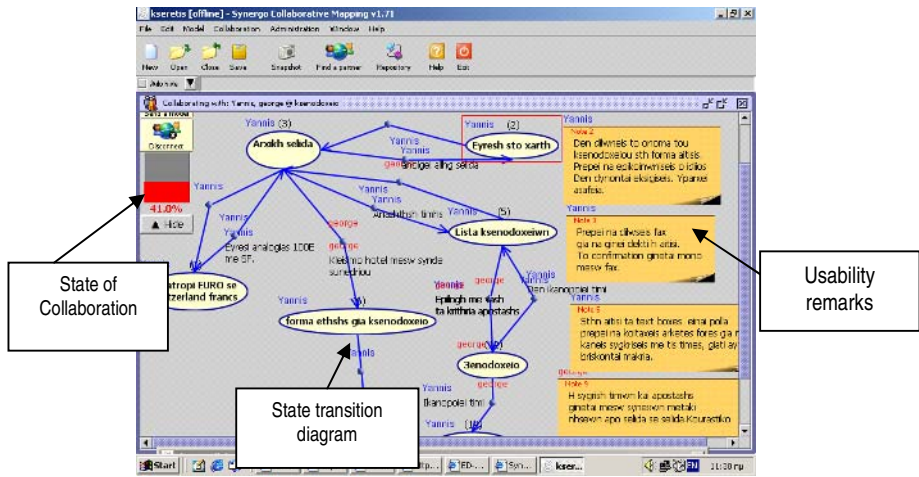


Fig. 6. Typical user workstation during collaborative problem solving

An extract of a typical solution produced by one of the groups is shown in figure 6.

It was found that there is significant difference between the mean value of the two sets (t-test: $p=0,0423 < 0,05$). The mean values of collaboration symmetry are 45% for set A and 36% for set B, with standard deviation 0.053 and 0.075 correspondingly. So the activity of the groups of set A who were aware of the collaboration state through the developed group awareness mechanism was more symmetrical.

Table 3. Group awareness State of Collaboration factor

| Set A: Groups with group-awareness mechanism | Set B: Groups without group awareness mechanism |
|--|---|
| 38 % | 36 % |
| 41 % | 47 % |
| 48 % | 34 % |
| 49 % | 37 % |
| 52 % | 26 % |
| 43 % | |

```
[U1] Let us start talking about ourselves in order to
increase the bar to 100%
[U2] U3, you should talk!
[U3] hhhm what to say :)
[U1] You see it went up to 42% by just doing that
[U3] hey hey hey
[U3] How do we start drawing?
[U2] As long as you U1 talk, it goes down...
[U1] OK I will shut up then..
```

Fig. 7. Example of dialogue extract about the collaboration factor

In addition, by examining more closely the dialogues in groups of set A it was found that in four (4) out of the six (6) groups there was an explicit discussion about the group awareness mechanism.

A side-effect of the group awareness collaboration mechanism was that in some occasions the partners attempted even to affect explicitly the value of this factor, as in the extract of figure 7. However this kind of dialogue events accounted for less than 5% of the overall exchanged messages.

Overall the dialogues were focused in the task and the participation of the partners in the groups of set A was more active and focused than those of set B. The discussion about the group awareness mechanism took place at the beginning and the end of the session in all four occasions and it did not affect the problem solving task. In groups of set B the participation of the partners in the activity was less symmetrical, due often to the existence of partners of limited contribution to the activity. This is an incident observed often in groups with more than two partners in synchronous groupware.

6 Conclusions

Building group reflection mechanisms for groupware systems, like the State of Collaboration (SoC) factor for the Synergo environment discussed in this paper, presents difficulties, since these factors are calculated from many diverse indices who are produced by the dispersed activity of the collaborating community. Use of just statistical aggregate measures is an approach that has been used effectively in the past, however there is an increasing need to capture the semantics of the evolving collaborative process in order to feed them back to the group of the partners providing them with more realistic group awareness view. Use of machine learning techniques for this purpose presents great advantages, since these techniques often require much less data and use less processing power than the statistical techniques, while they are more flexible in providing qualitative measures of the state of collaboration. However in order for such techniques to be proven effective, a tedious modeling phase should proceed followed by a careful training phase of the algorithms. In addition, rich data sets which depict many examples of collaborative or antagonistic situations should be used during the data mining process.

An overall conclusion of the study is that group awareness seems to play a significant role in the group activity, as it is easy to interpret, not requiring high cognitive load and focusing ability of the partners concerned, as is the case with individual partners' awareness mechanisms. Through a single graphic measure or a plot represents vividly the state of the group. The result in our case study was this mechanism to cause higher degree of involvement of the individual partners and lead to improved collaboration.

Acknowledgments. This research has been funded by the Hellenic Ministry of Education, program Herakleitos on Methods & Tools for CSCL and the European Research Team CAViCoLA in the Kaleidoscope NoE (Contract No. 507838).

References

1. Stahl, G. Rediscovering CSCL. In Koschmann, T., Hall, R., Miyake, N. (Eds.), *CSCL2: Carrying Forward the Conversation*, Lawrence Erlbaum Associates, Hillsdale, NJ (2001)
2. Kahrmanis G., A. Papasalouros, N. Avouris, S. Retalis, A Model for Interoperability in Computer Supported Collaborative Learning, Proc. ICALT 2006 - 6th IEEE International Conference on Advanced Learning Technologies. July 5-7, 2006 – Kerkrade , Netherlands
3. Stahl, G. Group cognition in computer-assisted collaborative learning, *Journal of Computer Assisted Learning* (2005) 79-90.
4. Dewan, P. Architectures for Collaborative Applications, Chapter 7 in *Computer Supported Cooperative Work*, Edited by Beaudouin-Lafon, pp.. 169-193, 1999 JohnWiley & Sons Ltd..
5. Michalewicz, Z.: *Genetic Algorithms + Data Structures = Evolution Programs*. 3rd edn. Springer-Verlag, Berlin Heidelberg New York (1996)
6. Dimitracopoulou, A., Komis, V., Design principles for the support of modelling and collaboration in a technology based learning environment”, *Int. J. Continuing Engineering Education and Lifelong Learning*, 15, (1/2), (2005), 30-55.
7. Xenos, M., Avouris, N., Komis, V., Stavrinoudis D, Margaritis, M. Synchronous Collaboration in Distance Education: A Case Study on a CS Course, in Proc. IEEE ICALT 2004, Joensuu, FI, (2004)
8. Voyiatzaki, E., Christakoudis, C., Margaritis, M., Avouris, N., Algorithms Teaching in Secondary Education: A collaborative Approach, in Proc. ED-Media 2004, pp. 2781-2789, Lugano, June 2004.
9. Harrer A., Kahrmanis G., Zeini S., Bollen L., Avouris N., (2006). Is there a way to e-Bologna? Cross-National Collaborative Activities in University Courses, 1st European Conference on Technology Enhanced Learning, Crete, Greece, October 1-4, 2006
10. Avouris, N., Margaritis, M., Komis, V. Modelling interaction during small-groups synchronous problem-solving activities: The Synergo approach, in Proc. ITS 2004, Maceio.
11. Avouris, N., Komis, V., Fiotakis, G., Margaritis, M., Tselios, N. “Tools for Interaction and Collaboration Analysis of learning activities”, Proc. CBLIS 2003, Nicosia, Cyprus.
12. Witten, I. H., Frank, E. “Data Mining: Practical Machine-Learning Tools”, Academic Press, San Diego, CA, (2000).
13. Hall, M. A., “Correlation-based Feature Selection for Machine Learning”. Ph.D. diss. Dept. of Computer Science, Waikato Univ., (1998).
14. Margaritis, M., Avouris, N., Komis, V. Methods and Tools for representation of Collaborative Learning activities. Proc. ETPE 2004, September 2004, Athens.
15. Avouris, N., Dimitracopoulou, A., Komis, V. On analysis of collaborative problem solving: An object-oriented approach, *Computers in Human Behavior*, 19, (2), 2003, pp. 147-167.
16. Winograd, T., A Language/Action Perspective on the Design of Cooperative Work, *Human-Computer Interaction* 3:1 (1987-88), 3-30.

Interaction Analysis for the Detection and Support of Participatory Roles in CSCL

José Antonio Marcos¹, Alejandra Martínez¹,
Yannis A. Dimitriadis², and Rocío Anguita³

¹ School of Computer Science Engineering,

² School of Telecommunications Engineering,

³ Faculty of Education,

University of Valladolid, Camino del Cementerio s/n, 47011, Valladolid, Spain
{jamarcos, amartine}@infor.uva.es, yannis@yllera.tel.uva.es,
rocioan@doe.uva.es

Abstract. Interaction analysis (IA) methods and tools aim to enhance collaboration, providing support for basic functions such as awareness, regulation or evaluation. The importance of these functions depends on the roles played by the participants in a collaborative experience. For this reason, IA tools need to recognize the dynamic role transitions that usually occur in authentic learning settings, as well as to interpret and manage the information needs required by these changing roles. We are working in the definition, developing and validation of a conceptual framework for characterizing roles in collaborative learning contexts that aims at supporting IA tools in achieving these goals. In this paper we present the main results obtained from an experience that illustrates how this framework, initially proposed in a previous paper, supports the definition of IA indicators and values for detecting role transitions in a dynamic way. This experience is part of a longitudinal validation process of the framework that we are carrying out in various authentic learning contexts.

1 Introduction

At present, the elaboration of advanced Interaction Analysis (IA) tools and methods for the study of collaboration is a research priority in the CSCL (*Computer Supported Collaborative Learning*) field [1], [2]. IA can support different functions (e.g., awareness, regulation and evaluation) based on the understanding of collaborative processes. The mentioned functions can be oriented to different types of users, which have different needs depending on diverse aspects related to the context, the specific task, the educational level of the participants and the IA purpose. For example, [3] identify different needs of a teacher in asynchronous and synchronous scenarios, and therefore suggest different types of support.

Following this idea, in the CSCW (*Computer Supported Cooperative Work*) field we can find some proposals of awareness systems that adapt their functionalities to the different participants' roles [4], [5]. These approaches consider that the key issue is to provide exactly the right amount and type of information for a given participant in a given role performing a given task. From these experiences we can state that IA tools would benefit from considering these role-based proposals, in order to improve

the collaborative processes they support. Moreover, in CSCL we can find a number of works that show how the pre-assignment of appropriate roles to the participants facilitates their interaction, improving the overall collaborative experience [6], [7].

From the aforementioned works, we can see that it would be very useful to *identify the roles* that can appear in collaborative processes and what are *their IA needs*, i.e., the information type and how to present it to these roles. Then, the problem faced in this paper consists in how to characterize the roles that participate in a collaborative activity, in order to facilitate the dynamic detection of role transitions during its development. With this information, an IA tool should be able to adapt its outputs to the needs of these evolving roles, in an automatic or semiautomatic way.

In a previous paper [8], we presented the outline of a framework for the structured description and characterization of roles. The framework faces the lack of a common taxonomy of roles in CSCL and the need of describing dynamic aspects, such as the mentioned shifts between roles that usually take place in real contexts.

This framework is not a final proposal, and it needs a complete validation process. The theoretical foundations of CSCL demand the use of authentic learning settings in order to achieve relevant evaluation results. In this paper we present one of these validation experiences, which was aimed to assess how the framework supports the detection of role transitions in a dynamic way. This experience also serves to illustrate how the framework can be applied to a concrete learning situation.

The rest of the paper is structured as follows: The next section introduces our proposal of a conceptual framework to describe roles in CSCL contexts. Section 3 presents the experience carried out in order to assess the capability of the framework to support the dynamic detection of roles and discusses next steps derived from this experience. The paper finishes presenting the main conclusions and an overview of our future work plans.

2 A Conceptual Framework for Describing Roles in CSCL

This section introduces the main features of our proposal of a framework for the structured description of roles, initially presented in [8]. We will focus here on the main aspects used in the experience presented in the section 3. This framework aims to enable IA tools to adapt their functionalities to the different roles played by participants in collaborative activities. This adaptation requires a description of these roles so that IA tools can interpret and manage computationally this information.

The description of a role in the framework includes four aspects: *definition*, *IA needs*, *context of application*, as well as *indicators and values for detecting it*.

The *definition of a role* includes its name and the description of its function. In this context, the name is a generic role such as a human, an agent or any combination of them (e.g. teacher or student) [9], and its description is a characterization of an actor in terms of activities, duties and responsibilities in the learning activity (e.g., facilitator: “a teacher performing a minimal pedagogical intervention in order to redirect the group work in a productive direction” [10]).

The *description of IA needs* specifies the IA information required for a role. These needs involve the purpose (e.g., awareness, regulation or evaluation), information content (e.g., participatory aspects, such as intra-group collaboration), information

type (e.g., numerical or graphical), complexity (e.g., elementary or advanced information) and presentation of information (e.g., bar chart or sociogram), as well as the timing (frequency) and type of medium which will be used to communicate the information to the user (e.g., by the teacher in the classroom or by mail).

These requirements are influenced by the context. The *description of context* includes diverse aspects collaborative activity such as the *scope*, that details the number of participants included in the learning activity (e.g., small group, large group), the *type of environment* (e.g., synchronous or asynchronous), the *educational level* of the students (e.g., university or K-12 students), the *collaborative experience level* of participants (e.g. elementary or expert), the *specific collaborative tasks* (e.g., collaborative edition), and the *tools* used to develop the activity (e.g. BSCW).

Finally the *specification of indicators and values* is meant to enable an IA tool to identify a possible change of role during the activity. Each indicator includes a specification of its:

- *Name*: An identification of the indicator, for example “participation rate”.
- *Description*: It refers to the generic aspect of interaction that it represents and its relation with the functions of this role.
- *Range of values and interpretation*: It explains the correspondence between the different values that the indicator can take and their interpretation with respect to the described role.
- *Relevance rank*: It is possible that we need to use more than one indicator. This aspect states the relevance of the indicator for detecting the role (e.g., some proportion as 50% or a rank such as first, second...).
- *Detecting mode*: How and when the indicator is calculated (i.e., directly in a specific moment, or between milestones).

All these aspects constitute the basis of the framework. However, this is not a final proposal, but it is under a process of cyclic refinement. As mentioned beforehand, the complexity of the CSCL domain and the generality of the framework itself, require a longitudinal study, where the ideas are incrementally applied to authentic learning scenarios. This way, we plan to assess specific aspects of the framework incorporating formative corrections to the proposal that are again assessed during the next cycle. During this cyclic process, we have already applied the framework to a case study where we could evaluate the capacity of the framework to successfully adapt the IA support to different roles, based on the descriptions of these roles made with the framework [11]. This was a *static adaptation*, where the roles played by the participants and their needs were pre-defined before the beginning of the collaborative activity. In this paper we are focusing on the possibility of identifying role shifts *dynamically*, so that an IA tool could adapt its output to these changes during the activity. We have carried out an experience in an authentic learning scenario to assess this possibility. Next section presents the results of this experience.

3 An Experience to Assess the Dynamic Detection of Roles

The study described in this section is part of a case study that has been taking place since February 2005 in the course of “ICT (Information and Communication

Technologies) applied in Education” at our University. Besides our goal of using it to assess the capacity of the framework to support the dynamic identification of role shifts in a collaborative activity, it also provides an example of the use of the framework in a concrete situation. Table 1 shows the main characteristics of the context of this experience, according to the five dimensions defined in the framework: scope, environment, educational level, experience and tools.

Table 1. Specification of characteristics related with the context of this experience

| Dimensions | | <i>Description</i> |
|---------------------------------|--------------------------|---|
| <i>Educational level</i> | University | A course of ICT applied in Education |
| <i>Scope</i> | Large group | Twenty-six students distributed in three groups |
| <i>Collaborative experience</i> | Students: None | |
| | Teacher: Expert | |
| <i>Environment</i> | Blended | Technology supports in-site or distance activities. |
| <i>Collaborative tools</i> | Synergeia [12] | This tool provides a workspace for sharing documents among all the actors in the course |
| <i>Specific tasks</i> | Theoretical phase | Students analyzed diverse aspects of the subject and elaborate in groups three reports (subtasks) |
| | Practical phase | Students created a <i>Webquest</i> , that could be used in a real school |

In this context we have applied the framework for detecting a limited set of emergent roles of learners (isolated and coordinator) and teachers (guide and collaborator). We employed Social Network Analysis (SNA) as a specific IA method, which is appropriate for the study of structural properties of individuals learning in groups [13]. We used SAMSA in order to produce the desired social network indicators. In this case study we considered the relationships composed by the indirect links between an actor that creates an object in a Synergeia shared workspace and those that access this object in order to read it. All the aforementioned roles were described and could be identified using the framework. For reasons of space, the rest of this section focuses on the detection of the *teacher-guide* and *teacher-collaborator* roles, but the discussed results are also applicable to the rest of the learners’ roles.

3.1 Description of the Indicators Associated with Each Role by Means of the Framework

This section explains how we employed the framework for specifying the set of SNA indicators and values for detecting the *teacher-guide* and the *teacher-collaborator* roles using IA. The selected indicators were: *degree centrality* ($C_D(i)$) and *closeness centrality* ($C_C(i)$). $C_D(i)$ is the most common measurement for the study of participatory aspects of learning. It indicates the activity of an actor in the network. Also, it is an index of the actor’s prestige [14]. $C_C(i)$ denotes the proximity of a node to the rest of nodes in the network. This index can be interpreted as a measurement of the influence of an actor in the overall network. In the case of relationships that consider the direction of the link, two degree and closeness indexes are defined. For example, for $C_D(i)$: *indegree* ($C_{D_i}(i)$), or the number of links terminating at the node;

and *outdegree* ($C_{Do}(i)$), or the number of links originating at the node. We have also selected the sociograms for the visualization of the detected roles in a very intuitive way. The sociograms represent the actors as nodes and the relationships among them as lines in the graph.

According to the dimensions specified in the framework, we defined the values of these indicators for detecting the *teacher-guide* and *teacher-collaborator* roles. We consider a *teacher-guide* as a leader that conducts the activity, detects participation problems and intervenes in order to improve the collaboration. For this reason, his SNA values are the highest among the actors in the network, and he has a central position in the sociogram. Table 2 details the concrete values associated to this role.

Table 2. Specification of the indicators and their values for the *teacher-guide* role (*teacher-collaborator* role values are not shown for space restrictions)

| Role: Teacher-Guide | | |
|--------------------------------------|--------------------------------|--|
| Indicators | | |
| Indegree $C_{Di}(i)$ | <i>Description</i> | Number of links terminating at this actor, in the sense measured by the network |
| | <i>Values / Interpretation</i> | A high value indicates a high actor's prestige into the group |
| | <i>Relevance rank</i> | First |
| Incloseness $C_{Ci}(i)$ | <i>Description</i> | Specifies the proximity of an actor to the rest of actors in the network |
| | <i>Values / Interpretation</i> | A high value indicates a high influence of the actor in the overall network |
| | <i>Relevance rank</i> | Second |
| Actor position in a sociogram | <i>Description</i> | A sociogram represents the actors as nodes and the relationships among them as lines in the graph. |
| | <i>Values / Interpretation</i> | A centered node in the graph indicates an prominent actor for the rest of participants |
| | <i>Relevance rank</i> | Third. Only for visual validation |

On the other hand, we defined the *teacher-collaborator* as a teacher that monitors the development of the activity but does not guide it. She participates only in specific moments, for example for reading the reports elaborated by the students. For this reason, her values for the selected indicators have to be lower than the majority of the actors in the network, and her position in the sociogram is not a central one.

The same procedure was followed to define the indicators for detecting the roles of the *isolated* and *coordinator learner*. With all these descriptions, we could analyze the networks and detect learners and teachers' role transitions. Next section shows how we could identify these transitions for the teacher, supported by the descriptions provided by the framework.

3.2 Results: Evolution of Teacher Role During the Collaborative Activity

We have analyzed the activity of the participants during the overall collaborative learning activity. Using SAMSA as the IA tool, and the specifications discussed in the

previous section we performed a study of participants' roles after the end of each subtask, approximately each four weeks.

During the elaboration of the first report, in the theoretical phase, we detected the role of the *teacher-guide*. His indexes $C_{Di}(teacher)$ and $C_{Ci}(teacher)$ were the highest of participants (29 and 10,57 respectively). Moreover, we can see in the sociogram associated to this phase (Figure 1(a)) how the teacher was the most centered node. Thus, we could conclude that the teacher was the leader of the activity in this phase.

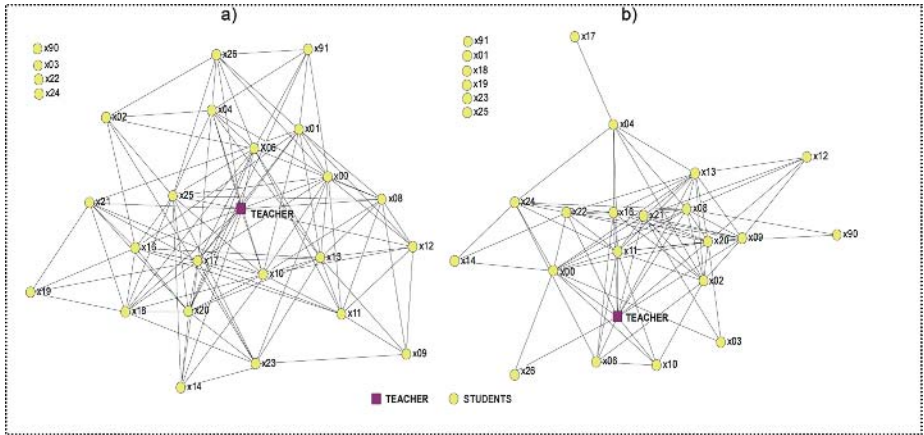


Fig. 1. Sociograms representing the participants interactions: (a) During the elaboration of the theoretical first report. (b) During the first part of the practical phase.

The values of these indexes decreased for the teacher during the next weeks. To the end of the theoretical phase, his indexes were lower ($C_{Di}(teacher)=19$ and $C_{Ci}(teacher)=13.17$) than some of the students indexes ($C_{Di}(x08)=39$, $C_{Ci}(x08)=13.73$; $C_{Di}(x00)=36$, $C_{Ci}(x00)=13.62$; $C_{Di}(x21)=28$, $C_{Ci}(x21)=13.39$; $C_{Di}(x20)=22$, $C_{Ci}(x20)=13.45$). Thus, the teacher was still one of the most important actors, but other participants had begun to acquire some autonomy.

Following this tendency, after the first part of the practical phase we could detect clearly that the teacher had lost his role of *guide* and he had become a *collaborator* in the activity. In this period his *indegree* and *incloseness* indexes show a notable decrease ($C_{Di}(teacher)=14$ and $C_{Ci}(teacher)=8.77$, respectively). More than 50% of the students presented higher values in these indexes (with C_{Di} values ranging from 132 to 21 and C_{Ci} values from 10.39 to 8.82). We can view the sociogram associated with the practical phase in Figure 1 (b). The teacher is not a centered node anymore.

These results were confirmed by triangulation with different sources of data and analysis methods, including questionnaires, focus groups of volunteers, and classroom observations, following the process described in the *Mixed Evaluation Method* [15]. This process confirmed the change of the teacher's relevance during the process. For example, 79% of the students confirmed the initial role of the teacher as a guide and justified her posterior evolution towards a less central role.

In conclusion, we can state that the indicators and values defined with the framework supported the detection of the teacher's role transitions during the

collaborative activity using IA. Once a role transition is detected, an IA tool could adapt its output to the specific needs of the emergent role. This adaptation has been already tested in a previous experience with the predefined roles, where the output of the SAMSA was adapted to these roles [11]. After the study presented in this paper, we can think of a next study where both functionalities are integrated, in order to provide a dynamic adaptation of IA support to the users of a CSCL scenario.

4 Conclusions and Future Work

This paper has presented an experience performed in an authentic learning scenario using IA methods for identifying dynamically a limited set of emergent roles, corresponding to teachers and learners, as part of the validation process of our proposal of a framework for the structured description of roles.

This dynamic detection of role transitions is aimed to allow IA tools to adapt their output to the needs of the described roles during the collaborative activities. A static adaptation of IA tool based on the framework has been already shown in [11].

The experience described in this work shows that the structured description of roles proposed in the framework provides appropriate information to describe and identify a limited set of roles. The fact that the indicators and values to detect these roles are described in computational terms allows IA tools to interpret and manage the information. This opens a space for the automatic or semi-automatic adaptation of IA tools to CSCL. Overall, these results aim to contribute to the evolution of the IA field in CSCL, which is currently more focused on the developing of research prototypes, and therefore, far from offering solutions for real users, as stated by [2].

Next iterations of the process of cyclic refinement of the framework will include its application to other authentic learning scenarios, where the dynamic identification of roles and the adaptation of the output provided by the IA tools will be integrated. Additionally, we will increment the number of roles to identify and support by the IA tools. This implies further work in the recognition and definition of adequate indicators to identify these roles.

Acknowledgments

This work has been partially funded by Kaleidoscope NoE (FP6-2002-IST-507838), Spanish Ministry of Education and Science (TSI-2005-08225-C07-04) and the Autonomous Government of Castilla y León, Spain (projects VA00905, UV31/04 and UV46/04). The authors would also like to thank the rest of GSIC/EMIC Group at the University of Valladolid for their support and ideas.

References

1. Dimitracopoulou, A. Designing Collaborative Learning Systems: Current Trends and Future Research Agenda. In: Proc. of the 4th Conf. on Computer Support for Collaborative Learning, CSCL 2005, Taipei, Taiwan. May 30 – June 4. (2005)

2. Soller, A., Martínez, A., Jerman, P., & Muehlenbrock, M.: From Mirroring to Guiding: A Review of State of the Art Technology for Supporting Collaborative Learning. *International Journal on Artificial Intelligence in Education*. 15 (2005), 261-290
3. Petrou A. & Dimitracopoulou A.: Is synchronous computer mediated collaborative problem solving 'justified' only when by distance? Teachers' point of views and interventions with co-located groups during every day class activities. In: Wasson and Hoppe (eds). *Computer Support for Collaborative Learning: CSCL 2003*. Bergen, Norway. 14-18 June. (2003)
4. Dourish, P. & Belloti, V.: Awareness and Coordination in Shared Workspaces. In: Proc. of the Computer Support for Collaborative Working (CSCW) Confer. Toronto, Canada (1992)
5. Drury, J. & Williams, M.G.: A Framework for Role-Based Specification and Evaluation of Awareness Support in Synchronous Collaborative Applications. Proceedings of the 11th Int. Workshop on Enabling Technologies for Collaborative Enterprises (WETICE02), Carnegie Mellon University, Pittsburgh (2002)
6. Mizoguchi, R. & Inaba, A.: Learners' Roles and Predictable Educational Benefits in Collaborative Learning. An Ontological Approach to Support Design and Analysis of CSCL. In: J.C. Lester (ed). Proc. of 7th ITS 2004. Maceió, Alagoas, Brazil (2004) 285-94
7. Strijbos, J.W.: Functional Versus Spontaneous Roles During Computer-Supported Collaborative Learning. In: Proc. of Second International Conference on Multimedia and ICTs in Education (m-ICTE 2003), Badajoz, Spain. (2003)
8. Marcos, J.A.; Martínez, A., and Dimitriadis, Y.. Towards Adaptable Interaction Analysis Tools in CSCL. 12th AIED International Conference. Workshop on Representing and Analysing Collaborative Interactions; Amsterdam, The Netherlands. 2005 Jul: pp. 70-74.
9. ISO/IEC JTC1 SC36 N0065: Expertise and Role Identification in Learning Environments. *Information Technology for Learning, Education and Training* (2001)
10. Chen, W.: Supporting Teachers Intervention in Collaborative Knowledge Building. In: Proc. of Elena Gaudioso & Luis Talavera (eds). Proc. of the Workshop on Artificial Intelligence in Computer Supported Collaborative Learning at the 16th European Conf. on Artificial Intelligence (ECAI'2004), Valencia, Spain (2004) 1-5
11. Marcos, J.A., Martínez, A., Dimitriadis, Y. & Anguita, R.: Adapting Interaction Analysis to Support Evaluation and Regulation: A Case Study. Proceedings of 6th IEEE Int. Conf. on Advanced Learning Technologies, (ICALT 2006), Kerkrade, The Netherlands, July 2006
12. ITCOLE Research Project. Synergeia Website. Retrieved in December, 2005 from <http://bscl.gmd.de> (2005)
13. Cho, H., Stefanone, M., & Gay, G.: Social Information Sharing in a CSCL Community. In: G. Stahl (ed.) *Computer Support for Collaborative Learning: Foundations for a CSCL Community*, Erlbaum, NJ (2002) 43-50
14. Wasserman, S. & Faust K. *Social Network Analysis: Methods and Applications*. Cambridge University Press., Cambridge (1994).
15. Martínez, A.; Dimitriadis, Y., and de la Fuente, P. Interaction analysis for formative evaluation in CSCL. *Computers and Education. Toward a lifelong learning society*. M. Llamas, M.J. Fernandez, L.E. Anido ed. Kluwer Academic; 2003; pp. 227-238.

ORCHESTRA: Formalism to Express Mobile Cooperative Applications

Bertrand David, René Chalon, Olivier Delotte,
Guillaume Masserey, and Matthieu Imbert

ICTT Laboratory
Ecole Centrale de Lyon
36, av. Guy de Collongue, 69134 Ecully Cedex, France
Bertrand.David@ec-lyon.fr

Abstract. Orchestra is a new formalism on which we are working in the field of cooperative systems design. In CoCSys methodology for Cooperative Capillary Systems design, we transform partial scenarios describing particular cooperative situations in a more comprehensive Cooperative Behaviour Model (CBM). In this paper, we describe our contribution to the need for a graphical formalism which would be able to express in a natural way, understandable by different actors (users, designers, developers,...) different cooperation situations in an ambient intelligence environment (mobile, context-aware, proactive and ubiquitous). ORCHESTRA is complementary to CTT and UML Use cases, and its objective is to express clearly cooperation situations (explaining easily synchronous or asynchronous cooperation activities) and the role (active or passive) played instantaneously by each actor. We take into account main concepts of “cooperative world” which are Actors, Roles, Groups, Tasks, Processes, Artefacts (Tools and Objects) and Contexts (Platforms, Situations and Users). With Orchestra formalism we try to express by a sort of music staff individual and collective behaviours. In this way we can model either individual works or organized collective activities. We present this formalism, its metamodel and its use for the description of two cooperative situations. We describe also a transformational process projecting a ORCHESTRA description on the cooperative architecture.

Keywords: Specific description language, MDA inspired elaboration process, transformation process, formalism meta-model and examples.

1 Introduction

CSCW [1] is a field of interactive computer-based systems which objective is to allow to several participants (actors) to work together via a computer-based system to solve cooperatively a problem which can be of different natures (design, management, production, learning, etc). Design of this kind of systems is relatively complex because it is not limited to individual activities, but also and mainly to cooperative work of several actors, which can be classified in co-operation, coordination and conversation activities in respect with the definition initially proposed by Ellis [10] and adapted by several other authors [8, 16]. This cooperative work can be done in

several cooperative situations characterized initially by Johansen and enhanced by Ellis [9]. At the moment CSCW systems are becoming more and more mobile, context-aware and proactive. We called this kind of cooperative systems Capillary Cooperative Systems (CCS) [7]. We use this term by analogy with the network of blood vessels. The purpose of the Capillary CS is “to extend the capacities provided by co-operative working tools in increasingly fine ramifications, hence their can use fixed workstations and handheld devices”. These systems become also pervasive, proactive and ubiquitous. Our final goal is to allow them to evolve in mixed reality environment (mixture of real and digital objects and tools) and to put into practice Ambient Intelligence (AmI) concept.

In the following sections we describe our methodology (section 2), we present CBM content (section 3), then we discuss the formalism features and present ORCHESTRA concepts and its meta-model and we give two illustrative examples (section 4). We also briefly sketch a transformational process from ORCHESTRA description to a Cooperative Architecture (section 5). We finish by conclusions and perspectives.

2 Our Approach: CoCSys Methodology

We are studying design of CSCW systems and we propose an approach and a process, called CoCSys (Collaborative Capillary System) engineering process. Main reason for this more comprehensive process is related to the necessity to allow the evolution of this kind of system during its use in relation with the users’ skills, expertise, and the evolution of their perception and the mastery of the system. Our approach is based on Model-Based approach [17], which is characterized by a different way of development: “Rather than programming an interface using a toolkit library, developers would write a specification of the interface in a specialized, high-level specification language. This specification would be automatically translated into an executable program, or interpreted at run-time to generate the appropriate interface.” This approach is used in HCI for several years and become more generally used in other development application fields. OMG adapted a similar approach as new paradigm of development which is called MDA Model-driven architecture [13]. Others acronyms describing similar ways are MDE (Model-driven engineering) or MDD (Model driven development). In each case specification at concrete, abstract or meta level is privileged before studying the way to produce an executable code. The production is done more or less automatically by transformation or translation of these models. The objective of our approach is to adapt this trend to CSCW. We are proposing a framework for design, implementation and evolution of CCS. Fig. 1 provides a general overview of this approach. It is based on 3 main parts: 1/Scenarios Collection, 2/Cooperative Behaviour Model (CBM), and 3/Collaborative Architecture; and 3 transformation phases: I/CBM Model Construction, II/CBM Projection on the Collaborative Architecture and III/Evolution. The methodology begins with the Scenario Collection phase: a list of scenarios [4] is collected during the discussions with potential users (Fig. 1, component 1). These scenarios are local, related to specific tasks or activities pointed out by different users. The Cooperative Behaviour Model (Fig. 1, component 2) for a specific collaborative application

contains concrete actors, artefacts, tasks and contexts that the cooperative application will take into account. Scenarios are studied in order to extract the actors' tasks or activities and to determine temporal and functional constraints defining the organization of these activities (Fig. 1, transformation I). The goal is also to study completeness, correctness and the coherency of this model by adding missing activities, by eliminating redundancies and by validating working process and participating artefacts.

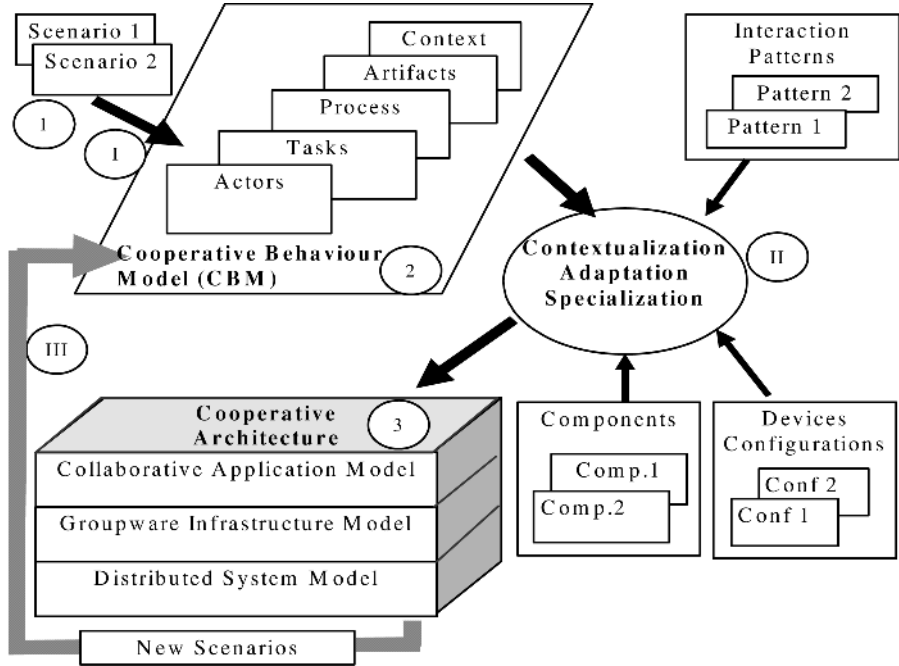


Fig. 1. CoCSys methodology

In the second transformation phase (Fig. 1, transformation II) contextualization, adaptation and specialization are undertaken. This process is a transformation of the CBM model to more operational description. It is not only a translation, but a projection on a predefined architecture (Fig. 1, component 3), i.e. from a software engineering perspective, it is clear that the development of a cooperative application cannot be carried out from scratch. We are using a three layers generic framework for cooperative systems (Fig. 1, component 3). Last transformation objective is to take into account evolution of scenarios and corresponding CBM evolution (Fig. 1, transformation III). These aspects are not described in this paper

3 Scenarios and Cooperative Behaviour Model

We consider that a scenario allows to final users and designers to meet them and discuss together. A scenario describes repetitive activity that should activate an

adaptation mechanism which will be recorded and reused. For us the scenarios are short stories describing precise working situations which occur for different actors. This analytical perception of working situations seems to be possible to catch and express observers or actors needs. We are asking to give as precise description as possible, i.e. to indicate, if possible, all actors evolving, artefacts used, activities executed and contexts characterising them (devices used, geographical location, temporal situation ...). We collect these scenarios for different collaborative situations. In this way we can consider that this formulation of scenarios is possible, meaningful and useful. If scenarios are short limited stories, expressed mainly by different actors, behaviour model objective is to discover overall organization of the cooperative system in which main elements are actors, artefacts, tasks, processes and contexts. The designers are in charge to study different scenarios and to construct gradually the Cooperative Behaviour Model (CBM). In the model we find comprehensive collections of actors, artefacts, activities and contexts and also all relations which allow materializing all necessary elements for each activity. Different processes are also explained carrying out dependencies between tasks and their temporal and organizational constraints. This comprehensive model is able to manage the cooperative system behaviour and will be used during the implementation process i.e. projection of this model on a particular hardware, network and software architectures. Main elements of the CBM model are:

- An actor, as instantiation of one or several roles, a role is a basic element of human behaviour in the system, which can be qualified as Acting (A), Observing (O) or Editing (E) i.e. observing and acting. We distinguish main actor (double arrow) and secondary actor (simple arrow).
- An activity, describing an identified work which a role can do, this activity can be also A, O or E, i.e. acting, observing or editing activity.
- A process is expressed as a network composed of process states (PS) and process transitions, which can also be qualified by A, O or E.
- An artefact can be either a tool or an object. The tool is an instrument used in the task; the object is either input, support or output of the task, qualified by A, O or E.
- A context is a collection of three aspects giving platform, situation (often logical, physical or geographical location) and user preferences characterising the context. We take into account several platform examples and elements: laptop, PDA, cellular phone, and also active environmental object (active RFID tag), passive environmental object (passive tag), ...

In the CBM model all these elements are expressed and interconnected. We can take as example a user's role, which is identified by a name, a type, its participation in different actors, the activities which can be done, the process states and transitions in which their can occur, the artefacts (tools and objects) manipulated and the contexts (platform, situations and user preferences) which applies the role. These interrelations are also needed for other elements of the model. They are explicitly or implicitly described and can change during the system life expressing its adaptation and evolution. List of activities is one of the main components of CBM. This list is obtained from the task tree which can be expressed by CTT [14], an interesting task formalism, and its environment (CTTE) proposed by Paterno. Its extension for

cooperative activities [12] seems appropriate to express cooperative situations. However, the choice to have a hierarchical structure as basic representation and to express everything on this structure doesn't appear appropriate to us. In CTT, collaboration is expressed by individual task trees and by a collaborative task tree. That is interesting to express tasks, but is insufficient for the more comprehensive view of collaboration, that we need. We consider that tree view of tasks is an interesting presentation during the task design phase. However, during the activities organization (definition of effective collaborations), mainly effective activities (leaves of the task tree) are important and their individual or collaborative scope is essential, in relation with effective actors, objects, tools, process states and transitions and contexts. To express in a more comprehensive way this view we propose a new formalism called Orchestra.

4 ORCHESTRA

The objective of Orchestra is to propose a more comprehensive formalism which is able to express together all main aspects of the CBM. ORCHESTRA adapts musical score notation [18] to our problem of CBM description. For us, the **5 lines of a staff** are expressing 5 main aspects of the CBM (Fig. 2), which are: user's role, activity concerned, process state or transition, artefacts involving in the activity and the context. On each line, we can situate one or several "notes" expressing names of corresponding items i.e. roles, activities, process states or transitions, artefacts and contexts. Each note can receive a stem which indicates the participation of the element (acting, observing or editing done by main or secondary actors). A bar line indicates the separation between independent **cooperation episodes**. Each cooperation episode expresses a state or a transition in the cooperation process description network. For each cooperation episode temporal organization is expressed either sequentially from the left to the right, by different types of parenthesis, or by explicit change of episode. These parentheses are used to express different situations:

(...) alternatives,
 {...} mandatory participation,
 [...] potential participation.

Different key signatures are expressing synchronous or asynchronous collaborations, collaboration modes and styles of coordination (computational ☐ or social ☺, implicit or explicit):

@ - Asynchronous with infinite answer delay
 @@ - Asynchronous with limited answer delay (on call)
 & - Synchronous "in-meeting" cooperation
 && - Synchronous "in-depth" cooperation

In synchronous collaboration two different participations must be distinguished:

- instantaneous, short term collaboration, i.e. vote activity,
- long term participation, long term collaboration, i.e. sketching activity.

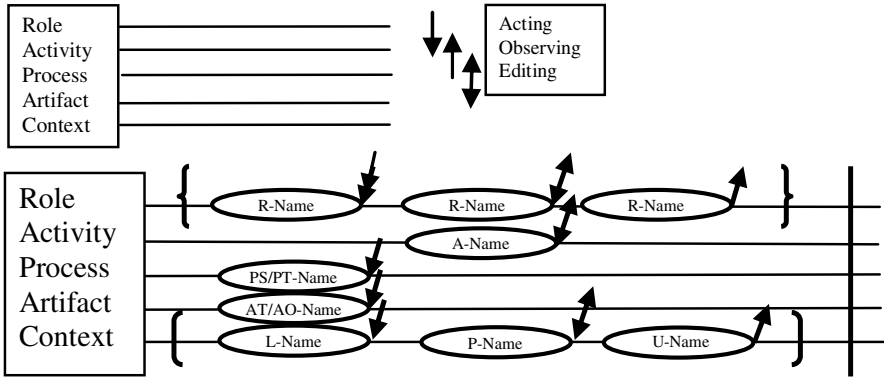


Fig. 2. ORCHESTRA main concepts

In the first case an implicit collaboration is appropriate (short exclusive access to the shared space), in the second case explicit participation must be asked and allowed (long-term access to the shared space) either by social coordination (☺), i.e. one of human actors is in charge of this coordination or a computational (☒) one i.e. the computer fulfil it. Graphically we express instantaneous collaboration by a **dot** over concerned chords, for long term collaborative we use a horizontal line and a symbol expressing social or computational coordination (☺, ☒) i.e. coordination made by one of the actors or by interaction (asking for, receiving and returning exclusive access right to shared space).

Another important notion in CSCW is awareness. Its objective is to allow to different actors to know (or not) what has been done by an actor. It is important to decide statically (by the designer) or dynamically by the actor himself the scope of information propagation to other actors. For static way we propose to express awareness in ORCHESTRA formalism. Special marks are proposed:

- 🙄 for no awareness,
- 🙇 for partial awareness (for specific actors),
- 🙋 for overall awareness (for all actors).

To explain more deeply ORCHESTRA formalism we give on Fig. 3 its meta-model, then we use it on two relatively simple examples.

4.1 Heating Equipment Maintenance Activities

We propose first to express heating equipment maintenance activities (Fig. 4) with six actors: client, secretary, technician, supervisor, expert and clerk. Main scenarios to put together are the following:

- A client (secondary actor), observing a problem with his heating equipment, phones to the repair company to ask intervention. The secretary (secondary actor) asks him his profile (address, equipment...) and finds him in the database. He organizes an appointment with a technician. State: **RV (RendezVous)**, Actors: **Client, Secretary, &**

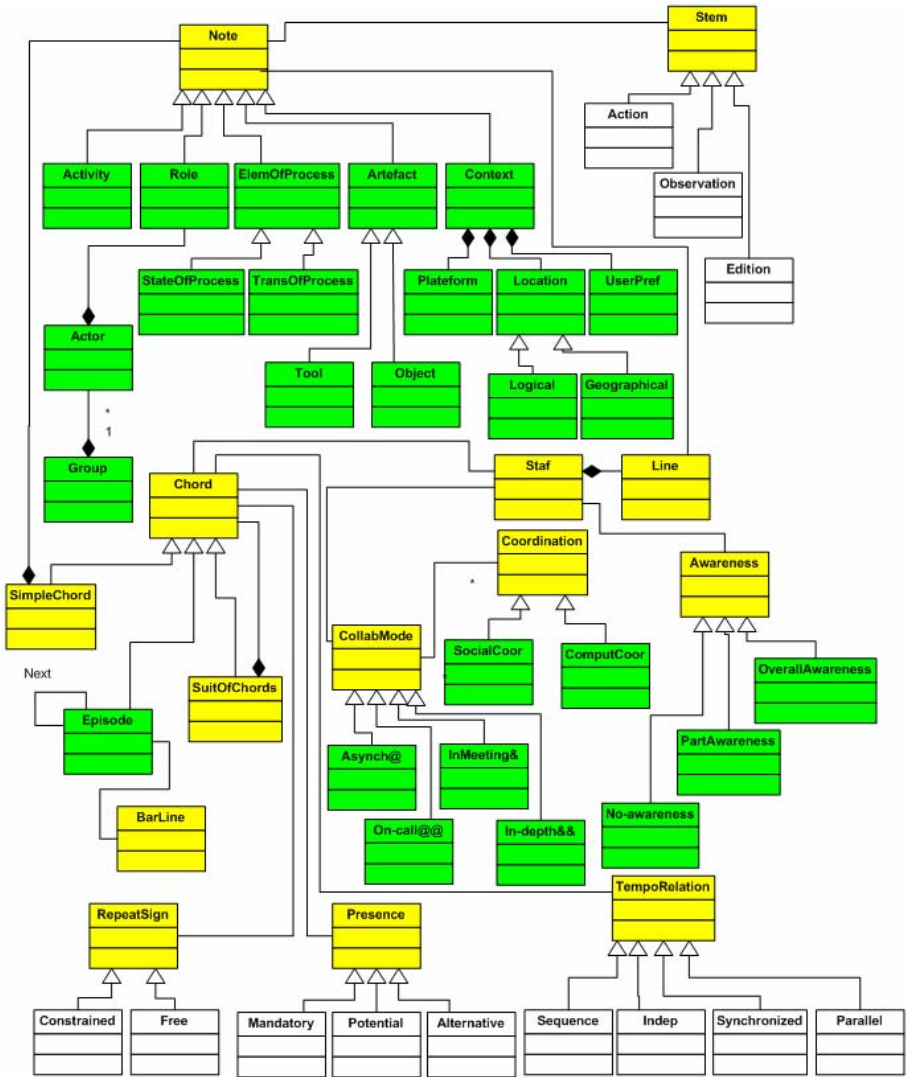


Fig. 3. ORCHESTRA metamodel with note, stem, chord, episode, collaborative modes, awareness, temporal relations, repetition signs and presence as main concepts

- In the morning, before leaving the company, the technician (main actor) loads on his PDA necessary information for his round with appropriate information (clients and their addresses, nature of intervention ...). **State: Init, Actor: technician, @**
- At client house, the technician works on maintenance process, he can study history file of the supplies, precise blueprints, elaborates a diagnosis using appropriate tools, and repair, or ask for spare parts. **State: Work, Actors: Client, Technician, &**

- In a situation of impossibility to diagnose alone, he can contact his manager (secondary actor) to ask him some helps and to exchange some information. He can also contact, in a synchronous manner the heating manufacturer expert (secondary actor) to study the situation with him. **State: Coop, Actors: Technician, Manager, Expert, &&**
- At the end of his round technician, back at the company, updates history file of visited equipments and gives his intervention statement. **State: End, Actor: Technician, @**
- Next day the clerk (secondary actor) produces the financial balance and statement of accounts and either sends the bill to the client or he integrates it in the client record. **State: FB (Financial Balance), Actor: clerk, @**

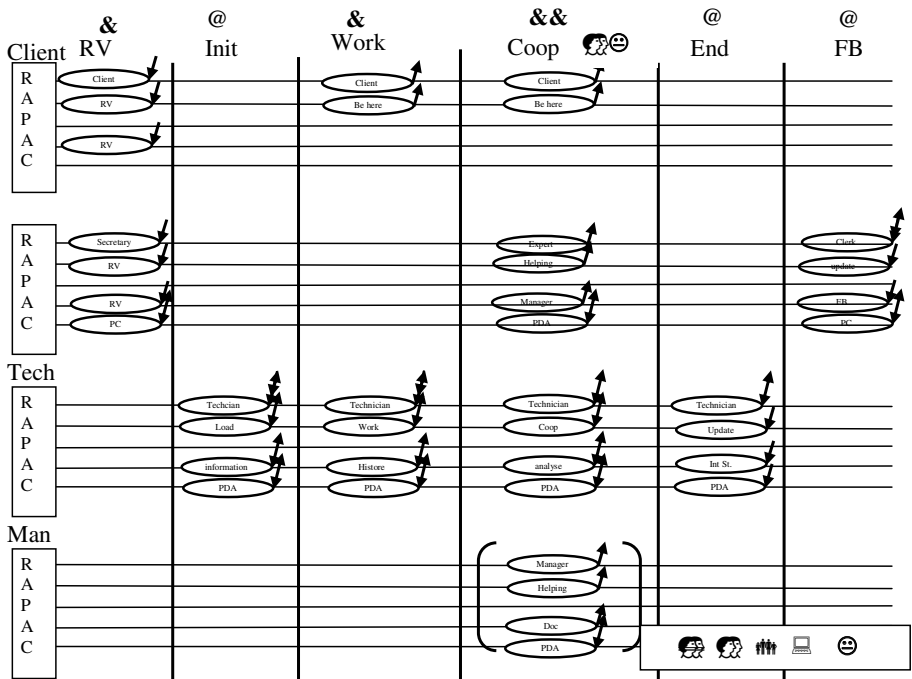


Fig. 4. Different ORCHESTRA description of heating maintenance activities example

In Fig. 4 we show ORCHESTRA modelling for this example. We are proposing several levels of description; individual activities are expressed by a staff (first staff). For collaborative ones either synthetic (second staff) or in-depth description are authorized. In-depth cooperative description is characterized by a precise description of each role of the same activity, activity set or activity period (period Coop). On the same sheet, for each role, a staff describes its situation. In this way it is possible to

understand globally all collaborative activities. A more synthetic view is obtained if we put together on the same staff several roles evolving simultaneously in the same (cooperative) activity. Corresponding cooperative process is presented on Fig. 5.

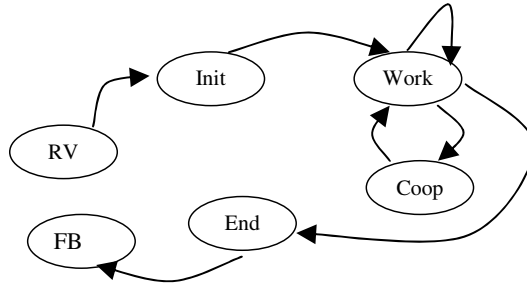


Fig. 5. Collaboration process, expressed as a state – transition network

4.2 Cooperation in Teaching in Large Class

As second example of ORCHESTRA modelling, we present our study on how to make teaching in large class livelier and more participative by using wireless devices. We describe briefly the challenge of teaching in large class and present the DRIM-AP project (French acronym for Multiple Interactive Radio Devices and Participative Lecture Theatres) at Ecole Centrale of Lyon (Graduate School of Lyon).

Lecture in large class is widely used in higher education as a teaching method but is also known to present some difficulties [19] like lack of exchanges, rarefaction of feedbacks, difficulty to motivate students. Indeed, teachers are often challenged by having to perform a lecture and to wonder about students understanding level at the same time. It needs from teacher a part of his attention and concentration to analyze student's behaviour and to encourage them to participate. On student side, lectures are sometimes perceived as boring and participation is complicated by shyness and fear of being ridiculous in front of teachers or fellow students.

Over the past few years, e-learning has hugely increased the use of technologies in educational area, providing software solutions to manage distant students through the use of learning platforms or virtual classrooms. Nevertheless, a few has been accomplished to really integrate the use of technologies in face to face lectures. But since the rise of wireless networks and mobiles devices, all components are now available to build solutions that could help teachers to increase participation and get more feedback from students in large class lectures [20]. That's the topic of our research program at Ecole Centrale of Lyon where we are working on the DRIM-AP [11] project whose aim is to provide a software solution of teacher-students interactivity based on the use of wireless devices and Wi-Fi local networks.

Implementation of mobile campus and student's laptops equipment policies provide a favourable background for new interactive teaching solutions. In a large class context, like shown in Fig. 6, each student can use his wireless device to

communicate with the teacher during lecture. On his side, the teacher displays the slides and monitors the class with the same computer.

The DRIM-AP project takes place in a research program, with a support from the Hewlett Packard company in the context of the 2004 HP EMEA Mobile Technologies for Teaching Grants. Our aim is to design a tool based on Cooperation Interaction and wireless technologies in the face to face approach of teaching and evaluate the usability of this kind of tool for students as well as for teachers. On the functional level, our work gathers the main orientations identified during the state of the art study:

- interactive tests, initiated by the teacher and addressed to all students in a synchronous way,
- individual and asynchronous students feedbacks concerning mainly two aspects: lecture speed (too slow, too fast) and lecture understanding,
- individual and asynchronous questions to the teacher,
- synchronous vote.

At the teacher side, we propose improved monitoring tools for managing the interactive class, broadcasting tests and polls, receiving and reading students' messages on the teacher laptop computer as well as slides control, tests elaboration and submission, answers collection, consolidation and synthetic presentation, vote organization, answer writing and sent to a particular student (individually) or broadcasted to all students (collectively).

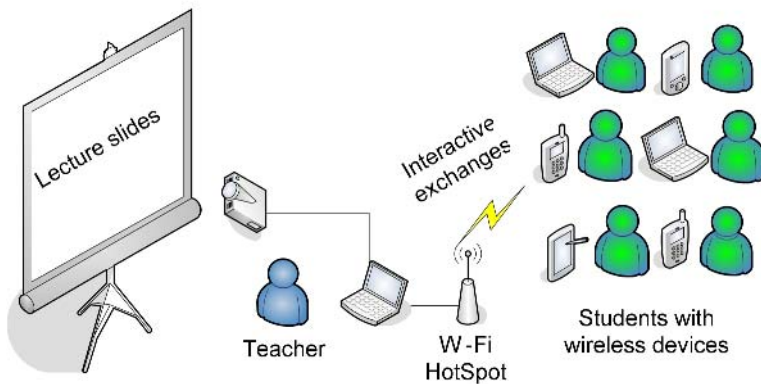
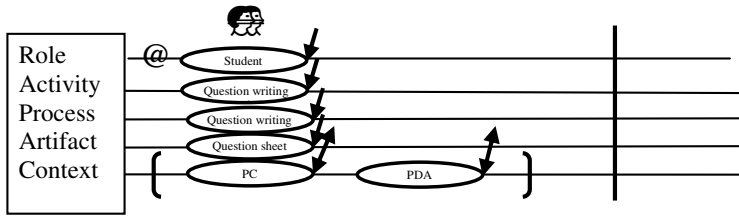


Fig. 6. Principle of an interactive large class using wireless devices

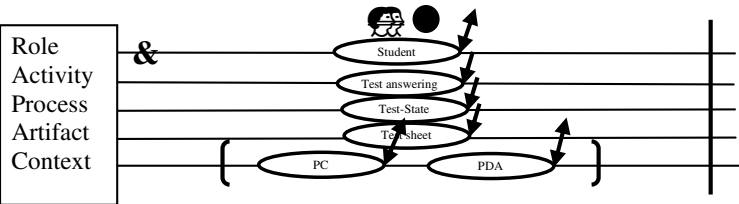
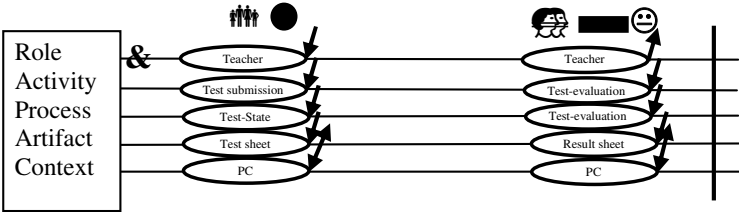
At the student side, communication feedback to the teacher about lecture, like speed (too slow, too fast), understanding, sound and visual ambiance is possible. Students can also write and submit questions or remarks, take notes, answer to full assessments or quick polls.

We show on Fig.7 several tasks and their context:

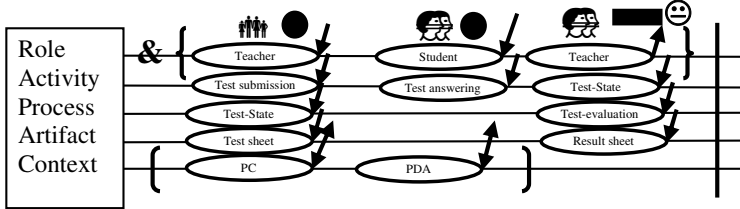
- 7A describes asynchronous individual student task “question writing”



A – Individual activity: question to teacher writing



B – Collaborative activity - In-depth view: test preparation, submission and evaluation



C - Collaborative activity - Synthetic view

Fig. 7. Different ORCHESTRA descriptions: individual, cooperative, in-depth or synthetic views for DRIM-AP case study

- 7B describes in in-depth manner synchronous cooperative task “interactive test”
- 7C gives a synthetic version of this task.

All events (question, feedback, slide display...) are tracked by DRIM-AP. In this way the lecture can then be replayed by the teacher in order to identify when the main problems (understanding, lecture speed ...) occurred. Corresponding cooperative process is presented on Fig. 8.

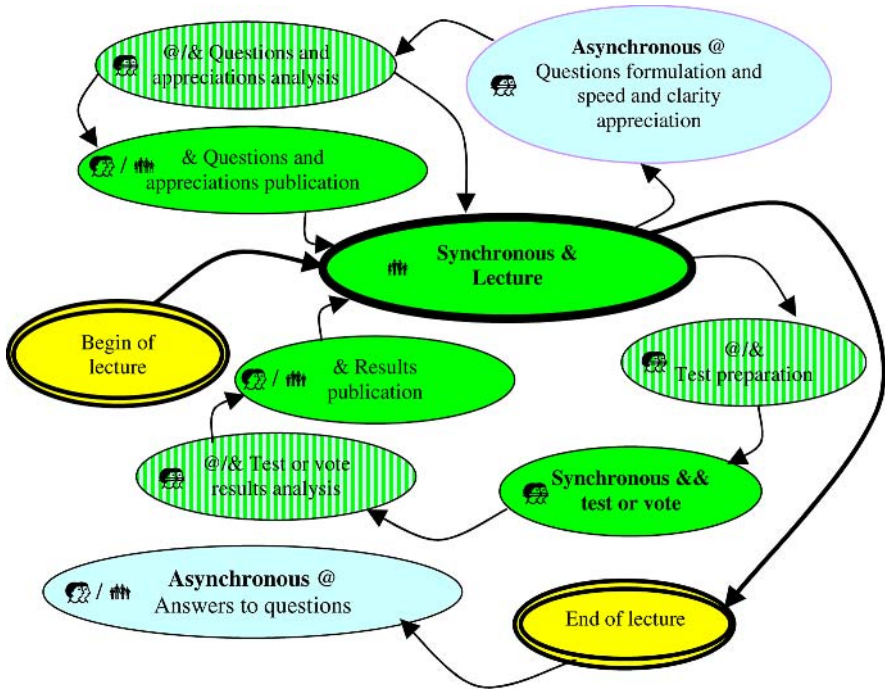


Fig. 8. DRIM-AP collaboration process, expressed as a state – transition network

5 Transformational Process from ORCHESTRA to Cooperative Architecture

As explained briefly in the CoCSys presentation, the target of our process is a generic software architecture composed of three layers decomposition as a generic framework for cooperative mobile pervasive systems. The top layer corresponds to the collaborative application level. It contains all the cooperative software used by the actors. This level is totally user-oriented, which means that it manages interaction control and proposes interfaces for notification and access controls. It uses multi-user services provided by a second layer. This one is a generic layer located between application layer and the distributed system layer. This layer contains common reusable elements of groupware activities and acts as an operating system dedicated to groups. It supports collaborative work by managing sessions and users, provides generic cooperative tools (e.g. telepointer) and is responsible for concurrency control. It also implements notification protocols and provides access control mechanisms. The last layer is essentially in charge of message multicast and consistency control. Usually, it is a computer-oriented layer which provides transparent mechanisms for communication and synchronization of distributed components.

Currently, we are developing a cooperative middleware called SMAC (Services for Mobile Applications and Collaborations) that implements the two lower layers (groupware services and distributed system) of this conceptual cooperative architecture (see Fig. 9).

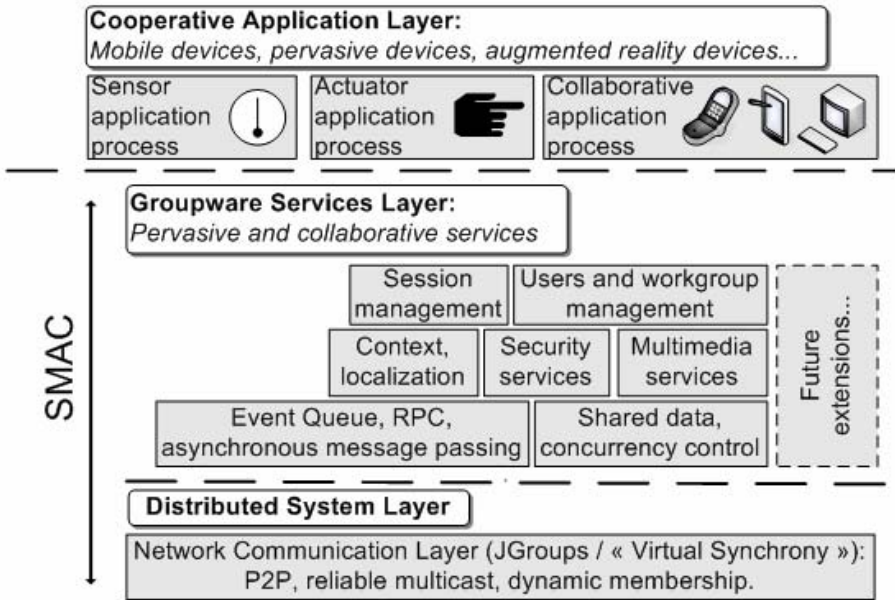


Fig. 9. Three layer collaborative architecture structure

As we target mobile devices, we have strong constraints for the choice of technology for the Distributed System Layer. On one hand, synchronous cooperation is hard to implement with lightweight clients, and on the other hand heavy distributed objects systems such as Corba or J2EE are not available on mobile devices. Currently we choose to base SMAC on the Virtual Synchrony distributed programming model [3, 6], by using a version of JGroups [2] specially implemented for the J2ME / CDC Java Virtual Machine. Although the Virtual Synchrony programming model does have some limitations regarding specific mobile CSCW scenarios (mainly: it does not scale well to a large number of concurrent users, and it is not very adapted to situations involving lots of connections and disconnections), it does fit well with the kind of scenarios that we are experimenting, and it provides convenient and powerful abstractions of cooperating processes that need to keep coherent states.

Above this layer, the Groupware Services Layer is composed of a core framework of Java classes onto which we can plug specific groupware services as needed. Currently, only a subset of these services is implemented, mainly the classes corresponding to the notions of collaboration, cooperation episode and session, and the classes representing users and groups. This provides a minimal system that handles synchronous collaboration, as well as persistence of collaboration states between sessions.

The relation between ORCHESTRA and the generic architecture is the following: Information coming from the ORCHESTRA description concerning roles, activities, process, artefacts and context is “projected” on this architecture. This projection

concerns either application layer or collaborative layer, whose core classes are summarized in figure 10. Information about role and actors is manipulated at the collaborative layer, as well as at the application layer, where the corresponding user interface is proposed. ORCHESTRA concept of activity is translated to the SMAC in two different ways. An application specific activity, called semantic activity, is located at the application layer, for generic activity its location is naturally at the collaborative layer. Concerning cooperation processes management expressed in ORCHESTRA by episodes and their orchestration, their corresponding SMAC classes are using an adaptive workflow engine. ORCHESTRA artefacts are either tools or objects, generic or semantic. Their mapping to SMAC is done either at application layer (for semantic artefacts) or at collaborative layer for generic ones. Tools are used or activated at application layer and objects are manipulated by services located either at application layer or at collaboration layer depending of their specificity or genericity. Context description expressed by ORCHESTRA is used at physical level concerning hardware platform description, at distribution layer concerning software level description and either at collaborative layer or application layer concerning location adjustment and user preferences. Main mechanisms used during this transformation are XML encoding and decoding of information manipulated in ORCHESTRA editor and interpretation engine, which is able to read these XML files and execute appropriate code either generated from this description or corresponding attachments doing the link with existing code at collaboration layer or specifically developed code at application layer.

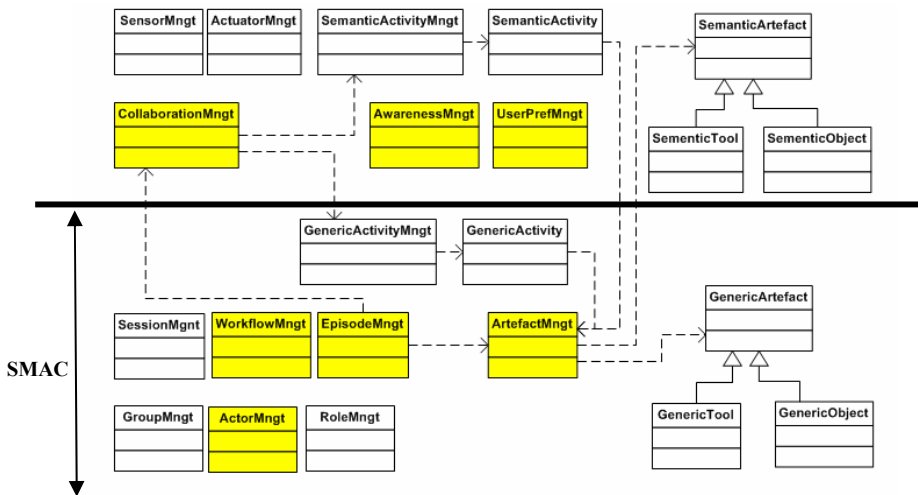


Fig. 10. SMAC and Cooperative Application Layer core classes

As illustrated in Fig. 11, according to platform adaptation mechanisms we are able to produce appropriate interfaces in regard with hardware platform used i.e. laptop, PDA or Smartphone.

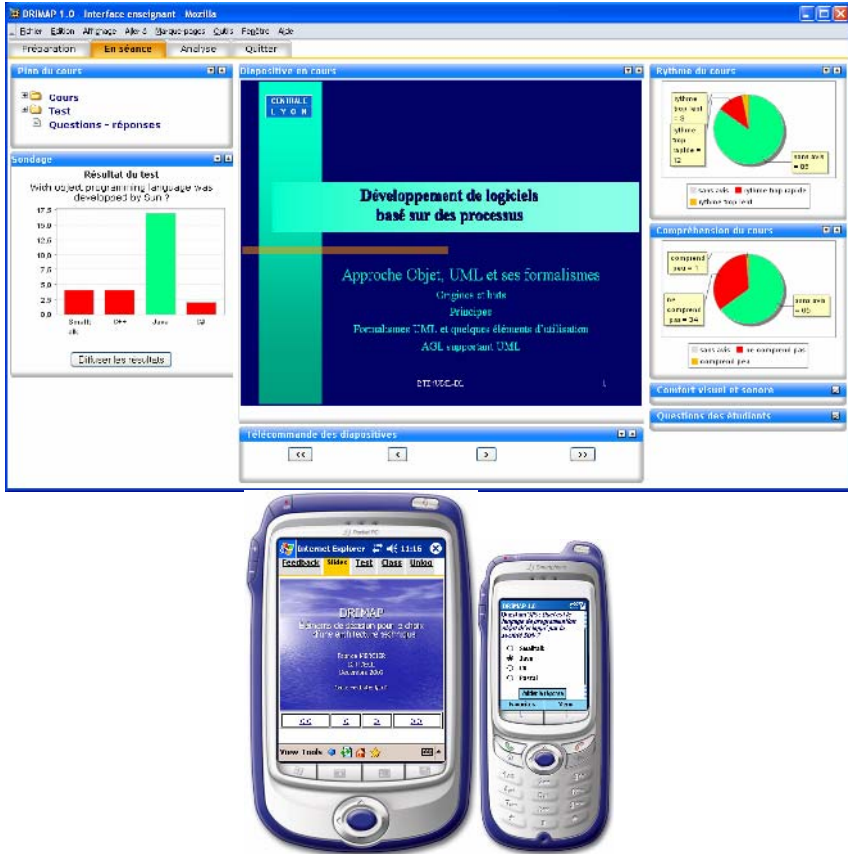


Fig. 11. Screenshots of DRIM-AP user interface on different devices

6 Conclusion

In this paper, we outlined a new formalism called ORCHESTRA, which objective is to provide a graphical expression of Cooperative Behaviour Model. CBM, elaborated from a collection of scenarios, as a reference for the transformation process allowing different implementations. As it is important to associate different actors to this constructive process, we propose a formalism which could be used during initial discussions as well as during the implementation and adaptation process. We expressed two concrete examples highlighting description capacity. We also explained the relation between the formalism and generic software architecture on which expressed mobile collaborative system can be implemented.

ORCHESTRA has been tested on several concrete examples and we continue to upgrade it by new concepts as result of these tests. The connection with mixed reality has not been described in this paper, when if we are currently working on it.

References

1. Andriessen J.H.E. *Working with Groupware: Understanding and Evaluating Collaboration Technology*, Springer, CSCW Series 2003
2. Andriessen J.H.E. *Working with Groupware: Understanding and Evaluating Collaboration Technology*, Springer, CSCW Series (2003)
3. Ban B. *Design and Implementation of a Reliable Group Communication Toolkit for Java*. Cornell University (1998)
4. Birman K. *Reliable Distributed Systems: Technologies, Web Services, and Applications*. Springer (2005)
5. Carroll J. M., *Making Use: Scenario-Based Design of Human-Computer Interactions*, the MIT Press (2000)
6. Chalou R., David B.T. IRVO: an Interaction Model for designing Collaborative Mixed Reality Systems, HCI International 2005, Las Vegas, USA, 22-27 July (2005)
7. Chockler G.V., Keidar I. and Vitenberg R.. *Group Communication Specifications: A Comprehensive Study*. ACM Computing Surveys, 4(33):1-43, December (2001)
8. David, B.T., Chalou, R., Vaisman, G., Delotte, O., *Capillary CSCW*, In Proc. of IHCI, Crète (2003).
9. David B. T. IHM pour les collecticiels. In Réseaux et Systèmes Répartis, Hermès, Paris, vol. 13 (2001) 169-206
10. Ellis C., Gibbs S.J., Rein G.L. *Groupware: some issues and experiences*. In Communications of the ACM, vol. 34, n° 1, (1991) 38-58
11. Ellis C., Wainer J. *A conceptual model of Groupware*. In Proc. CSCW'94, ACM Press, (1994) 79-88.
12. Mercier F., David B., Chalou R., Berthet Jp. *Interactivity in a large class using wireless devices in Mobile learning anytime everywhere*, Attewell J., Savill-Smith C. (Editors) *A book of papers from MLEARN 2004*, LSDA (Learning and Skills Development Agency) (2005) ISBN 1-84572-344-9
13. Mori G., Paternò F., Santoro C. *CTTE: Support for Developing and Analyzing Task Models for Interactive System Design*. In IEEE Transactions on software engineering, vol. 28, n. 9 (2002)
14. Object Management Group, <http://www.omg.org/mda/>
15. Paternò, F., *Model-Based Design and Evaluation of Interactive Applications*, Applied Computing Series, Springer -Verlag (2000)
16. Rosson, M.B., Carroll, J.M.. *Usability engineering scenario-based development of human-computer interaction*. Morgan Kaufmann (2002)
17. Salber D., Coutaz J., Decouchant D. Riveill M. *De l'observabilité et de l'honnêteté dans la communication homme-homme médiatisée*. In Proceedings of Septièmes Journées sur l'Ingénierie de l'Interaction Homme-Machine IHM'95, Toulouse, France, Toulouse: Cépaduès (1995) 27-33
18. Szekeley P., *Retrospective and Challenges for Model-Based Interface Development*, CADUI'96, Namur, 5-7 June 1996 J. Vanderdonck (ed.) *Collection "Travaux de l'Institut d'Informatique" n°15 Presses Universitaires* (1996)
19. Stewart D., *The Musician's Guide to Reading and Writing Music*, Backbeat Books (1999)
20. Truong, T. M., Griswold, W. G., Ratto, M., and Star, S. L. *The ActiveClass Project: Experiments in encouraging classroom participation*. Technical Report CS2002-0715, UC San Diego, Department of CSE (2002) <http://www.cs.ucsd.edu/users/wgg/Abstracts/aclass.pdf>
21. VanDeGrift, T., Wolfman, S. A., Yasuhara, K., and Anderson, R. J. *Promoting interaction in large classes with a computer-mediated feedback system*. Technical Report 02-12-02, University of Washington, Computer Science & Engineering (2002)

A Decentralized and Flexible Tool Supporting Extreme Programming Software Development

N. Baloian¹, F. Claude¹, R. Konow², and M. Matsumoto³

¹ Universidad de Chile, Departamento de Ciencias de la Computacion, Blanco Encalada 2120, Santiago, Chile

{nbaloian, fclaude}@dcc.uchile.cl

² Universidad Diego Portales, Ejército 441, Santiago, Chile
rkonowkr@al.udp.cl

³ GITS, Waseda University, 1011 Okuboyama, Nishi-Tomida, Honjo-Shi, Saitama 367-0035, Japan
mmatsumoto@waseda.jp

Abstract. This paper presents a system called CodeBreaker for supporting small and medium size software development based on an extreme programming principle. The system follows a decentralized model of development, which means, it does not requires a central repository. A set of rules for code ownership maintains the synchronization of the work among all members of the developing team which can work on- or offline. It allows fine-grained locking of parts of the code.

Keywords: collaborative software development, peer-to-peer, extreme programming.

1 Introduction

The development of systems for supporting distributed programming teams has attracted the attention of many authors in the past. Most of these systems are developed for supporting a particular software development style. For example, in [1] a system for supporting distributed teams in extreme programming is presented. In [2] the authors describe a system for supporting distributed software development based on a peer-to-peer architecture, in opposition to the most common centralized repository architecture. Version control systems like CVS [3] or SourceForge [4] are perhaps the most frequently used today for supporting collaborative, distributed programming. Although the development of this kind of systems has been very prolific in the past, there are many reasons to believe that there is still room for improving software development support and that the last word is far from being said. This is especially true when we consider new situations that arise from new scenarios created by the existence of pervasive computing enabled by mobile technology like wireless LANs and smaller, lighter and more powerful notebooks. This scenario promotes the emergence of small programming teams, which may start developing a small to medium size project in a brain-storming like meeting. Such kind of situations is becoming more common, as most people use their own notebook computer as their

working machine anywhere, be it at home, at work, or even during a coffee break, which has been characterized as “nomadic” computing. To more concretely illustrate the requirements of the software developing scenario we want to support, let’s take the example of two or three programmers that get together and exchange ideas about a new system they have just conceived in a planned or spontaneous brain storming session. They open their laptops and start developing a new project, which they more or less outline by creating new classes which contain just some sample code or comments. A wireless network may be present, permitting them to work synchronously. If not, they will have to exchange files to merge their work asynchronously, maybe by email or by pen drives. They decide to continue separately dividing the job and responsibilities. All or some of the original members may meet again, new members who joined the project may be also present and they will have to merge their work. They will certainly welcome a tool for coordinating their work satisfying the following requirements:

- **Work on a peer-to-peer architecture without having a central repository.** As we want to support people who may start a new development without previous preparation, a central repository may not be always available for all members at that moment. Because of this, every member of the developing group should have a copy of the project as updated as possible, even when working alone.
- **Allow synchronous and asynchronous collaborative working.** The system should support the synchronous collaboration work when two or more users are online, providing adequate tools. But it should also allow synchronizing the work with other participants which are offline in the best possible way, and provide mechanisms for merging the code developed offline.
- **Allow the inclusion of new unforeseen participants** Because the system is aimed to support flexible and changing teams, there should be a way to include unforeseen participants and assign them tasks. However, the system should avoid an uncontrolled explosion of participants and maintain a certain order in the versioning of the code.
- **Allow fine grained and logical oriented locking of code.** In a less formal and flexible team everyone may have access to all the code and be able to modify it. However, this condition may introduce too much complexity for synchronizing the work. A good trade-off solution may be that the system should give the possibility of locking finer parts of the code inside a file, like a class, a instance variable or a method. It also should allow reserving names for allowing the locking of code which has not been written yet. In this way, participants may distribute the work among themselves by just locking names of classes, methods or even variables which are still not written or used.

Of course, for this scenario we have an extreme programming style of development in mind. Extreme Programming (XP) is a software development methodology, which emphasizes bringing the project to the beta testing phase as quick as possible, reducing the time of planning phase and increasing the priority for the beta testing phase [5]. Currently, the cost and time to develop small or medium-size software using the classic software engineering methods is too high. XP stresses the

collaborative work and distributed programming. Most of the systems claiming to support software development according to XP are focused on supporting the synchronous work. Some of them are mere collaborative editors while others include support for coordinating the work, like awareness and versioning mechanisms. Some authors have already pointed out to the necessity of not having a centralized repository to coordinate the work of a software developing team [6], while others also have stressed the necessity of having a fine grained, logical oriented locking of the code [7]. The decentralized model is certainly the most flexible and suitable model for the requirements of the situation we are going to address.

However, there is still no system which meets all the requirements mentioned. Developing such a system represents a challenge of high complexity, in the design and in its implementation. In this work we will present a system called Codebreaker for supporting small and medium size software development based on an extreme programming principle, meeting the requirements mentioned above which corresponds to a specific subset of what is known as XP.

Because of the logical locking of the code requirement we will develop it for supporting a particular programming language which is Java. However, most of the restrictions that this language imposes are easily transferable when implementing the same system for any other object oriented language or even a modular programming language with some modifications.

2 Related Work

Back in the late 80's and early 90's when the Internet was rapidly expanding, there was a great interest in the distributed systems. It was then predicted that such systems will be the dominant technology for the synchronous collaborative work in the future [8]. We can nowadays confirm those predictions and add that these systems have also deeply influenced the working style in all fields. Of course, computer system programming was one of the first, and many systems have been developed since very early. We can classify those systems in two categories according to the aspect they stress with their support.

2.1 Versioning Management Systems

In the 1990's perhaps the most used tool for collaborative work synchronization was created, CVS, [3] initiating a wave of development of tools supporting Version Management. CVS problems are well known [9]: it uses a centralized model, a central data repository and only few operations or commands which can be executed offline. This makes this structure really unsuitable for synchronous collaborative programming development. All developers need access to the central server for almost all operations. Today, there is a whole family of CVS-like tools: GNU-Arch, Subversion, CSSC, PVCS, etc. These applications are frequently used in the Open Source community and also in large business environments. All of them follow the same schema: one central repository, and file-level permissions. (Check in, out).

2.2 Collaborative Development Environments

One of the first approaches to the implementation of collaborative development environments is the Orwell system [10]. This system allows the Smalltalk programmers to develop programs using a common library. An interesting aspect of this system is that it organizes the developing system code in methods and classes instead of files, thus using a more logical approach to present the code. Another Collaborative Environment that follow the same idea of the Orwell system is Tukan [11]. This synchronous distributed team programming environment for Smalltalk claims to solve the problems that Extreme Programming teams have. Tukan incorporates a version management system and adds awareness information, communication channels and synchronous collaboration mechanisms. It also provides a shared code repository with a distributed version management and the code integration can be made in a centralized or decentralized way. The IBM Rational ClearCase System [12] provides real time support for collaboration between developers located anywhere on the Internet. It uses a central server that manages user's permissions and differences between the source code versions. The server has also support for multiple repository server deployments for large-scale enterprise teams. Another tool to which supports the collaborative editing of source code is the *Collab* add-on for the Netbeans 5.0 [13]. This add-on allows the NetBeans users to edit files collaboratively, share files and provides space to communicate with other developers. An interesting system which does not rely in a central repository but has the ability of use multiple repositories was recently presented in [14]. The principle behind it is that of using and re-using software components from different repositories while also offering the own local to the rest of the community.

3 The CodeBreaker

3.1 Rules for Code Ownership

In order to allow the synchronization of the code being developed among the members of the group in an asynchronous scenario CodeBreaker imposes that any existing code in any of the participants' computer should be "owned" by someone. A CodeBreaker code development project starts with one person defining the project and others joining it. Each new member including the one who created the project has to register an e-mail address and receives a digital signature. All members can develop new code which is owned by him/her. Other members will receive the code and can use, modify, and even share it with others, but the only "official" version can be distributed or approved by the owner. In this way, there will be always a final version of the entire software which will be the sum of the code pieces each participant owns and has released. The rules for sharing the work should guarantee that there will be no inconsistencies about the final version (this will be discussed in section 3.4). In order to allow users to delegate their work, users can pass the ownership of the code among each other. Figure 1 shows an example how ownership of code may develop during a project involving three programmers.

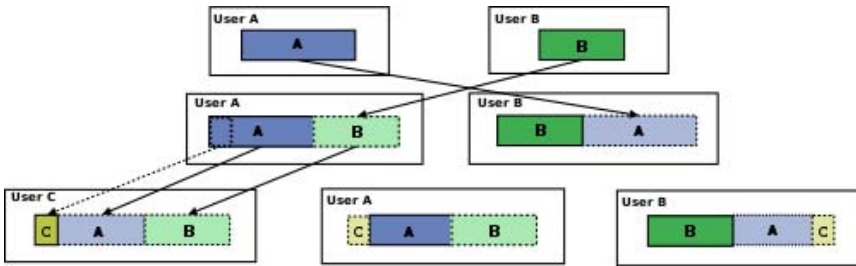


Fig. 1. The different colors show the ownership of the code. In the first row, A and B start a new project writing both a part of the code. In the second, they merge their works and keep the ownership. In the third row, C joins the project and A grants ownership rights to part of the code.

3.2 Exceptions to the Rules

It is important to maintain the rights of the owner of the code and the order of the project itself in order to avoid an uncontrolled explosion of versions. It is also known that in many projects it is sometimes impossible to maintain and respect every rule because of the emergence of unforeseen situations, so an alternative should exist for bypassing the rules in exceptional cases. For example, it could happen that a certain user cannot work on the project anymore and that he is not reachable to ask him to delegate the work to other users. In this case there are two mechanisms that can be applied and the two coexist giving more flexibility to the system. The first one is that a user can ask the rest of the team to approve or reject by voting a petition for becoming the owner of a certain code piece that is owned by a third member of the team and/or to force the acceptance of a given modification.

3.3 Logical Locking

As we already said, the entities of the code which can be owned are logical more than physical one. Logical entities which can be locked are organized according to the JAVA organization of the code. The locking is done over a name of a class or interface, a method inside a class or a class variable. In this way, it is possible to lock code which still not exists. The scope of the locking also follows the class hierarchy of Java: If a class name is locked all the extended classes will be locked as default. In the same way, if an interface file is owned, the implementations of those methods are also owned. This may be necessary in cases when one of the participants should write the same method for different classes, for example, the same programmer developing a drawing method for different objects representing graphical elements. An exception to the past rules is for example what happens with packages. Even if they are owned by a certain user, any other user should have the chance to create classes inside it.

As the system is aimed to support the development in Java and is implemented as a plug-in for NetBeans 5.5, the logical separation of the code is based on the same granularity provided by this development tool. Every part of the code is assigned to a user and it appears locked for the rest of the development team. It is important to notice that locking a part of a code means that a specific snipe of code is owned by a

specific user, so other users can not distribute modified code as a final accepted code. They need the permission of the actual owner. However, they have the chance to modify it for personal use or to present it to the owner or the rest of the team for being accepted as final in the future.

By automatically locking the inherited classes of a locked class, i.e. the user that owns a specific class, owns by default the subclasses that extend it, a better control of the whole system is achieved. For example, a class that has been implemented to fit a small set of requirements and is not completely defined could have many changes in their implementation issues, the data representation, and many similar details. This functionality ensures that the users that try to inherit from such classes must have the permission from the owner of the parent class, preventing inconsistencies

It is certain that having temporary code or avoiding modifications completely is not possible, but this option of the system allows giving a little more control to the process and as it is based on the rules defined for the system, they are still flexible enough to support a more relaxed working style.

3.4 Synchronizing the Work

Synchronization must be possible when working synchronously as well as asynchronously. When working synchronously the information about changes of any type is sent to all connected participants. When a latecomer joins a working session with one or more other participants, their records are compared to update information about changes. Only code changes which are issued by the owner of the code are forcibly exchanged. There is no conflict about which is the latest version, since the owner issues a correlative number when its code is being distributed. This number is also used to check if the change has been incorporated already. So if two users A and B meet and have two different versions of a code owned by a third member C, the version of this part of the code with the highest correlative number will be copied to the file system of the user with the lower correlative number. When an owner wants to publish a new version of a code a file with an XML content containing metadata and data for the code is generated and signed with his digital signature. The same is done for distributing information about changes to the code ownership and new members.

For participants who are seldom online with the rest of the group or if various subgroups do not meet each other frequently CodeBreaker offers an asynchronous mechanism based on the use of e-mail. The XML files with the changes are sent to all email addresses of the project. Users can download them and process them offline.

As the system is planned to work on an XP-like environment, the option of pair programming [15] is a very important issue. To allow pair programming, a user should ask for being watched by another user. The user that begins to watch should have permission of modifying parts of the source code and to see real-time the modifications made by the user that sent him the invitation. When both ended to work as a pair, the source code should be saved on both workstations, but the modification should be marked as from one user only, so that the owner receives only one confirmation of a given code.

3.5 Assigning Roles

CodeBreaker is aimed to support more a flat project structure in which every participant has the same rights and responsibilities. However, sometimes even in small projects there may be a need for having a certain hierarchy in order to maintain the synchronization among the participants. CodeBreaker introduces two mechanisms which allow this with flexibility. The first one is, when a user is created it may or not receive the right of accepting new participants for the project. The number of participants which is allowed to invite can be also specified. This rule helps to keep the control about the number of participants in the project. The second one is about receiving the ownership of a code. A user may receive or not the permission of passing the ownership of a code to a third one. This may be used to assign responsibilities to certain members of the team which they will not be able to avoid by granting rights to another member.

4 Conclusions

With the system presented in this document it should be possible to support a real XP project development based on the conditions described in this work, giving the chance to the development team to use a tool that has the flexibility enough to develop the software without having troubles because of a complicated tool. The simplicity behind this idea gives the real chance to give a competitive tool. The rules the system implements about ownership of the code for controlling the coordination of the participant's work also support this fact and add more flexibility, so that the user can create a project that works under the rules that are most similar to the way his/her team really works.

The usage of a widely known IDE is very important, not only because there is no need to build one from scratch, but also because it does not represent a real adaptation to new software for a development team.

Another important aspect about this work is the fact that many small projects developed in real life, such as small software for limited purposes, internal utilities for companies and also including many small open source projects, are developed under an XP-like methodology under the conditions we described.

In order to implement the peer-to-peer communications among the online participants the system uses the JXTA™ [16] technology, which provides libraries and APIs aimed at implementing peer-to-peer systems.

Codebreaker use this technology to discover the participants of the developing team in the LAN and to establish a connection between them. JXTA also allows the system to be extended for many users, so that they can be connected from anywhere in the Internet, even through firewalls.

CodeBreaker is still in the prototype implementation phase. To continue the work over this idea, we plan first to finish the development in order to test of the system in real environments. With all this information and the information of every tested team about its past projects, the efficiency of this tool could be really measured and it could be possibly to conclude about its effectiveness.

References

1. Schümmer,T. ,Schümmer,J. : Support for Distributed Teams in eXtreme Programming,In eXtreme Programming Examined, edited by Succi, Giancarlo, Marchesi, Michele , Addison Wesley, 2001.
2. Bowen,S.,Maurer,F. : Designing a Distributed Software Development Support System Using a Peer-to-Peer Architecture, 26th Int. Comp. Software and Apps. Conf. (COMPSAC 2002), pp. 1087-1092, 2002.
3. Berliner,B : CVS II:Parallelizing Software Development, 1989.
4. SourceForge,<http://www.vasoftware.com> , last visited on 14 February 2006.
5. Beck, K. : Extreme Programming Explained. Addison-Wesley, 2000.
6. Van der Hoek, A. , Heimbigner,D. , Wolf, A.L. : A generic, peer-to-peer repository for distributed configuration management, icse, pp. 308, 18th International Conference on Software Engineering (ICSE'96), 1996.
7. Magnusson,B. ,Asklund,U.,Minör,S. : Fine-grained revision control for collaborative software development, Proceedings of the 1st ACM SIGSOFT symposium on Foundations of software engineering, pp. 33 – 41, 1993.
8. Xu1,B., Lian,W., Gao,Q. : A General Framework for Constructing Application Cooperating System in Wind, ACM SIGSOFT Software Engineering Notes ,Volume 28 , Issue 2 (March 2003), pp. 15
9. Neary,D. : Subversion - a better CVS, <http://www.linux.ie/articles/subversion/> last visited on 13 February 2006
10. Thomas,D. , Johnson,K. : Orwel, a configuration management system for team programming, Conference on Object Oriented Programming Systems Languages and Applications, pp. 135 – 141, 1988.
11. Schümmer,T. ,Schümmer,J. : TUKAN: A Team Environment for Software Implementation. OOPSLA'99 Companion. OOPSLA '99, Denver, CO, pp. 35-36, 1999.
12. IBM Rational ClearCase, Integrated SCM for Rational Developer products and Eclipse, <ftp://ftp.software.ibm.com/software/rational/web/whitepapers/int-scm-rad-eclipse.pdf>, White papers of IBM, December 2004
13. Netbeans, Sun Microsystems, <http://www.netbeans.org>, last visited on 13 February 2006.
14. Warzee, Xavier The Valtech Collaborative Cockpit, to appear in the proceedings of the CCE workshop, Prague, Czech Republic, April 2006
15. Padberg,F. , Muller,M. M. : Analyzing the Cost and Benefit of Pair Programming, metrics, p. 166, Ninth International Software Metrics Symposium (METRICS'03), 2003.
16. JXTA Technology: Creating Connected Communities, Sun Microsystems, <http://www.jxta.org/docs/JXTA-Exec-Brief.pdf>, last visited on 13 February 2006.

The PoEML Proposal to Model Services in Educational Modeling Languages

Manuel Caeiro-Rodríguez, Martín Llamas-Nistal, and Luis Anido-Rifón

University of Vigo, Department of Telematic Engineering
C/ Maxwell S/N E-36310, Spain
{Manuel.Caeiro, Martin.Llamas, Luis.Anido}@det.uvigo.es

Abstract. This paper proposes a framework to support the modeling of services in *Educational Modeling Languages* (EMLs). EMLs have been proposed to support the modeling of educational units (e.g. a theoretical distance learning course, a lab practice, a discussion-based course). Their modeling approach is based on the featuring of the elements involved in educational units (e.g. participants, data, tasks) and the coordination among these elements (e.g. the order between tasks, the data flow, the assignment of participants to tasks). A key issue in EMLs is the modeling of environments where participants are intended to interact. This part of the modeling involves the featuring of the services and the coordination required to obtain appropriate service interactions. The paper describes the perspectives of a new EML proposal named PoEML (*Perspective-oriented Educational Modeling Language*) devoted to the modeling of services and their coordination: operational, interaction, awareness and authorization.

1 Introduction

Educational Modeling Languages (EMLs) [1], specifically the de facto standard IMS *Learning Design* (IMS LD) [2], were proposed some years ago to support the computational modeling of 'Units of Learning' (UoLs) in accordance with different pedagogical approaches. To do it, EMLs enable the featuring of the elements that participate in educational units and their coordination. They are usually arranged in accordance with a 'Task scheme' involving three main entities: (i) the *Goal(s)* that have to be achieved in each *Task*, which are usually related with an *Object* that need to be produced; (ii) the *Person(s)* that have to carry out each *Task*, who participate in the task playing specific *Role(s)* (e.g. learners and staff); and (iii) the *Environment* where each *Task* has to be carried out (composed by artifacts, applications, services, etc.). Eventually, these UoLs are executed by EML-compliant LMSs (*Learning Management System*), and their activities are eventually supported by tools/service that the LMS integrates. In this way, EMLs offer an approach to enable the development of tailorable groupware.

A key issue in the modeling of UoLs is the featuring of e-learning environments. These environments will be arranged during run-time to support interaction among participants. The modeling of environments in EMLs needs to

consider two main components: artifacts and services. This paper is focused on the modeling of services and their coordination. In order to enhance the reuse of UoLs at design time EMLs should not fix the services to be used during runtime. Accordingly to this approach, services should be described in an abstract way, indicating the behavior required in the e-learning environment. During runtime, different tools providing the required behavior can be used. As a result, the same e-learning environment can be arranged using different tools.

This paper proposes a *service description framework* for EMLs based on the previous ideas. In addition, it proposes to generalize EMLs coordination support to enable the modeling of service control and management issues. This is performed in the context of a new EML proposal, PoEML: *Perspective-oriented Educational Modeling Language*. The main concern of this language is to provide a modular solution that facilitates the modeling of UoLs in an incremental and flexible way.

The next section introduces the modeling of services in IMS LD. Afterwards, the PoEML proposal is briefly introduced to provide an overview of the whole proposal. The following four sections describe the specific components of the *service description framework*. The paper finishes with some conclusions.

2 The Modeling of Services in IMS LD

A main concern in the modeling of educational units is related with learning environments. Learning environments are made up of artifacts (e.g. properties, documents) and services. These elements are intended to be used by participants to perform the tasks. The modeling of the great variety of educational units introduces a requirement for the use of a wide range of services [3].

EMLs could approach the modeling of services in two ways. An initial solution may involve that the EML enables the modeling of all the functionalities required on any educational unit. This approach does not seem feasible. Firstly, it is very difficult if not impossible to provide the wide range of required services. Secondly, as reusable components, UoLs need to remain neutral in terms of software requirements. If they only work with a specific tool then they are not reusable. As a consequence IMS LD has proposed the description of four services and the need to support more services in the future. The SLeD proposal [4] to extend IMS LD has adopted a *Service Oriented Architecture* (SOA) approach to perform this extension, proposing environments configured in accordance with abstract services descriptions. Anyway, currently the IMS LD support to model services is very limited. It only includes four services: email, discussion forum, monitor and search/index. In order for complex designs to be created a greater range of services needs to be described. In addition, service coordination is not considered at all. Therefore, this paper introduces a *service description framework* dealing with the modeling of services and their coordination.

3 Proposal Context: PoEML

This paper is in the context of a larger work whose main purpose is to enhance the modeling support provided by current EMLs [5]. In this initiative the modeling of educational units is performed following a separation of concerns approach. The different concerns are named perspectives. The proposal identifies twelve perspectives: *Functional, Social, Informational, Structural, Operational, Organizational, Process, Temporal, Authorization, Awareness, Interaction, and Causal*. This decomposition in separate perspectives enables to approach the modeling of UoLs in a structured way. Simple educational units can be modeled using the basic perspectives, while complex units may require the more advanced ones. Eventually, the computational modeling complexity is reduced for a large amount of educational units. In addition, the obtained models are more flexible, as changes in a certain perspective do not affect to other perspectives (or they affect in a controlled way).

PoEML is organized in several packages in accordance with the perspectives and aspects identified. Figure 1 illustrates the *EducationalScenario (ES)* element. It is the core component of the proposal. An *ES* is intended to support the modeling of any kind of educational practice at different aggregation levels, from simple lessons, to complete courses. The *ES* element is the aggregation point where all other elements are anchored. Each *ES* constitutes a context of elements not accessible from other *ES*, except from a *Parent ES* to its *Child ESs*. As it is represented in the figure, an *ES* is intended to: (i) achieve a certain *Goal* or set of *Goals*; (ii) that have to be attained by a particular *Participant* performing in a given *Role*; in a particular *Environment* composed by (iii) a set of *Artifacts*, (iv) and *Tools* that represent Applications and Services; (v) in the context of a certain *Organizational Structure*; (vi) considering a certain *Order* in the way in which *Tasks* are intended to be attempted; and (vii) *Temporal Restrictions* on their performance; and involving a set of rules that control and manage (viii) the *Authorizations* of the involved participants to invoke operations; (ix) the *Awareness* they receive during execution; and (x) the *Interaction* through applications and services. Most of the considered elements enable the inclusion of self-aggregations. As example, a role may contain other roles to support the modeling of hierarchical groups.

The next sections introduce the service description framework included in PoEML. The perspectives composing this description framework are: *operational* (about the modeling of services) and *interaction, awareness* and *authorization* (about service coordination with other elements of educational units).

4 The Operational Perspective

This perspective involves the description of services to be included in *ESs' Environments*. It considers the featuring of functional and non-functional requirements. In addition, we propose to model specific service components in order to support their management by the LMS. The solution is related to current

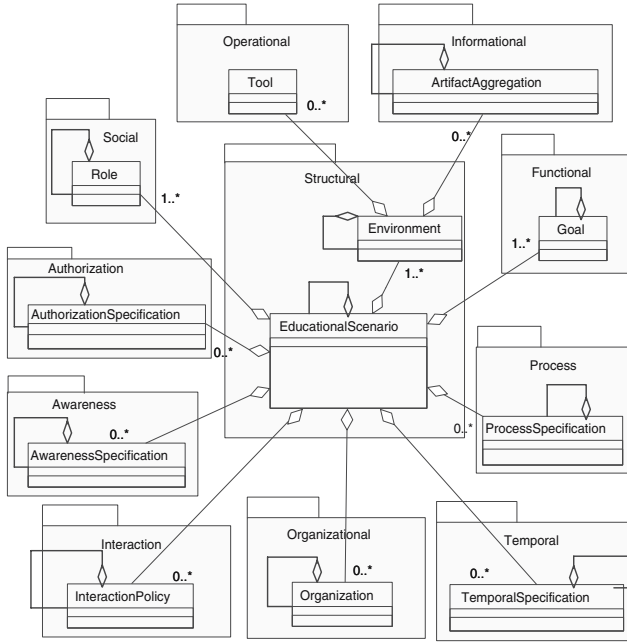


Fig. 1. The *EducationalScenario* element and its main components

technologies that describe services in the context of the *Services Oriented Architecture* (SOA). The next section introduces this technology and then the operational perspective is presented.

4.1 Service Object Architectures

The SOA approach [6] is based on the description of abstract services interconnected in a specific way. During run-time the abstract services are provided by concrete instances that provide the required behavior. This approach is intended to facilitate the development of applications comprised of a number of components which can plug together, instead of the more integrated, monolithic systems. Currently, it is possible to identify two approaches, one developed by the industry and the other one by the academia [7].

In the industry approach the main idea is to encapsulate a certain functionality within an appropriate interface and provide it as a Web service. These proposals only consider the syntactic featuring of Web services. The more important specifications were *Web Services Description Language* (WSDL) [8], that enables the syntactic description of Web services in a language neutral manner, and the *Universal Description, Discovery and Integration* (UDDI) [9], that enables the advertisement of Web services in a universal register. In these registries Web services are published in accordance with categories of a controlled vocabulary. The vocabulary indicates the different types of services.

The academic approach is initiated under the consensus that syntax alone is not enough to develop definitive solutions. At this point, semantic Web services are being proposed to allow the semi-automatic and automatic annotation, discovery, selection, composition and execution of Web services. Three main approaches have been developed to bring semantics to Web services: WSDL-S, OWLS and WSMO.

4.2 Operational Specification

This section presents a proposal to support the modeling of the services that should be included in the *EMLs' Environments*. The *Operational Specification* is only concerned with the description of the features of the desired services. The perspectives presented in the next three sections are concerned with their coordination. This perspective proposes the modeling of the following issues:

- *General Service Description*. It should be possible to specify the desired functional and non-functional requirements of services. We do not propose any model to perform this description but support the available possibilities: both semantic and non-semantic. In this way, it is possible to indicate the services required in accordance with a term in a controlled vocabulary, a taxonomy or an ontology. In general, we support the use of semantic Web services proposals. Anyway, a more realistic approach is to use a UDDI approach as it does not require the development of ontologies. This general service description involves the description of the functional capabilities, but also non-functional capabilities (e.g. quality of service, cost, availability) and user-interface aspect.
- *Operations Description*. In addition to the *General Service Description* we consider the featuring of service operations. These operations are intended to be used by the LMS that executes EMLs models (in accordance with the interaction perspective). As an example, we can require that a conference service will provide an operation that enables the LMS to invite a participant.
- *Events Description*. Services can generate events that need to be captured by the EML LMS. In the *awareness perspective* these events are intended to be processed in order to capture specific situations. Therefore, it is possible to indicate that an application providing certain events is required. As an example, we may require a simulator service that provides events to indicate the start, pause, stop and finish of a simulation.
- *Permissions Description*. Services' permissions also need to be described in order to demand its provision during runtime. These permissions are assigned to participants in order to constraint the available functionality for users. Many times, different users are assigned different privileges to use the functionalities of services (e.g. an expert learner has access to more functions in a simulator than a novice learner). In the section about the *authorization perspective* we describe the management of permissions in ESs.

Operations, Events and Permissions descriptions may be provided both in a syntactic or semantic way. The main problem of the syntactic description is that

different services may offer the same syntax but with different behaviors. By the way, the provision of semantic annotations is more complicated.

5 Interaction Perspective

The *operational perspective* indicates the services that should be included in an environment. Anyway, it does not provide any support to coordinate them. The *interaction perspective* is concerned with the invocation of operations in services. We consider *interaction specifications* to control and manage service behavior. We are considering the interaction perspective to support the modeling of policies in collaborative applications: session management, membership management, floor control, conversation control.

5.1 Interaction Specifications

The modeling of the interaction perspective is approached through *Interaction Specifications* (ISs). An IS indicates which operations have to be invoked, how they have to be arranged and the entity they are offered to. The modeling of the ISs involve three components: interaction sources, interaction processing and interaction sinks. These components are described in the next sections.

Interaction Sources. They are the basic operations that may be invoked in the UoLs. These operations are named as *primitive operations*. The *interaction source* description deals with the featuring of the operation and its parameters. As an example, an operation provided by a conference service to add participants to sessions: *'add-participant (session-id, participant-id, initial-status)*. The services *primitive operations* are featured in the *operational perspective*. In addition, the EML LMS also provides operations that may be invoked during UoL execution. For example, there exists a set of operations that enable to perform modifications in a running instance of an ES schema: *create-new-sub-ES, remove-sub-ES, create-new-role, assign-participant-to-role*, etc. These operations are included to support the dynamic modification of the UoLs during run-time.

Primitive operations can be of two types: a synchronous request/response or an asynchronous one-way operation. A one-way operation requires only the input parameters of the operation because it does not expect any response. A synchronous request/response operation requires both input parameters and output parameters. As an example: the operation *send-mail* in an e-mail service is an asynchronous operation. An operation *get-participant-contributions* on a bulletin board service includes input and output parameters.

Interaction Processing. The *interaction processing* is about how *primitive operations* are performed. This processing involves three issues: (i) the specification of the particular element in which the operation have to be produced (e.g. a particular instance of a tool, all the instances of a tool, all the tools in a certain environment); (ii) the management of the input and output parameters (it is

about the population of the parameters with actual values. This management is modeled with elements of the *informational perspective* as it deals with data transfer and copy); and (iii) the composition of operations.

The main point in *interaction processing* is operation composition. Many times an operation is invoked in isolation, independently of other operations. Nevertheless, in some situations several operations have to be performed in conjunction, arranged in a certain way. As a result *composite operations* are obtained involving several *primitive operations* and other *composite operations* using appropriate operators. Figure 2 depicts the operation constructs considered for the composition of operations. The proposed constructs are described as follows:

- The *sequence* construct allows to define a collection of operations to be performed sequentially in lexical order, namely, in the order in which they are listed within the *sequence* element. The *sequence* construct is considered finished when the final operation were completed.
- The *parallel* construct provides concurrency and synchronization. A *parallel* completes when all of the operations in the construct have been completed. Completion of an operation in a *parallel* construct includes the possibility that it will be skipped if its enabling condition (see next item) turns out to be false. The *parallel* construct completes when all the enable operations have been completed.
- The *if-expression* construct enables to introduce a control to decide if an operation should be invoked. Therefore, this construct supports conditional behavior. In case the if condition is true the *then* branch is taken and the corresponding operation is performed. In other case, the else branch is taken (if available. If the else branch is not available it is considered as an empty operation). The *if-expression* construct is complete when the operation of the selected branch completes.
- The *while* construct allows to indicate that an operation is to be repeated as long as a certain success criteria is met. It supports repeated performance of a specified operation.

Interaction Sinks. The *interaction sinks* involve the entities to which the operations are offered. In POEML, *operations* are invoked as a result of an event produced during the execution. Tools offer a main part of their functionality to users through their own graphic interfaces. The main purpose of *composite operations* is to be invoked directly by the EML LMS. In this way, they are involved in the modeling of ECA (*Event-Condition-Action*) rules. For example, it is possible to indicate that when an ES is instantiated (namely, an event *ES-new-instance* triggers) to invoke the creation of an instance of a certain service.

5.2 An Interaction Specification Example

As an example of an *interaction specification* we provide the following XML code (for the sake of simplicity we have renewed some points like service instances or event processing which are not relevant). The desired behavior is that when a new

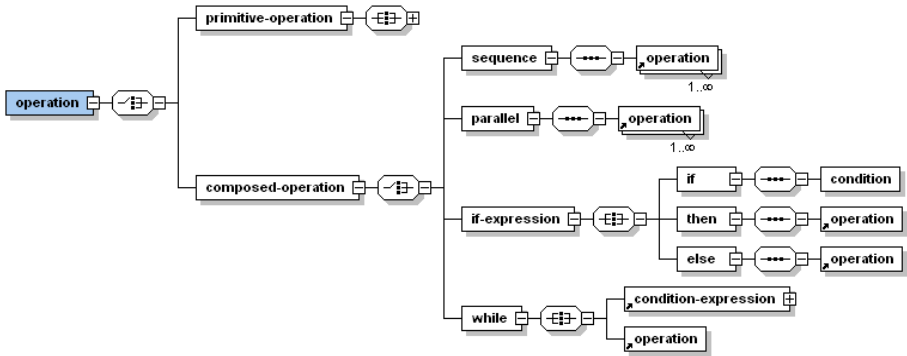


Fig. 2. The set of operation constructs available in PoEML

participant enters in an ES it is added to the chat session. In addition, if the new participant is a teacher and the current moderator in the chat is a learner then the new participant is assigned the moderator role in the chat. The specification represents an ECA rule that is evaluated when a participant enters in an ES (it is represented by the participant-connect-event). The rest of the XML involves a *composite operation* made up by a *if-expression*. The condition evaluates if the new participant is a teacher and the current moderator a learner. If this condition satisfies, the participant is added to the chat and the moderator role is assigned using a *sequence* construct. In other case, the participant is simply added to the chat.

```

<event>
  <filter>
    participant-connect-event
  </filter>
</event>
<operation>
  <if_expression>
    <if>
      <AND-condition>
        participant-enter-ev.participant.role='teacher'
        chat-s.get_moderator().role='learner'
      </AND-condition>
    </if>
    <then>
      <sequence>
        chat-s.add_user(participant-connect-ev.participant)
        chat-s.set_moderator(participant_connect-ev.participant)
      </sequence>
    </then>
  </if_expression>

```

```

    <else>
      chat-s.add_participant(participant-connect-ev.participant)
    </else>
  </if-expression>
</operation>

```

6 Awareness Perspective

The *awareness perspective* is intended to provide timely and highly relevant information about what is happening during an UoL execution. Information in this perspective is delivered as *awareness events*. We propose to model the way in which events produced by services should be processed and delivered to appropriate users and services. An important property of the *awareness perspective* is to provide the right amount of awareness. If too much information is provided the user may be saturated and the information is useless. If not enough information is provided the user does not acknowledge many situations. Therefore, awareness should be constrained and focused in order to be provided to the right participant the right awareness at each time.

6.1 Awareness Specifications

This perspective is intended to model *Awareness Specifications* (AwSs). An AwS defines patterns of events, describes how information from constituent events is to be digested, and dictates to whom the result is to be delivered.

Awareness Sources. The *Awareness Sources* involve the set of events that can be produced during the execution of a UoL. Typically, these events are called as *primitive events*, since they are the basic events produced in a system. *Primitive events* may be considered for every situation that is of interest.

Similarly to the distinction between types and instances of types in programming languages we distinguish between *event instances* and *event types*. An *event type* describes at an abstract level the essential factors that unambiguously identify the occurrence of an event of that type. Each concrete occurrence of an *event type* is represented within the system by its specific *event instance*, whose main task is to save all relevant information about each the event. The *event type* specifies the parameters that sufficiently describe the specific features of the event. Each *event type* will provide a different set of features. These parameters will take particular values in each *event instance*. As an example, an *event instance* of the *event type* 'simulation-abort' is generated each time a simulation is aborted. This *event type* specifies particular features, such as: the time point at which the abort is produced, the simulation point, the participant involved, etc. Each *event instance* will provide particular values of the features (e.g.: '13:30, 189, maria, ...').

This paper is focused on the events generated by the *services* included in the *environments*. These events are featured in the *operational perspective*. Each

service will provide a particular set of events, each one of them with particular parameters involving different content (e.g., a *simulator tool* generates events like: *simulation-init*, *simulation-pause*, *simulation-resume*).

In addition to the events generated by services, during the execution of a UoL other events may be generated (e.g. events related with the presence of participants in environments). These events are generated by the LMS during the execution of the UoL. The *awareness perspective* is also intended to control these events to provide a comprehensible support.

Awareness Processing. *Awareness Processing* is concerned with the capture and processing of *event instances* in order to detect particular situations. The purpose is to detect occurrences of simple primitive events. In addition, usually we are not interested on a single event but on a combination of multiple events, that are called in the literature as *composite events*. *Composite events* are defined by composing *primitive* or *composite events* with a set of operators. Similarly to *primitive events*, *composite events* are featured by a set of parameters.

A technique is needed to detect occurrences of combinations of multiple events, i.e., to detect *composite events*. This has long been studied in the active database field where *event algebras* have been proposed [10]. More recently, XML-based approaches involved in distributed Web systems have also considered this problem [11]. As a result, the literature describes several techniques that can be used to process events (e.g. Petri nets, automata, tree-based, graph-based). In this work, the approach adopted is based on *Directed Acyclic Graph* (DAG). A *composite event specification* is a rooted DAG where the leaves of the DAG are primitive event producers, the non-leaves are event operator instances, and the edges are connections, i.e., typed event streams. An *event operator* is a self-contained, reusable algorithm for recognizing *event instances* and calculating the parameters of the resulting composite events. During execution of the specification, primitive events will enter the DAG at their associated leaves and flow to the input slots of operators connected to those leaves. As composite events are generated, they flow to their consumers, which are usually slots of other event operator instances.

The *event operators* are intended to process the *event instances* to capture *composite events* [12]. We consider three basic categories of *event operators*:

1. *Filtering* is about selecting events of the same event type that satisfy a certain condition on their content. It allows to select events of interest based on the values of their parameters. In this way, it is possible to select events related with: a particular instance of a service; all the instances of a service; a certain participant; etc. In addition, awareness specifications are situated in particular context. The operators in this category involve typical arithmetic comparison to check the value of event parameters: equal, greater than, minus than, value in a interval. The operators included in this category are:
 - *Selection*. The selection operator enables to select a particular *event instance*. As example: (i) to take the average instance during an interval T; (ii) to take the most recent instance during an interval T.

- *Single comparison.* It generates an output event when the considered parameter value of the input event satisfies a comparison expression. For example, it enables to capture the events produced in a interval T.
 - *Double comparison.* It takes two producers of events as inputs (but with the same *event type*) and generates a *composite event* as output if inputs have occurred and the parameters of both events satisfy a certain comparison.
2. *Aggregation is about to summarize events of the same event type detected in the system.* It allows to count the number of events produced. We consider different kinds of operators to aggregate events:
- *Counter.* It maintains a count of the number of input events seen and it delivers a *composite event instance* with a parameter indicating the value of the counter. This operator is useful when combined with filter operators described previously (e.g. a *counter operator* computes the number of *compilation-error* events produced and a *comparison operator* enables to capture when such number is greater than 10).
 - *Interval Counter.* It maintains a count of the number of input events seen in a certain temporal interval. This operator enables to capture the frequency of generation of events. It can be used to detect if an excessive number events are produced in a certain interval of time (e.g. to detect if there are more than 5 *compilation-errors* by minute. This may indicate a deficiency on the learner programming capabilities).
3. *Event correlation addresses relationships among instances of different event types.* The operators considered to process composite events are:
- *Conjunction.* A complex event $C = \text{conjunction}(A, B)$, based on the conjunction operator and two events A and B, is triggered whenever events A and B occur in any order. It is possible to include a temporal parameter T indicating the maximal length of the interval between the occurrences of A and B. If T = infinite it indicates that there is no restriction.
 - *Disjunction.* A complex event $C = \text{disjunction}(A, B)$, based on the disjunction operator and two events A and B, is triggered whenever event A or B occurs.
 - *Concatenation.* A complex event $C = \text{concatenation}(A, B)$, based on the concatenation operator and two events A and B, is triggered whenever B occurs, provided that A did already occur. Event B begins before event A is finished.
 - *Sequence.* Similar to the previous operator, but in this case event B begins when event A has finished. It is possible to indicate: (i) that is no occurrence of any event between event A and B; (ii) that there is an interval T between event A and B; (iii) that event A and event B occur contiguously.
 - *Concurrency.* A complex event $C = \text{concurrency}(A, B)$, based on the concurrency operator and two events A and B, is triggered whenever events A and B occur in parallel.

- *Negation*. This (unary) operator enables to indicate that a *event instance* of the specified type has not to be produced. This operator is combined with the other ones, enabling to indicate: no event B occurs during A's occurrence; no event B occurs after starting A's occurrence within an interval T; event A is followed by event B and there is no event C in the middle between both events, etc.

Awareness Sinks. The *awareness sink* is about the entity that receives the events processed in the *composite events* specifications. These entities may be human participants, services or ECA (Event-Condition-Action) rules (obviously, the event part of an ECA rule). The sink specification means that when a certain event is produced in a resource the *awareness sink* entity is acknowledged.

The specification of the *awareness sink* can involve a single instance of the described element or a set of instances. As an example, we can indicate that an event has to be notified to a teacher role (namely, to all the participants assigned to the teacher role) or to a certain participant performing such teacher role (e.g. a teacher related with the learners that triggered the event). This kind of assignment is specified using conditions involving the properties of the *awareness sink* and the *composite event*.

The assignment of an *awareness specification* to a human participant can be of two types: synchronous and asynchronous. Synchronous assignment indicates that the participant is assigned just in the moment in which the event is produced. Asynchronous assignment indicates that the event is stored. In this way, the participant can obtain the event anytime, even if she is not connected when the event is produced.

6.2 An Awareness Specification Example

Figure 3 depicts a example of a *composite event* DAG. This *composite event* is considered in the context of a computer programming laboratory where a learner has to produce a computer program. Learners have an editor to type the program source code and a compiler to check the program and to compile it. A typical behavior of learners when they have problems during programming is that they compile the program several times without making any change in the program source code. In this example we are interesting in detecting this situation to inform to a supporting tutor.

In the figure we consider two types of primitive events. The *Program Modification* event triggers when the learner modifies the program source code. The *Compilation Error* event triggers when the learner compiles and a compilation error is produced. This event is provided to an *Aggregation Interval Counter* operator that computes the number of times that such event is produced in a interval T (T is short enough to compute events produced consecutively, because we are interesting in this situation that reflects learner exasperation). Then a *Filter single comparison* operator is included to detect if the previous counter is greater or equal than 3. Also, a *Correlation negation* operator is included to

check if a *Program Modification* event triggers. Finally, a *Correlation Conjunction* operator triggers the desired *composite event* if 3 instances of *Compilation Error* trigger in the period and no *Program Modification* event triggers.

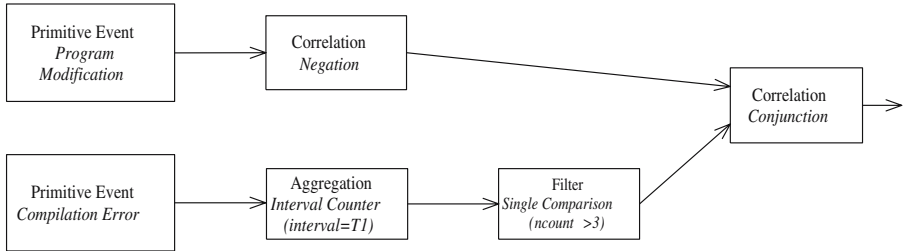


Fig. 3. A composite event DAG representation

7 Authorization Perspective

The *authorization perspective* involves the modeling of authorizations to support the management and assignment of permissions.

7.1 Authorization Specifications

We consider the modeling of the *Authorization Specifications* (AuSs) distinguishing three main components.

Authorization Sources. The *Authorization Sources* involve the set of permissions supported by the elements involved in a UoL. These permissions are called *primitive permissions*, since they are the basic permissions.

In this paper we are focused on the permissions supported by services. These *primitive permissions* are featured in the *operational perspective*. In addition, PoEML considers permissions in other elements:

- *Artifacts.* These are permissions about the management of data fields and files: *read*, *modify*, *destroy*.
- *ES schema.* This is a set of permissions that enable the introduction of modifications in an ES schema during run-time. There are different levels of permissions depending of the desired modification capability: *tutor* (it enables to perform to select alternatives considered during the design-time); *author* (it enables to model new schema elements and to eliminate existing schema elements during the run-time).

Authorization Processing. The authorization processing involves the management of *primitive permissions* in order to facilitate their assignment to

participants. This processing involves the combination of permissions and the introduction of conditions to obtain *composite permissions*.

A *Composite permission* is an abstraction that encapsulates two or more permissions or the same permission in several elements. This abstraction is convenient to express permissions at a high (abstract) level. We distinguish two kinds of *composite permissions*:

- *Composition of different permissions*. It involves the combination of two or more permissions into a single permission abstraction. The permissions may belong to the same element or to different elements. For example, consider the permission to transfer data from one application to another application (e.g. the transfer of the results of a simulator to an analysis tool). Such an action typically involves two permissions. The first permission is for exporting the data from the first application to an appropriate artefact. The second permission is for importing the data from an artefact to the second application. However, it is useful for modeling purposes to think of a more composite abstraction called *transfer-permission* that consists of the individual permissions.
- *Composition of the same permission in different elements*. This involves the combination of the same permission in several elements. As an example, consider an *ES Environment* that includes synchronous and asynchronous communication services: a chat tool, a whiteboard tool, a bulletin board, etc. These applications usually involve a permission (namely, an application role) named as '*moderator*'. We can consider an '*environment-moderator*' permission that includes the '*moderator*' permission in all the services included in the environment. In this way, a higher level abstraction is provided. Furthermore, it is assured that if a participant is assigned the '*moderator*' permission in the chat service it also owns the same permission in the bulletin board. The combination of permission elements can be performed in accordance with: (i) the elements that belong to an environment; (ii) the elements of a certain type (e.g. text files); (iii) the elements of a certain class (it is possible to assign elements to different classes or categories, e.g. restricted and unrestricted resources). It is also possible to combine these selections (e.g. elements of a class and of a certain type).

In addition to aggregate permissions *composite permission* processing can also include conditions. These conditions indicate constraints to determine if the combined permissions should be transferred (provided to the *authorization sink* or not. As an example, the previous *transfer-permission* may depend on a property about the UoL execution mode (e.g. demo, review, master).

Authorization Sinks. The *authorization sink* indicates the entity that receives the permission composed in the *permission processing* part. These entities may be human participants or services (e.g. a service may be authorized to use another service or an artifact).

Similarly to the *awareness sinks* the specification of the *authorization sink* can be referred to a single instance of the described entity or to a set of the available instances. As an example, we can indicate that a write permission is assigned to all the learners or only to the learners that have obtained an A grade in a previous questionnaire.

8 Related Works and Conclusions

EMLs have been proposed to enable the modeling of UoLs in accordance with different pedagogical approaches. This is a very ambitious goal that requires the support of many different issues. Particularly, the support of collaborative learning requires the provision of appropriate learning environments and their coordination in appropriate ways. Therefore, the modeling of learning environments is one of the more critical points. A main part of this modeling deals with the integration and coordination of external systems. In this way, the paper introduces a *service description framework* proposal to support the modeling of the services and their coordination in environments of educational units. It is an original proposal that contributes to the development of EMLs modeling capabilities, but that also introduce interesting solutions to the provision of tailorable groupware.

The development of a *service description framework* is a main concern in the current research of EMLs. The SLeD (*Service Based Learning Design*) project has developed a LMS that integrates IMS LD UoLs with final tools [4]. The results of this project are being considered by the IMS LD developers to propose a CCSI (*CopperCore Service Integration*) module [13]. Particularly these works are focused on the integration of IMS QTI questionnaires and IMS LD UoLs. Other initiatives related with the modeling of services are [14] and LAMS [15]. Nevertheless, to the best of our knowledge, there no exists any proposal for a service description framework. In a different way, several initiatives related with the description of services frameworks for e-learning have been proposed during the last years (e.g. the JISC ELF [16], the IMS Abstract Framework [17]). Anyway, these proposals are more related with the development of software systems than with the execution of EMLs.

This paper proposes a comprehensible *service description framework*. It contains proposals to model the features and services together with their coordination. The solution proposed can be considered as a generalization of the current IMS LD service approach as it considers services' operations, events and permissions. Eventually, we hope this proposal may be considered for the future extension of IMS LD. It will be presented in the future in the appropriate community. In addition, the general PoEML approach, based on separation of concerns, can facilitate the development of EML languages, as it provides a structured approach to the modeling and use of UoLs models. As a result, the computational modeling of educational units can be solved in a simpler and more flexible way.

Acknowledgements

We want to thank Spanish *Ministerio de Educación y Ciencia* for its partial support to this work under grant *MetaLearn: methodologies, architectures and languages for E-learning adaptive services* (TIN2004-08367-C02-01).

References

1. Rawlings, A., van Rosmalen, P., Rodríguez-Artacho, M., Lefrere, P.: Survey of educational modelling languages (EMLs). Technical report, CEN/ISSS Workshop on Learning Technologies (2002)
2. Koper, R., Olivier, B., Anderson, T., eds.: IMS Learning Design Information Model. IMS Global Learning Consortium, Inc. (2003)
3. Dalziel, J.R.: From re-usable e-learning content to re-usable learning designs: Lessons from LAMS. Technical report (2005)
4. Weller, M., Little, A., McAndrew, P., Woods, W.: Learning Design, generic service descriptions and universal ACID. IEEE Educational Technology and Society (2006)
5. Caeiro, M., Anido, L., Llamas, M.: Towards a benchmark for the evaluation of LD expressiveness and suitability. Journal of Interactive Media Education (4) (2005)
6. Curbera, F., Khalaf, R., Mukhi, N., Tai, S., Weerawarana, S.: The next step in Web services. Communication of the ACM **46**(10) (2003) 29–34
7. Cardoso, J.: Approaches to developing semantic Web services. International Journal of Computer Science **1**(1) (2006) 8–21
8. Chinnici, R., ad A. Ryman, J.J.M., Weerawarana, S.: Web Services Description Language (WSDL). Technical report, W3C (2006)
9. Clement, L., Hatley, A., von Riegen, C., Rogers, T.: Uddi version 3.0.2. Technical report, OASIS (2004)
10. Zimmer, D., Unland, R.: The formal foundation of the semantics of complex events in active database management systems. Technical Report Technical Report 22/1977, C-LAB, Paderborn Germany (1997)
11. Bernauer, M., Kappel, G., Kramler, G.: Composite events for xml. In: Proceedings of WWW2004. (2004)
12. Yoneki, E., Bacon, J.: Unified semantics for event correlation over time and space in hybrid network environments. In Meersman, R., Tari, Z., eds.: Proceedings of CoopIS/DOA/ODBASE 2005. Volume LNCS 3769., Berlin Heidelberg, Springer-Verlag (2005) 366–384
13. Vogten, H., Martens, H., Nadolski, R., Tattersall, C., van Rosmalen, R., Koper, R.: Coppercore service integration. Integrating IMS Learning Design and IMS Question and Test Interoperability. In: Proc. ICALT, (Kerkrade, The Netherlands)
14. Vega-Gorgojo, G., Bote-Lorenzo, M.L., Gomez-Sanchez, E., Dimitriadis, Y.A., Asensio-Perez, J.I.: Semantic description of collaboration scripts for service oriented cscl systems. Artificial Intelligence in Education (2005)
15. LAMS: Learning Activity Management System (2005)
16. S. Wilson and K. Blinco and D. Rehak: An e-learning framework. a summary. Technical report, JISC (2004)
17. Smythe, C.: The IMS Abstract Framework: Applications, services, and components. Technical report, IMS Global Consortium (2003)

A Framework Designed for Synchronous Groupware Applications in Heterogeneous Environments

Axel Guicking¹ and Thomas Grasse²

¹ Fraunhofer IPSI, Dolivostrasse 15, 64293 Darmstadt, Germany
axel.guicking@ipsi.fraunhofer.de

² Jeppesen GmbH, Frankfurter Strasse 233, 63263 Neu-Isenburg, Germany
thomas.grasse@jeppesen.com

Abstract. The recent proliferation of using mobile devices in collaborative scenarios increases the need for sophisticated and flexible groupware frameworks for heterogeneous environments. This paper presents the architectural design of Agilo, a groupware framework that has been designed explicitly for synchronous groupware applications involving the use of heterogeneous devices. By respecting device heterogeneity from the ground up, the framework provides an architectural design that is highly flexible along different architectural dimensions on the one hand and simple yet powerful to use on the other hand. Two applications from different application domains based on Agilo are described together with first usage experiences from the developer's point of view.

1 Introduction

During the last decade, the use of mobile devices in daily work scenarios has massively increased. Although mobile devices have found their way into business work settings the application areas still are most often limited to individual services like synchronizing personal calendars, note-taking, and browsing the web. More recently, the research on the integration and use of mobile devices in collaborative settings is constantly growing. It has been pointed out that the use of application frameworks is an adequate way to simplify the design and development of applications in general and groupware in particular [1,2]. Allowing for the increasing trend of mobility in CSCW (Computer-Supported Collaborative Work) scenarios, new groupware application frameworks have been developed or existing groupware frameworks have been extended in order to support mobile devices.

Heterogeneous environments comprising mobile devices exhibit specific characteristics [3,4,5], most notably these are (a) the limitations of processing and battery power, memory and user interface capabilities, (b) unreliable network conditions, and (c) a highly dynamic environment during application runtime including, for example, changing user and device locations).

These characteristics affect all parts typically available in a groupware framework—communication abstractions, framework and application layer, and

user interface support. The belated extension and adaptation of frameworks therefore leads to conceptual as well as implementation-related mismatches between the framework parts addressing classical groupware scenarios and the parts addressing support for mobile devices. Groupware frameworks that have been designed explicitly to support mobile or heterogeneous devices usually focus on specific application domains, such as meeting environments where mobile devices are used as input devices and to share data (e.g. Pebbles [6] and SharedNotes [7]) and collaborative face-to-face learning environments (e.g. ConcertStudeo [8]).

This paper presents the groupware development framework Agilo that was explicitly designed to support heterogeneous devices from the ground up. The consideration of device heterogeneity from the very beginning of the framework design phase has led to a framework architecture that avoids the above mentioned mismatches while providing support for the different characteristics of heterogeneous environments. By providing a high degree of flexibility along several architectural dimensions the framework is suitable for applications in very different application domains.

The focus of this paper lies on the presentation of the architectural design of the framework and how it meets the characteristics of heterogeneous devices. The framework has been used to build two applications from the domains of public safety organizations and meeting support systems. Besides the description of the applications first usage experiences from the developer's point of view are presented as well.

The remainder of this paper is organized as follows: section 2 motivates the need for a highly flexible groupware framework to include heterogeneous devices. In section 3 groupware frameworks supporting heterogeneous devices are presented and analyzed according to their flexibility. In section 4 the architectural design of the Agilo groupware framework is explicated. Section 5 describes two applications built using the Agilo framework. In section 6 experiences from application development and runtime execution are presented. Section 7 concludes the paper with a short summary and several open issues that need to be addressed in subsequent research.

2 Motivation

In order to provide comprehensive support for application development in general, it has been pointed out that application frameworks need to provide flexibility appropriate to the application domain [1]. For the development of groupware applications, several architectural patterns (or "variation points") have been identified [9,10].

These variation points can be divided into static and dynamic variation points. Variation points addressing static characteristics of groupware architectures are the following: (a) The *Distribution Architecture* addresses the distribution in the collaborative application. Prominent examples for different distribution architectures are Client-Server and Peer-To-Peer distribution models. (b) The *Communication Infrastructure* addresses low-level communication issues such as network

and messaging protocols. (c) The *Sharing Model* specifies how data accessible by different users and components is shared and manipulated, for example by exchanging messages or by manipulating replicated objects.

Variation points addressing dynamic characteristics are the following: (d) The *Concurrency Model* addresses the design how multiple concurrent processes and threads execute in the collaborative application and framework. (e) The *Synchronization Model* specifies the coordination of concurrent access of shared data in order to avoid or resolve conflicting changes. Although the latter two variation points usually address issues that are part of the framework, application developers should be able to easily adapt or even exchange the according framework components according to specific application needs.

It is rather obvious that the tight integration of different realizations of variation points simplifies the development of more complex applications [10]. For example, when considering a meeting scenario where the participants are equipped with notebooks in order to provide input for brainstorming and voting sessions: during the meeting, after one of the voting sessions, the facilitator notices that another brainstorming session should be performed next. He updates the meeting agenda accordingly and changes the configuration of several subsequent voting sessions. While the input of participants is usually entered once and never changed again, the agenda and session configurations need to be synchronously updated in an atomic way at each device. For participant submissions a message-based approach is convenient since they are atomic by themselves and no concurrency conflicts can arise. However, atomic manipulations of multiple data instances necessitate the use of transactions, and, depending on the frequency of concurrent (and maybe conflicting) changes of the data objects, specific concurrency control mechanisms might be necessary as well.

Coming from this example it is only a small extension of the application scenario to include heterogeneous devices which puts other constraints and requirements on the application and, as stated above, on the underlying application framework as well.

3 Related Work

There exists a wide variety of frameworks that provide comprehensive support for the development of groupware applications. However, the support of heterogeneous devices often has been added belatedly to existing frameworks that originally have been designed to support the application development for desktop and PC-based groupware applications, e.g. Pocket DreamTeam [11] or Manifold [12]. During the last few years frameworks have been proposed to support the development of groupware applications using either mobile devices exclusively or using heterogeneous devices. According to the focus of the paper, we focus on the discussion of groupware frameworks for heterogeneous environments.

The DOORS system has been designed for asynchronous collaboration in heterogeneous environments which has been extended to support synchronous collaboration as well [13]. Its object framework provides replicated data objects in

order to allow working on shared data while disconnected (so-called coobjects). DOORS offers flexibility according to the Distribution Model by providing replicated servers and according to the Concurrency and Synchronization Models by encapsulating the according framework functionality and providing different implementations. In order to provide different Sharing Model implementations, the coobject notion needs to be extended. However, these implementations are based on replicated objects which complicates more low-level implementations, such as plain message-passing.

In [12], Marsic presents the Manifold framework, an extension of the DISCIPLINE framework [14] to support heterogeneous devices. It uses a data-centric approach for sharing: while data is shared among all collaborators using XML (Extensible Markup Language) it is presented and adapted according to device-specific capabilities using XSL (Extensible Stylesheet Language). Manifold uses a multi-tier architecture by separating concerns in a presentation layer, domain logic, and collaboration functionality. While DISCIPLINE already provides support for heterogeneity on the networking level, Manifold makes use of Java Beans¹ in order to provide support for heterogeneous devices on the application and interaction level. Although Manifold provides an extensible architecture and flexibility according to the communication infrastructure, flexibility related to the other variation points is limited.

Pocket DreamTeam [11] is an extension of the Java-based DreamTeam platform [15] in order to support mobile collaboration. DreamTeam is a Peer-to-Peer based platform for synchronous collaborative applications that makes use of so-called “resources” which form the basis for collaborative applications, e.g. shared texts or shared web pages. Each resource can communicate with their corresponding peer resources using synchronous remote method calls. Pocket DreamTeam handles the restrictions of wireless connections by using remote proxies that mediate state changes between peer resources. These proxies are located on stationary parts of the network and therefore provide reliable network connectivity to other peers that may act as proxies themselves. Another extension of DreamTeam addresses flexibility concerning the Sharing Model, called DreamObjects [16]. However, DreamObjects does not support devices with limited capabilities. Furthermore, flexibility according to the Distribution Model as well as most other variation points is limited. In addition, as described in [11], a DreamTeam application has to be ported to C++ for use in Pocket DreamTeam.

QuickStep is a toolkit designed to support data-centered collaborative applications for handheld devices [17] based on record-based shared data (like to-do lists, calendars etc.). In order to avoid conflicts, only the creator of a record is allowed to modify it which in turn allows fast synchronization of replicated objects. However, it does not provide typical groupware services like session management (all users connected to a server implicitly join a session). Furthermore, QuickStep primarily addresses collaboration of co-located users, e.g. in meeting scenarios to synchronize personal calendars. In fact, according to the variation points described above, QuickStep only provides flexibility according

¹ <http://java.sun.com/products/javabeans/>

to the Communication Infrastructure by supporting different communication protocols.

The BEACH environment has been designed to support synchronous collaboration using heterogeneous devices [18]. As an example application for asynchronous brainstormings using limited devices (in this case Palm Pilot V), PalmBeach has been implemented [19]. However, PalmBeach is a separate application that has been implemented from scratch using a proprietary messaging protocol in order to allow for communication with more capable devices running the BEACH platform. The BEACH platform itself has never been designed for application development involving mobile devices with limited capabilities.

4 Framework Design

The Agilo framework combines approaches of so-called “white-box” and “black-box” application frameworks [1]. White-box frameworks support extensibility by providing base classes to be inherited and pre-defined hook methods to be overridden by application developers. Black-box frameworks provide interfaces to plug-in components into the framework by using object composition and delegation. While white-box frameworks usually require application developers to have intimate knowledge of the internal structure of the framework, they provide better support for the developer in order to adapt internal framework functionality than black-box frameworks. Black-box frameworks, on the other hand, are generally easier to use and extend but hide most of the framework functionality. Agilo provides both, template methods and framework base classes to be extended on the one hand as well as interfaces in order to plug-in application components on the other hand. This approach leads to a major benefit over frameworks that strictly follow one of the two approaches: The black-box parts of the framework are fully sufficient to build less complex applications that can be easily accomplished by less-experienced developers. However, the white-box parts allow the fine-tuning of framework-internal structures and behavior by more experienced developers in order to meet application-specific needs and requirements which have not been foreseen during the framework design phase.

The Agilo framework architecture is composed of three tiers: The bottom tier consists of a network abstraction interface and protocol-specific implementations, the middle tier consists of the mandatory framework core and the upper tier consists of application as well as optional framework components (see figure 1).

The upper tier is the framework part application developers usually are faced with. By providing most of the framework functionality as optional components in this tier Agilo becomes much more flexible and customizable compared to frameworks, where all or most of the framework functionality is contained in the middle tier. The upper tier follows a component-based approach which leads to several advantages: (a) components can be easily reused, (b) components can be configured and adapted independently which simplifies testing and increases flexibility, (c) the API (Application Programming Interface) of the core framework is small and compact and therefore easy to learn and memorize which is

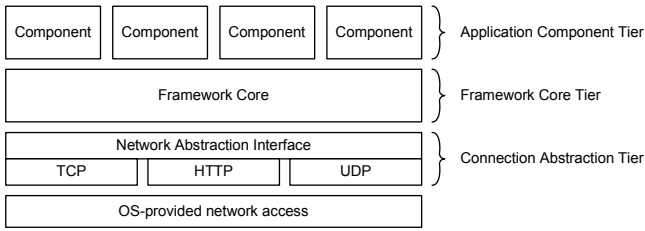


Fig. 1. The Three Tiers of the Agilo Framework Architecture

important especially for less experienced developers, (d) since unnecessary components don't have to be deployed and instantiated, application deployment can be tailored according to specific device capabilities more easily.

The remainder of this chapter details the architectural design of the Agilo framework according to the variation points depicted in section 2.

4.1 Conceptual Model

The Agilo framework is based on three main concepts: *modules*, *messages*, and *connections*. Modules are software components either on framework or application level that are responsible for processing incoming messages (they constitute the upper tier, see figure 1). Messages are application-specific data chunks that are sent between clients and server.² The delivery of messages between clients and server is performed by connections that hide low-level implementation details of network protocols (connections are the upper edge of the bottom tier). While the concepts of messages and connections can be directly mapped to the variation points Sharing Model and Communication Infrastructure, respectively, the concept of modules is cross-cutting to the different variation points. The following paragraphs explain the three concepts in more detail.

According to the Sharing Model, Agilo provides different data sharing realizations. As basic communication abstraction Agilo provides messages that are application-specific data chunks sent asynchronously between clients and server. Synchronous messages are realized on top of asynchronous messages that can be used by clients to send a request to the server and block until a response from the server arrives. Although not often required, messages can have an explicit priority in order to be able to process more important messages earlier than other messages. On top of both types of messages and provided as optional components in the upper tier, transactions for atomic sending and processing of multiple messages as well as a generic transaction-based object replication mechanism can be used in application scenarios with a high number of concurrency conflicts and frequent data access and manipulations. The different realizations can be used tightly integrated in a single application.

² For the sake of clarity we use the terms Client and Server since a Peer-To-Peer distribution model can be realized on top of the Client-Server distribution model, where each peer acts as both, client and server, at the same time [10].

According to the Communication Infrastructure, Agilo provides a high-level abstraction of network connections (the bottom tier in figure 1) that allows the implementation of applications independent of underlying network and transport characteristics. Typical connection implementations are TCP (Transmission Control Protocol) sockets and—to support nodes secured by a firewall—HTTP (Hypertext Transfer Protocol) connections, where clients constantly poll the server to send and receive accumulated messages. Each connection uses a marshaller that converts messages into a byte sequences and vice versa. By exchanging the marshaller of a connection the messaging protocol can be easily adapted in order to support the integration of third-party clients and devices into Agilo applications or to meet specific security requirements.

An Agilo application usually consists of several modules, each running either on client- or server-side. Modules listen to incoming messages and process them by performing some kind of activity. Which messages a module is interested in is specified by a message filter of the module. The message filter arbitrarily defines a boolean expression to accept or reject incoming messages. This way, a single module can listen to different kinds of messages and different modules can get notified about the same incoming message. Modules are registered at a local *ModuleRegistry* that allows retrieving local module instances using node-wide unique lookup names in order to access application logic of other local modules by direct method calls.

Figure 2 shows the static relationships of the conceptual core of Agilo.

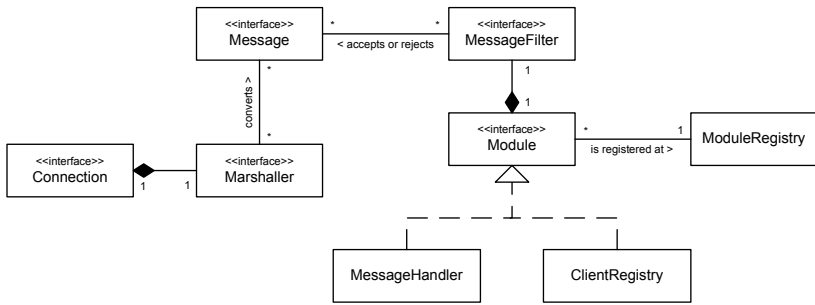


Fig. 2. The Classes and Interfaces Constituting the Agilo Conceptual Model

4.2 Execution Model

During application runtime, several aspects of the Agilo framework core functionality are required in order to provide message delivery and processing. Besides the two variation points addressing dynamic concerns, the Concurrency and Synchronization Model, the typical message flow is of central importance of the Agilo architecture. The following paragraphs describe the two core modules that are related to the dynamic variation points, the Concurrency Model and the

Sharing Model, and that are necessary for the execution of Agilo-based groupware applications before the general message flow in the system is presented.

The server-side *ClientRegistry* module manages the dynamic grouping of clients. By storing a mapping from arbitrary identifiers to a collection of client identifiers, multiple clients can be easily addressed at once to deliver messages. Using the general-purpose *ClientRegistry*, application-specific session and client management can easily be realized. Since the *ClientRegistry* is an ordinary module that can be accessed via the *ModuleRegistry*, the framework does not limit whether the client mappings are entered by clients or directly by the server.

In order to improve network performance the default implementation of the *ClientRegistry* can be exchanged to support message delivery to multiple clients on network level using, e.g., IP multicast. In combination with a toolkit for reliable multicast such as JGroups³, the typical drawback of multicast—potential packet loss—can be avoided. However, for scenarios involving widely distributed clients the benefit and performance gain of network-level multicast decreases.

The second core module that needs to be present for processing messages is the *MessageHandler* that supports different realizations of the Concurrency Model. By default it enqueues incoming messages according to their priorities; messages with the same priority are enqueued in FIFO (First In First Out) order. A single active object, the *MessageRouter*, dequeues messages and forwards them sequentially to the modules that are listening for this message (according to their *MessageFilter*). In case other message delivery orders are sufficient⁴, these can be realized by either configuring the *MessageHandler* module or, for proprietary concurrent message processing strategies, by replacing it with a proprietary implementation.

Regarding the Synchronization Model, the default implementation of the *MessageHandler* avoids any conflicts because all messages are processed sequentially, which can be seen as an implicit transaction handling. The use of transactions to bundle multiple messages and process them atomically does not necessitate conflict resolution strategies as well: the messages that are part of the transaction are enqueued one after the other similar to enqueueing single messages—the only difference is that the framework guarantees that no other messages not belonging to the transaction are enqueued in between. Analogously, the execution of transactions that manipulate replicated objects does not require synchronization if manipulations of replicated objects can occur independent of the current object state. Hence, as long as data manipulations cannot fail, no synchronization mechanisms are necessary.

Nevertheless, if manipulations of shared data can fail, synchronization strategies such as locking or automatic conflict resolution are inevitable. To provide support for applications that require this kind of data manipulation behavior, a module providing transaction management based on the Java Transaction API⁵ is currently under development.

³ <http://www.jgroups.org/>

⁴ For a survey and comparison of different message orders, see, for example, [20].

⁵ <http://java.sun.com/products/jta/>

Figure 3 shows how messages are processed by Agilo. In order to provide a “complete picture,” the figure shows a situation where a message is delivered as reaction on an incoming message.

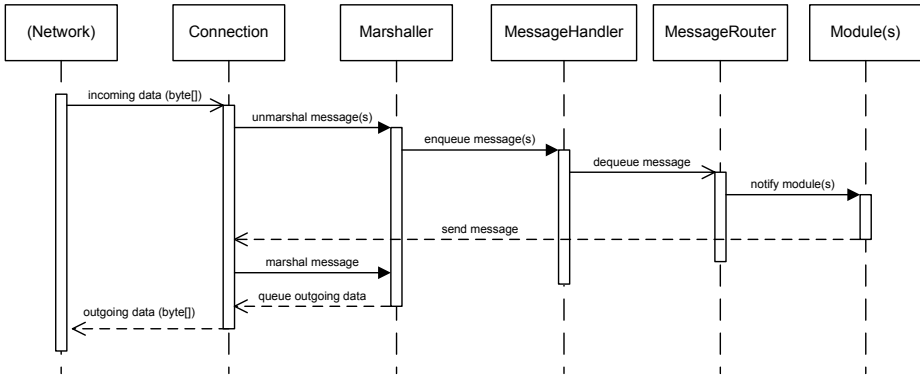


Fig. 3. Message Delivery and Processing in the Agilo Framework

4.3 Immanent Support for Heterogeneous Devices

The Agilo framework addresses the heterogeneity of devices by the following features:

1. The framework provides appropriate device-independent abstractions to free application developers as much as possible from device-specific implementation details.
2. The framework is highly tailorable according to device characteristics and usage purposes.
3. The network abstraction interface provides configurable network protocol-independent reliability support.

The first feature is realized by using Java as framework programming language. For many of the more capable devices on the market today Java Virtual Machines (JVM) based on the Java 2 Micro Edition (J2ME) are available. Devices for that no JVM is available or that do not provide enough resources to execute J2ME-based applications, client applications based on other programming languages can be integrated into Agilo applications by using customized messaging protocols.

The second feature is enabled by the modularity of the framework design. The capabilities of different devices used in a single application scenario often differ. Therefore, the devices are used for specific purposes that best match their individual characteristics. By providing the modularity as integral part of the architectural design of the framework, this massively simplifies application development involving devices with different capabilities:

- Best-matching modules can be chosen by application developers as needed for the purpose of specific devices while modules that are not used by a specific device do not need to be deployed to it. This is a necessary prerequisite for using devices with very limited processing power and memory.
- The runtime performance increases because unnecessary code execution overheads are avoided.
- Since the API of the framework is inherently segmented into the core API and separate module APIs, especially less experienced application developers benefit by not getting overwhelmed by a huge and complex API.

The third feature is part of the network abstraction tier. According to the fact that wireless network connections can be highly unreliable Agilo has to provide a reliable messaging service. In order to provide reliable network connections, the different connection implementations need to be equipped with a guarantee for (a) lossless message transmission, (b) correct message reception, and (c) correct message arrival order without duplicates. These requirements are implemented on framework level instead of completely relying on network protocol characteristics which on the one hand simplifies the connection implementation using other network protocols and increasing code reusability and, on the other hand, enables reliable message exchange independent of the underlying network protocols and marshalling of messages. However, different protocols per se already ensure some of the required reliability issues. For example, TCP provides correct message reception and arrival order, while UDP (User Datagram Protocol) does not provide lossless transmission and correct arrival order.

In order to avoid unnecessary overhead on the framework connection layer, the different connection implementations only make use of the reliability features if necessary. For that, an “optimized ACK” protocol is provided by the framework where lost or corrupt messages are explicitly requested by the receiving from the sending node. In order to be able to use third-party messaging platforms that provide reliable messaging on their own (for example, JGroups or JMS⁶), the reliability features of the framework can be easily switched off.

5 Applications

Based on Agilo, two applications for heterogeneous environments have been implemented: First, an application supporting communication and coordination in emergency missions of public safety organizations, called OPUS. Second, a commercial application for sophisticated large-scale meeting support, called Digital Moderation⁷, has been extended to support the use of heterogeneous devices.

5.1 OPUS

The communication during emergency missions as they are performed today by public safety organizations is based on analog trunked radio which leads to several

⁶ <http://java.sun.com/products/jms/>

⁷ <http://www.ipsi.fraunhofer.de/digital-moderation>

problems [21]: (a) A partner has to follow the whole communication in order to decide which information is dedicated to him. (b) In order to communicate a partner has to interrupt his current work context. (c) In high noise areas the understanding of the communication partner can become difficult and may lead to delays in case of explicit inquiries. (d) Messages that contain a high amount of information probably have to be written down. (e) No private information can be exchanged between two communication partners. (f) Finally, the access to the communication media is not easy as the number of participants increases.

In order to address these problems caused by the trunked radio technique as communication medium, the synchronous groupware OPUS has been proposed in [21]. The requirements for the software were derived from several typical scenarios in missions of public safety organizations. The functional requirements can be divided into the domains task management and resource management. The task management comprises all activities of generation, assignment, and maintenance of tasks. A task is a problem which a resource has to work on. A resource, in turn, is every unit, single man, or equipment that can perform or can be used to perform the work to solve a task. The resource management comprises all activities to control the relationship among resources.

Besides the functional requirements, two non-functional requirements have been identified as well. First, to adequately support the work context the system has to run on handheld devices. This avoids additional weight to be carried by relief units besides their regular equipment. Depending on the user-specific work context, PDA, SmartPhones as well as cell phones need to be supported. Second, during a mission, the device may be not always connected to the network. Thus, interrupted communication links need to be taken into account.

To meet the denoted requirements, the OPUS software architecture has been designed as described in [21]. By applying the patterns “Replicate For Freedom” and “Mediated Updates” [22], the architecture follows a replicated approach, where applications and shared data objects are replicated to client devices. In case of local modifications the client notifies a central server component that propagates the changes to the affected clients. This architecture ensures that a user can still keep on working while the communication link to the server is temporarily interrupted. To support limited devices, the provision of a central server exempts client devices from maintaining lots of communication links.

Figure 4 shows the overall architecture of the OPUS system. The consistency module is responsible for communicating local data changes to the server and for updating local data replicas in case of data update messages received from the server. The data model holds the domain-specific application data which is accessed and manipulated either by the consistency module or locally by the user via the user interface. The task and resource management modules contain the application logic which connects the application data with the user interface.

Details about the implementation of OPUS can be found in [21]. Table 1 shows the realization of the static variation points in the OPUS system. The transaction-based messages are used in order to allow reassigning resources to another supervisor which has been realized by using the Agilo ClientRegistry.

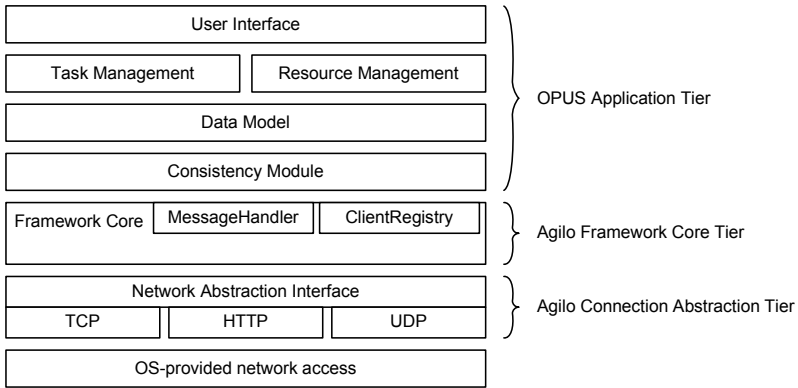


Fig. 4. Top-Level Software Components in the OPUS System

Table 1. Realizations of Static Variation Points in the OPUS Application

| Variation Point | Realization |
|----------------------|---|
| Distribution Model | Client-Server |
| Comm. Infrastructure | Different Network Protocols (TCP, HTTP) |
| Sharing Model | Asynchronous messages, synchronous messages, transaction-based messages |

5.2 Digital Moderation

The Digital Moderation system is a commercial meeting support system for facilitated and co-located meetings providing conceptual as well as technical scalability with respect to the number of meeting participants (meetings with up to several hundreds of users are supported). The main characteristics of the Digital Moderation system are easy adaptability and extensibility to accommodate different facilitation methods, dynamic runtime extensibility to allow changes of the agenda during the meeting, automatic meeting report generation as well as sophisticated facilitation services in order to increase meeting performance. Facilitation methods are realized by providing a meeting tool API, e.g. to provide brainstorming, ranking and voting tools.

Digital Moderation supports different meeting scenarios; besides large-scale meetings, workshop scenarios with about 20 participants are supported that usually are performed by an external facilitator equipped with several WiFi-capable notebooks at a company site (see figure 5). Both scenarios essentially require a very high system reliability. The system itself needs to be robust against hardware and network failures either because no sophisticated failure-tolerating hardware is available or because of financial reasons in case of a large number of meeting participants.

Digital Moderation is implemented using Agilo with a Client-Server Distribution Model which provides better technical scalability for large-scale meetings

compared to a Peer-to-Peer Distribution Model and which simplifies automatic meeting report generation. Network failures are already avoided by the network abstraction tier of the framework while hardware failures, especially in case of server failures, are currently addressed by a generic recovery module based on message logging.

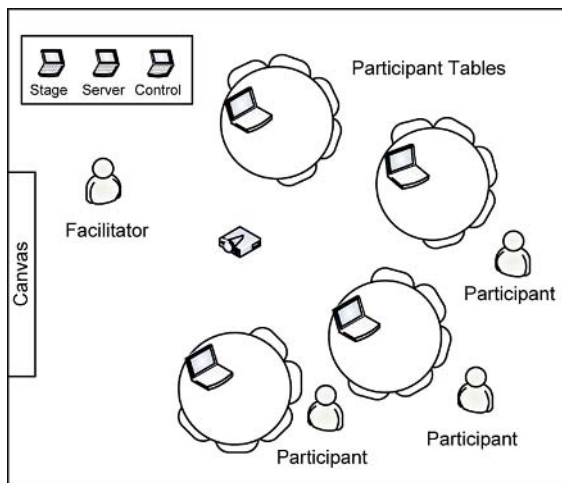


Fig. 5. A Typical Digital Moderation Workshop Setup

Recently, the Digital Moderation meeting scenarios have been extended in order to provide support for using heterogeneous (especially mobile) devices during meetings; the implementation of these extensions are currently ongoing. One of these extensions addresses the technical support during a large-scale meeting: technicians are equipped with PDA during meetings in order to get continuously informed about device characteristics and device-specific connectivity details. Additionally, participants can ask for technical support via the user interface which automatically shows up on the user interface of the technicians' PDA. This way, technical issues can be handled more efficiently and user satisfaction can be improved because of the smooth integration of the technicians and their responsibilities into the meeting execution.

Another extension addresses the use of different devices in a meeting according to specific meeting task characteristics. Depending on the task, devices with different interface capabilities are more appropriate than others. For example, for a meeting of a design team, the use of devices with pen-based input is more applicable e.g. to provide scribbles during a brainstorming session, while other participants use keyboard-based input devices to submit new ideas.

Table 2 shows the static variation points in the Digital Moderation application. While asynchronous messages are used for participant contributions, synchronous messages are mainly used during client startup in order to retrieve

Table 2. Realizations of Static Variation Points in the Digital Moderation System

| Variation Point | Realization |
|----------------------|---|
| Distribution Model | Client-Server |
| Comm. Infrastructure | Different Network Protocols (TCP, HTTP) |
| Sharing Model | Asynchronous messages, synchronous messages, replicated objects |

meeting and tool configuration data. The transaction-based manipulation of replicated objects is used by the facilitator to update meeting data, such as tool configurations and the meeting agenda. Since no concurrent data manipulations can happen (only the facilitator is allowed to update the meeting data), no sophisticated concurrency control and synchronization mechanisms are needed.

6 Experiences Gained

Up to now, the Agilo framework has been used by ten application developers whose expertise ranges from less-experienced Java developers to expert Java developers with comprehensive experience in developing distributed systems. The Digital Moderation application has been used successfully to perform meetings with up to 200 participants involving more than 50 devices.

The Agilo framework core consists of 112 classes for clients and 132 classes for the server which result in framework binaries of around 200 KB and 230 KB, respectively, in size. Optional modules, e.g. the transaction-based generic object-replication without sophisticated synchronization and concurrency-handling, consists of 54 classes resulting in around 50 KB binaries on clients and server.

Compared to Agilo, other groupware frameworks, e.g. COAST [2] and DyCE [23], provide more functionality as part of the non-dividable framework core which leads to two immanent limitations. On the one hand, larger frameworks put more constraints on how to use and extend the framework which massively increases the learning time of application developers. On the other hand, these frameworks require more system resources which limits the applicability for heterogeneous environments.

The experiences gained during the development of the applications sketched in the previous section confirm these conclusions. The small framework core and the modular architecture of the framework lead to a very quick understanding of how to implement applications using Agilo (typically far less than a single day)—even for less-experienced developers.

To our experiences, the modularity and flexibility of Agilo substantially supports the evolution of applications. New functionality can be implemented by introducing new kinds of messages and developing independent modules. This way, new functionality can be easily added without affecting stability and correctness of existing application logic. The flexibility to combine different realizations of variation points in a single application massively simplifies the development of applications combining different complexity levels without overly increasing the overall system complexity.

7 Conclusions and Future Work

This paper presented the architectural design of the groupware framework Agilo that has been explicitly designed to support the development of groupware applications in heterogeneous environments. By providing a highly modular architecture where most of the groupware functionality itself is provided as separate modules, the framework offers a high degree of flexibility, which imposes only few usage constraints on the application development. In fact, applications can be built faster due to increased reusability of software components and faster understanding of the provided framework core concepts. The inherent extensibility of the framework provides support for device-specific development and tailoring for a wide range of devices—from desktop PCs and full-featured notebooks to handheld PDA and devices providing only a very limited set of capabilities like SmartPhones. The experiences gained during the design and development of two different systems (one of them a large commercial meeting support system) based on Agilo have shown that even unexperienced Java developers can comprehend the conceptual framework design very quickly and implement applications within the first one or two days after starting to work with Agilo.

Although the framework has been used to implement two different systems, there are three main areas that require further research. One area addresses the implementation of other applications that especially differ with respect to the dynamic variation points. Since both applications presented in this paper do not put strong requirements on synchronization and concurrency control mechanisms (because of their application domains), the experiences regarding the Synchronization as well as the Concurrency Model are still in an early stage.

Another area is to improve runtime support for heterogeneous devices. In the literature of Ubiquitous Computing it has been emphasized that heterogeneous environments are highly dynamic [3]. While some of the dynamic characteristics are already supported in Agilo (e.g. fault tolerance against intermittent network failures), others are not yet supported in an application-independent way, for example, changing user and device locations and moving stateful applications to other devices. In order to support this kind of context awareness, the approach most often suggested is automatic self-adaptation of the system or framework (e.g. in [3]). For convenient support of application developers the Agilo framework should provide according services, especially to support device and application mobility.

Finally, Agilo needs to be evaluated quantitatively regarding runtime performance, stability and scalability. Albeit there have been several performance and load tests conducted for the Digital Moderation system, concrete statements about throughput, failure frequency and scalability of the framework need to be determined in a more systematic and reproducible way.

References

1. Fayad, M., Schmidt, D.C.: Object-oriented application frameworks. *Communications of the ACM* **40**(10) (1997) 32–38
2. Schuckmann, C., Kirchner, L., Schmitter, J., Haake, J.M.: Designing object-oriented synchronous groupware with COAST. In: *Proc. CSCW '96*, ACM Press (1996) 30–38

3. Raatikainen, K.E., Christensen, H.B., Nakajima, T.: Application requirements for middleware for mobile and pervasive systems. *ACM SIGMOBILE Mobile Computing and Communications Review* **6**(4) (2002) 16–24
4. Roth, J.: Seven challenges for developers of mobile groupware. In: Workshop “Mobile Ad Hoc Collaboration” of CHI '02. (2002)
5. Weiser, M.: The computer for the 21st century. *Scientific American* (September) (1991) 94–104
6. Myers, B.A.: Using handhelds and PCs together. *Communications of the ACM* **44**(11) (2001) 34–41
7. Greenberg, S., Boyle, M., LaBerge, J.: PDAs and shared public devices: Making personal information public, and public information personal. *Personal Technologies* **3**(1) (1999) 54–64
8. Wessner, M., Dawabi, P., Fernandez, A.: Supporting face-to-face learning with handheld devices. In: *Proc. CSCW '03*, Kluwer Academic Publishers (2003) 487–491
9. Avgeriou, P., Tandler, P.: Architectural patterns for collaborative applications. *International Journal of Computer Applications in Technology (IJCAT)* **25**(2–3) (2006) 86–101
10. Guicking, A., Tandler, P., Avgeriou, P.: Agilo: A highly flexible groupware framework. In: *Proc. CRIWG '05*, Springer Verlag (2005) 49–56
11. Roth, J.: The resource framework for mobile applications: Enabling collaboration between mobile users. In: *Proc. ICEIS '03*. Volume 4. (2003) 87–94
12. Marsic, I.: An architecture for heterogeneous groupware applications. In: *Proc. ICSE '01*, IEEE (2001) 475–484
13. Preguia, N., Martins, J.L., Domingos, H.J.L., Duarte, S.: Integrating synchronous and asynchronous interactions in groupware applications. In: *Proc. CRIWG '05*, Springer Verlag (2005) 89–104
14. Marsic, I.: DISCIPLINE: a framework for multimodal collaboration in heterogeneous environments. *ACM Computing Surveys* **31**(2es) (1999) Article No. 4
15. Roth, J.: DreamTeam – A platform for synchronous collaborative applications. *AI & Society* **14**(1) (2000) 98–119
16. Lukosch, S.: Transparent and Flexible Data Sharing for Synchronous Groupware. PhD thesis, University of Hagen, Germany (2003)
17. Roth, J., Unger, C.: Using handheld devices in synchronous collaborative scenarios. *Personal and Ubiquitous Computing* **5**(4) (2001) 243–252
18. Tandler, P.: Synchronous Collaboration in Ubiquitous Computing Environments. PhD thesis, Darmstadt University of Technology, Germany (2004)
19. Prante, T., Magerkurth, C., Streitz, N.: Developing CSCW tools for idea finding: empirical results and implications for design. In: *Proc. CSCW '02*, ACM Press (2002) 106–115
20. ter Hofte, H.: Working Apart Together. Foundations for Component Groupware. PhD thesis, Telematica Instituut, Enschede, NL (1998)
21. Grasse, T.: Eine Systemarchitektur zur effizienten Steuerung von mobilen Einsatzkräften [in german]. Master’s thesis, FernUniversität Hagen, Germany (2005)
22. Lukosch, S., Schimmer, T.: Patterns for managing shared objects in groupware systems. In: *Proc. EuroPLoP '04*. (2004) 333–378
23. Tietze, D.: A Framework for Developing Component-based Cooperative Applications. PhD thesis, Darmstadt University of Technology, Germany (2001)

Implicit Plasticity Framework: A Client-Side Generic Framework for Collaborative Activities

Montserrat Sendín¹ and César A. Collazos^{1,2}

¹ GRIHO: HCI research group
University of Lleida, 69, Jaume II St., 25001- Lleida, Spain
msendin@eps.udl.es

² Department of Systems, FIET
University of Cauca, Campus Tulcan, Popayán, Colombia
ccollazo@unicauca.edu.co

Abstract. We are interested in integrating and exploiting the *shared-knowledge* from a group by an existing infrastructure of plasticity, as another parameter more to be embedded in the adaptation process. The aim is to offer the benefits from *plasticity* and *awareness* jointly, providing a systematic support in both issues. In this paper we focus on the proactive adaptation to contexts of use under a plasticity viewpoint. The aim is to promote interaction and real time coordination, contributing to real collaboration in multiple and changing groupware scenarios.

1 Introduction

We are no longer tied to our desktop computer due to the wireless technology, the mobile networking capabilities, and a plethora of new computing technologies. New advances provide us freedom to move around and to access to the technology in new and changing environments, keeping in permanent contact when we are working on groups. However, current CSCW approaches focus on the restrictions and affordances that mobile devices and mobility provide, but they do not address the huge heterogeneity¹ and the adaptation to changing contexts of use at the same time. Real time constraints related not only to the shared-knowledge between group members, but also other related to the user (changing needs and preferences), to the environment (daylight, localization, etc.), and even related to network constraints (bandwidth, server availability), which describe the *context of use*², are volatile and require sophisticated capturing and adaptive capabilities that today are still challenging. In a broad perspective, the variability and multiplicity of parameters introduced by all the previous issues are collected under the term *plasticity*. It was coined as the ability from systems

¹ Diversity and versatility in the decision about the device or platform to use, taking into account the significant differences in their physical, graphical and interaction features.

² Our context of use conception encompasses five components: the environment, the user, the platform, the particular shared knowledge and the task at hand.

to mold their own UI to a range of computational devices, conditions and environments in order to tackle the diversity of contexts of use in an economical and ergonomic way [12], offering a great flexibility. We propose to adapt plasticity tools to novel groupware scenarios. Another handicap of CSCW is the lack of evidence in the use of complex social dynamics where the group activity takes place. In this line, groupware designers have included aspects related to *awareness*, which has become a cornerstone in computer systems design in several ways. Awareness reduces the meta-communicative effort needed to collaborate across physical distances in CSCW environments [8] and promotes real collaboration among group members. The shared knowledge should be appropriately captured, represented, integrated and promoted. However, awareness support is not systematic and developers must build it from scratch every time.

Groupware systems must be highly adaptable to new changing conditions involving not only constraints related to mobility as described above, but also to their distributed shared knowledge. In this paper we deal with the dynamic adaptation at runtime. It is necessary to implement some mechanisms to obtain a twofold benefit: (1) reaction in a proactive manner to contextual changes and (2) shared-knowledge awareness, contributing to make collaborative work successful. The infrastructure presented in this paper reuses some existing tools addressed to provide plasticity in order to integrate awareness information and exploit it as an integral part of the plasticity process.

This paper is structured as follows. Section 2 discusses some related work. Keeping in mind that our goal is to integrate awareness mechanisms in a certain infrastructure of plasticity, in section 3 we describe the approach and infrastructure of plasticity we consider the most appropriate. Section 4 presents a description of the software architecture we propose and reports some guidelines of abstraction towards a major flexibility to compose the generic *application framework*³ we pursue. Finally, some conclusions and further work are explained.

2 Related Work

Many authors have proposed different elements necessary for obtaining a real collaboration. We present some related works to support collaborative systems in which proactive aspects and/or awareness mechanisms are treated.

Collaborative environments supporting proactive adaptation. It is worthy pointing up the research on collaborative work based on the Collaborative Filtering approach. In this line, some authors have emphasized their research on adaptive systems (proactive adaptation) based on the group member's interests. In particular, Barra [1] describes an adaptive system for group navigation on the web whose goal is to provide collaborative and adaptive navigation to users groups sharing a "common interest" on the web. However, the most part of these

³ A semi-complete application that can be customized to produce particular applications.

systems is only focused on different static user aspects, providing different kinds of prefixed profiles.

Collaborative environments supporting heterogeneity and proactive adaptation. One of the most relevant and complete work is the one developed by Favela et al. [4]. Their research is applied in the healthcare field. They combine interactive public displays together with handhelds, towards the development of a pervasive hospital environment. The integration of proactive components in an agent-based architecture offers information relevant to the case at hand, apart from context-aware and personalized information to the user. We can point up that their approach is totally dependent on the server, implying that some problems like network failures could have serious consequences in critical environments like hospitals. In that way, our approach intends to reduce this degree of dependence, tending towards client-server architectures that obtain an operational balance between both sides. Its idea is to make client devices autonomous to a great extent by means of low resource-consuming programming techniques that can be supported in small devices, as explained in section 4.1.

There are many works related with frameworks to dynamically support context-awareness. Marsic [7] has developed a data-centric framework for synchronous collaboration of users with heterogeneous computing platforms, allowing clients with different computing capabilities to share different subsets of data based on XML. The interface is customizable according to the context and the user needs. However, the level of shared information is quite limited, and what is more important, it is static. The shared-knowledge awareness [2] is not handled.

Collaborative environments supporting awareness and proactive adaptation. One of the most relevant works that integrate awareness mechanisms is the one proposed by Correa and Marsic [3]. They have developed an extensible and generic architecture to support awareness in heterogeneous collaborative environments under a semantic consistency approach, what is their main innovation. Their architecture not only provides awareness, but also allows the adaptation, although only related to resource constraints. They show that awareness support implies a trade-off between the degree of awareness and the network usage. In fact, our work is in the line of pursuing this balance. Another common point with our work is the development of an application framework. They plan the development of an adaptive version of their architecture to awareness issues.

As we have observed, there is a lack of guidelines about how to integrate aspects as adaptivity, context-awareness and shared-knowledge in the same tool. Our work intends to support the development of computing collaborative environments that integrate all these aspects.

3 Initial Approach and Infrastructure of Plasticity

The infrastructure chosen for our proposal is based in our dichotomic view of plasticity [9], which divides the plasticity problem in two different challenges

with two clearly delimited goals that make up an extension to the Thevenin and Coutaz concept of plasticity [12]. They are called *explicit plasticity* and *implicit plasticity* [9]. We match these two issues respectively with the stages of design (construction, sometimes a reconfiguration of an existing UI) and execution (specific readjustments at runtime) that the UI has to withstand over the whole system's lifetime. To be more precise, *explicit plasticity* tackles relevant changes in the UI, caused by unforeseen situations that involve a reconfiguration of the UI (e.g. changes in the computing device). Due to the considerable scope involved, it needs to be solved in a server, where it is brought into operation under an explicit request by the client. This is why we call it "explicit". *Implicit plasticity* is in charge of providing proactive adaptation (also called adaptivity) at runtime, as the user goes through new contexts of use. It tackles specific modifications in the UI, originated by predictable contextual changes (e.g. changes in the daylight level or in the user's location), which can be solved by an automatic readjustment on the client side, without any express action or request. This is why we call it "implicit". Clearly both challenges require different modelling, strategies and tools; hence they need to be studied and developed in different frameworks. This division in two goals is what we call a "dichotomy".

Under this twofold perspective, the infrastructure of plasticity consists of combining two different engines framed in a client-server architecture. These engines are called *Explicit Plasticity Engine* (EPE henceforth) and *Implicit Plasticity Engine* (IPE henceforth), respectively. The EPE consists of an automatic tool of production of plastic UIs, as in a design stage. The IPE consists of a runtime adaptive engine with the capacity to detect the context and react in order to adapt the UI to the contextual changes on the fly, providing thus the proactive adaptation pursued on the client side. In this line, our interest is focused on developing a generic framework easily customizable to IPEs for particular systems. This is what we call *Implicit Plasticity Framework* (IPF henceforth).

This architectural framework allows delimiting clearly both goals, to be solved in both engines, which are managed in an alternative, iterative and complementary manner. The goal is to give feedback to the plasticity process without discontinuities, keeping both sides in continuous updating. Under this approach, the client only resorts to the server when he/she needs a reconfiguration of the UI -unsolvable locally by the IPE-, propagating to the server the contextual changes that require to be accommodated to the new situation. We can sum up the benefits of our infrastructure in these three ones: (1) an operational balance between both sides; (2) autonomy to perform adaptivity on the client side (that reduces dependence to the server and possible communication failures); and (3) real time reaction to certain contextual changes, contributing to a proactive (implicit) adaptation. Figure 1 shows the overview of the process described, as well as the delimitation between the two sub-concepts of plasticity. The EPE is out of the scope of this paper. A detailed description can be looked up in [10].

In order to integrate the awareness information, the key point is to include the evolving group state and the real time group constraints in the characterization of the context of use. This information is captured and represented in the

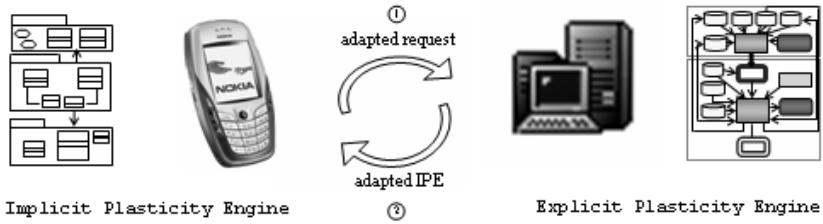


Fig. 1. Overview of the plasticity process

client-side in order to be considered at runtime. However, it is essential sharing it with the rest of the group. This is why he/she propagates it to the server, which will be able to gather and model an overall group knowledge, through the EPE.

4 Software Architecture for the Client-Side

4.1 Design Requirements and General Structure for the IPE

Taking into account that our final goal is to develop a generic framework to easily derive the suitable IPE for a particular system (the IPF), we apply the most orthogonal design strategies to obtain the most challenging design requirements. Particularly, we must guarantee certain properties such as: (1) transparency in adaptation; (2) reusability; and (3) orthogonality. In particular, orthogonality is essential in order to the adaptive mechanisms be handled independently, so that they can evolve individually, avoiding conflicts and promoting reusability. This property is especially important in systems where a lot of dimensions are presented, such as collaborative environments. In order to guarantee these properties we need to apply a separation of concern technology.

We conceive an IPE as a software architecture divided into three layers. The two first are: (1) the *logical layer*, which contains the application core functionality; (2) the *context-aware layer*, which controls and models the real time constraints (the contextual model); in the case of collaborative applications, all the information related to the communication and coordination actions that affect to the whole group state: the *shared-knowledge* [2] from the perspective of the user at hand. This information will be kept updated for further use, in order to provide awareness. Finally, the third layer is an intermediate layer responsible for applying the adaptation over the core system according to the contextual model (*context-aware layer*) following, as mentioned before, some sort of separation of concern technology. The approach chosen for this layer is the Aspect Oriented Programming [6] (AOP henceforth). The reasons that justify this decision are out of the scope of this paper. It can be looked up [11] for a detailed justification, as well as an introduction to AOP. AOP guarantees the three properties mentioned before, obtaining the maximum modularization for the adaptive mechanisms, and what is more important, without affecting the software structure of the underlying system (transparency). AOP offers these goals encapsulating the

treatment of each real time constraint (e.g. the related to groupware) in separated and concentrated program units called *aspects*⁴ (orthogonality). Hence we call the intermediate layer (3) *aspectual layer*.

From a general collaborative viewpoint, the *aspectual layer* has the responsibility of promoting communication events and actions focused on improving the coordination of the group activities. To do so, it is in charge of interfering implicitly the situations along the system execution in which these initiatives could be beneficial for the group, and augment there the core functionality triggering this kind of actions, as well as catching the context in order to construct the *shared-knowledge* from the particular viewpoint of each user. These actions are triggered by means of the AOP mechanisms, trying to improve collaboration, and they make up the extra-functionality that the IPE introduces without affecting the underlying system. Furthermore, due to we use a combination of metadata and *aspects* -whose explanation is out of the scope of this paper-, coupling between the base system and the adaptation mechanisms is removed (reusability). We can say that the *aspectual layer* acts as a transparent link between the other ones. Figure 2 depicts a sketch of an IPE for a generic collaborative system. We include two *aspects* to tackle the two goals mentioned: the *Communication aspect* and the *Coordination aspect*. The role of the *coreAppAnnotator aspect* in figure 2 is also out of the scope of this paper. The Shared-Knowledge-Model component corresponds to a representation of the *shared-knowledge* from the particular user viewpoint.

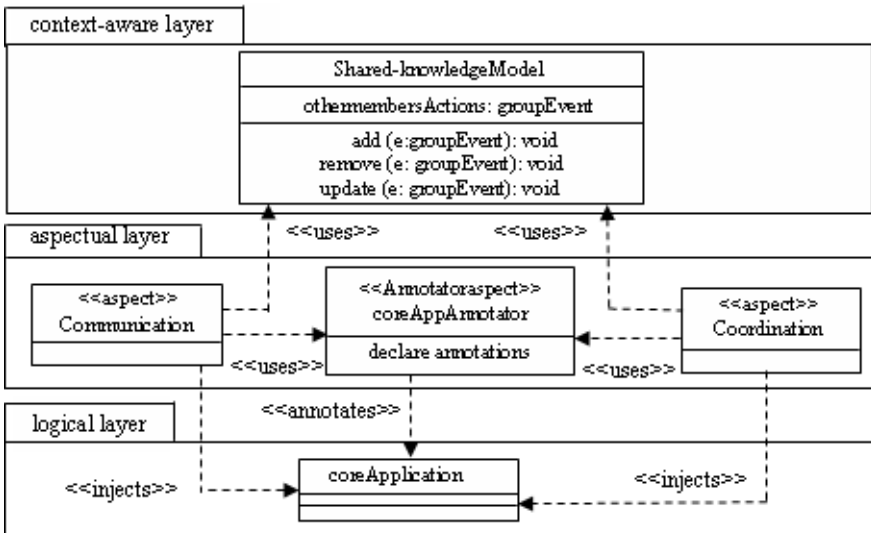


Fig. 2. IPE for a generic collaborative system

⁴ Program units that interfere the core functionality injecting new code or modifying the base code.

4.2 Further Guidelines Towards Abstraction: The IPF

In the design of our IPF, according to the experience extracted from the IPEs built up to now, we take into account some considerations in order to achieve system-independence and reusability. Let us see them according to each issue pursued: different adaptation mechanisms, different contextual needs and different domains of application.

Adaptation mechanisms. In order to avoid system-dependences, and to obtain "universal" adaptation mechanisms, we resort to *aspect* hierarchy and to some refactoring steps. Thus, references to the name of certain methods, classes or particular APIs are encapsulated, redefining the associated elements in *sub-aspects* conveniently specialized. This is also the appropriate strategy if we need to define different ways to interfere the base code behaviour following the AOP mechanisms. Another strategy to achieve abstraction is refactoring in the *aspects* code (the *advices*⁵). In effect, sometimes is not necessary to define the complete advice in *sub-aspects*, but only a method that encapsulates special needs or variabilities. Then, we use advice refactoring. This idea in particular corresponds to the *Template advice idiom* [5]. We can use other types of refactoring or AOP-specific patterns to make good aspectual designs.

Application domains. We are planning to reuse our IPF for different domains deploying libraries of *aspects*. Thus, each particular application would be able to establish the set of concerns it needs to manage. For example, in an archaeological site considering the daylight constraint to adjust the UI is required. However, in an indoors museum guide this concern is useless. In a tele-aid system another kind of concerns are required to assist high-mountain rescues. It would be useful to build a package of *aspects* related to mountain conditions.

Contextual needs. In a similar way, we need to adapt the *context-aware layer* to the *aspectual* one, in order to map *aspects* with data stored in the contextual model. We need to obtain flexibility also in the *context-aware layer*. As long as this layer is based on the object-oriented programming, we use hierarchy of classes to build their components in a generic way.

5 Conclusions and Further Work

Assuming that mobile solutions can offer large-scale solutions in supporting coordinated work, it is recommendable to reuse as far as possible the work already realised to solve problems inherent to mobility in the groupware work. In this line, plasticity tools intend to offer a solution to most of the issues related to groupware. Moreover, groupware activities need to be designed providing awareness mechanisms to share the group understanding among group members. We have presented an infrastructure of plasticity in which to integrate awareness

⁵ The code to be executed when aspects intercept the base code. Equivalent to methods in classes.

mechanisms, as well as how to accommodate the client-side software architecture to support collaborative activities. The aim is to combine plasticity and awareness goals, until now following separated ways, making group issues an integral part of the plasticity process.

The infrastructure of plasticity proposed follows a client-server architecture model not centered in the server as usual is presented in the literature, but adjusted to our dichotomic view of plasticity, which looks for an operational balance client-server. This approach promotes a better collaboration between devices and less constrained mobility conditions, contributing to autonomy and robustness at the client side, as well as to a real time reaction during the task performing. This approach is in the line of obtaining a trade-off between the degree of awareness and the network usage, identified by Correa and Marsic [3]. Moreover, the software architecture for the client-side is based on low resource-consuming program units (*aspects*) that support awareness mechanisms and adaptivity in compact limited appliances, scarcely increasing the size of the final code. As a result, this architecture becomes suitable for the pervasive design. Additionally, the integration of the adaptive mechanisms with the base system becomes a seamless process because of: (1) any refactoring step or modification in the software structure from the initial system is needed; and (2) coupling with the base system is also removed due to we use a combination of metadata and *aspects*, promoting so reusability.

As further work we plan to arrange and deploy a hierarchical library of generic *aspects* that might be included in the groupware design. In a similar way, we are developing the groupware facet in the server side, constructing an EPE to tackle the design stage. Additionally, we want to specialize our work in the construction of collaborative systems for ambience intelligence and pervasive computing scenarios. Finally, we are interested to develop some visualization mechanisms to provide shared knowledge awareness information to the whole group, using the same infrastructure proposed in this paper.

Acknowledgements

Work partially funded by Spanish Ministry of Science and Technology, grants TIN2004-08000-C03, Colciencias (Colombia) Project No. 4128-14-18008 and Project No. 030-2005, and finally to ParqueSoft-Colombia (DaVinci's Laboratory).

References

1. M. Barra. Distributed systems for group adaptivity on the web. *Adaptive Hypermedia and Adaptive Web-Based Systems (AH 2000)*, Italy, 2000.
2. C. Collazos, L. Guerrero, J. Pino, and S. Ochoa. Introducing shared-knowledge-awareness. *IASTED Conf.: Inform. and Knowledge Sharing*, pages 13–18, 2002.
3. C. Correa and I. Marsic. A flexible architecture to support awareness in heterogeneous collaborative environments. *CTS'03*, pages 69–77, 2003.

4. J. Favela, M. Rodríguez, A. Preciado, and V. González. Integrating context-aware public displays into a mobile hospital information system. *IEEE transactions on information technology in Biomedicine*, 8(3):279–286, 2004.
5. S. Hanenberg and A. Schmidmeier. Idioms for building software frameworks in aspectj. *2nd ACP4IS*, 2003.
6. G. Kiczales, J. Lamping, A. Mendhekar, C. Maeda, C. Lopes, J.M. Loingtier, and J. J. Irwin. Aspect-oriented programming. *11th ECOOP'97*, 1241:220–242, 1997.
7. I. Marsic. A software framework for collaborative applications. *Collaborative Technology Workshop*, pages 10–11, 1999.
8. K.A. Palfreyman and T. Rodden. A protocol for user awareness on the world wide web. *Proc. of CSCW 96, Boston, MA, USA*, pages 130–139, 1996.
9. M. Sendín and J. Lorés. Plasticity in mobile devices: a dichotomic and semantic view. *Workshop on Engineering Adaptive Web*, pages 58–67, 2004.
10. M. Sendín and J. Lorés. Remote support to plastic user interfaces: a semantic view. *Selection of HCI related papers of Interaccin 2004*. Springer-Verlag, 2005.
11. M. Sendín and J. Lorés. Towards the design of a client-side framework for plastic uis using aspects. *Int. Workshop on Plastic Services for Mobile Devices*, 2005.
12. D. Thevenin and J. Coutaz. Plasticity of user interfaces: Framework and research agenda. *Proc. of Interact 99, Edinburgh*, pages 110–117, 1999.

Supporting Mobile Collaboration with Service-Oriented Mobile Units

Andrés Neyem, Sergio F. Ochoa, and José A. Pino

Department of Computer Science, Universidad de Chile.
Blanco Encalada 2120, Santiago, Chile
{aneyem, sochoa, jpino}@dcc.uchile.cl

Abstract. Advances in wireless communication and mobile computing extend collaboration scenarios. A current strategy to address productive, educational and social problems is to incorporate mobile workers using computing devices into work practices. Typically, collaborative applications intended to support mobile workers involve some type of centralized data or services. This situation constrains the collaboration capabilities, particularly in ad-hoc communication scenarios. We propose an autonomous software module able to provide and consume services from others units. We call it a Service-Oriented Mobile Unit (SOMU). A SOMU has been implemented as a middleware running on laptops and PDAs. Collaborative mobile applications developed on this middleware are then able to interact among them almost in any communication scenario. Availability of this tool is particularly relevant to support mobile collaboration when there is no stable communication support or no communication at all.

Keywords: Middleware for Mobile Groupware, Service-Oriented Mobile Units, Web services Platform, Ad-hoc Collaboration Scenarios.

1 Introduction

Fast development in the area of information and communication technology and especially in broadband internet access and mobile computing has changed the established ways of communication, learning, entertainment and work in professional and private lives. The *mobile* and *mobility* concepts have a strong link to wireless technologies [1]. Most often a mobile worker is conceived as a person moving and executing tasks anywhere and anytime, using mobile computing devices with wireless communication capabilities. Provided the current mobile computing devices have wireless communication capabilities, any place becomes a potential scenario to support mobile work. Examples of these scenarios are: parks, coffee shops, hospitals, universities, schools, shopping malls, offices and airports.

Mobile workers are on the move to carry out their activities. Usually, they have some instances for data synchronization or collaboration with other people. Mobile workers are frequently not sure which is the next collaboration scenario and its characteristics. Therefore, they need autonomous and flexible collaborative solutions independently of the availability of centralized resources or fixed wireless communication infrastructure (access points). When two or more mobile workers meet, the physical scenario must not be a limitation to collaborate.

Collaboration activities involving mobile workers can be supported by mobile networks, also called MANETs (Mobile Ad-hoc NETworks) [16]. However, it means solutions including MANETs to support the computer-based collaborative activities should be designed and implemented. Most collaborative applications intended to support mobile workers involve some type of centralized data or services. This situation constrains the collaboration possibilities, particularly in ad-hoc communication scenarios. A software piece which is able to provide and consume services from others units is proposed. It is called *Service-Oriented Mobile Unit (SOMU)*. The solution is fully distributed. Each unit has been implemented as a middleware running on laptops and Personal Data Assistants (PDAs). Collaborative mobile applications developed on this middleware are then able to interact among them almost in any communication scenario. Thus, mobile workers using such applications can collaborate when there is no stable communication support or no communication at all. Two application scenarios are briefly described below to illustrate the role of MANETs in mobile collaboration.

- *Disaster Relief*: Activities to resist and recover from natural, hazardous and intentional eXtreme Events (XE) are highly dynamic and demand effective collaboration among a broad range of organizations. First responders (police, firefighters and medical personnel) deployed in the work area need to know the information about the site and affected buildings (e.g. maps, probable people locations and vulnerable points), exit routes, resources deployed in the area and tasks assignment. Mobile workers from several organizations need to be autonomous, interoperable and carry diverse shared information to do the assigned activities. Sometimes they also need to update such information and communicate the updates to the partners, leaders and other organizations in order to support decision-making processes. Typically, this collaboration scenario has minimal or no communication capabilities [6]. However, collaboration among first responders is required. Government authorities in charge of macro-decisions should be able to access information from the mobile workers (e.g., police, firefighters and medical personnel) to monitor the activities evolution and make corrections on previously made decisions.
- *Building and Construction*: The building and construction industry is characterized by: (a) dispersed teams working on the development of a new site, (b) teams do not belong to the same company, (c) they are not able to use fixed communication infrastructure and (d) they need to be on the move to carry out the assigned work. For example, electrical engineers (mobile workers) belonging to a company need to be on the move in order to inspect and record the status of the electrical facilities being developed by the company employees at a construction site. During the inspection, each engineer updates the information recording the current status of the electrical facilities. After the inspection and before leaving the construction site, the engineers meet to check agreement on the updated information and review it. If they detect incomplete or contradictory data, some of them can inspect the facilities again in order to solve such case. Similarly to the previous scenario, mobile workers need autonomy, interoperability and they also need to be able to collaborate no matter the features of the physical scenario.

Mobile computing devices and mobile ad-hoc wireless networks (MANETs) offer a wide range of new collaboration possibilities for mobile workers. However, the design and implementation of the mobile collaborative solutions for ad-hoc scenarios imply several challenges in terms of the following aspects.

Autonomy: Collaborative mobile applications should function as autonomous solutions. Communication availability in the physical scenario and access to centralized shared data and services cannot be a limitation to support collaboration among mobile workers in ad-hoc scenarios. Therefore, solutions able to work in Peer-to-Peer (P2P) settings are required.

Interoperability: Provided mobile workers may need to do casual or opportunistic collaboration, the collaborative mobile applications they use should offer data and services interoperability.

Shared information management: Shared information supporting collaborative applications in these scenarios need to be highly replicated since there are frequent disconnections in wireless networks (even using access points). Keeping the shared information coherence in a P2P network is not a trivial problem to solve.

Use of hardware resources: The collaborative mobile applications should operate, in many cases, with constrained hardware resources; e.g., the case in which these solutions need to run on PDAs.

Next section describes the challenges and opportunities offered by service-oriented computing to support collaboration in ad-hoc wireless settings. Section 3 presents related work. Section 4 describes the way to overcome the stated challenges with SOMUs. Section 5 shows two application scenarios, and section 6 presents the conclusions and future work.

2 Service-Oriented Computing in Ad-Hoc Wireless Settings

Ad-hoc networking refers to a network with no fixed infrastructure [24]. When the nodes are assumed to be capable of moving, either on their own or carried by their users, these networks are referred as MANETs. The nodes of the network rely on wireless communication to collaborate with each other. The advantage of ad-hoc networking is that the absence of a fixed infrastructure reduces the cost, complexity and time required to deploy the network. It also allows users to be on the move transporting their communication capabilities [23]. Unfortunately, most of these MANETs have a small communication threshold in terms of allowed distance between two mobile workers. In addition, the lack of a fixed infrastructure introduces challenges for using and maintaining ad-hoc networks. Knowledge of various factors will help to motivate understanding of the protocols that have been developed for ad-hoc networks. A brief explanation of these properties follows.

- *No pre-existing infrastructure:* By definition, ad-hoc networks do not have any infrastructure. The nodes in the network rely on wireless communication for information dissemination and gathering. This lets ad-hoc networks be used in

remote environments, and mainly to support mobile workers. Moreover, the MANETs are attractive because of the reduced effort to set up and use them.

- *Small communication threshold:* Mobile computing devices provide communication services without using a base station when they are part of a MANET. Thus, each device may function as a base station to act as a gateway between peer devices or to access other networks. The current wireless communication norms supporting mobility have a limited communication threshold (or communication range). For example, the IEEE 802.11b/g (Wi-Fi) threshold is about 200 meters in open areas and 20 meters in built areas.
- *Power-scarce devices:* Mobile devices making up the ad-hoc network have a physical environment that is assumed to be devoid of resources such as power. In fact, because of the absence of any underlying infrastructure, power outlets generally are not available. For this reason, mobile devices that form the ad-hoc network use either battery power or passive power sources, such as solar energy. This fact further reduces the communication threshold of this type of networks.
- *No centralized mechanisms:* Since ad-hoc network do not have any underlying infrastructure and wireless communication is employed, centralized routing algorithms are not applicable. The cost of transmitting data from all nodes in the network to a central location becomes prohibitively expensive in terms of power usage. Furthermore, centralized components become critical failure points and then there are the typical problems with scalability and fault tolerance for processing all the information.

On the other hand, *Service-Oriented Computing* (SOC) is a new paradigm gaining popularity in distributed computing environments due to its emphasis on highly specialized, modular and platform agnostic code facilitating interoperability of systems [22]. A key issue with SOC in ad-hoc networks is to mitigate the problem of frequent disconnection and to ensure that some channel between the user and the provider of a service is maintained for a significant period. Furthermore, SOC helps decouple concerns about network availability and connectivity and it also implies simplifications in the software development process.

The service model is composed of three components: services, clients and a discovery technology. Services provide useful functionality to clients. Clients use services to support complex functionalities that will be available for users. The discovery process enables services to publish their capabilities and clients to find and use needed services. As a result of a successful lookup, a client may receive a piece of code that actually implements the service or facilitates the communication to the server offering the service. The implementations of service-oriented models may have some limitations in terms of functionality because of the peculiarities of the ad-hoc wireless settings.

The idea of using mobile computing devices as hosts for service registries is very appealing. However, overloading simple devices belonging to a work session may lead to a defensive behavior from a collaborative system, e.g., terminating advertisement broadcasts or completely ignoring client communication.

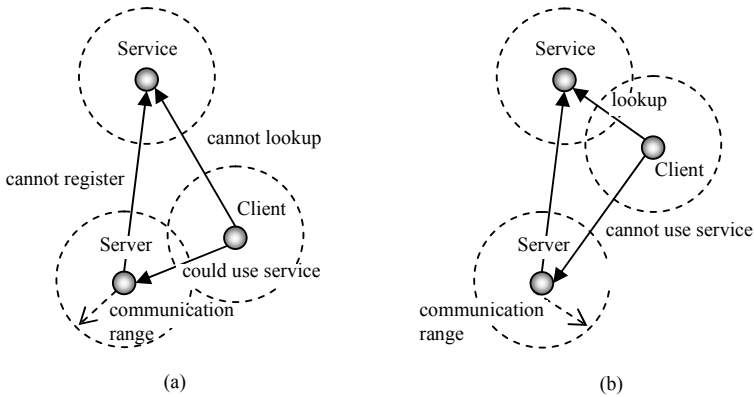


Fig. 1. a) The client could use the service but it cannot discover it because the service registry is not accessible; b) A client discovers a service which is no longer reachable

Failure of a mobile computing device implies a complete lack of communication between users in a collaborative session and between clients and services whose communication is routed via this device, even if they could communicate directly. Therefore, the service model needs to adapt itself to the new networking conditions. For example, if the node hosting a service registry suddenly becomes unavailable, the advertising and lookup of services becomes paralyzed even if the pair of nodes representing a service and a potential client remains connected (Fig. 1a). Furthermore, due to frequent disconnections and mobility of nodes, there is another problem when the advertisement of a service is still available in the lookup table until its lease expires (Fig. 1b).

As a summary, high degree of freedom and a fully decentralized architecture can be obtained in MANETs at the expense of facing significant new challenges. MANETs are opportunistically formed structures that change in response to the movement of physically mobile hosts running potentially mobile code. New wireless technologies allow devices to freely join and leave work sessions and networks, and exchange data and services at will, without the need of any infrastructure setup and system administration. Frequent disconnections inherent in ad-hoc networks lead to inconsistency of data in centralized service directories. Architectures based on centralized lookup directories are no longer suitable. Therefore, the model and technologies addressing these issues should consider all nodes as mobile units able to provide and consume services from other mobile units.

3 Related Work

Several collaborative solutions have been proposed to support mobile workers [2], [8], [17], [18], [26]. Although the proposals have shown to be useful to support specific collaborative activities, they were not designed as general solutions. Therefore, the capability to reuse these solutions in various work scenarios is relatively small.

On the other hand, there are several interesting initiatives in the middleware area, which propose reusable functions to support collaboration in P2P networks. One of them is LaCOLLA [14]. This middleware has a P2P architecture and provides general purpose functionalities for building collaborative applications. LaCOLLA works well in networks with important signal stability, such as fixed or one-hop wireless networks. However, the middleware does not support autonomous members of a group and does not have components and mechanisms that will allow mobile devices become LaCOLLA peers.

Unlike LaCOLLA, the iClouds framework offers spontaneous mobile user interaction and file exchange in mobile ad-hoc networks [11]. This framework also provides independence of a server doing a full replication of any shared file, which is appropriate in MANET scenarios. However, it does not provide support to exchange shared objects, just files. In addition, iClouds does not distinguish among copies of the same shared file (e.g. master and slave copies) and it does not support distributed operations on those files either. The functions provided by iClouds are focused just on data sharing.

There are frameworks that provide, through an API, specific functionalities to support mobile collaboration, such as YCab [5] and YCab.NET [21]. These frameworks implement their own protocol and they provide just the following generic services: session manager, text chat, image viewer, GPS and client info. Probably, the most popular framework to support P2P collaboration is JXTA [13]. This framework provides a common platform to help developers build distributed P2P services and applications. Here, every device and software component is a peer and can easily cooperate with other peers. Although JXTA has shown to be useful to support collaboration in P2P networks, it also requires a fixed or one-hop wireless network (similar to LaCOLLA). Therefore, it is not well suited to apply it in ad-hoc mobile work settings.

On the other hand, Nokia has developed a services-oriented framework that could be used to support mobile collaboration. This framework includes a set of APIs and an SDK (Software Development Kit) allowing developers to create service-oriented applications that act as consumers of Web services on mobile devices [12]. Provided the mobile applications can just consume services, their autonomy is limited and they require a service provider, which is not suitable for ad-hoc mobile work scenarios.

Currently, there are several proposals to share information in P2P networks, even considering mobile computing devices [10], [20]. Typical examples are tuple-based distributed systems derived from LINDA [7], such as: FT-LINDA, JINI, PLinda, T-spaces, Lime, JavaSpaces and GRACE [9], [19], [3]. Despite the fact these implementations work in P2P networks, they use centralized components that provide the binding among components of the distributed system. Other middleware, such as XMIDDLE [15] and PASIR [20], allow mobile hosts to share documents across heterogeneous mobile hosts, permitting on-line and off-line access to data. Nevertheless, these middleware are just focused on data sharing and they do not support the autonomy and interoperability capabilities required by mobile workers.

4 The Services-Oriented Mobile Unit

The need to support mobile collaboration in ad-hoc work scenarios and the limitations of current solutions to support it motivated the development of the SOMU software module. SOMU is a lightweight platform able to run on PDAs and notebooks. It enables each mobile computing device to produce and consume Web services from other peers. Such functionality is implemented in a lightweight Web server called μ WebServer (Fig. 2). Thus, the autonomy and part of the interoperability required by mobile workers is supported.

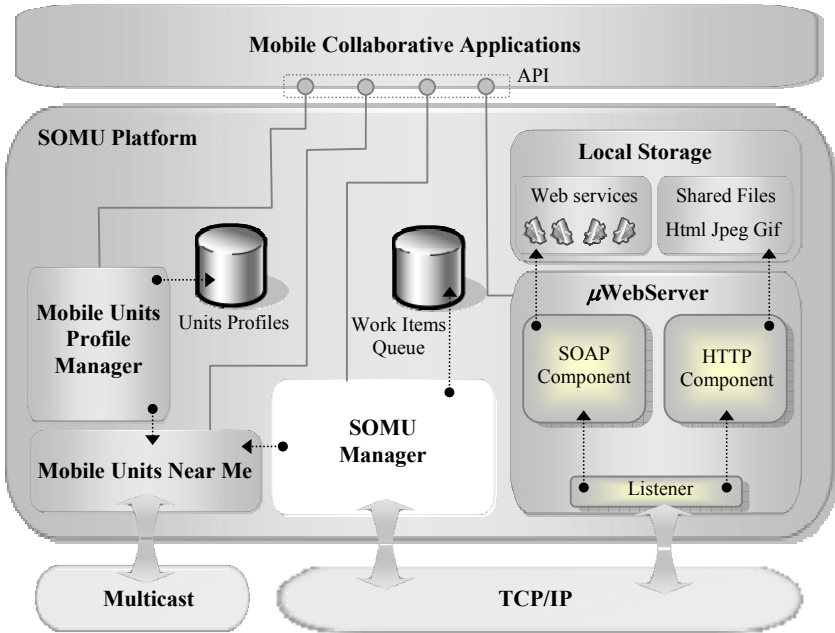


Fig. 2. SOMU Architecture

SOMU also implements a *local storage* which is composed of (1) a shared storage space to allocate the files the mobile unit wants to share, and (2) a space to allocate those Web services exposed by the mobile unit. By default, SOMU provides basic Web services for Web services description and discovery.

The *SOMU Manager* is the component in charge of creating, storing and dispatching work items when a mobile collaborative application invokes Web services exposed by other mobile units. The work items stored in a mobile unit represents the Web Services (WS) invocations that such unit needs to perform. Each work item is composed of a ticket, a mobile universal unit, the WS proxy, WS input and WS output. The *ticket* is the work item identifier. It is used to communicate the results of a WS invocation to a mobile collaborative application. The *Mobile Universal Identification (MUI)* identifies each mobile unit. This identifier allows the

SOMU Manager to make direct invocations to WS running on other mobile units. *WS Proxy* contains the information required to coordinate the invocation and the response of WS exposed by other mobile units. *WS Input* contains the invocation parameters to be sent by the WS Proxy when it invokes the remote WS. *WS Output* contains the results of a WS invocation.

The *Mobile Units Near Me* is the component in charge of discovering and recording the mobile units that are close to the current mobile device. This information is used to decide a good time to start an interaction with a specific mobile unit. This component uses a multicast protocol. It involves discovering the name, universal identification and the IP address of the mobile units belonging to the MANET.

Since Web services are typically accessed from different kinds of mobile computing devices, interoperability and personalization play an important role for universal access. The *Mobile Units Profile Manager* stores and manages information related to mobile units, such as the universal identification, hardware, software, and network capabilities. Web service can use this information to provide optimized contents for various clients. The two main components of the platform, i.e., the *μWebServer* and the *SOMU Manager*, are explained in the next two sub-sections.

4.1 *μWebServer*

The *μWebServer* has the capability of exposing Web services and executing HTTP requests from Laptops and PDAs. The *listener* is responsible for managing client requests on a particular port. It performs validations and determines the most appropriate supporting components to carry out a request. The supporting components represent the implementation of a particular Internet protocol. The *μWebServer* implements supporting components for HTTP and SOAP.

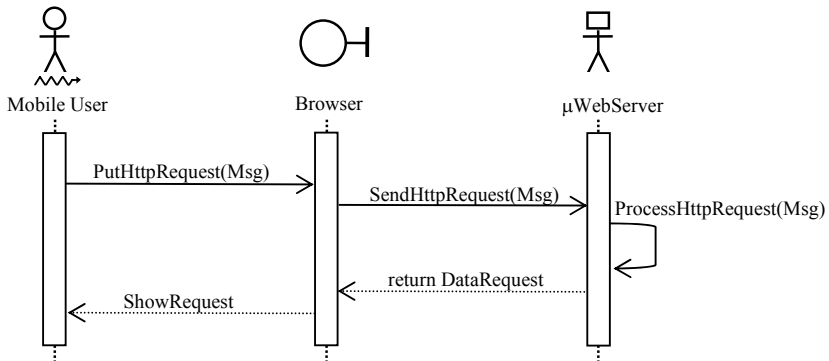


Fig. 3. Sequence diagram of result request service over HTTP

The HTTP component supports the processing of HTML, GIF and JPEG Web requests and GET and POST through SOAP components. As client requests are received, the required file is retrieved from local storage. Then, this file is converted

into a stream of bytes and sent back to the mobile unit client. Figure 3 shows the sequence diagram to invoke Web services over HTTP GET operations.

Figure 4 (a) presents the results of invoking the “Mobile UDDI” Web service (included by default in SOMU), which provides information about all Web services hosted in a remote mobile unit. Figure 4 (b) presents the results of a similar invocation. In this latter case, the invoked remote Web service is the “Mobile Info Profile” Web service (also included by default), which informs the WSDL document related to it.

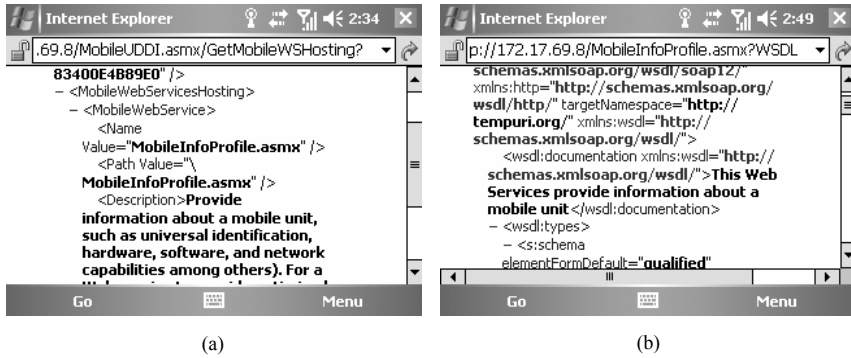


Fig. 4. (a) List of Web services hosted in a remote mobile unit; (b) WSDL of a remote Web service

On the other hand, the SOAP component addresses the requirements of processing Web services remote invocations by clients. The current implementation supports GET, POST and SOAP action operations. Typically GET and POST operations are used for browser requests. These operations return an XML string representing the results. Meanwhile, SOAP actions are used to identify SOAP packets sent by applications using a particular Web service. Additionally, the SOAP component provides facilities to automatically generate WSDL (Web Service Definition Language) files from a requested Web service.

4.2 SOMU Manager

This component creates, stores and dispatches work items when a mobile collaborative application wants to invoke remote Web services. If the destination mobile unit is online, the SOMU manager picks up the work item and processes it, by creating a proxy client instance that interacts with the remote Web service. When the SOMU manager receives the results of the Web service invocation, it notifies to the mobile collaborative application and it delivers the results.

On the other hand, if the remote mobile unit hosting the Web services is not reachable, the mobile collaborative application switches to offline mode. Internally, the mobile collaborative application calls the manager to create a work item. The work item is stored in the *work items queue*. Periodically, the *mobile units near me* verifies if the destination mobile unit gets online. When the destination unit is online, the SOMU manager from the requester unit retrieves the work item from the queue.

Then, the manager sends an invocation of the remote Web service using the proxy functions. After processing the request, the remote Web service returns the results back to proxy client. Finally, the manager then returns the results to the mobile collaborative application.

4.3 SOMU Components Dynamic Interaction

In order to understand the SOMU platform functionality, Figure 5 shows the dynamics of the interactions between two mobile collaborative applications, when an application “A” requires a Web service exposed by a remote application “B”. The first step of this interaction requires the application “A” make a local request to invoke the remote service from “B”. “A” states this requirement through a work item which is created and stored by the SOMU manager (2nd step). Then, this manager asks to the *mobile units near me* component if the application “B” is in online mode and if “B” is within the “A” communication range. If the answer is negative, then the SOMU manager waits and retries until it gets a positive answer (3rd step).

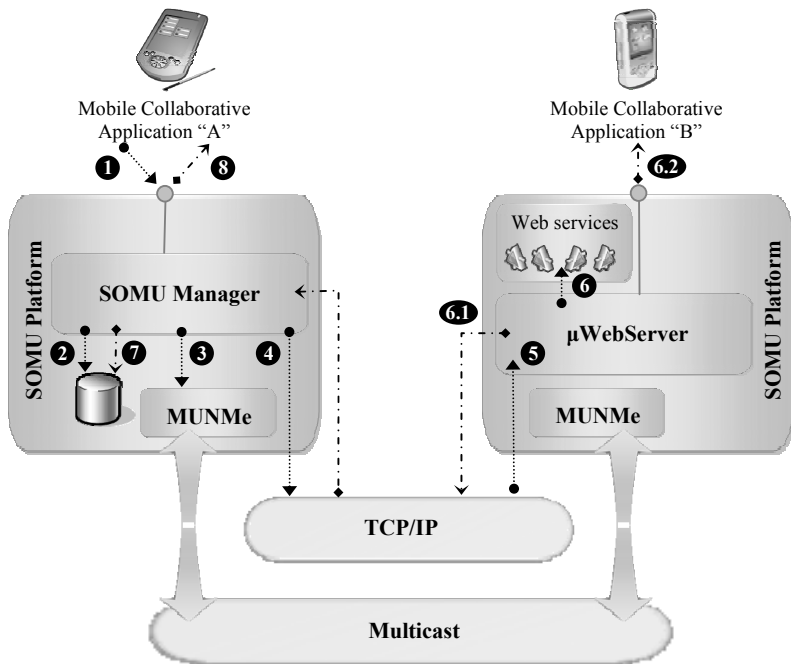


Fig. 5. Interactions among SOMU main components

When the mobile application “B” gets online and in the “A” communication range, the SOMU manager creates the proxy using reflection from the context information. Such information is in the WS Proxy field which is part of the work item. Then, the

SOMU manager invokes the remote service hosted in “B” (4th step). The invocation is received by the μ WebServer (5th step). Since the request is a Web service invocation, the μ WebServer SOAP component activates the corresponding Web service through reflection and it invokes the method implementing the service (6th step). The μ WebServer returns the results to the mobile application “A” (6.1 step). If the application “B” is subscribed to receive the events related to a specific Web service, the μ WebServer will send the corresponding notification (6.2 step). When the SOMU manager from “A” receives the results, it removes the work item from the queue (7th step). Finally, the SOMU manager from “A” notifies the mobile application “A”, indicating the work item with the specific ticket has finished its processing (8th step).

4.4 SOMU Implementation Aspects

SOMU was implemented in C# using the .NET Compact Framework; however, it can also be implemented using the J2ME SDK for mobile devices. The .NET platform was chosen since it offered rapid prototyping and a rich development environment including live debugging on emulators. The .NET libraries natively support XML manipulation, Web service description and reflection. This allows us to implement basic services for Web services description and discovery.

5 Application Scenarios

The following scenarios show how the actions taken by a mobile collaborative application are translated into the actions that occur within the Services Oriented Mobile Units. Two mobile collaborative applications which use the services provided by the platform were developed in order to test SOMU. These applications represent a proof-of-concept and they illustrate the feasibility of the proposed approach. One of them concerns one of the application scenarios mentioned in Section 1. They are briefly described below.

5.1 Mobile Electronic Meeting System

The implemented mobile electronic meeting system, called *Meeting Space*, is an interactive mobile computer-based system for supporting decision meeting processes. Like other Electronic Meeting Systems (EMS), the application goal is to support group members to be effective and make good decisions. The application was designed to be used by mobile users working online and offline. Provided the application running on each mobile unit is autonomous, uncoupled and independent of centralized components and networking infrastructure, users can meet in almost any place and carry out an ad-hoc work meeting (Fig. 6). The tool supports just some pre-meeting and meeting processes [4]. Specifically, it allows to: (1) create and share a meeting agenda, (2) specify, make private annotations and provide feedback about shared problems and solution ideas, (3) detect peers near the current device and generate notifications, and (4) share documents with peers.

During a pre-meeting, users can work alone in order to collect information and make private annotations about each item of the meeting agenda. When the *mobile users near*

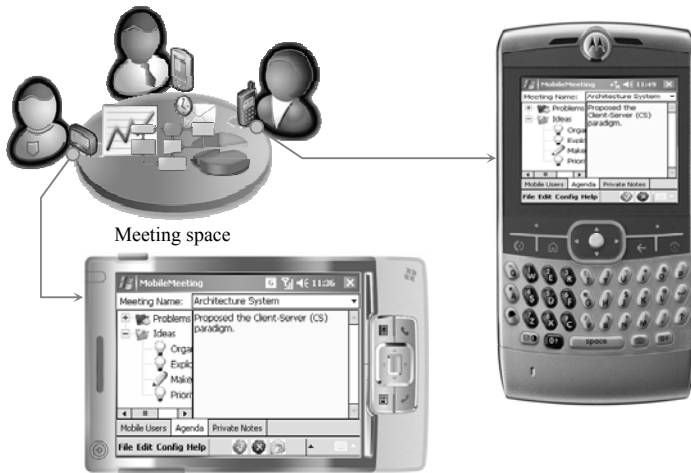


Fig. 6. Meeting Space Application

me component detects two or more users in the same communication range, it notifies the local SOMU manager. Then, this manager notifies the others SOMU managers running on the remote mobile units, which deploy a visual notification on the screen of the mobile computing devices. Thus, the application provides these persons the opportunity to hold an ad-hoc meeting to discuss in a face-to-face setting. For such discussion, the users can share documents and annotations by using the SOMU Web services and also specific Web services developed just for this application. The preliminary conclusions or results of the pre-meeting can be recorded in a shared file. Then, this file can be distributed among the mobile units by using the SOMU Web services.

Users can propose and share ideas (and annotations related to them) to discuss each item of the meeting agenda. Users can also provide feedback about the proposals and interact with other users in order to refine an idea, problem or any other item. The current application does not support rich computer-supported interaction mechanisms, such as full mediated discussion forums or brainstorming tools.

The meeting documents are registered and distributed to the corresponding members at the end of the meeting. Since the meeting place is almost any available place, the service-oriented solution proposed by this application becomes suitable to address the physical scenario constraints.

Figure 7 presents a possible sequence diagram of a process to show how the system supports an idea discussion. When a mobile user A proposes a public idea, the SOMU manager creates a work item. These public ideas can be delivered to other peer members as soon as possible. If a mobile user B is in A's communication range, then A's SOMU manager invokes a Web service from B in order to communicate the idea. When B's μ WebServer receives the proposal, it communicates the idea to the local application. The B's application makes the idea available for the user to process it. If B rejects the idea, then such communication is received by the A's application via the

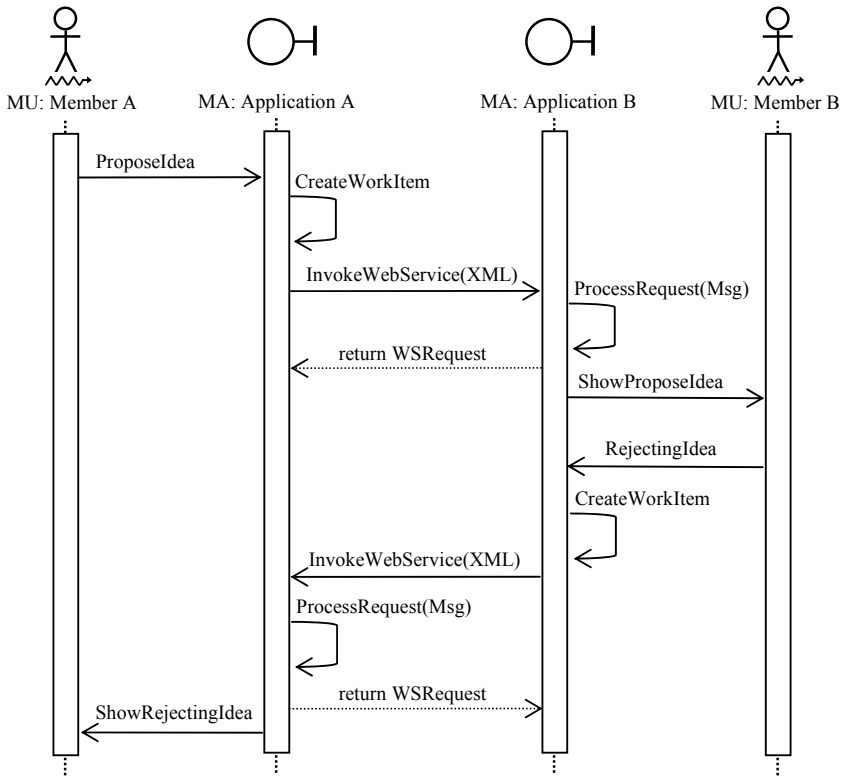


Fig. 7. Sequence diagram of typical activity of accepting/rejecting/refining ideas between two meeting members

SOMU manager. Afterwards, the A user can redefine the idea and submit it for consideration again. Thus, a new interaction cycle begins.

A next version of the *Meeting Space* application is planned to have support for voting. Users will be able to cast anonymous votes, a desirable feature in certain types of decision meetings. This feature will make the application valuable in physical settings otherwise unsuitable for these decision meetings.

5.2 Group Decision-Support System for Disasters in Urban Areas

Disasters affecting urban areas have shown the need to improve the group decision-making processes and the coordination of efforts done by organizations participating in disaster relief activities [6]. Typically, police is in charge of isolating and securing the affected area, firefighters are the initial responsible for protecting human life and physical infrastructure, medical personnel are responsible for healthcare of the affected people, and government authorities are responsible for coordinating the efforts of the participant organizations in order to reduce the impact of the extreme event on Society [6]. One key aspect to consider is that critical activities must be

carried out in a short time period, such as ensuring the safety of the disaster area and conducting search and rescue procedures [25].

Initially, first responders deployed in the disaster scenario have to coordinate their efforts in order to support these activities. The developed application allows mobile first responders to access and distribute shared geographical information of the disaster area and the resources available to support the mitigation process. The information about the available resources is deployed on a map in order to get a visual identification of the resource allocation (Fig. 8). This information is divided in several layers. Each organization involved in the activities can update a specific layer. These information layers can be shared among mobile workers deployed in the disaster area and also among the disaster managers. Typically, a mobile worker uses a PDA or a Tablet PC.

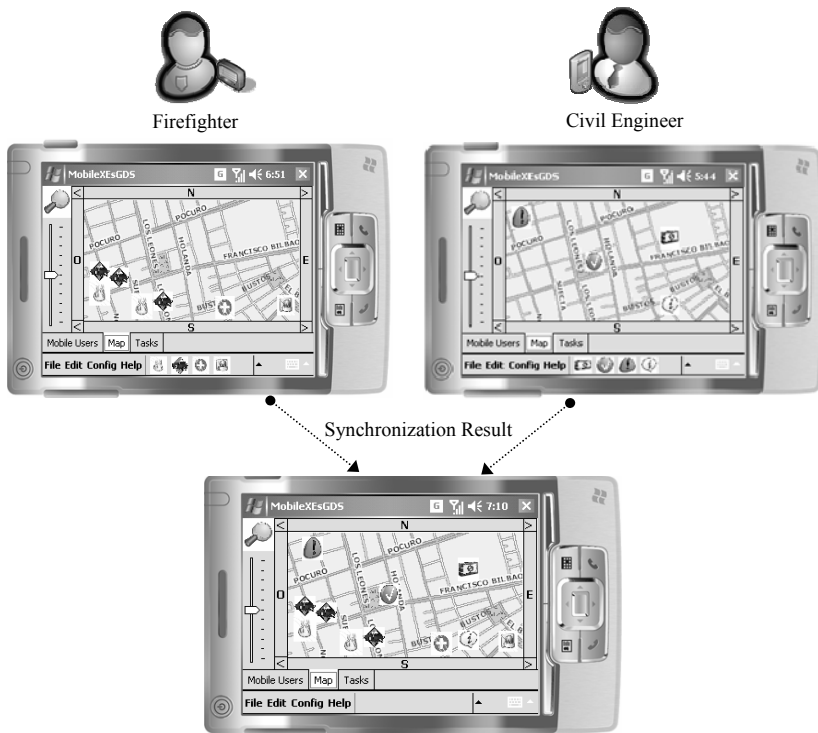


Fig. 8. Information Synchronization between a firefighter and a civil engineer

In order to illustrate how this application works, let us consider the following situation. A firefighter team needs to get updated information related to the stability of the physical infrastructure of an area, because they need to conduct search and rescue activities in such place. Therefore, the most direct way is to get an updated information layer from civil engineers evaluating the area. Two or more firefighter team members can use PDAs to get information from civil engineers, other partners and the command post.

If the communication in the disaster area is based on MANETs, then the firefighter team members need to be aware about the presence of civil engineers within their communication range. The *mobile units near me* component can notify these firefighters about such situation. Firefighters synchronize structural information from the civil engineer and get an updated view of the disaster area (Fig. 8). Thus, these first responders can make better decisions about where and when to conduct the search and rescue procedures. The decisions made and the results of the search and rescue activities are recorded in the firefighters’ information layer. Now, a new update of the shared information is available.

This synchronization process uses not only the Web services provided by SOMU by default, but also other Web services created just for this application. One of these Web services is SyncXML that synchronizes to XML files following a policy similar to the one proposed by XMIDDLE [15]. This Web service is essential for the application because all basic information is represented in XML. In order to illustrate how SOMU components interact in this application, let us consider the same case described in Figure 8. In this case, Figure 9 presents a sequence diagram of the interactions between SOMU and application components.

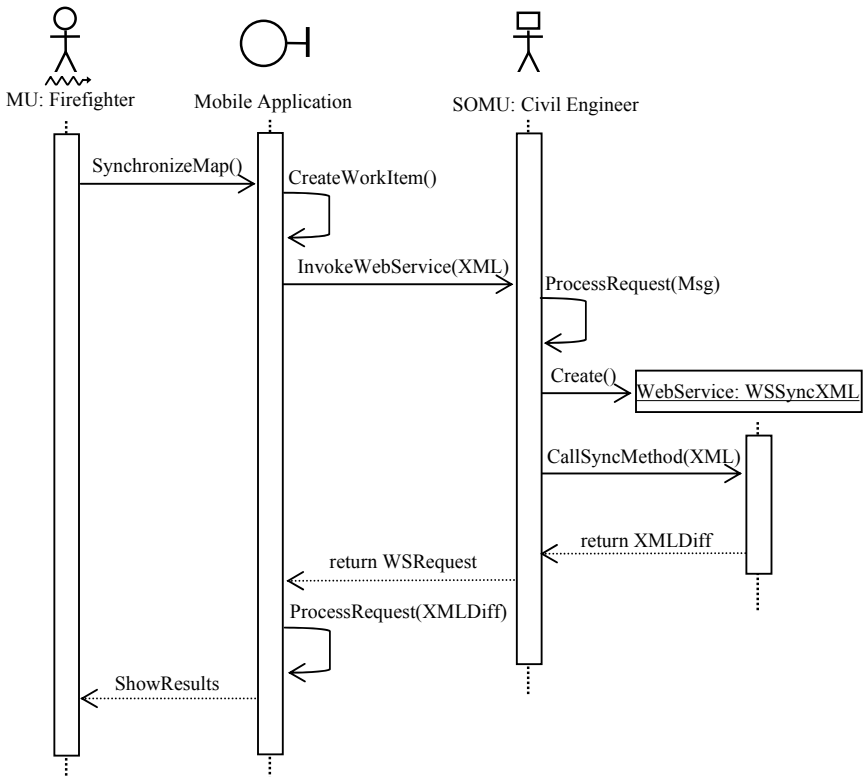


Fig. 9. Sequence diagram of synchronization structural information between a firefighter and a civil engineer

When a civil engineer gets in the communication range of the firefighter team, or vice versa, firefighters using PDAs are notified through a user event launched to the device screen. Therefore, a mobile unit used by a firefighter requests a synchronization operation in order to get updated information related to the physical infrastructure information layer. Then, the SOMU manager creates a work item and invokes a Web service exposed by the civil engineer's mobile unit, by indicating the version of the layer information the firefighters have. When, μ WebServer in the civil engineer mobile unit receives the request, it launches the SyncXML service to process the request. Since the information the firefighter has is outdated, a local process is launched in the civil engineer mobile unit to retrieve the information updates from the local layer. Then, an XML file indicating the information updates is sent to the firefighter SOMU manager as response to the invoked Web service. The receiver mobile unit processes such information and shows it on the device screen.

6 Conclusions and Future Work

Most frameworks and platforms proposed to support collaborative activities of mobile workers use some type of centralized data or services. This centralization jeopardizes the application capabilities to support collaboration in ad-hoc communication settings. This paper presents a platform called SOMU (Service-Oriented Mobile Unit) intended to support the collaborative activities carried out by mobile workers in ad-hoc scenarios. Unlike the previous related works, SOMU proposes a fully decentralized architecture allowing mobile devices to act as autonomous units. The platform lets mobile computing devices expose and consume Web services in order to carry out an activity. Collaborative mobile applications developed on this middleware are then able to interact among them almost in any communication scenario. Availability of this tool is particularly relevant to support mobile collaboration when there is no stable communication support or no communication at all.

This middleware was implemented in C# using the functionality provided by the .NET Compact Framework. However, the same functionality could be implemented using J2ME. The type of implementation allows SOMU to run on a wide range of computing devices from PDAs to desktop PCs.

The platform provides a basic foundation for the development of mobile collaborative applications. This platform intends to increase the technical feasibility of solutions in the area and to reduce the development effort of MANET-based mobile collaborative applications. These issues have not been fully analyzed yet for the two developed applications, but the initial findings support these hypotheses.

Future work includes, in the short future, formal experimentation to study the possible contributions and limitations of SOMU and the consequences on the applications developed on it. Furthermore, the functionality provided by Web services will be tested in order to determine if the uncoupled interaction proposed by SOMU represents a limitation for mobile workers when collaborating in ad-hoc communication scenarios. As a second step, the functionality of SOMU will be extended to integrate (by default) P2P sessions management, standard WS discovery mechanisms (such as WS-Discovery), and enabled support for the new stack

specification, such as WS-Security, WS-Trust, WS-Federation, WS-Addressing, WS-Routing and WS-Attachment.

Acknowledgments. This work was partially supported by Fondecyt (Chile), grant N°: 1040952 and by MECESUP (Chile) Project N°: UCH0109.

References

1. Adelstein, F., Gupta, K.S., Golden, R., Schwiebert, L.: *Fundamentals of Mobile and Pervasive Computing*. McGraw-Hill Publisher, (2005).
2. André, P., Antunes, P.: SaGISC: A Geo-Collaborative System. Proc. of the 10th International Workshop on Groupware (CRIWG), Springer, LNCS 3198, San Carlos Costa Rica, (2004) 175-191.
3. Bosneag, A.M., Brockmeyer, M.: GRACE: Enabling collaborations in wide-area distributed systems. Proc. of the 14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise (WETICE), Workshop on Distributed and Mobile Collaboration (DMC), IEEE CS Press, Linkoping University Sweden, (2005) 72-77.
4. Bostrom, R., Anson, R., Clawson, V.: Group Facilitation and Group Support Systems. In *Group Support Systems: New Perspectives*, Jessup and Valacich (Eds.), (1993) 146-168.
5. Buszko, D., Lee, W., Helal, A.: Decentralized Ad-Hoc Groupware API and Framework for Mobile Collaboration. Proc. of ACM International Conference on Supporting Group Work (GROUP), ACM Press, Colorado USA, (2001) 5-14.
6. Comfort, L.: Coordination in Complex Systems: Increasing Efficiency in Disaster Mitigation and Response. *International Journal of Emergency Management* 2(1), (2004) 62-80.
7. Gelernter, D.: Generative Communication in Linda. *ACM Transactions on Programming Languages and Systems* 7(1), (1985) 80-112.
8. Guerrero, L., Pino, J., Collazos, C., Inostroza, A., Ochoa, S.: Mobile Support for Collaborative Work. Proc. of the 10th International Workshop on Groupware (CRIWG), Springer, LNCS 3198, San Carlos Costa Rica, (2004) 363-375.
9. Handorean, R., Payton, J., Julien, C., Roman, G.: Coordination Middleware Supporting Rapid Deployment of Ad Hoc Mobile Systems. Proc. of the 23rd International Conference on Distributed Computing Systems (ICDCS), Workshop on Mobile Computing Middleware (MCM), IEEE CS Press, Rhode Island USA, (2003) 363-368.
10. Hauswirth, M., Podnar, I., Decaer, S.: On P2P Collaboration Infrastructures. Proc. of the 14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise (WETICE), Workshop on Distributed and Mobile Collaboration (DMC), IEEE CS Press, Linkoping University Sweden, (2005) 66-71.
11. Heinemann, A., Kangasharju, J., Lyardet, F., Mühlhäuser, M.: iClouds: Peer-to-Peer Information Sharing in Mobile Environments. Proc. of the 9th International Euro-Par Conference (Euro-Par), Springer, LNCS 2790, Klagenfurt Austria, (2003) 1038-1045.
12. Hirsch, F., Kemp, J., Ilkka, J.: *Mobile Web Services: Architecture and Implementation*. Nokia Research Center. John Wiley & Sons Publisher, (2006).
13. JXTA Project, URL: <http://www.jxta.org>. (2003).
14. Marquez, J., Navarro, L.: Autonomous and Self-sufficient Groups: Ad Hoc Collaborative Environments. Proc. of the 11th International Workshop on Groupware (CRIWG), Springer, LNCS 3706, Porto do Galinhas Brasil, (2005) 57-72.

15. Mascolo, C., Capra, L., Zachariadis, S., Emmerich, W.: XMIDDLE: A Data-Sharing Middleware for Mobile Computing. *Journal on Personal and Wireless Communications* 21(1), (2002) 77-103.
16. Meissner, A., Musunoor, S.: Group Integrity Management Support for Mobile Ad-Hoc Communities. Proc. of the 4th International Middleware Conference, Workshop on Middleware for Pervasive and Ad-Hoc Computing (MPAC), ACM Press, Rio de Janeiro Brazil, (2003) 53-59.
17. Menchaca-Mendez, R., Gutierrez-Arias, E., Favela, J.: Opportunistic Interaction in P2P Ubiquitous Environments. Proc. of the 10th International Workshop on Groupware (CRIWG), Springer, LNCS 3198, San Carlos Costa Rica, (2004) 349-362.
18. Muñoz, M.A., Rodriguez, M., Favela, J., Martinez-Garcia, A.I., Gonzalez, V.M.: Context-aware mobile communication in hospitals. *IEEE Computer*, 36(9). (2003) 38-46.
19. Nemlekar, M.: Scalable Distributed Tuplespaces. MSc. Thesis. Department of Electrical and Computer Engineering, North Carolina State University, Chapter 5, (2001).
20. Neyem, A., Ochoa, S., Guerrero, L., Pino, J.: Sharing Information Resources in Mobile Ad-hoc Networks. Proc. of the 11th International Workshop on Groupware (CRIWG), Springer, LNCS 3706, Porto do Galinhas Brasil, (2005) 351-358.
21. Procopio, M.J.: YCab.NET: Decentralized Collaboration Groupware for Mobile Devices using the MS .NET Framework. Master's of Science Thesis, University of Florida. (2002). Available at http://etd.fcla.edu/UF/UFE1000155/procopio_m.pdf
22. Sen, R., Handorean, R., Roman, G-C., Gill, C.: Service Oriented Computing Imperatives in Ad Hoc Wireless Settings (Book Chapter), *Service-Oriented Software System Engineering: Challenges And Practices*, Stojanovic and Dahanayake (Eds.), Idea Group Publishing, Hershey, USA, (2005) 247-269.
23. Stojmenovic, I., Wu, J.: Ad-hoc Networks. *IEEE Computer*, 37(2), (2004) 9-74.
24. Tschudin, C., Lundgren, H., Nordström, E.: Embedding MANETs in the Real World. Proc. of the 8th International Conference on Personal Wireless Communications (PWC), Springer, LNCS 2775, Venice Italy, (2003) 578-589.
25. Turoff, M.: Past and Future Emergency Response Information Systems. *Communications of the ACM*, April, (2002) 29-32.
26. Zurita, G., Baloian, N.: Handheld Electronic Meeting Support. Proc. of the 11th International Workshop on Groupware (CRIWG), Springer, LNCS 3706, Porto do Galinhas Brasil, (2005) 341-350.

SAGA: A Web Services Architecture for Groupware Applications

Benjamim Fonseca¹ and Eurico Carrapatoso²

¹ UTAD/CETAV, Quinta de Prados, Ap. 1013
5001-801 Vila Real, Portugal
benjaf@utad.pt

² FEUP/INESC Porto, Rua Dr. Roberto Frias
4200-465 Porto, Portugal
emc@fe.up.pt

Abstract. To improve their efficiency and competitiveness, organizations are increasingly interested in applications that support team work, usually know as groupware. Beside interoperability, familiarity with the application and users' mobility support, a feature that is of outmost importance in groupware is the notification of events produced by cooperative activities. Web Services have emerged recently to support the exchange of data in distributed environments using common Internet technologies and have been used mainly to build business-to-business applications. However, Web Services have capabilities that make them suitable to meet the requirements posed by groupware applications, a field where little work has been carried out. This article describes a model for developing cooperative applications based on Web Services technology and using asynchronous notification of events, and presents a brief description of the implementation of the support services for that model and of a prototype application that uses them.

Keywords: CSCW, Groupware, Web Services.

1 Introduction

One technological development that revolutionized the professional and leisure activities in the last decade was the Internet, particularly the World Wide Web (WWW). Its ease of utilization and its potential regarding information retrieval and business and leisure activities have led to an exponential growth of the users' community, fostered by the recent availability of mobile access to the Web. The dawn of this century witnessed the emergence of a new technology that enables the exchange of messages between remote applications or services, using Web protocols and data formats widely supported. This technology, known as Web Services, has capabilities that ease the interaction between peers in heterogeneous environments.

In the current organizational context of companies and institutions, one success factor is the ability to effectively realize team work. This fact has raised the interest of organizations in applications of Computer Supported Cooperative Work (CSCW). In this kind of applications, usually referred as groupware, interoperability issues,

application's familiarity and users' mobility support assume significant importance. A common requirement for cooperative applications is the ability to asynchronously notify the occurrence of events produced as the result of cooperative activities.

During the last decade, several frameworks have been developed to enable the construction of groupware applications and the majority of them were implemented using Java technology. Examples of such frameworks are CBE [1] MetaWeb [2], Mushroom [3], Collaboration Bus [4], Habanero [5], Agilo [6], AORTA [7], DOORS [8], ANTS [9] and Artefact [10]. A few other frameworks were implemented using other programming languages, like GroupKit [11] (C++) and COAST [12] (SmallTalk). These frameworks usually communicate through TCP/IP, HTTP or even proprietary protocols, except the Artefact framework that relies on CORBA [13]. The major flaws of all these frameworks are the lack of interoperability support and the inability to integrate legacy applications. Only Artefact can theoretically address these issues, but the CORBA implementations never achieved the desired degree of interoperability and only a few of the specified CORBA services were really implemented.

A technology emerged recently and promises to efficiently address interoperability and legacy support: Web Services. The main field in which Web Services are currently used is business-to-business (B2B) applications and in the vast set of groupware categories, Workflow Management is the only one that shows significant developments in the use of Web Services. This fact, along with the issues referred above, motivated us to define a model for the creation of cooperative applications that uses Web Services technology to provide the mechanisms that support the cooperative activities, relying on the asynchronous notification of events to reflect them. The main objective of this model was to define a set of core services that enabled the development of cooperative applications that used these services to provide features such as the sharing of cooperative events and shared data consistency. Along with the innovation issue referred before, Web Services rely on highly accepted and disseminated technologies such as the HyperText Transfer Protocol (HTTP) and the eXtensible Markup Language (XML), providing a high level of interoperability, application's integration and user's familiarity with the execution environment.

The main goal of this article is to present SAGA, a framework composed by a set of Web Services that provide the features required by cooperative activities, namely the asynchronous notification of events. For this purpose we first discuss some theoretical issues concerning Web Services and CSCW. Then we present the SAGA architecture and describe the main implementation issues of the prototypes of SAGA's services and of an innovative cooperative video editing application, COVIEW, developed to validate the SAGA architecture. Finally, we present some experimental results concerning the use of SAGA and COVIEW.

2 Web Services

The concept of service is frequently associated with the idea of an application accessible through interfaces, which tell us how to use the operations they provide. These applications are usually referred as applications with a Service Oriented

Architecture (SOA).[14, 15] The Web has a utilization paradigm and a protocol set for communications and data representation that were easily accepted and are widely supported. The ability to build services accessible using Web protocols is very attractive for a significant portion of the software industry and the international scientific community. The convergence of SOA and Web protocols, under the guidance of the World Wide Web Consortium (W3C),[16] produced the technology currently known as Web Services.[15, 17] A simple definition for a Web Service is that of an application that is accessible through an interface, using common Web protocols, such as HTTP,[18] and use data representations that follow the *de facto* standard XML.[19]. As a component, a Web Service represents a functionality that can be reused without knowledge of its implementation details.

The use of protocols and data representations widely adopted gives Web Services a highly appreciated and desirable feature, that other distributed processing architectures had difficulties to efficiently achieve: interoperability. Indeed, the use of Web protocols for communication provides platform independence and the use of XML for data representation provides independence at the programming language level. The latter also has the ability to transform legacy applications into services accessible through Web Servers, facilitating the interaction among systems. This feature gives organizations the ability to increase the profitability of their investments in information systems and to expand business opportunities. Thus, the trend will be for any kind of application to be offered as a Web Service and become accessible anywhere.

Web Services use XML to describe their service interfaces and to encode the messages exchanged in the invocations. The description of the interfaces is contained in a file using the Web Services Description Language (WSDL).[20] That description contains information regarding the available operations, the data types manipulated, the format of the exchanged messages, the protocols that are supported and at least one access point (an address), known as a Uniform Resource Identifier (URI). The messages exchanged in the invocations use a packet format and a data encoding

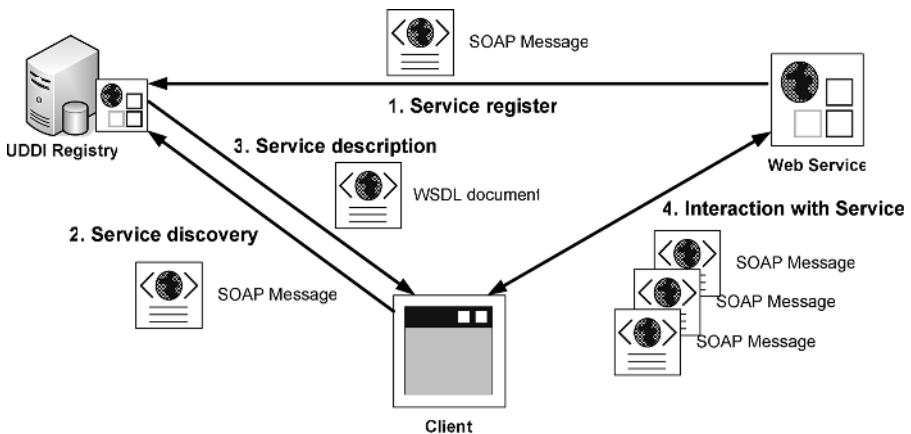


Fig. 1. Web Services based system architecture

mechanism defined by the Simple Object Access Protocol (SOAP).[21] Beside the message exchange model, SOAP formalizes a remote procedure call model. Another important feature of the Web Services is the availability of a service, Universal Description Discovery and Integration (UDDI),[22] that enables other services to register and publish their interfaces, making it possible for a user to discover and know how to use them. The classification of services is also possible, facilitating the processes of discovery and utilization. For each Web Service registered, UDDI stores its name, its operations and its access point, i.e., the information contained in the WSDL description. The description of a service interface can be used to build the service's client applications. Figure 1 shows the generic architecture of a system based in web services and the sequence of activities since a service registers until it is used by a client.

3 CSCW

The scientific field known as CSCW [23, 24] investigates how team work can be supported by information and communications technologies, in order to improve the performance of a group of persons involved in the execution of common or inter-related tasks. CSCW is an inter-disciplinary scientific domain, involving the scientific areas of distributed systems, multimedia communication, telecommunications, information science and socio-organizational theory. The impact of the utilization of CSCW applications (usually referred as groupware) is not always positive, being very important the consideration of socio-professional issues. Indeed, the utilization of CSCW applications can modify substantially work practices or dissolve organizational aspects of the team, which can bring negative consequences to their adoption. As far as possible, it is recommendable to use methodologies that enable the understanding of the way people usually work or that enable the discovery of ways to improve it. Some examples of successful groupware applications are workflow applications, like IBM Lotus Notes[25] and Microsoft Exchange.[26]

An important issue in groupware is the existence of an environment that is shared among team members. This environment may include documents, shared whiteboards and shared pointers, among others. Groupware applications must have mechanisms to distribute cooperative events produced by members in the shared environment. Usually, the shared environment coexists with the private environments of each member, imposing the availability of diverse management mechanisms to control information access. To ensure consistency of the data being shared, special care must be taken regarding concurrency control, implementing mechanisms like atomic transactions, locks, versioning, token passing or voting systems. Potential targets for groupware are software project and engineering teams, coordination of work processes in large organizations, distance learning, telemedicine and cooperative editing.

4 SAGA Architecture

The diversity and complexity of work methodologies in organizations is increasing and the participants' responsibilities are not always statically defined. Hence, the execution of tasks by several persons, which may not always play the same role in the

execution of a certain type of tasks, is relatively frequent. Furthermore, during the execution of a task, one person may want to consult others, regarding specific issues or to obtain approval from upper levels of the organization.

Usually, groupware tries to support team work in the most successful way, providing means for information sharing, for its joint manipulation and for communication between cooperating participants. However, the architecture of cooperative applications is based often on proprietary solutions, which do not address issues of flexibility, interoperability and support of legacy systems.

The Internet is now present in nearly all organizations, namely through popular applications such as e-mail and web browsers. Java technology enables the construction of applications that make the most of this situation, but does not satisfy several security and privacy issues, neither the interoperability between diverse systems, written in different programming languages. Developing cooperative applications based on Web Services can be an attractive choice, enabling applications to take advantage of the potential of the technology regarding distribution issues, such as interoperability, security and legacy systems reuse. Furthermore, the Web Services technology is supported by the major actors in the software industry and academic community. Usually, Web Services are used mainly in business-to-business applications. In the CSCW domain, there is some work carried out concerning the Workflow Management field, under the siege of the Workflow Management Coalition (WfMC).[27] But the features of Web Services make them suitable to support other classes of cooperative applications. This led us to define a model for building cooperative applications based on Web Services, which we designated SAGA – web Services Architecture for Groupware Applications.

The main goal of SAGA is to constitute a framework that enables the development of cooperative applications through the composition of several core functionalities available through Web Services. These must provide a set of operations suitable to fit the requirements of every class of cooperative applications.

Because the tasks assigned to each participant in a cooperative session may vary more or less frequently, it is desirable to have the ability to download applications as needed and execute them immediately. This feature provides the system with a high degree of flexibility and the users with the latest version of the required application. Figure 2 depicts SAGA's overall architecture, showing the generic services that provide support for several kinds of cooperative applications and their interactions with the client-side components.

The Group Storage module is a database that stores the applications and information resources that will be manipulated by cooperative users. Local Storage is used to store the resources downloaded from the server and the resources produced by the local user. The resources information contained in the Group Storage is stored in object databases, which manage data such as the name of the resource, its type, its version, a brief description and some keywords to ease queries. For information resources the identification of its creator is also stored.

The Client module refers to a simple application that serves as an access point to the system and corresponds conceptually to the client of both the Authentication Service and the Applications Repository Service. Indeed, clients of Web Services

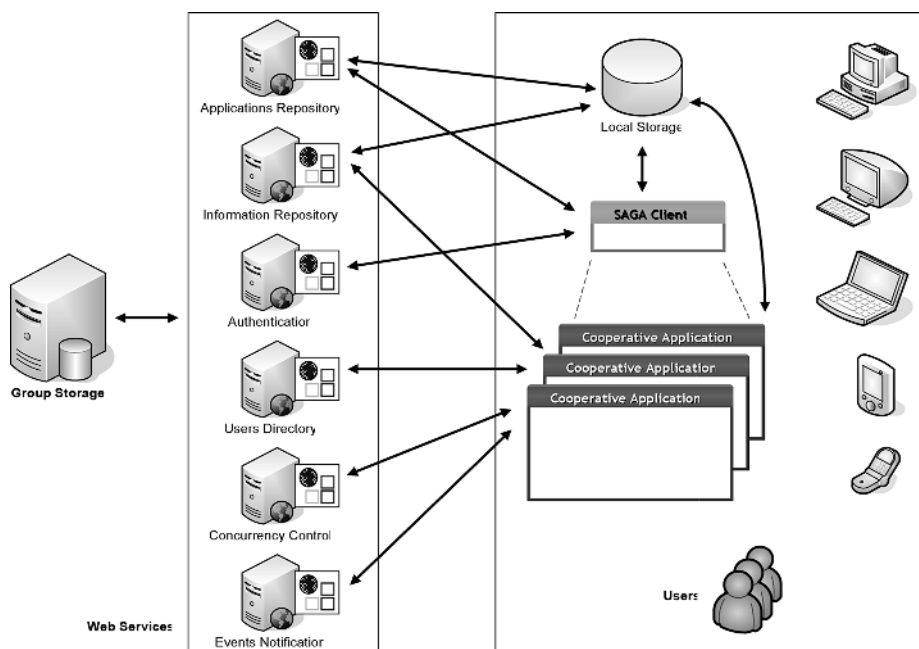


Fig. 2. SAGA architecture

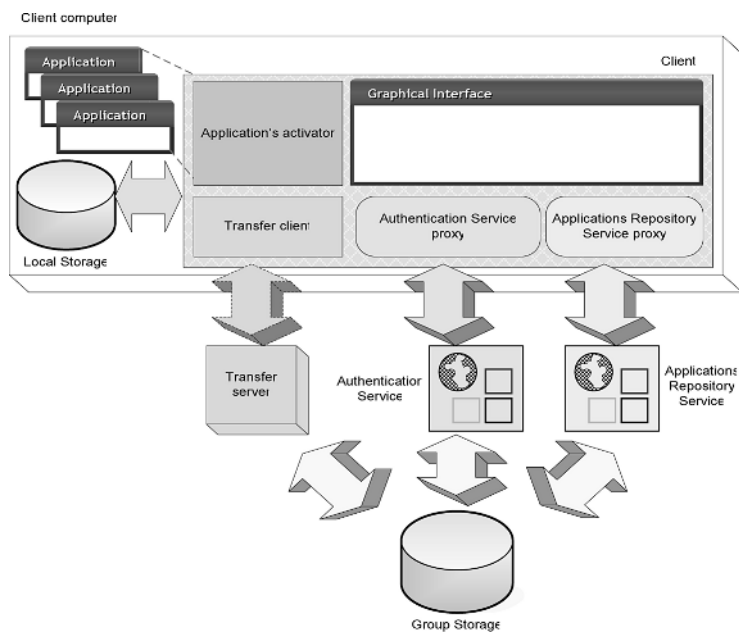


Fig. 3. Client architecture

interact with service proxies instead of interacting directly with service instances. These proxies run on the client computer and intermediate communications between clients and services. Hence, the Client module in the SAGA architecture uses proxies for the Authentication and Applications Repository services. Figure 3 shows the block diagram of the Client module.

Cooperative activities usually incorporate at least one of the following functionalities: communication among team elements, information sharing and joint visualization of activities or work environments. For the latter two it may be highly relevant to ensure exclusive access to shared resources, using concurrency control mechanisms. These mechanisms include free manipulation of resources by several simultaneous processes and restriction to a few or even to a single process (e.g. locks). For this purpose, the SAGA architecture contemplates a Concurrency Control Service. This service, whose architecture is shown in Figure 4, has methods to obtain and release locks and tokens, to assign version numbers and to manage voting systems.

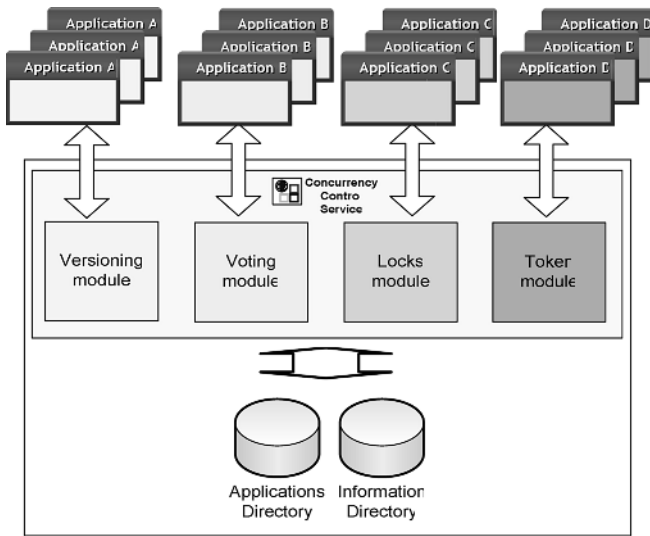


Fig. 4. Concurrency Control Service’s architecture

The user, after his authentication, visualizes the list of the applications available in the Applications Repository Service and may then select the ones he wants, download and execute them immediately. These applications may, in runtime, use the Information Repository Service to access the information resources and download the needed ones. When a user wants to cooperate with another one he accesses the Users Directory Service to see the list of available potential partners. Then, the user sends an invitation to the potential partner, which is informed of that fact and has the choice to accept or reject the invitation. Upon acceptance, users register their interest in being notified of cooperative events and can start a cooperative work session. The invitation process and the notification of cooperative events are intermediated by the

Events Notification Service. This service is an interface to the functionalities provided by an events notification system, allowing the adoption of the most convenient one (e.g.. an organization can have a legacy system that it wants to reuse) and its architecture is shown in Figure 5.

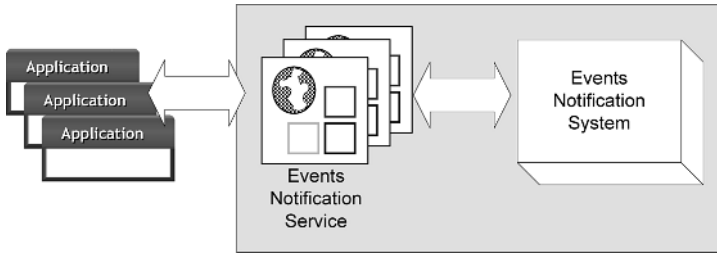


Fig. 5. Events Notification Service architecture

The Events Notification Service interface provides operations to register interest in receiving cooperative events and to remove that interest, as well as to notify of the events produced during a cooperative session. For each user joining a cooperative session an instance of the Events Notification Service is created. This instance is registered in the Events Notification System as an event consumer. Each cooperative event fires the invocation of the notification operation, which delivers it to the Events Notification System, which in turn will propagate it to all registered consumers. These notifications are asynchronous to avoid the need for applications to be web services with public operations. All cooperative events are distributed as strings, making the system suitable for any kind of event produced by any kind of cooperative application.

Since SAGA adopts generic architectural solutions, it has features that make it suitable for a vast set of cooperative applications, namely:

- the features provided by Web Services offer a high degree of interoperability and the ability to integrate legacy systems;
- the Events Notification Service enables the creation of multiple cooperative sessions and the propagation of any type of events;
- the Concurrency Control Service provides diverse mechanisms that enable the adoption of the concurrency control policy that best suites the application needs.

5 Implementation Issues

To validate SAGA we implemented prototypes of the services described above, as well as a prototype application to test them. It was not our intention to have an implementation covering the whole system, but to introduce some simplifications that would make the implementation feasible and would still address the main features required to validate the SAGA architecture. All services and applications were developed in Java, since it is an object oriented language with good semantics and syntax capabilities and it has a set of frameworks in areas crucial for our purposes,

namely those related with events notification (JSDT – Java Shared Data Toolkit),[28] media playback (JMF – Java Media Framework) [29] and object database management (JDO – Java Data Objects).[30] For the Web Services infrastructure we chose Systinet WASP Developer,[31] because it has a mature implementation, a vast programmers’ community and can be integrated with major Java development environments, such as Sun Microsystems NetBeans [32] and IBM Eclipse [33], enabling automatic generation of several useful code fragments. Fig 6 shows the generic class diagram of the SAGA prototype implementation we made, where we can see:

- the client applications – validator, COVIEW (cooperative video editing tool), InformationList, ApplicationsList;
- the proxies to the web services – Authentication, EventsNotification, InformationRepository, ApplicationsRepository;
- the web services – authentication, eventsNotification, informationRepository, applicationsRepository;
- management applications – SystemManager, operDBUsers, operDBResources;
- databases’ classes – User, Client, Manager, Resource, Information, Application.

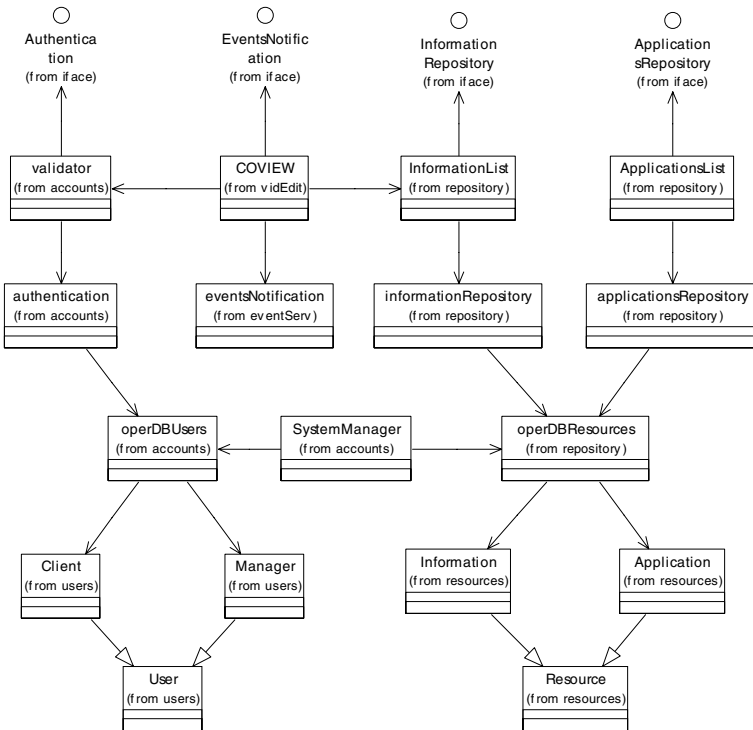


Fig. 6. SAGA generic class diagram

To manage the data required by services and applications we have two main object databases: one for storing users' information and other to store both information and application resources. The information stored in the users' database is used in the authentication process. The resources database stores resource objects that can be either applications or multimedia information resources, as well as the associated metadata. There are 2 types of users: the common user (*Client*) and the system manager (*Manager*). Both have specific attributes and operations and inherit others from the superclass *User*. A similar situation occurs for the resources database, which has a superclass (*Resource*) and 2 subclasses, *Information* and *Application*, as can be seen in Figure 7. The repositories provide operations to list, add, remove and search for resources in the databases. Only a manager can remove resources (of both types) from the database or add *Application* resources to it. Informational resources can be added by any registered user. The prototype implementation of SAGA contemplates search operations by name and keywords, which are performed using Object Query Language (OQL) [34] statements.

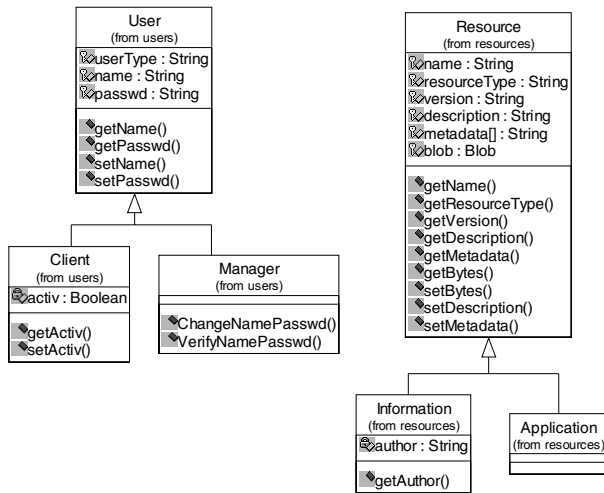


Fig. 7. Databases class diagram

A management application allows system managers to create, destroy or change application and informational resources, as well as user accounts. Its graphical interface is shown in Figure 8.

In Figure 9 we can see the window that is shown when a user wants to add a new resource to the database. This window is similar for both applications and informational resources. The “Type” field does not refer to the type of resource in the sense of being an application or an informational resource. This field is manually inserted by the user with expressions like “MPEG”, “DivX” or “MP3”, helping further resources’ identification. Its value is stored in the *resourceType* attribute of the database class *Resource*.

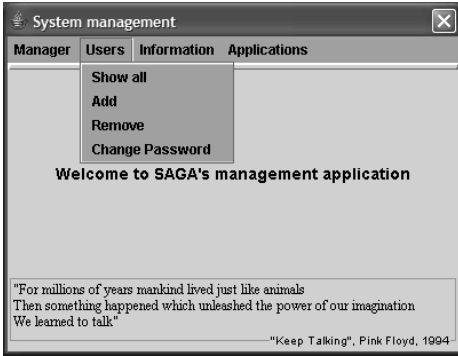


Fig. 8. System management window

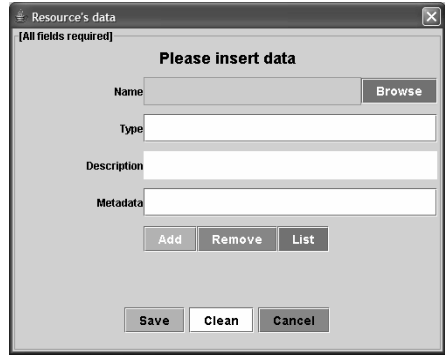


Fig. 9. Adding new resources

Applications can be downloaded as JAR files (because graphical interfaces usually produce several class files) and instantly activated in the client side, using a class loader for this purpose. Figure 10 shows the window containing the list of informational resources available to the users. Clicking the resource entry and then the “OK” button opens a window showing the characteristics of the selected resource. This window is similar to the one of Figure 9, with 2 extra fields: version number and author’s name. A similar window is shown for the list of application resources (without author’s name).



Fig. 10. List of informational resources

The Concurrency Control and the Events Notification services are particularly important for cooperative activities. The prototype of the Concurrency Control Service implements a versioning mechanism, since it was sufficient for the application we chose to validate SAGA. Moreover, we were particularly interested in testing the performance of the system and the Events Notification Service is crucial to this issue. This service follows a publish-subscribe model, where applications (event producers) publish their interest in propagating their events and other applications manifest their intention (subscribe) to receive these events. When an application starts its execution, it registers as a consumer of system events, so that its user may be invited for a cooperative session. Upon acceptance, it is also registered as a consumer of the specific events of that session. The registration of one application creates an instance of the Events Notification Service that is used to asynchronously notify it of

the events it is interested in. This instance mediates the registration and notification processes between the applications and the events notification system (supported by JSDT). Each user may be engaged with several other users in the same cooperative session or in separate ones. Indeed, a user A may have a cooperative session with user B and another cooperative session with users C and D, without mixing cooperative events from distinct sessions.

Figure 11 shows the main classes involved in the event notification process, namely the Events Notification Service (`eventServ`) and its client-side proxy (`I_Event`), some JSDT classes and interfaces that help to implement the notification mechanism, the test application (`COVIEW`) and the classes related with the asynchronous notification feature of the Web Services infrastructure (`GenericAsyncCallback` and `AsyncConversation`). Beside the operations to register in sessions, to quit them, to invite/accept cooperation and to send/receive events, there is an operation that shows the list of active users. This operation is used to search for potential partners to invite for cooperation. It corresponds to the Users Directory Service, which we choose not to implement separately, but to integrate it with the Events Notification Service, because their functionality is closely related.

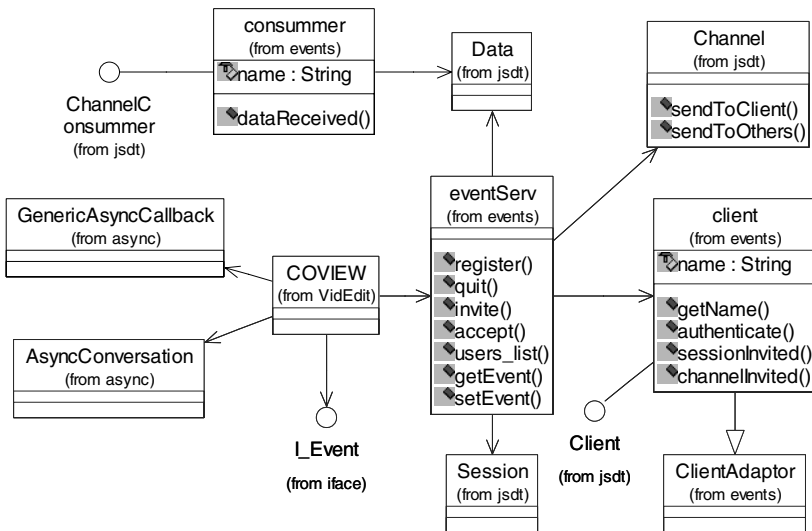


Fig. 11. Events Notification's class diagram

6 Cooperative Video Editing Application - COVIEW

To test the behavior and performance of SAGA with an innovative application, we built a prototype of a cooperative video editing tool, COVIEW (COoperative Video Editing on the Web). This prototype is a simplified application in terms of the functionality it offers, because our aim was mainly to check if cooperative video

editing is viable and if SAGA can support it. Therefore, COVIEW only implements the definition of editing points (IN and OUT) and the reproduction of both the original clips and the clips produced as a result of the editing process. These operations fire events to be propagated to cooperators, which can then properly manipulate them. The implementation of a complete cooperative video editing tool is a task quite complex and would require the involvement of a large team. Indeed, the choice of events to be propagated and what to do with the events that are received is a subject that deserves appropriate attention by a team with professional expertise in the field of video production and also with social sciences background, which is not our case. Nevertheless, for the purposes exposed above we believe it was sufficient.

Fig. shows the main window of the COVIEW application, where we can observe the various menus and buttons available and a video clip being played. The manipulation of video clips is achieved with the help of the JMF framework. Cooperation can be started by choosing the corresponding entry in the “Cooperation” menu. This action fires the invitation process described previously. Fig. shows the window that is displayed when a user is invited to join a cooperative session and the one used to notify the reception of a cooperative event when the user is already in cooperation mode. Similar windows exist for the acceptance and rejection of an invitation and for notifying that someone has left the cooperative session.



Fig. 12. COVIEW: main window



Fig. 13. Windows shown for invitation and for event notification

7 Experimental Results

The operation of the system was subject to both qualitative evaluation and quantitative measurements. The qualitative evaluation of the system being used by several cooperators was very positive since the system performed quite well regarding response time and did not show any errors or locks related with cooperative activities. We also made some measurements to register the delay introduced by the propagation of cooperative events. These measurements were carried out building small applications that simulate the production of cooperative events at predefined time intervals. The results of this performance test are summarized in Table 1, where it can be observed that the system performed quite well in almost all situations, except for the case of a production of events at time intervals of only 0.1s, which is an unlikely situation in most utilization scenarios. Furthermore, for almost all situations the average delay was less than 100 ms and the minimum delay was below 1ms and the test applications were not able to record it.

Table 1. Results of the performance tests

| | Δt - time interval between consecutive events (s) | | | | | |
|----------------|---|-------|-------|-------|-------|-------|
| | 0.1 | 0.5 | 1 | 2 | 5 | 10 |
| Average | 49.498 | 0.326 | 0.073 | 0.059 | 0.055 | 0.061 |
| Maximum | 111.701 | 2.433 | 0.320 | 0.281 | 0.241 | 0.331 |
| Minimum | 1.713 | 0 | 0 | 0 | 0 | 0 |

Figure 14 shows a chart with the performance results produced for a time interval of 1 s between consecutive cooperative events. Other time intervals, except for 0.1s, exhibit a similar chart.

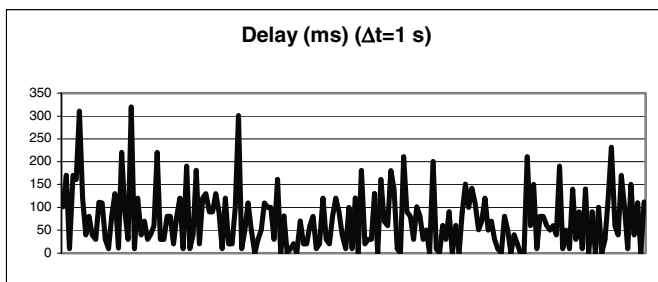


Fig. 14. Performance results for a time interval of 1 s

8 Conclusion

This paper described an architectural model for a framework that enables the creation of several kinds of cooperative applications. The architecture, which we called

SAGA, uses web services to provide a set of core functionalities that can be composed to build the cooperative applications. To validate the architecture we built prototypes of the specified web services and also of an innovative cooperative video editing tool, COVIEW. Web services have features that make them suitable to support cooperative applications, namely those related with interoperability and the fact that the message-oriented approach is adequate to the exchange of event notifications.

SAGA is an open, distributed, interoperable, modular and evolutionary architecture that proved to be viable, allowing the interaction between applications and support services as well as the exchange of events produced during cooperative sessions. The prototype services that were built based on the SAGA architecture performed robustly and their architectural solutions are generic enough to allow their usage in diverse cooperative application scenarios. COVIEW has capabilities that can turn it into a powerful tool since remote and collaborative video editing is an activity that could be envisaged in many situations where a reporter sends the raw material to the TV headquarter but wants to be involved in the final editing.

We are planning several improvements of the services and applications described in this article, namely the addition of metadata to the resources stored in databases and a more accurate specification of COVIEW requirements, resulting from the inclusion of multidisciplinary teams (with sociologists and audiovisual professionals) in the development process and the testing of the final product in real world situations.

References

1. Lee, J.H., et al. *Supporting multi-user, multi-applet workspaces in CBE*. in *CSCW'96*. 1996. Boston, USA: ACM Press.
2. Trevor, J., T. Koch, and G. Woetzel. *MetaWeb: Bringing synchronous groupware to the World Wide Web*. in *ECSCW'97*. 1997. Lancaster, UK: Kluwer Academic Publishers.
3. Kindberg, T. *Mushroom: a framework for collaboration and interaction across the Internet*. in *ERCIM W4G workshop on CSCW and the Web*. 1996. Sankt Augustin, Germany: ERCIM/W4G.
4. Munson, J. *Collaboration Bus Infrastructure: Bus Agents*. Available from: <http://www.cs.unc.edu/~munson/DARPA/busagent.html>.
5. Chabert, A., et al., *Java object-sharing in Habanero*. *Communications of the ACM*, 1998. **41**(6): p. 69-76.
6. Guicking, A., P. Tandler, and P. Avgeriou, *Agilo: A Highly Flexible Groupware Framework in Groupware: Design, Implementation, and Use: 11th International Workshop, CRIWG 2005, Porto de Galinhas, Brazil, September 25-29, 2005. Proceedings* H. Fuks, S. Lukosch, and A.C. Salgado, Editors. 2005, Springer-Verlag: Berlin. p. 49.
7. Orozco, P., et al., *A Decoupled Architecture for Action-Oriented Coordination and Awareness Management in CSCL/W Frameworks*, in *Groupware: Design, Implementation and Use: 10th International Workshop, CRIWG 2004, San Carlos, Costa Rica, September 5-9, 2004. Proceedings* G.-J.d. Vreede, L.A. Guerrero, and G.M. Raventós, Editors. 2004, Springer-Verlag: Berlin. p. 246.
8. Preguiça, N., et al., *Integrating Synchronous and Asynchronous Interactions in Groupware Applications* in *Groupware: Design, Implementation, and Use: 11th International Workshop, CRIWG 2005, Porto de Galinhas, Brazil, September 25-29, 2005. Proceedings* H. Fuks, S. Lukosch, and A.C. Salgado, Editors. 2005, Springer-Verlag: Berlin. p. 89.

9. García, P. and A. Gómez-Skarmeta, *ANTS Framework for Cooperative Work Environment*, in *IEEE Computer*. 2003. p. 56-62.
10. Brandenburg, J., et al. *Artefact: A Framework for Low-Overhead Web-Based Collaborative Systems*. in *CSCW 98*. 1998. Seattle, USA: ACM.
11. Roseman, M. and S. Greenberg. *GroupKit A Groupware Toolkit for Building Real-Time Conferencing Applications*. in *CSCW'92*. 1992. Toronto, Canada: ACM Press.
12. Schuckmann, C., et al. *Designing object-oriented synchronous groupware with COAST*. in *CSCW'96*. 1996. Boston, USA: ACM Press.
13. Vinoski, S., *Distributed Object Computing with CORBA*, in *C++ Report Magazine*. 1993.
14. Associates, B. *Service-oriented architecture (SOA) definition 2005*; Available from: http://www.service-architecture.com/web-services/articles/service-oriented_architecture_soa_definition.html.
15. W3C. *Web Services Architecture*. 2004; Available from: <http://www.w3.org/TR/ws-arch/>.
16. W3C. *W3C*. 2004; Available from: <http://www.w3c.org/>.
17. Newcomer, E., *Understanding Web Services: XML, WSDL, SOAP, and UDDI*. 1st edition ed. Independent Technology Guides, ed. D. Chappell. 2002, Boston: Addison-Wesley Professional.
18. W3C. *HTTP*. 1999; Available from: <http://www.w3.org/Protocols/>.
19. W3C. *XML*. 2004; Available from: <http://www.xml.org/>.
20. W3C. *WSDL*. 2001; Available from: <http://www.w3.org/TR/wsdl>.
21. W3C. *SOAP*. 2003; Available from: <http://www.w3.org/TR/soap/>.
22. OASIS. *UDDI*. 2004; Available from: <http://www.uddi.org/>.
23. Borghoff, U.M. and J.H. Schlichter, *Computer-Supported Cooperative Work*. 1998: Springer-Verlag.
24. Beaudouin-Lafon, M., et al., *Computer Supported Co-operative Work*. Trends in Software, ed. B. Krishnamurthy. 1999; John Wiley & Sons.
25. IBM. *Lotus Notes*. 2004; Available from: <http://www.lotus.com/>.
26. Microsoft. *Microsoft Exchange*. 2003; Available from: <http://www.microsoft.com/exchange/default.asp>.
27. W3C. *WfMC*. 2004; Available from: <http://www.wfmc.org/>.
28. Sun. *Java Shared Data Toolkit*. 2002; Available from: <http://java.sun.com/products/java-media/jsdt/>.
29. Sun. *Java Media Framework*. 2002; Available from: <http://java.sun.com/products/java-media/jmf/>.
30. Sun. *Java Data Objects (JDO)*. 2004; Available from: <http://java.sun.com/products/jdo/>.
31. Systinet. *Systinet WASP*. 2003; Available from: <http://www.systinet.com/>.
32. Sun. *NetBeans*. 2003; Available from: <http://www.netbeans.org/>.
33. IBM. *Eclipse*. 2003; Available from: <http://www.eclipse.org/>.
34. ODMG. *ODMG OQL User Manual*. [PDF] 2004; Available from: <http://www.odmg.org/oqlg.zip>.

Towards a P2P-Based Active e-Learning Space*

Xianghua Xu and Jian Wan

Institute of Software and Intelligent technology
Hangzhou Dianzi University, Hangzhou, P. R. China, 310018
xianghuaxu@yahoo.com.cn, wanjian@hzieee.edu.cn

Abstract. In this paper, an active and autonomous e-learning system--Active e-Learning Space (ALS) is presented. ALS is a P2P-based learning environment that supporting dynamic construction of hierarchical and self-managed learning community. In ALS, student can construct or join a learning community according to the learning requirement, learning collaboratively with others. ALS is composed of three parts: (1) ALS server accomplishes e-learning information management and services; (2) Participating sites constitute a pastry-based p2p overlay network supporting message multi-casting and uni-casting, and resources sharing; (3) ALS e-Learning application, which is running on the top of the p2p network, realizes an active learning community.

Keywords: active e-learning space; e-learning; p2p computing; CSCL.

1 Introduction

With the advance of information technology, significant changes have been taken place in the ways that people acquire and disseminate knowledge. CSCL, as a new learning medium has been widely developed [1]. However, many E-learning applications are in nature a learning content publish/subscribe system, which focuses on content dissemination, and just to produce more and more learning contents for the interested students[2]. In such e-Learning environment, students are considered mainly as a passive role to consume a great deal of learning materials, and the learning process is mainly controlled under instructor[3].

Obviously, the deficiency of such e-Learning schema can not reach a promising pedagogic effect as expected. Education is not an affair of 'telling' and being told, but an active constructive process [4]. In a real learning process, students are playing an active learning role that they are not only a receiver of education, but also a generator of contents of a learning context, and always keep active mutual interaction with teachers and other students.

Therefore, an ideal e-learning environment should supports active learning community: (a) *multiple role*: students are "taught" by others, not only "taught" by teacher, and are educating others in the community as well; (b) *autonomy*: how, where, what of learning is controlled by the students in some extent instead of dictated by a teach; (c) *active*: the learning content is partly controlled by students,

* This paper is supported by research funding from Natural Science Foundation of China (60573176), and Science Foundation of Hangzhou Dianzi University (KYS041505044).

they should have the ability to contribute, modify the content that directly impact their learning context. We call such e-learning environment as *Active Learning Space*.

In this paper, we present a prototype system (called ALS) partially realizing Active Learning Space. The system has two features: firstly, the e-Learning sessions and learning communities are constructed on a Pastry-based P2P network infrastructure [5], and running on the distributed sites; Secondly, learning community or sub-community can be constructed by teacher or student according to their needs in an ongoing learning session.

The rest of this paper is organized as follows. In section 2, we present the system architecture of the environment. Section 3 discussed the design of Active e-Learning Community. In section 4, we introduce the state of art of related works of e-Learning systems. Finally, we draw the conclusions and point out our future research directions on section 5.

2 System Architecture

The ALS is a distributed e-Learning system which is comprised of two categories of node, as shown in Fig.1. First type is an ALS server as a supporting platform on which management modules are running, including user manager, course manager, learning space manager, learning material manager; Second type is a group of peer sites which constitute the ALS e-Learning environment supported by pastry p2p overlay substrate. Teachers and students are participating in e-Learning session from their distributed sites.

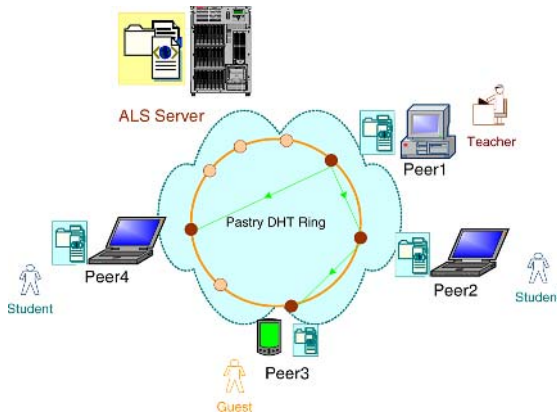


Fig. 1. Active Learning Space system architecture

The ALS server functions are composed of following primary modules:

User Manager: There are three classes of user role: teacher, student, and guest. Each role provides corresponding access rights to the e-Learning resources, such as course files, learning materials and course session.

Course Manager: Course information, such as course list, course schedule, course-student relationship and course-teacher relationship, are maintained by Course manager.

Active Learning Space Manager: This module is responsible for maintaining the information of hierarchical community’s structure and its member node list, addressing the community management operations, such as create/destroy a community, join/leave a community, and query community info.

Users enter into the ALS e-Learning environment from their dispersed peer sites. These sites constitute a fully decentralized p2p network. As shown in Fig.2, the pastry substrate layer constitutes the peers’ infrastructure, realizes multi-casting and unicasting of e-Learning message among the peers, and supports distributed learning material sharing. The application layer, which realizing the actual Active e-Learning Space, is composed of topic manager, and collaboration tools, such as whiteboard, chat, file sharing and courseware sharing.

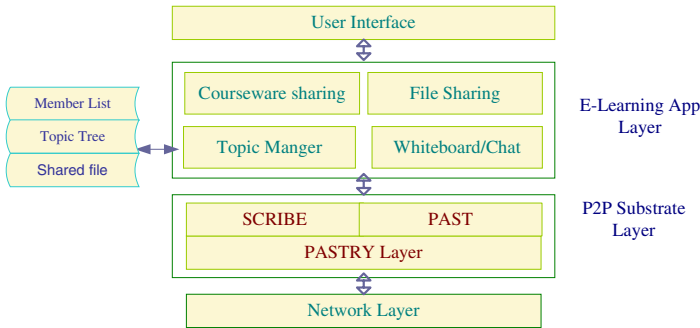


Fig. 2. Peer site architecture

At the peer site, ALS e-Learning application is building on the pastry p2p substrate. It supports a scalable, distributed object location and routing substrate for p2p applications. SCRIBE serves a scalable message publish/subscribe support layer for above application layer. PAST serves a distributed file sharing support layer for shared document manager.

The application layer is composed of four application modules, including file sharing, topic manager, community manager and whiteboard/chat. File sharing maintains a distributed file-sharing pool. The shared file directory provides a global view among buddy members in the active community. For each learning topic session of a topic, community manager maintains a community’s member list. Topic manager maintains current topic list and topic tree to log topics, sub-topic. In an active community, teacher and student member can create a new topic or sub-topic as his needs in the learning process.

3 Implementation of Active e-Learning Space

3.1 Hierarchical Community Structure

In ALS, users of online learning are dynamically divided into different group (called as community) according to there studying topic. There are two types of community

that can be constructed: course community and topic community. Course community is a virtual classroom of the course teaching and learning in ALS learning environment. Topic community is a virtual discussing board beyond the class in ALS learning environment. Community member can create a sub-community at any time as needed in course of studying or discussing. The community manager is responsible for community maintenance issues, such as create/destroy community/sub-community, member join/leave. As shown in Fig.3, each site maintains a hierarchical community structure. Each structure node represents a community or sub-community. Community maintains its course/topic, member list, root node ID (owner). ALS provides several collaborative tools to user, such as chat, whiteboard, file sharing tool, and courseware broadcasting tool. Users of same community can study collaboratively by means of these tools. For example, users in same community can communicate message through chat/whiteboard tool, share file through file sharing tool. By using courseware broadcasting tool, users can also view PPT/PDF slides broadcasted synchronously from a member of same community, and make real time communication by means of collaborative marking.

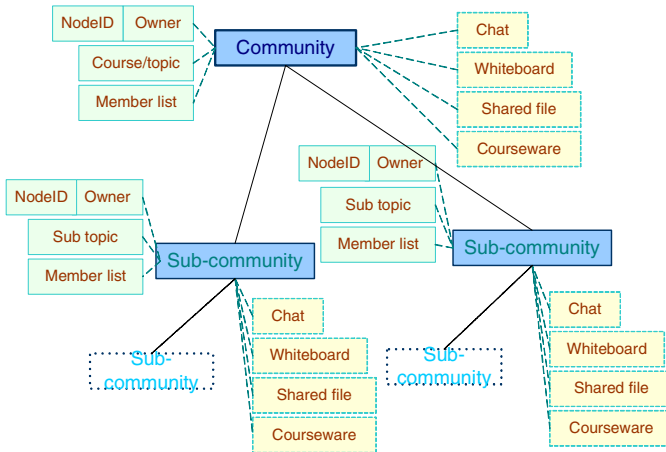


Fig. 3. Community structure

3.2 Dynamic Community Construction

In the ALS runtime environment, students will dynamically form multiple learning communities and sub-communities in course of learning session according to their learning topic.

The course community is created by the corresponding instructor of the course's topic. Community's members are composed of students that have selecting the course. Students can attend the course community–virtual class, and become a member of the class. In the process of course learning, members can create sub-community as needed. For instance, a teacher creates a sub-community, relating to the questions, outline, concepts, exercises, chapter/section, in the class according to his teaching needs. Then, he can invite students to participate in the sub-community, discussing

around the topic, or answering questions relating to the topic, in order to strengthening students’ understanding of the topic. Members are attending the topic sub-community according their interesting freely. In a community or sub community, member sites maintain a multicast tree for messaging among members.

Fig.4 shows a scenario of community composition in an ALS runtime environment: there is “Grid computing” community, “P2P” community, and three sub-community (“Grid introduction”, “Grid scheduling” and “P2P File sharing”). John and Mike are current in same sub-community -- “P2P file sharing”.

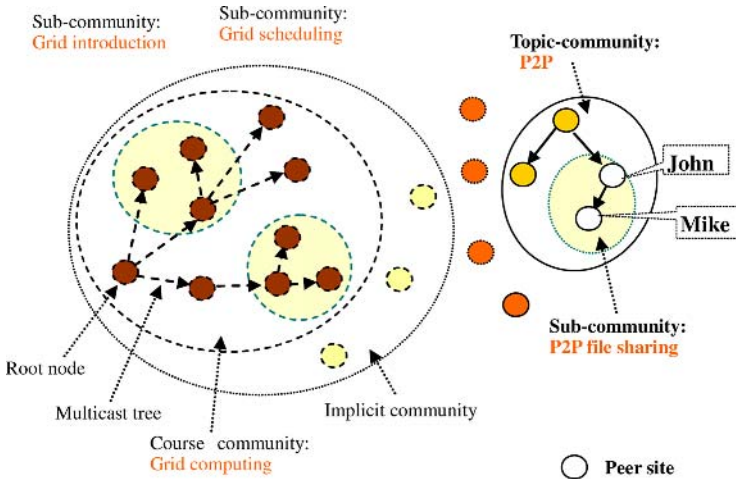


Fig. 4. An example of community and sub-community at ALS running time

3.3 Characteristic of Active Community

Autonomy in ALS community

Autonomy refers to the fact that the how, what, and why of learning is controlled by the learner instead of dictated by a teacher or a school. Firstly, community is created, controlled and administrated by its members. A topic community or sub-community can be constructed by any user at any time in the ALS learning context. Furthermore, user can join/quit any community freely in the ALS e-learning environment. Finally, user can share a document, discussing a question, sharing an application with other user freely according to their study objective, personal interests and preference in ALS.

Static role and member relationship

In current implementation, there are three categories of role supported in ALS: teacher, student and guest. ALS assigns a role to user according to the corresponding responsibility in e-learning environment at the member initialization phase. Member’ role reflects static relationship with other members in ALS, for example teaching-studying relation between teacher and student, classmate relation between two students, colleague relation between teachers. In a course community, the members’ role is predefined.

Dynamic implicit role and member relationship

Besides static role and static relationship in course community, ALS can define dynamic role and member relationship in topic community or topic sub-community. Topic community is constructed dynamically according to teacher's teaching objective or student's interests. Hence, topic community defines a dynamic learning community, and reflects a dynamic role relationship among their members. Users can create/join/quit a community dynamically according to his interest in current context. A topic community implicitly maintains dynamic member and role relationship, such as questioning-answering relation, discussion relation, learning material sharing relation and so on, according to members' interests, experience, knowledge background, emotion and etc.

Implicit member

Implicit member refers to the member, who had taught or studied a course. ALS sites provide a searching tool which help user to find potential implicit members and their Pastry node-IDs by searching information of history topic or history course preserved in the ALS server. Then the Pastry substrate will help user notifying and inviting them to attending current community if they are online. If the potential member accepts the invitation to joining, then he becomes a temporary member, and participates in discussion with others instantly.

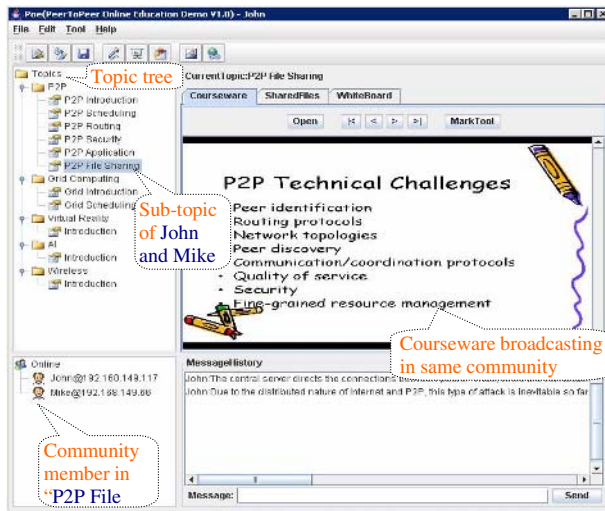


Fig. 5. A runtime snapshot of the prototype system

3.4 The Prototype System

The demo prototype system is developed to demonstrating the active learning space. Currently, functions, such as topic/community management, courseware broadcasting, file sharing, chat/whiteboard, are implemented. Users can create topic, invite members to join topic, synchronously broadcast courseware to members. Fig. 5 shows a runtime snapshot of the prototype system. In the system, there are several topics that

user can participate in, such as “P2P”, “Grid computing”, “Virtual reality”, “AI” and “Wireless”. In “P2P”, there are six sub-topics, such as “P2P Introduction”, “P2P File sharing”, etc. John and Mike are currently within “P2P File sharing”, chatting and sharing PPT slides about P2P technology.

4 Related Works

In recent years, P2P technology has been used in many projects to supporting e-Learning and collaborative application. Edutella [6] is a RDF-based framework for representing information in the P2P network environment. It focuses on exchanging information about learning objects over the P2P network. However, there are no collaborative e-learning functions, and does not intend to support interactive learning activity. ELENA [7] is a service-based architecture for personalized distributed e-Learning environment. It mainly emphasizes on providing the personalized access to learning resources in an e-Learning network. The works of Planet [8] is mainly on designing a distributed content repository which built on a structured p2p grid layer. It provides an alliance of content repositories throughout the p2p grid, and allows researchers to locate content by their own keywords of interest, which are stored and looked up in the decentralized p2p-grid infrastructure.

Gridcole [9] is an OGSA-based CSCL system. The primary work is focused on configurable feature. By employing IMS-LD scripts to describe the sequence of learning activities and required tools, users can configure a suitable environment supporting their own CSCL scenarios. GlobalEdu [10] intends to develop a large-scale pervasive learning environment. It aims at supporting learner’s mobility at global scale. The ActiveCampus [11] is a wireless location-aware e-learning environment, and mainly used for supporting location-aware instant messaging and maps of the user’s location annotated with dynamic hyperlinks of nearby buddies. Leite, F.G. [12] presented a PDA-based mobile learning architecture based on P2P network. His work is mainly addressing the issues of mobile data management, replication and physical failing in the mobile learning system. APPLE [3] builds a virtual classroom environment based on WSRF.NET and Gnutella-like P2P network. The main work is to providing live broadcasting services for peer site in a p2p e-learning environment.

The previous works are mainly focused on two aspect of problem in p2p-based e-learning systems: representing, exchanging, personalized access and distributed repository of learning resources; configurability, mobility, location-awareness, and mobile data management support in e-learning system architecture and design. Our work is mainly focused on designing of active e-learning environment, in which user can collaborates or communicates actively with other member in the process of online studying in a dynamic constructed community.

5 Conclusions and Future Works

In this paper, an active and autonomous e-learning system--Active e-Learning Space (ALS) is presented. ALS is a project that intends to exploring the benefits of active, dynamic and autonomy e-learning community supported by a p2p environment. In

ALS, student plays an active role that he can construct a learning community according to the learning scenario in course of online studying, communicates and collaborates with others. The ALS e-learning system is composed of three parts: (1) ALS server realizing e-learning information management and services; (2) A fully decentralized pastry-based p2p network that is consisted of user sites, realizes multi-casting and uni-casting of e-Learning message, and supports distributed learning material sharing; (3) ALS e-Learning application, that is running on the top of the p2p network, realizes construction and management of active and dynamic learning community, and supports whiteboard, chat and courseware sharing.

Currently, topic/community management, courseware broadcasting, file sharing, and chat/whiteboard tool, have been implemented in the system. We are planning to implement learning material recommending function by means of semantic searching, and research member recommending issue about a user needs to communicate with other member familiar with same topic. Finally, the evaluation of performance and usability of the ALS system is also our future work.

References

1. Adelsberger, H.H., Collis B., and Pawlowski J.M., Handbook on Information Technologies for Education and Training. (2002): Springer-Verlag, Berlin.
2. Brusilovsky, P. and Miller P., Course Delivery Systems for the Virtual University. Access to Knowledge: New Information Technologies and the Emergence of the Virtual University. (2001): Elsevier Science. 167-206.
3. Jin, H., et al., APPLE: A Novel P2P Based e-Learning Environment. Lecture Notes in Computer Science, 3326, 2004: p. 52–62.
4. Thomas, S., Pervasive, persuasive eLearning: modeling the pervasive learning space. Proceedings of the 3rd International Conference on Pervasive Computing and Communications Workshops (PerCom 2005), 2005.
5. Rowstron, A. and Druschel P. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms (Middleware). 2001.
6. Nejdil, W., et al. EDUTELLA: A P2P Networking Infrastructure Based on RDF. In Proceedings of WWW'2002. Honolulu, Hawaii, USA.
7. Dolog, P., et al. Personalization in Distributed e-Learning Environments. In Proceeding of WWW'2004. 2004.
8. Pairot, C., et al. The Planet Project: Collaborative Educational Content Repositories on Structured Peer-to-Peer Grids. In Proceeding of IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2005). 2005.
9. Bote-Lorenzo, M.L., et al. A Tailorable Collaborative Learning System That Combines OGSA Grid Services and IMS-LD Scripting. In Proceedings of International Workshop on Groupware (CRIWG'2004). 2004.
10. Barbosa, D.N.F., et al. Learning in a Large-Scale Pervasive Environment. In Proceedings of the Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW'06). 2006.
11. Griswold, W.G., et al., ActiveCampus - Experiments in Community-Oriented Ubiquitous Computing. IEEE Computer, 2004. 37(10): p. 73-81.
12. Leite, F.G., Ramirez M.R., and Souza J.M.d., Learning Communities Support by Mobile Systems Based on Peer-to-Peer Networks. Proceedings of WISE 2004.

Understanding the Trade-Offs of Blending Collaboration Services in Support of Contextual Collaboration

Roberto S. Silva Filho¹, Werner Geyer², Beth Brownholtz², and David F. Redmiles¹

¹ Department of Informatics
Donald Bren School of Information and
Computer Sciences, University of California
Irvine, CA, 92697 USA

² IBM T.J. Watson Research
One Rogers Street
Cambridge, MA 02142, USA

rsilvafi@ics.uci.edu, werner.geyer@us.ibm.com,
beth_brownholtz@us.ibm.com, redmiles@ics.uci.edu

Abstract. Contextual collaboration seamlessly integrates existing groupware technologies into a uniform user experience that combines synchronous and asynchronous interactions. This user experience is usually supported by a contextual collaboration infrastructure that needs to efficiently cope with the fast switching and integration of different modes of interaction. This paper experiments with a new model for contextual collaboration based on the notion of generic shared objects. We describe a native implementation of this model and evaluate its behavior under different media traffic conditions. We compare the native implementation with an alternative implementation that integrates existing notification and meeting servers to deliver the same model behavior. We discuss trade-offs and limitations of those two implementations.

1 Introduction

Contextual collaboration promises new levels of productivity by seamlessly integrating content sharing, communication channels, and collaboration tools into a unified user experience. One form of contextual collaboration embeds collaborative features, such as presence awareness, instant messaging, real-time conferencing, file exchange, and virtual workspaces into other business applications [10, 14]. For example, through the integration of communication channels and office tools, users can easily switch between individual and collaborative work. Through a single click of a button, they can start a chat from within their document editors, share a document on their desktops by dragging it on their buddy lists, or start a remote presentation by right-clicking on a presentation file on their desktop. Contextual collaboration lowers the end-user's barrier to engage in collaboration by transparently integrating existing groupware technologies. By doing so, it reduces end-users' cognitive cost of switching between collaboration tools and applications, providing contextual points of access to a set of inter-related applications and the artifacts they produce. A highly contextualized user experience entails frequent changes in work mode and modalities. From an infrastructural perspective, this requires the use of different services, for

example, meeting servers to support synchronous collaboration, notification servers to support timely delivery of messages, or document repositories to allow sharing of content.

In this paper, we study a model for contextual collaboration that supports multiple modalities of media collaboration. Our model is based on generic shared objects that provide building blocks for supporting contextual collaboration applications. We present a native implementation of this interaction model and study its behavior under different interaction patterns, representing different kinds of media collaborations. We compare our native service implementation with an alternative integrated implementation where existing services such as meeting and notification servers are used. Our goal is to characterize and understand the trade-offs and limitations that exist in different implementations of services supporting contextual collaboration with respect to the responsiveness of the infrastructure and its ability to support the traffic requirements of different collaboration tools.

This work was motivated by previous research on Activity Explorer (AE) [6, 8]. AE provides a highly contextualized user experience integrating synchronous and asynchronous types of collaboration. AE is built on top of our collaboration model using generic shared objects. Previous works, however, did not analyze the limitations of the model in terms of scalability, support for different media interaction, and the trade-offs involved in building such an infrastructure using existing technologies. Hence, with this work, we expect to understand the applicability of the model to different traffic conditions, and to assess the use of existing services in supporting this blended collaborative model. The lessons learned can be applied to the development or improvement of contextual collaboration infrastructures.

Section 2 of this paper discusses related work. In Section 3 we describe the contextual user experience in AE in more detail. Section 4 introduces the contextual collaboration model used as the basis for our study. Section 5 describes the two implementations of this model. In Section 6 we describe our simulation environment, the experiments performed, and the experimental results comparing both implementations. Section 7 discusses general trade-offs and lessons learned.

2 Related Work

The concept of using shared objects to support collaboration is similar to the Tuple Space work, proposed by Gelernter as part of the Linda coordination language [5]. Tuple Spaces are currently implemented in IBM's TSpaces system [18] and SUN's JavaSpaces [3]. They provide a persistent shared memory accessed through an API that allows distributed processes to read, write, and remove information represented as tuples. Compared to our shared objects, Tuple Spaces are rather a programming paradigm that helps developers with concurrency control and other issues, while we focus on offering a shared object service that can be used to build collaborative applications. As such, membership, notifications, and service-oriented communication are an integral part of our model.

Notification servers, as defined by Patterson et al. [12], provide a simple common service for sharing state in synchronous multi-user applications. They address the problem of maintaining consistency in real-time applications and supporting awareness. Compared to Tuple Spaces and our shared objects, state is usually not persistent.

Publish/subscribe systems are similar to our work since they offer general purpose event notification functionality based on the observer design pattern [4]. Notification servers such as Elvin [2] or YANCEES [16] are usually employed as event routing infrastructure to support the development of awareness applications. Elvin provides a relatively simple but optimized set of functionalities, efficiently processing large quantities of events based on content-based routing of tuple-based events. In such systems, however, event persistency is usually not supported. Moreover, those systems are not usually designed to support synchronous real-time interaction. The insufficiency of the publish/subscribe model in supporting different groupware applications is also discussed in [17] and [9], where new services around this model are proposed to address some of the deficiencies such as the lack of flexibility in the notification model, and support for end-user subscriptions.

The technical aspects of blending of synchronous and asynchronous collaboration have been also addressed in [13] and [8]. Pregoça et al. [13] provide a very good description of the general problem space. Compared to our work, they mainly address consistency control issues.

3 Activity Explorer

Activity Explorer (AE) is a contextual collaboration application based on the paradigm of activity-centric collaboration [7]. AE runs as a stand-alone desktop application that connects to a contextual collaboration server implementing our collaboration model. In AE an activity is a set of related, shared objects representing a task or project. The set of related objects is structured as a hierarchical thread called *activity thread*, representing the context of the task at hand. Users create new activity threads by creating root objects from any type of content or communication. Users add items to an activity thread by posting either a response or a resource addition to its parent object. Activity threads combine different types of objects, membership, and alerts. The context (membership and content of the activity thread) is made persistent through the use of shared objects. AE supports sharing of six types of objects: message, chat transcript, file, folder, annotated screen snapshot, and to-do item.

Fig. 1 shows the main AE user interface. My Activities (A) is a multi column “inbox-like” activity list that supports sorting and filtering of activities and shared objects. Selecting a shared object in this list populates a read-only info pane (B). The Activity Thread pane (C), maps a shared object as a node in a tree representing an entire activity thread. Activity Thread and My Activities are synchronized by object selection. My People (D) is a buddy list showing all members the current user shares activities with. Users interact with objects or members, as displayed in these views, through right-click context menus. Representative icons are highlighted green to cue users of shared object access and member presence (2a, 2b).

The following scenario illustrates a contextual user experience in which shared objects are used in a collaborative context, as part of an activity. The activity starts from a document. The outcome of the activity is shown in Fig. 1.

Bob and Dan are working on a project (a file) using Activity Explorer. Bob right clicks on the file object in his list to add a message asking Dan for his comments (1b). A few hours later, Dan returns to his desktop (2a). In the system tray, Dan is alerted to the new activity. Clicking on the alert, he is taken to the activity thread. He opens

the message and while he is reading it, Bob perceives Dan is looking at the message due to the turning of the object icon to green (2b). Bob then seizes the opportunity to expedite their progress; he right clicks on the initial message and adds a chat to this activity (2c). A chat window pops up on Dan's desktop and they start a chat session (2d). Bob refers to a detail in the project description; for clarity he wants to show Dan what he would like changed. By right clicking on the chat object, Bob creates a shared screen object (3a). A transparent window allows Bob to select and "screen scrape" any region on his desktop. He freezes the transparent window over the project text. The screen shot pops up on Dan's desktop (3b). Bob and Dan begin annotating the web content in real-time like a shared whiteboard (3c).

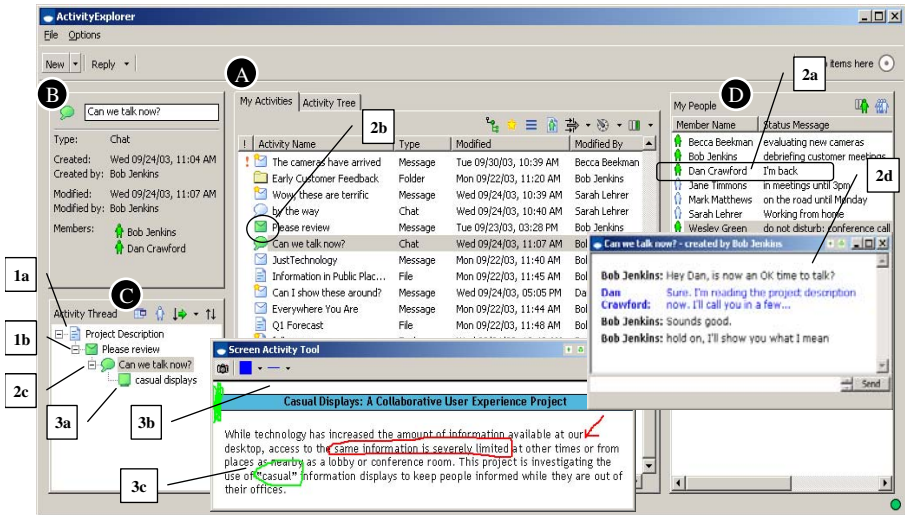


Fig. 1. Activity Explorer User Interface

4 Contextual Collaboration Model

The contextual collaboration model behind AE is based on the concept of *Generic Shared Objects (GSO)* [8]. GSOs are persistent collaboration objects that can be used as building blocks for new collaborative applications that require a seamless, contextual user experience with blended synchronous and asynchronous collaboration. This generic model provides both simplicity and uniformity, allowing the extension of the service to new media types, and the uniform composition of artifacts into hierarchies such as activity threads. GSOs combine various collaborative functions such as group communication, content management, notifications, and membership-based access control policies into objects that can be hierarchically composed.

In this paper, we assume a client/server architecture in which many clients interact with each other through a collaboration server (or service) implementing the concept of GSOs. This architectural style was selected for being currently supported in the AE prototype, as well as in existing technologies such as notification servers and meeting

servers used in our experiments in the integrated implementation described later on in the paper. Note that the GSO model can be also implemented in different architectural styles (e.g. see [8]).

The GSO communication protocol is based on three basic primitives: *Request*, *Response*, and *Notification*: A client interacts with a GSO by issuing a *Request* to that object (for example, reading an attribute, adding a new member, reorganizing the object hierarchy and so on). The object then replies with a *Response* to the requesting client. Depending on the type of request, the object can also send out Notifications to currently online clients as illustrated in Fig. 2 (b).

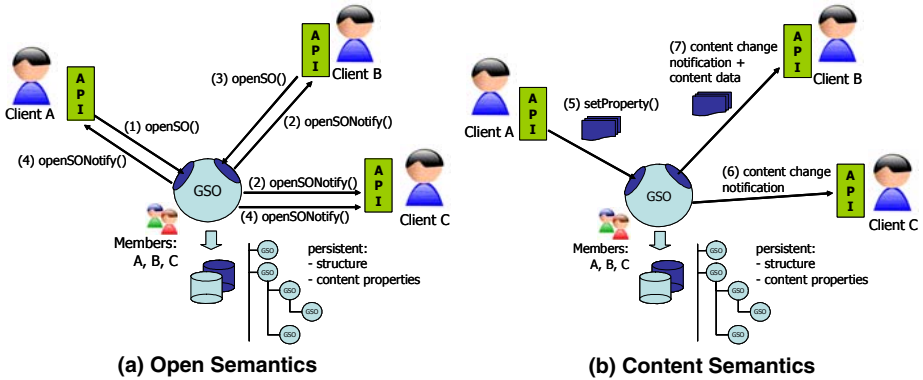


Fig. 2. Generic Shared Object behavior

Our contextual collaboration service manages a collection of GSOs and their relationships, i.e. by containment and/or reference. This facilitates the aggregation of GSOs into hierarchical structures, thus modeling complex collaborations such as the previously mentioned activity threads in AE (see Fig. 1 C).

Each GSO provides a simple content model based on a set of properties. The content model describes what kind of data an object shares and stores, for example, chat transcripts, e-mails, file contents, streaming media and so on; e.g. each Shared Object in AE is represented by a GSO. Jazz [1] and C&BSeen [11] are other examples of applications that use GSOs in a less direct way. Note that a GSO does not provide any means for semantically describing the content. Content is associated with a GSO by adding arbitrary numbers of <name, value> pairs. The interpretation and use of the <name, value> pairs is left to client applications, which provides flexibility to the model. For example, the persistent chat object in AE, stores each chat message as an arbitrary long String property.

Every GSO represents a “persistent conferencing session” between its members. The distribution of content (synchronous or asynchronous) is performed through the use of notifications. Any modification to the set of properties of a GSO is not only stored in the underlying data store, but also automatically sent as notifications to all the other members of that GSO. Hence, our model provides a different paradigm for real-time collaboration based on persistent state and state change notifications.

Each GSO also manages a list of members (e.g. A, B, and C in Fig. 2). The GSO member list controls the access to its content and represents a distribution list for

sending notifications about the creation and modifications of a GSO. The member list is dynamic, allowing the addition and removal of existing members at runtime. Since the member list is also a property of the GSO, any modification to this list, triggers notifications that are sent to all online GSO members.

Notifications of content change come in two different modalities controlled by the use of *open* and *close* requests. Change notifications (without the actual content) are sent to all online members of the object whose open status for that object is false. Notifications with the actual content (or a delta change) are sent to all online members whose open status for that object is true. This semantic is important to prevent members that are not interested in certain objects from receiving unnecessary information each time a change is made in the object.

Since all GSO content changes persist, GSO properties are still available when clients disconnect and later reconnect to the service. This allows members of an object to interact asynchronously. In summary, the described behavior of GSOs inherently merges real-time conferencing with content management and asynchronous collaboration modes.

5 Implementation

In order to study and better understand the implications and trade-offs of combining various interaction modes of collaboration in a common model, we have built two implementations: (1) a server that implements the GSO collaboration model natively; and (2) a server that uses existing collaboration technologies to deliver the same functionality offered by our model.

5.1 Native Implementation

In our native implementation, the GSO concept is directly mapped to persistent objects (using the OO programming paradigm). The implementation of the GSO manages every aspect of the model, i.e. content management, membership, access control, notifications, data transfer and persistency. The GSO service manages a collection of GSOs and their aggregation into hierarchical structures (trees). Clients access the GSO service through a client side API (see Fig. 2).

In the example of Fig. 2 (a), clients A, B, and C are all members of a GSO object. Client A and B open the object for real-time interaction by submitting an *openSO()* requests to the server (1, 3). The server GSO then sends open notifications to all its members, by iterating over the member list and invoking the registered callback interface methods (2, 4). The open state of the GSO is now changed to true for clients A and B. Sending notifications to every member of the GSO keeps all connected clients in a consistent state (i.e. with the latest view of the GSOs they are members of). Client C, for example, knows that A and B are currently working on the GSO content. Based on this information, client C can decide to open the GSO object and start receiving the actual new content as it gets changed. In Fig. 2 (b), client A changes the content of the GSO by submitting a *setProperty()* request (5); client B receives a content change notification including the content data (7). Client C is online but receives only a content change notification without the data because its open state is false (6). However, knowing that the content has changed, Client C could

now read the updated content of the object by submitting a *getContent()* request to the server.

The server is implemented in Java and communicates via Remote Method Invocation (RMI) with its clients. Notifications are sent to clients through RMI also. Upon logon, each client registers an RMI callback interface with the server. Since we assume storage to be a constant throughout this paper, we did not implement a particular storage mechanism in our prototypes.

5.2 Integrated Implementation

In our alternative integrated implementation, the initial native implementation was modified to perform synchronous interaction through meeting servers and to deliver events using a notification service. The integration of the two new backend technologies was completely transparent to the end users. Clients interact through the same GSO service API. In the backend, however, the implementation complexity increased significantly. A more detailed description of the service integration and the data flow can be found in [15].

For example, in order to integrate the meeting server with our model, we introduced the concept of a server-side client (SSC) that acts as a connector between the synchronous meeting and the persistent aspects of the model. A SSC is a special client in a meeting session. A meeting is a session created between two or more participants/clients that provides a non-persistent shared space where messages are sent to all the meeting members. The SSC is responsible for storing session data in a persistent repository by updating the respective GSO when content is changed. For example, when a chat message is posted to a meeting session, the SSC for that session stores the message in the GSO, which itself triggers a notification. This approach provides a generic mechanism that can be used to transparently integrate any meeting server.

Note that using meeting servers to support real-time collaboration entails setting up a meeting session with the meeting server every time a client opens a shared object for real-time interaction (see Fig. 2 (a)). Likewise the meeting session needs to be disposed every time the client closes the GSO. For each session, a SSC also needs to be created in the beginning and disposed in the end.

We integrated a notification server into the service to support asynchronous change notifications. Whenever a GSO's property or content is changed, a single notification is produced. Differently from the previous native implementation, that produced one notification per GSO member, a single message is now relayed to a notification server that is responsible for distributing the notification to all the members of the object. The subscription style used was topic-based: each client subscribes/un-subscribes to a global GSO notification topic when logging on and off the service. In this approach, the notification server acts as a broadcast channel; a bus connecting all online clients. Notifications are subsequently filtered in the client side API, i.e. the client API ignores notifications that are not addressed to that particular client.

The integrated solution was also completely implemented in Java. We used YANCEES [16] as the notification server because of its ability to be configured with

a simple topic-based core, and for having a simple API, similar to Elvin [2]. We used a simple Java-based meeting server from the TeamSpace project [7]. We sought to keep the two implementations as similar as possible in order to get meaningful results for a comparison, e.g. both implementations share the same common GSO model and externalize the same GSO API. However, given the number of different existing publish/subscribe and real-time collaboration systems, our simulation results may vary depending on the backend technologies used.

6 Experimental Results

The model described in Section 4 unifies characteristics of publish/subscribe systems, synchronous collaboration servers, and content management in a uniform and flexible way. As such, it facilitates the development of collaborative applications that have contextual collaboration characteristics. This blending of synchronous and asynchronous collaboration, however, requires the compromising of different requirements from these two interaction modalities. For example, traditional synchronous communication infrastructures, such as meeting servers, are usually designed to support the collaboration of small groups, under more strict timing and bandwidth conditions such as audio or video. Notification servers, on the other hand, generally are employed in applications with less strict timing and real-time constraints, focusing on awareness and messaging, where the number of clients is potentially large and the data traffic is relatively small. When those two different interaction modes are combined in a single collaboration model, different trade-offs involving scalability, responsiveness, robustness, and implementation complexity have to be considered. We conducted a series of experiments to understand these trade-offs and answer the following questions: How well do the two different implementations of the model handle the blending of synchronous and asynchronous collaboration? What is the impact of different data rates and data sizes depending on the type of media interaction? How is the response of the infrastructure to different combinations of those factors?

6.1 Experimental Setup

Since we wanted to understand the behavior of the model under regular use conditions and have strict control of the number of clients connected, we developed an automated client simulator that interacts with our service implementation using different patterns. Those patterns simulate the use of different collaborative tools with their traffic conditions, number of users and data size. The simulator client exercises the server APIs performing regular actions such as: create new object, set properties, open, close, add member and so forth. For the purpose of our tests, we defined four different patterns approximating the traffic conditions of chat, file sharing, message exchange, and streaming media. The streaming media pattern was defined to analyze the server behavior under heavier load, testing its scalability limits. Note that these patterns are only approximations of actual interaction patterns. Table 1 describes the different patterns with their data characteristics and probabilities.

The main differences between the four media traffic patterns are in the size of the data, the number of messages exchanged by each member, and the frequency (defined by the interval between messages). For example, a typical chat session in our simulator client corresponds to an interaction with a GSO with two members on average exchanging an average of 10 messages each member. Each message has an average length of 40 characters. Each chat GSO also has an average of seven properties that are modified with 16 characters on average. Chat messages are exchanged at every 15 seconds on average. During this interaction, periods of inactivity may also occur with an average duration of 15 seconds.

Table 1. Media pattern programming used in our experiments

| Media Pattern | n° Mem bers | Data | | | Content change probabilities | | |
|-------------------------|-------------|--------------|--------|----------|------------------------------|-----|-----|
| | | Size (chars) | n° msg | interval | Set | Add | Del |
| Streaming | 5 | 64K | 100 | 50 ms | 0.5 | 0.5 | 0.0 |
| Chat | 2 | 40 | 10 | 15 sec | 0.0 | 1.0 | 0.0 |
| File Sharing | 4 | 100K | 10 | 5 min | 0.7 | 0.1 | 0.2 |
| Message Exchange | 8 | 1K | 1 | 1 sec | 1.0 | 0.0 | 0.0 |

In our GSO model, a property can be set (overwritten or created), added (appended to the end of the current content), or deleted. Table 1 also shows the probabilities for these content change actions. In the chat pattern, for example, all chat content changes are of type “Add” because chat transcripts are typically not randomly modified, but they grow over time as new messages are exchanged.

For each pattern, we reproduce the actions of a typical work day of 8 hours. We programmed our automatic client to perform those actions in a simulation time of 4 minutes. This setup is similar to [8] and allow us to stress test the infrastructures using a reduced number of clients. During one simulated workday, the following actions are performed by the client: A total of 15 shared objects are created on average with five objects being root objects (representing a new activity thread). Each client listens to an average number of 10 objects. 15 open and 15 closed objects on average are modified that day. The interaction patterns also differ with respect to the time span that each client is working either online or offline.

All experiments were carried out on three client machines (IBM T30, 1.6GHz, 512MB) and one server machine (IBM MPro, 3 GHz, 1.5 GB). The client machines and the server were connected on an isolated 100Mbps Ethernet local network to eliminate interference with other network traffic. Client machines were equally loaded with a set of client simulators in steps of one, i.e. the first test starts with 3 clients (one in each client machine), then 6 clients (two per client machine) and so forth. Please note the number of simulator client processes running on a single client machine impacts the overall simulation results. Based on tests, we decided to limit the number of automated clients to eight per client machine in order to minimize this effect.

6.2 Results: Native Implementation

In order to understand the overall service behavior to the different media patterns, we plotted the total average execution times for each one of the four patterns against the number of clients interacting with the system. In this experiment, each client process executes a typical work day, using a single interaction pattern which includes open and closing objects, logging in and out, offline times and content changes.

Fig. 3 shows that the system has a linear response to the increase in the number of clients, for low-frequency traffic patterns such as chat, message exchange, and file sharing. The graph also shows that the size of the data, as in the case of file sharing, does not impact performance as much as the frequency of the messages. The main characteristic of streaming media is its high frequency of relatively large data messages. As can be seen in Fig. 3, our reference implementation does not scale as well for this pattern (it grows in a non-linear fashion). This can be explained by the fact that we send out content change notifications (with or without the actual content) to every member of the GSO. Given the high data frequency of streaming media, the server load increases quickly, since each data message triggers a series of content change notifications, typically one for each member of the objects involved.

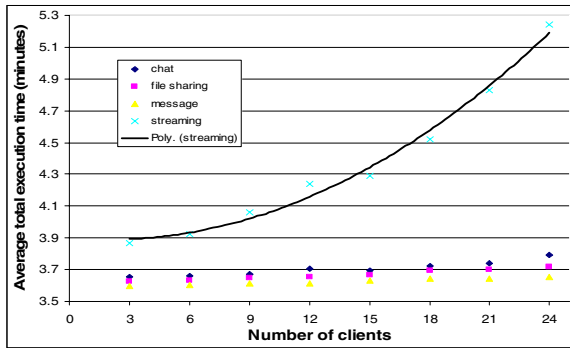


Fig. 3. Average total simulation execution times of the native implementation under different activity patterns for a typical workday

In another experiment, under the same experimental conditions, we sought to understand the responsiveness of our implementation. The responsiveness of a collaborative system is defined by its response and notification times. The response time describes how fast the system reacts to user input, i.e. how fast actions are reflected in the user interface of the clients executing the action and receiving responses. The notification time describes how fast a collaborative system updates remote clients. In a collaborative setting, it is desirable to keep this number as low as possible in order to keep all clients in sync with each other minimizing lag. Response time in our model is determined by the execution time of the client API calls. Fig. 4 shows the average method execution times for setting the content property of a GSO performed by the *setContent()* API call.

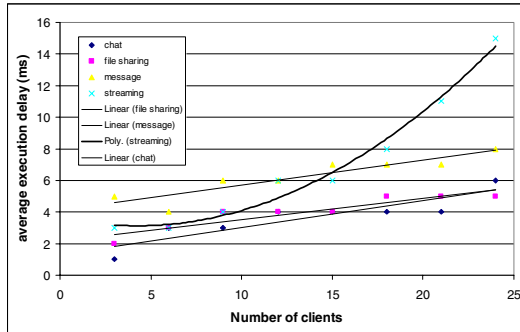


Fig. 4. Average execution time of the *setContent()* call in the native implementation with different media patterns

Fig. 4, shows that the execution times for the *setContent()* API call are relatively low (in the order of milliseconds). They grow linearly with the number of clients for all interaction patterns, except for the streaming media pattern. For a small number of clients, and consequently a small number of method calls on the server, the streaming media pattern is comparable to the other patterns but, as the number of method calls increases with the number of clients, the response time of the system to this pattern grows quadratically. Note that the message pattern initially has a relatively high execution time compared to streaming media. The reason is the higher number of members in that pattern (eight on average). This demonstrates the low impact of notifications (without data) relative to the frequency of interaction with the system.

In the same experiment, we also measured notification times: the period of time from calling a method in the client API to the delivery of its notification to the other members of a GSO.

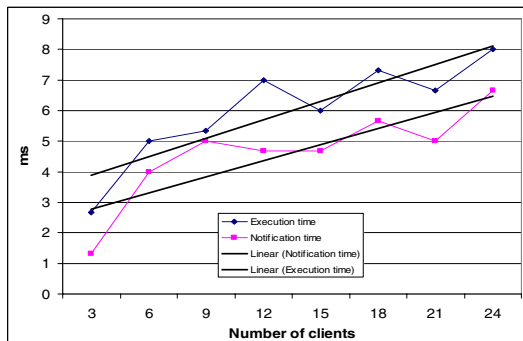


Fig. 5. Average execution vs. notification time for creating new GSOs in the native implementation

Fig. 5 shows the average execution vs. notification times for creating new GSOs. In this experiment, the notification times are slightly lower than execution times. At an almost constant difference of about 1 ms (in the trend lines), each local user

interaction is made visible to remote clients at about the same time. Except for streaming media *setContent()* calls, the response times of the native implementation are relatively high (i.e. below 10ms) and the notification delays are extremely low.

As a general conclusion, our experiments show that the performance of the model is a function of the data frequency of the interaction pattern (number of data messages/second), and the number of members of a GSO. For general traffic (low frequency and low bandwidth) the model scales very well having good responsiveness. However, for streaming media traffic, with a relatively medium number of members, and an average volume of information, the system delays increase quadratically.

6.3 Results: Integrated Implementation

Existing real-time collaboration servers are optimized for online meetings with a smaller number of participants but relatively high data volume, e.g. audio, video. Given the results in the previous section, it seems reasonable to apply real-time meeting servers to support frequent and high volume property changes in a GSO. We hypothesized that the implementation of the synchronous aspects of our model with a meeting server would increase the overall system performance.

Notifications are another aspect of our model that we believed to be well understood today. Publish/subscribe systems provide general-purpose event notification services. Notification servers receive anonymous notifications and route them to interested parties. This routing is orchestrated by subscriptions. These systems are typically optimized for a very large number of subscribers and small to medium data volumes for each subscriber. We hypothesized that GSO events such as create / delete GSO, add/remove member, or infrequent property changes (e.g. changing the presence status of a member on an object) would be well supported by a publish/subscribe system.

Hence, we expected that our integrated implementation of the model using meeting and notification servers, would result in better scalability of both the notification process (asynchronous mode in our model), and the synchronous collaboration through content exchange (the synchronous mode of our model). An expected price to be paid, however, would be the extra cost of integration and the increased complexity of the architecture. In order to verify this hypothesis, we repeated the same set of tests with the integrated service implementation.

Fig. 6 (a) compares the cost of the set/add content calls in both implementations for the streaming media pattern. As expected, the integrated implementation scales better, in a more linear fashion, than our original native implementation. In other words, using a dedicated meeting server seems to pay off for this type of traffic.

The chat and the file sharing media patterns did not expose any significant differences in the integrated implementation with regards to the cost of the *setContent()* call. The message exchange pattern, however, yielded some interesting results. Fig. 6 (b) shows that the use of our meeting server was more costly, in terms of performance, than the native implementation for this pattern. Both implementations though seem to expose linear behavior as indicated by the trend lines. One of the major differences between the message exchange pattern and the other patterns is the number of members per GSO (eight on average for the message pattern). While our meeting server seems to handle high bandwidth, high frequency traffic well, performance seems to degrade with an increased number of meeting participants.

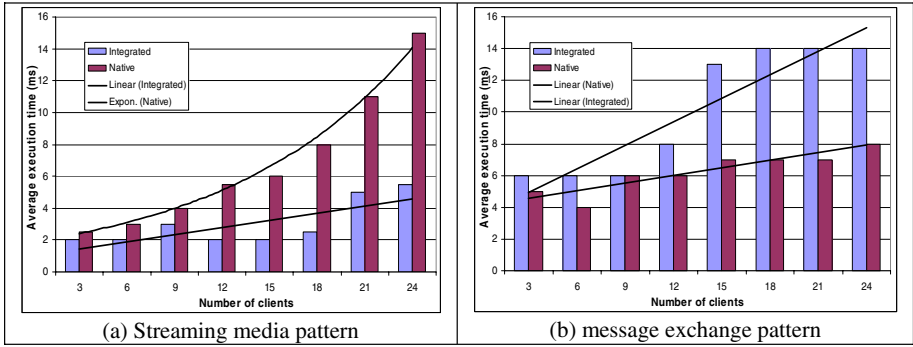


Fig. 6. Comparison of the average execution times for *setContent()* calls

Since the use of a meeting server introduces additional complexity (see Section 0), we expected that the price for better scalability during the synchronous interaction phase of a GSO would come with additional delays in the start up of the shared meeting that handles it. The data in Fig. 7 compares the cost for opening GSOs in both implementations. The data confirms that the open call, where a new meeting session is started, has become one of the most costly calls in the integrated implementation. However, it still scales in a linear fashion indicated by the trend line.

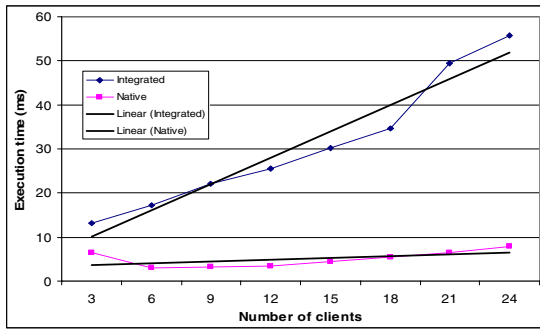


Fig. 7. Comparison of the average execution times for *openSO()* calls

When comparing the average execution times of other GSO API calls for both implementations, we noticed that the *registerMember()* and *loginMember()* calls also impose high delays in the integrated implementation. The reason for these delays is our notification server. Creating subscriptions when registering members and when logging in comes at an additional expense. Note that subscriptions in our native implementation were implicit through the member list.

While we expected that subscription management would come at an extra cost, we were surprised to see that the notification server introduced high delays in delivering notifications. Fig. 8 compares execution times for creating GSOs against the notification time. The integrated implementation has low response times but does not

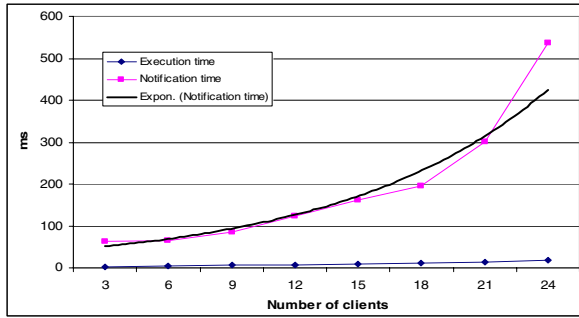


Fig. 8. Average response vs. notification times for creating new GSOs in the integrated implementation

scale well with regards to notifications. On average, under a load of 24 clients, remote clients are updated only 0.5 second after the GSO was created locally. The notification times seem to grow exponentially according to the trend line.

One could argue that the use of the notification server as a shared bus is one of the reasons for the notification server behavior observed in Fig. 8. In another alternative implementation, we tested server-side filtering of events, i.e. the configuration of the notification server with more accurate subscriptions that filter out events that are not of interest of the client. This approach, however, required constant update of the subscriptions (each client manages one or more subscriptions filtering out events that do not belong to the objects they are members of). Subscriptions need to be updated when new objects are created, members log on/off, or members are removed/added to objects. Given the high subscription costs impacting the *registerMember()* and *login()* operations, this solution did not scale well. These membership and object life-cycle dynamics resulted in similar or worse delays than the ones observed in Fig. 8.

7 Lessons Learned

Interference of conflicting requirements. The support of synchronous and asynchronous interaction in a common and simple model is not a trivial task. While the native implementation of the GSO model supported well the majority of traffic patterns, it did not scale well for high frequency, high-bandwidth data as in our stream media pattern. The use of meeting servers can improve the performance of synchronous message exchange under those circumstances. However, the notification server in our integrated implementation became a bottleneck, impacting the scalability of the entire model. This demonstrates how a combination of different services can interfere with one another, limiting the performance of the overall infrastructure.

Integration complexity. Our initial hypothesis, that the integration of existing services to support contextual collaboration, would combine the strengths of both services, showed not to be completely true. It had shortcomings in the form of extra complexity. Even though an integrated solution, that uses specialized services, can perform better than a more simple implementation, the integration of those off-the-shelf components usually demands special attention to matters such as timing,

synchronization, and adequacy to the model. It also makes the implementation of the system more prone to errors and additional setup delays, such as startup times, as observed in our experiments, during member log-in and opening objects.

Mismatch of programming models. Another issue elucidated in our experiments was a mismatch of the programming models of the different components used. For example, the extension of the meeting server to support persistency was not trivial; our solution was to use a server-side client acting as meeting recorder. Another example was the inadequacy of the notification server in handling frequent subscription changes. In our experiments, we tested the integrated GSO implementation with two subscription models: server-side filtering and client-side filtering. Client-side filtering was the approach that better scaled in our implementation. Both approaches, however, had their own trade-offs and limitations: client-side filtering moves part of the processing to the client side, but requires the delivery of extra notifications through the network. Server-side filtering limits the amount of traffic to the clients and relieves them from discarding unnecessary notifications. However, the latter approach results in an extra burden to the notification server, that needs to deal with constantly changing subscriptions in order to accommodate changes in the GSO membership.

Impact of distribution. An advantage of using separate components such as a meeting server and a notification server is the ability to distribute processing throughout different hosts in a network. In additional tests, we distributed the notification and meeting servers across different machines in the network. We found that, with more than 30 clients, the distributed configuration begins to perform better than the centralized approach. This shows that with a significant number of clients, the distribution of main system components is a good approach for scalability.

8 Conclusion

In this paper we studied two implementations of a new collaboration model that seamlessly integrates different collaboration modalities into a single interaction model. Our model facilitates the development of contextual collaboration applications such as Activity Explorer. Our experiments show the trade-offs of developing contextual collaboration systems based on existing collaboration services such as meeting and notification server. The simultaneous support for synchronous and asynchronous interaction in a single model tends to work well in a native implementation for the average case, where neither the synchronous nor the asynchronous aspects of the model are put to exceeding stress. The low complexity of a native implementation together with high responsiveness might satisfy the requirements of the majority of contextual collaboration applications today. The integration of meeting servers restricted to only media traffic can significantly improve the scalability of the implementation. The use of generic notification servers to support the model, however, was problematic because mapping GSO behavior onto publish/subscribe semantics caused additional overhead.

Acknowledgements. We would like to thank John Patterson, the Pesto team in Haifa, and the Activity Explorer product and research teams for their inspiring discussions.

References

1. Cheng, L.-T., Hupfer, S., Ross, S. and Patterson, J., Jazzing up Eclipse with collaborative tools. in *OOPSLA'03 workshop on eclipse technology eXchange*, (Anaheim, CA, 2003), 45-49.
2. Fitzpatrick, G., Mansfield, T., Arnold, D., Phelps, T., Segall, B. and Kaplan, S., Instrumenting and Augmenting the Workaday World with a Generic Notification Service called Elvin. in *(ECSCW '99)*, (Copenhagen, Denmark, 1999), Kluwer, 431-451.
3. Freeman, E., Hupfer, S. and Arnold, K. *JavaSpaces Principles, Patterns, and Practice*. Book News, Inc, 1999.
4. Gamma, E., Helm, R., Johnson, R. and Vlissides, J. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Publishing Company, 1995.
5. Gelernter, D. Generative communication in Linda. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 7 (1).
6. Geyer, W. and Cheng, L., Facilitating Emerging Collaboration through Light-weight Information Sharing. in *conference supplement CSCW'02*, (New Orleans, LA, 2002).
7. Geyer, W., Richter, H., Fuchs, L., Frauenhofer, T., Daijavad, S. and Poltrock, S., A Team Collaboration Space Supporting Capture and Access of Virtual Meetings. in *ACM 2001 International Conference on Supporting Group Work*, (Boulder, CO, USA, 2001), ACM.
8. Geyer, W., Vogel, J., Cheng, L. and Muller, M., Supporting Activity-Centric Collaboration through Peer-to-Peer Shared Objects. in *ACM GROUP*, (Sanibel Island, FL, 2003), 115-124.
9. Kantor, M. and Redmiles, D., Creating an Infrastructure for Ubiquitous Awareness. in *Eighth IFIP TC 13 Conference on Human-Computer Interaction (INTERACT 2001)*, (Tokyo, Japan, 2001), 431-438.
10. Mahowald, R. From ICE Age To Contextual Collaboration, IDC, retrieved at http://www.cio.com/analyst/062901_idc.html, June 29, 2006.
11. Moody, P. and Feinberg, J., C+B Seen Project. in <http://domino.research.ibm.com/cambridge/research.nsf/pages/projects.html>.
12. Patterson, J.F., Day, M. and Kucan, J., Notification servers for synchronous groupware. in *ACM conference on Computer supported cooperative work (CSCW'96)*, (Boston, Massachusetts, 1996), 122-129.
13. Preguiça, N., Martins, J.L., Domingos, H. and Duarte, S., Integrating Synchronous and Asynchronous Interactions in Groupware Applications. in *11th International Workshop, CRIWG 2005*, (Porto de Galinhas, Brazil, 2005).
14. SearchDomino.com. Contextual Collaboration, http://searchdomino.techtarget.com/sDefinition/0,,sid4_gci934929,00.html, June 26, 2006.
15. Silva Filho, R.S., Geyer, W., Brownholtz, B., Guy, I., Redmiles, D.F. and Millen, D.R. Architectural Trade-Offs for Collaboration Services Supporting Contextual Collaboration - RC23756, IBM T. J. Watson - Cambridge, Cambridge, MA, 2005.
16. Silva Filho, R.S. and Redmiles, D., Striving for Versatility in Publish/Subscribe Infrastructures. in *5th International Workshop on Software Engineering and Middleware (SEM'2005)*, (Lisbon, Portugal., 2005), ACM Press, 17 - 24.
17. Souza, C.R.B.d., Basaveswara, S.D. and Redmiles, D.F., Using Event Notification Servers to Support Application Awareness. in *IASTED International Conference on Software Engineering and Applications*, (Cambridge, MA, 2002), 691-697.
18. Wyckoff, P. TSpaces. *IBM Systems Journal*, 37 (3).

Leveraging the Linda Coordination Model for a Groupware Architecture Implementation

José Luis Garrido¹, Manuel Noguera¹, Miguel González²,
Miguel Gea¹, and María V. Hurtado¹

¹Dpt. Lenguajes y Sistemas Informáticos, University of Granada,
E.T.S.I.I., c/ Saucedo Aranda s/n, 18071 Granada, Spain
{jgarrido, mnuquera, mgea, mhurtado}@ugr.es

²Tecnologías de la Información, Autonomous University of Madrid,
E.P.S., c/ Tomás y Valiente 11, 28049 Madrid, Spain
miguel.gonzalez@uam.es

Abstract. Functional and non-functional requirements must be taken into account early in the development process of groupware applications in order to make appropriate design decisions, e.g. spatial distribution of group members and group awareness, which are related to the main characteristics exhibited by CSCW systems (communication, coordination and collaboration). This research work presents a proposal intended to facilitate the development of groupware applications considering non-functional requirements such as reusability, scalability, etc. In order to achieve these objectives, the proposal focuses on the architectural design and its implementation, with emphasis on the use of a realization of the technological Linda coordination model as the basis for this implementation. The outcome is a distributed architecture where application components are replicated and event control is separated. This work is part of a conceptual and methodological framework (AMENITIES) specially devised to study and develop these systems.

1 Introduction

Groupware has been defined as “a computer-based system that supports groups of people engaged in a common task (or goal) and that provides an interface to a shared environment” [7]. To date, groupware has comprised various systems: Workflow Management Systems (WfMS), Computer-Mediated Communication (CMC) (e.g. e-mail), Decision Support Systems (DSS), shared artifacts and applications (e.g. shared whiteboards and collaborative writing systems), etc. These systems include common and specific requirements in relation to the following group activities [7]:

- *Communication.* This emphasizes the exchange of information between remote agents by using available media (text, graphics, voice, etc.).
- *Collaboration.* Effective collaboration requires people to share information in the group context.
- *Coordination.* The effectiveness of communication/collaboration is based on coordination. It is related to the integration and harmonious adjustment of the individual work effort towards the accomplishment of a greater goal.

The inherent complexity of CSCW (Computer Supported Cooperative Work) systems requires a great deal of effort in specifications and development [3]. The development of groupware applications, the technological part supporting collaboration processes in CSCW systems, is more difficult than that of a single-user application; social protocols and group activities must be taken into account for a successful design [13]. Methodologies and implementation techniques aimed at enhancing group interaction activities (especially for synchronous groupware [23]) should therefore be applied. On the other hand, there is a lack of methodological proposals for addressing and integrating the study and development phases of a CSCW system. Furthermore, we argue that special attention must be paid to the software architecture, which is defined by the recommended practice for architectural description of software systems [2] as “the fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution”.

This article shows the implementation of a high-level architectural design for groupware applications [11]. This architecture guarantees important software quality properties since the main design criterion centers on mutual component independence. To this aim, we propose the use of the data-driven programming model [20] provided by the JavaSpaces technology [29], a realization of the Linda coordination language [5], in order to accomplish its implementation. The proposal based on this technological data-driven coordination has two aims: firstly, to fulfill common and specific functional requirements in CSCW systems related with human communication, coordination and collaboration, such as to provide group awareness in order to support and enhance these group activities; and secondly, to be able to build this kind of distributed applications taking also into account non-functional requirements such as reusability, scalability, etc. This proposal is part of AMENITIES [10], a conceptual and methodological framework that is specially devised to study CSCW systems and develop groupware applications.

The paper is organized as follows. Section 2 briefly introduces the AMENITIES methodology, providing a general description of its models and stages. Section 3 focuses on how an architectural design enables us to address the development of groupware applications. Section 4 introduces the Linda coordination model and describes how it is used to implement group awareness in this architectural design. In Section 5, the physical architecture and its corresponding deployment are described briefly for the current implementation. Section 6 references related work and the main conclusions are provided in Section 7.

2 AMENITIES

AMENITIES [10] (an acronym for A Methodology for aNalysis and desIgn of cooperaTive systEmS) is a methodology based on behavior and task models, specially devised for the analysis, design and development of CSCW systems. The methodology stems from cognitive frameworks (activity theory, distributed cognition, etc.) and methodological proposals (requirements [17] and software engineering [27]),

task analysis [31] and modeling [21]). Thereby, AMENITIES provides a conceptual and methodological framework that seeks to avoid the main deficiencies found in approaches traditionally applied to this kind of system, by focusing on the group concept and covering the most relevant aspects of its behavior (dynamics, evolution, etc.) and structure (organization, laws, etc.). Another objective is to allow us to systematically address the analysis and design of CSCW systems and to facilitate subsequent software development. Therefore, it also proposes a concrete methodology including concrete phases, models, notations, etc. Fig. 1 provides a general scheme of the methodology, showing the main models (boxes) and stages (dashed lines) involved. Just like most methodologies, AMENITIES follows a simple iterative process allowing us to refine and review these models.

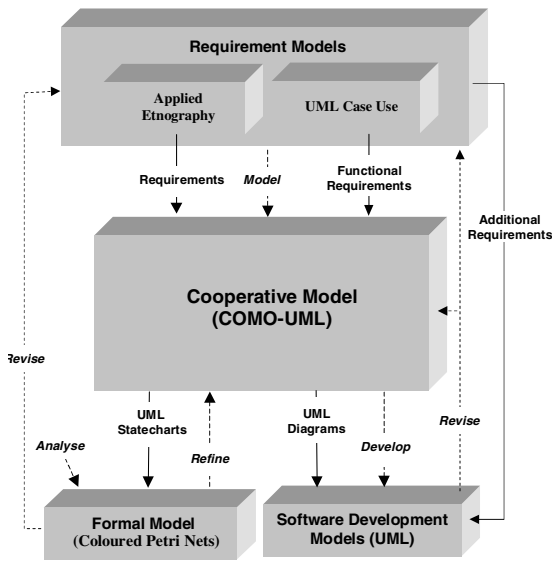


Fig. 1. General scheme of AMENITIES

The cooperative model (called COMO-UML in Fig. 1) is the core of the methodology and enables us to represent and connect instances of concepts defined in the conceptual framework of AMENITIES, according to the requirements for each specific system. The cooperative model [8] describes the system (especially on the basis of coordination, collaboration and communication) irrespective of its implementation. It therefore provides a better understanding of the problem domain. In order to build this model, a structured method (comprising four stages) is proposed: specification of the organization, role definition, task definition, and specification of interaction protocols. This method has been specifically devised to make easier connections between all the concepts (for instance, tasks to be performed under each role). The modeling notation proposed is based on both UML state and activity diagrams [19], but with a semantics specially defined for this problem domain [9].

3 An Architecture for the Groupware Development

The architectural design of the groupware application is the starting model for the software development phase in AMENITIES, and therefore, for the proposal introduced in this paper. This section shall first present the motivations and foundations for our architectural proposal; it shall then make the proposal concrete by using a real case study.

3.1 Motivations and Foundations

Most requirements to be considered for the development of each specific groupware application are specified in the cooperative model described above, for instance, tasks requiring various actors to be accomplished (i.e. cooperative tasks), or constraints (specified by means of the law concept) preventing an actor from being involved in more than one task. However, apart from specific requirements for each groupware application, common design issues for this kind of system should be taken into account at an abstract level. An architecture guiding the organization of architectural elements (basically composition, interfaces and interactions) can just provide this desired abstraction level covering the following general objectives:

- Groupware applications are inherently distributed. It is therefore important to obtain an implementation stemming from a set of subsystems that communicate with each other through well-defined interfaces. The division/partitioning of the whole system into components (called subsystems) facilitates its development, evolution and maintenance.
- Appropriate organization and mapping of functionality onto subsystems in order to achieve certain desired software properties such as reusability, portability and interoperability.
- A groupware system should be able to increase the number of subsystems because new applications supporting other activities could be added for the same group.

In order to achieve these objectives, some guidelines of the Unified Software Development Process [15] for specifying the architectural view of the design model have been adopted. For this purpose, the design process is carried out using the UML language, providing the three following architectural views:

1. *Component view*. In order to represent the system partitioning, package diagrams with the stereotypes system and subsystem are used in conjunction with the composition relationship (a form of the aggregation association that considers bound parts by lifetime).
2. *Functional view*. The functional aspects (static structure) of the system are specified by means of both class and interface diagrams associated with subsystems, and creating connections between them on the basis of the use relationship.
3. *Behavioral view*. System behavior (dynamic view) (i.e. how the subsystems collaborate by means of interactions) is specified on the basis of the functional structure (described in the previous paragraph), using collaboration diagrams.

3.2 Case Study

A collaborative appointment book application for group work has been developed according to the architecture to be described in the next subsection. This application allows lecturers/researchers (people playing these organization roles) within the same department at the University of Granada to coordinate in proposing meetings in a common forum. This human coordination must be supported on the basis of providing group awareness [25]; users currently connected to the system can observe each other (both presence and activity). An additional requirement for the system is to allow participants to share information in real-time (synchronous) and asynchronous modes.

The application (see Fig.2) consists of:

1. A panel for possible roles to be played, which also highlights the role currently being played by the participant.
2. A panel showing the other participants who are playing the same role.
3. A panel including the calendar and the messages sent.
4. A popup window to show the activities of each participant we are interested in.

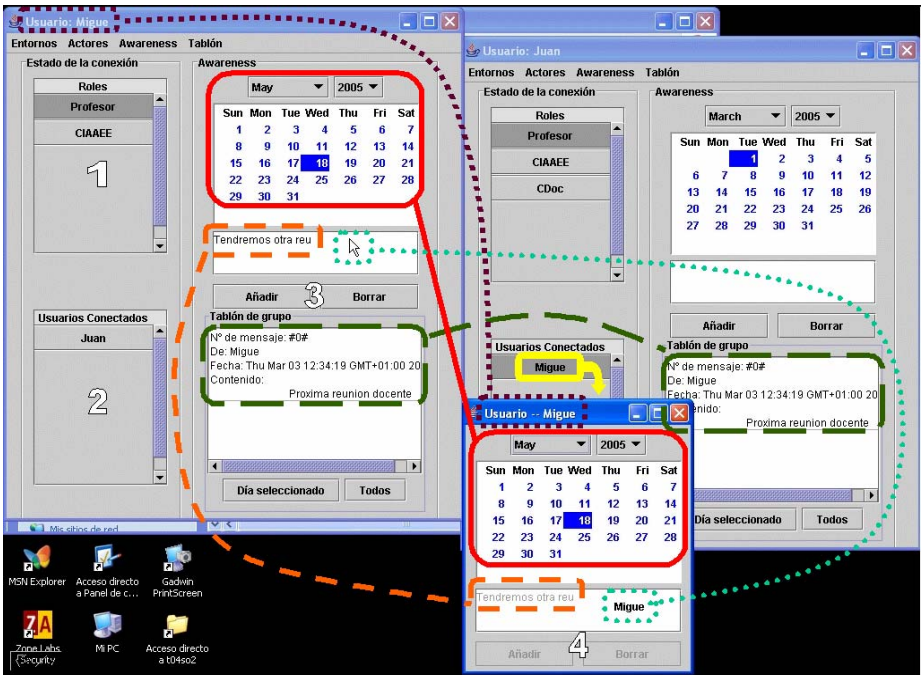


Fig. 2. General view of the collaborative appointment book application

3.3 Architecture

According to the objectives and the three views described in the subsection 3.1, the architecture is made concrete as follows:

1. *Component view*. The whole system is divided into several parts. A basic groupware application consists of four subsystems: *Identification*, *Metainformation*, *Awareness* and the application itself (in this case, the *Appointment book*). Fig. 3 shows the component diagram for the case study, illustrating the UML package diagram and the *System* and *Subsystem* stereotypes. Although there is only one application in this example, other applications can be easily integrated while this design is maintained.

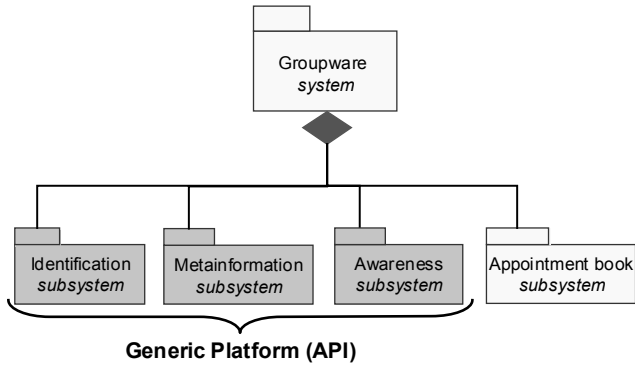


Fig. 3. UML package diagram for the component view

The *Identification*, *Metainformation* and *Awareness* subsystems are always present for every groupware application. The application subsystem is obviously specific for each groupware application. All the subsystems are described in more detail in the next views, according to the functionality they provide and the behavior exhibited.

2. *Functional view*. By means of a UML interface diagram, Fig. 4 (left) shows the four mentioned subsystems and their use relations on the basis of the associated functionality. Each subsystem provides an interface designed to achieve independence between the subsystems and other applications making use of them.

In particular, the *Metainformation* subsystem supports all the functionality for checking metadata (possible roles to be acquired, laws applied by the organization, etc.) specified in the cooperative model. The *Identification* subsystem is used to start the application and to control users' access to the system. The *Awareness* subsystem is intended to maintain shared and contextual information (telepointers, list of participants playing a specific role, etc.) in charge of providing group awareness that the participants need for an effective collaboration. Finally, the *Appointment book* subsystem provides both an extended functionality as that of a single-user appointment book and slightly different semantics.

3. *Behavioral view*. Fig. 4 (right) shows how the subsystems collaborate to resolve interactions between users and the *Appointment book* subsystem. A UML collaboration diagram is used to model this behavior.

In this case, a typical user interaction starts at the *Appointment book* subsystem. Firstly, users must identify themselves in the system (messages 1, 1.1, and 1.2). Once they have been correctly identified, they can choose to register their presence

in the system so that other users can choose to observe them (messages 2, 2.1, and 2.2). They themselves want to know which users are currently connected under the same role (messages 3, 3.1, 3.2, and 3.3). The system therefore supplies the necessary infrastructure for group awareness and we are able to collaborate in real time.

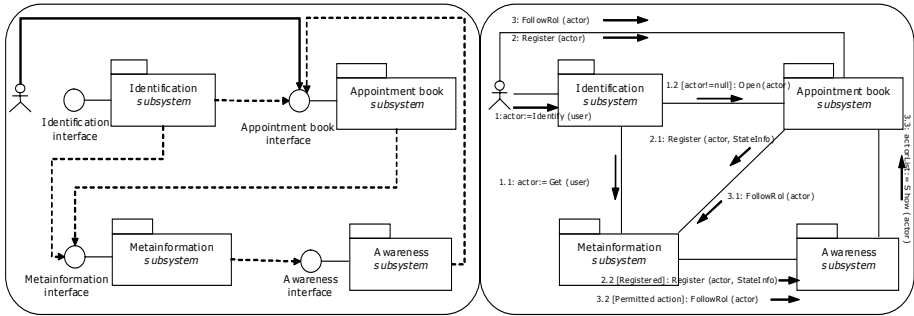


Fig. 4. UML interface diagram (left) and collaboration diagram (right)

4 Architecture Implementation

There are basically three ways of communicating/sharing information between group members in order to provide effective group awareness: “*explicit communication, where people tell each other about their activities; consequential communication, in which watching another person work provides information as to their activities and plans; and feedthrough, where observation of changes to project artifacts indicates who has been doing what*” [14].

In the following subsections, we will briefly describe Linda and JavaSpaces (Section 4.1), and also how the Linda coordination model is applied to the implementation of the *Awareness* subsystem (Sections 4.2 and 4.3) within the architectural design presented in Section 3.3. The focus is on how object-oriented tuple space features can be used to support these three ways of communicating/sharing information.

4.1 The Linda Coordination Model and the JavaSpaces Technology

The aim of distributed computing is to design and develop each distributed application as a set of processes and data which are distributed over a computer network, and to interact with them in an integral way. Computation and involved element coordination are usually addressed separately in distributed computing. A technological coordination model establishes the relations between components and in turn provides the mechanisms needed to enable interaction between them. These mechanisms are orthogonal to the computation model [12].

The space-based distributed computing model is derived from Linda [5]. A space is an object (or tuple in Linda terms) store for data shared through a computer network. A tuple is a data structure with several typed fields set to particular values, e.g.

<10,"Madrid">, consisting of an integer number and a string. Only a few atomic, basic functions are provided to operate on them (see Table 1). Operations *in* and *rd* use an especial template tuple as a pattern with which to match those tuples to be retrieved. If several tuples match the searching template, only one is retrieved non-deterministically. Since a tuple space is "global" (i.e. visible from any location), the processes on any computer can insert tuples into or take tuples from the space concurrently.

Table 1. Tuple space operations

| Operation | Description |
|-----------|---|
| out | Insert a tuple into the space |
| in | Take a tuple from the space |
| rd | Inspect (read) a space tuple, without removing it |

JavaSpaces [29] has been chosen as a tuple space implementation based on Linda. In JavaSpaces, tuples and their fields are typed as objects in the Java programming language. This provides a richer type system than Linda does. Since tuples are objects which belong to a particular class, they may have methods which are associated with them.

There are four main operations which can be invoked in a JavaSpace tuple space: *write*, *read*, *take* (can be blocking or not) and *notify*; the first three operations correspond to the Linda operations *out*, *rd*, and *in*. Henceforth, we will use the operation names provided by JavaSpaces. Objects use the *notify* invocation to ask the space to inform them that one kind of tuple matching a given template has been inserted into the space. The mechanism involved in this operation is called distributed notification of events [28]. When an object wants to be notified about some kind of tuple insertion, it must register the corresponding matching pattern of the tuple together with the *notify* operation. Notification will be provided when a tuple matching this pattern has been inserted into the space. This mechanism will be very useful as basis of providing context awareness in groupware applications.

In JavaSpaces, the operations mentioned above are not limited to a single space but may be carried out in turn on different spaces. As far as the information space (i.e. the tuple space) is concerned, this also enables centralized, replicated, dynamic, hybrid, etc. architectures to be devised. On writing a tuple in the space, the time it will remain in the space before being deleted is also defined. This time may be indefinite so the tuple in question will remain until it has been withdrawn by the *take* operation.

4.2 Data Sharing

While the cooperative task is being performed, it is necessary to share data. This section discusses the basic mechanism to share data stored in tuples. Depending on when data are shared, it is possible to distinguish between synchronous and asynchronous modes of communicating information [23]. Synchronous communication shows the changes that occur in the shared environment as soon as they take place. Asynchronous communication can be obtained for instance, when

logging into the system and observing the changes that other users have made on the shared objects at another time.

In the collaborative appointment book application, both types of communication can be found in the way the user interacts with the board saving appointment messages. When a user decides to log into the system and download the corresponding board messages, he/she can see the contributions that other group members have submitted previously (asynchronous communication). If he/she wants to be informed of any message arriving at the shared board, he/she will immediately see any contribution from the members playing their current role (synchronous communication). Similarly, users can choose what kind of system interaction they desire at any time and effortlessly change from one to another.

Fig. 5 shows a sample scenario for the collaborative appointment book introduced in Section 3.2. Let us imagine that there are several users in the system (e.g. *Anna*, *Peter*, *John* and *Martha*). Each one is part of at least one group. For example, all are members of the group *Prof* (*Professor*) and *Peter*, *John* and *Martha* are also *TC* (*Teaching Committee*) members. Let us imagine that user *John* has just logged into the system. At present, only *Anna*, *Peter* and now *John* are connected. If *John* decides to load his group board, he will see all the proposals that other group members have previously submitted. In this example, we can see how he would read *Peter's* and *Martha's* messages (*Martha* is not even connected now). If he decides to “register” to be notified about the submission of new messages by other members, he will need to insert a template tuple into the space (through his *Awareness* subsystem) in order to show that he is interested in the tuples related to his group board.

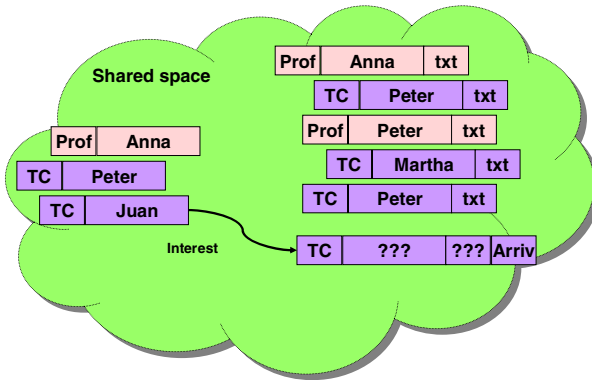


Fig. 5. Synchronous and asynchronous data communication

4.3 Feedthrough and Consequential Communication

The JavaSpaces mechanism for distributed notification of events (*notify* operation described above) and the definition of the time a tuple may remain in the space are also used in our approach to support the other ways of group awareness, namely feedthrough and consequential communication. This section discusses how events are propagated in the system.

We have found that a groupware system can be basically modeled using two types of tuples: tuples containing steady information and which are inserted into the space for an indefinite time; and tuples used to announce events and which are more volatile since information relating to their associated events is relevant at a given time and in a particular context. This is shown through the two following scenarios.

Actor’s Presence

Let us consider that the actor *John* has logged into the system and is playing the role of *Teaching Committee (TC)* member. He is reading this committee board to see what messages have been sent by other members. *Peter* is currently connected and he is playing the same role as *John* (i.e. *TC*). *Anna* is also connected and playing the role *Prof*. In addition, *John* would like to know which other members are connected now in order to discuss the date of an online meeting with them or to see which other proposals they are currently submitting. In order to accomplish this, user *John* asks (from his appointment book application, and therefore, the appointment book subsystem) to be informed of which other users playing the same role as him are connected. In turn, the *Appointment book* subsystem will use the awareness subsystem to insert appropriate template tuples into the space so that he will be notified whenever any user playing the role *TC* arrives or leaves. This scenario is depicted in Fig. 6. The same applies to the notice board messages that are represented on the right-hand side of the figure.

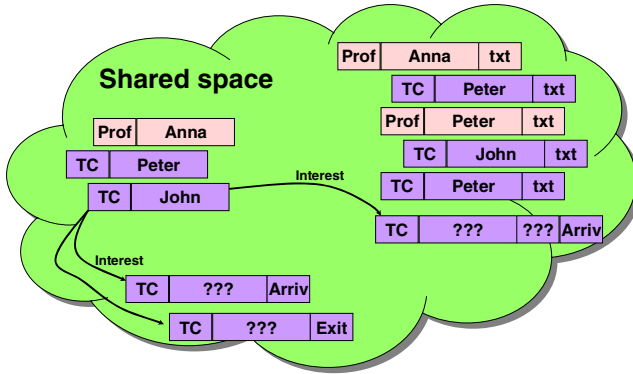


Fig. 6. Starting scenario

If professor *Martha*, who is also a *TC* member, enters the system playing this role, the space would notify *John’s Awareness* subsystem of this event. Previously, *Martha’s Awareness* subsystem would have inserted the appropriate tuples into the space (see Fig. 7) when she logs in.

In this way, *Martha’s Awareness* subsystem would insert a (1)-type tuple (as mentioned at the beginning of this Section) indicating more stable information such as her name, role she is playing, connection time, etc. (for the sake of simplicity, this tuple has the form <TC, Martha> in Fig. 7); and another (2)-type tuple in the form <TC, Martha, Arriv> that provides “event” information such as “Martha has just

logged in". The (2)-type tuples would have a brief remaining time in the space associated since this kind of information is only meaningful or valid to the connected users that have asked to be notified of other group member activity. If after half an hour, another *TC* member connects (e.g. *Martin*), his *Awareness* subsystem would inform the other members playing his role by reading <TC,??> tuples.

The same applies to the exit notification. When *Martin* logs in, the tuples used by other members logging out before he connects (e.g. <TC,Peter,Exit>) are useless and must therefore be deleted. By deleting these tuples, we also decrease the information overload in the space. We will see this in detail in the following scenario.

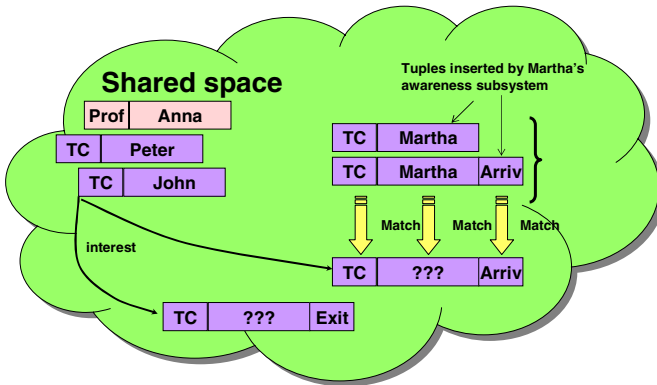


Fig. 7. Arrival of a member with the same role

Actor's Activity

The need for tuple elimination is more evident if we consider the telepointer mechanism. The coordinates of other users' mouse pointers have a very brief validity (a hundredth of a second at most) since they are continuously changing. In this regard, the programmed removal of the tuples in the space is a very efficient way of maintaining space consistency and preventing obsolete information being stored.

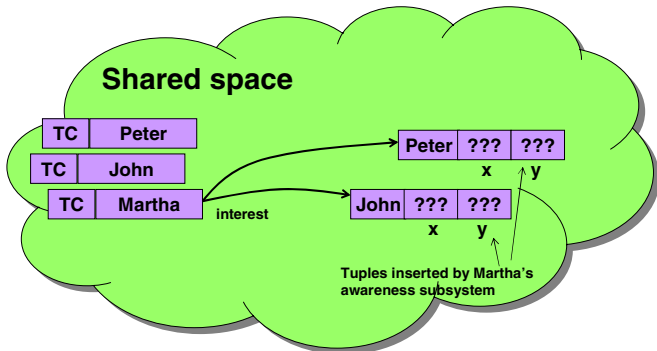


Fig. 8. Remote mouse pointer tracing

The starting point for the following scenario is Scenario 1, although some tuples have been hidden for readability. Let us imagine that *Martha* has just arrived and wants to see what *John* and *Peter* are doing. For this purpose, the appointment book application allows other members' application interfaces to be replicated and represented remotely.

Martha's awareness subsystem will be in charge of inserting template tuples in the space so as to trace *John's* and *Peter's* pointers (see Fig. 8).

Since *John* and *Peter* are registered in the system and their activity can be observed, their *Awareness* subsystems are continuously inserting their mouse pointer coordinates (see Fig. 9) in the space. This kind of tuple lifetime is very brief as the information they provide is very punctual. It does not matter whether *Martha's* *Awareness* subsystem captures all *John's* or *Peter's* mouse pointer coordinates.

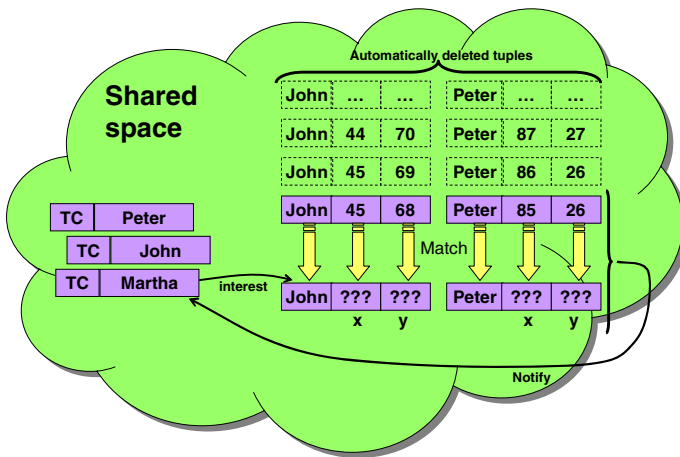


Fig. 9. Remote mouse pointer tracing

5 Physical Architecture and Deployment

JavaSpaces allows several spaces to be “running” on the same or different machines. This means that the tuple space need not be allocated on a specific machine and may also be distributed and partitioned between different sites. In any case, all the tuples in all the spaces form a single logical space of data. The space partitioning also allows pieces of information and tuples used by certain services (e.g. *Awareness* subsystem) to be allocated on certain machines depending on system performance. The way a space client application is “advised” about the entire tuple space distribution and the space distribution itself are beyond the scope of this work.

Each user has a replica of the application, i.e. the one he/she interacts with. In order to represent the behavior of other group members remotely, it is only necessary to replicate the events that the space notifies in the remote user interfaces that every user maintains for each user he/she is observing. This way of programming remote state replication is particularly efficient since a user need not send his entire state by means

of complex data structures or objects; this state is instead defined separately on the basis of very short messages announcing the occurrence of certain events.

Current implementation has been carried out by using standard internet technology (web browsers and servers, Java, etc.). Fig. 10 shows how client applications and the space are not necessarily placed in any particular network node. The clients and services can be allocated in the same or different nodes. Clients interact with the system by inserting information into the space or retrieving information from it without being aware of where the tuples (or services) are.

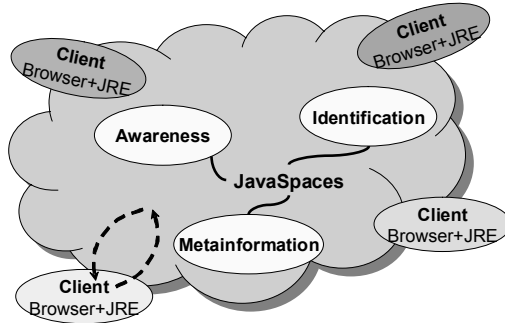


Fig. 10. Abstract view of the system and deployment

6 Related Works

The CSCW and groupware community have developed in the last years several experimental collaborative systems. Most of them have been developed for particular applications; this fact requires a great effort in implementations. Others have identified basic services for groupware systems developing toolkits to build these applications. For example, Groupkit [24] provides a component library for building multi-user interfaces. The main problems of this kind of proposal are:

- interfaces cannot interoperate between them, and
- difficulty of adaptation to the user's needs.

Pounamu [18] is a collaborative editing tool for software system design that permits work to be carried out both synchronously and asynchronously. It is built in Java using RMI and can be installed using different architectural designs, although it is limited to peer-to-peer collaboration scenarios.

EventHeap [16] provides a software infrastructure for interactive workspaces by means of an extension of TSpaces [32]. In [26] a flexible notification framework for describing and comparing a range of notification strategies is introduced; it can be very useful to guide the design of notification components. NSTP (Notification Service Transfer Protocol) [22] is a basic service (no semantics) for sharing state in synchronous groupware, therefore, it abstracts out the problem of state consistency from any application. The mechanisms implemented in these three proposals are similar to the presented in this paper, but in our case, the emphasis is mainly on the

methodological context for these implementations (global and abstract architectures, methodologies, ...) instead of technological one (failure isolation, algorithms, consistency...).

MARS-X [4] and XML-Spaces [30] are coordination models that extend Linda model features using XML document properties for information sharing. Another architectural pattern for web services based on the Linda model (which also uses XML documents) is proposed in [1]. None of these addresses remote event replication in remote interfaces.

7 Conclusions and Future Work

Technological and social aspects influence human collaboration; being aware of other group members' activity diminishes the risk of work duplication and inconsistencies in proposed tasks deriving from their concurrent accomplishment. This paper proposes an implementation for fulfilling functional requirements related to group awareness. It is part of a conceptual and methodological framework.

The state of the system is described by means of tuples in a space; replicas of the same groupware application and interactions between subsystems can be technologically coordinated exchanging tuples through spaces instead of communicating directly. Another benefit is the temporal and spatial dissociation inherent to this paradigm since a space and its tuples may remain in the system, even after the process that created them has ended. At this level, implementation and deployment issues are abstracted thanks to the coordination model that eases the building of distributed applications.

Whether it is distributed or not, a tuple space (such as the one our collaborative appointment book interacts with) imposes no restriction about the computer where it must be placed. This enables us to fulfill certain non-functional requirements related to profitable transparency characteristics in distributed systems: localization, access, replication, concurrency and scalability. This degree of transparency simplifies architecture development and facilitates the application of concrete guidelines in groupware systems building. As outcome of this research work a hybrid collaboration architecture [6] has been proposed; application is replicated in each network node, and control is logically centralized (actually distributed or centralized depending on the tuple space implementation). The main advantage of this approach is that users keep local replicated copies of other participant interfaces without sending the objects which define them (the interfaces).

There are other benefits of the proposed solution made apparent from the experience acquired in the development of a real example. Although performance analysis has not been carried out yet, apparent results obtained executing the collaborative appointment book application on internet seem promising.

By way of future work, we have started to develop a graphic component library (toolkit) for building groupware applications following the hybrid architecture mentioned above. The aim is to include and encapsulate the implementation philosophy described in this work within these graphical components in order to simplify the subsequent groupware systems development. In addition, portability in the current implementation is Java-dependent, that is, the applications which interact

with the space must be programmed in this language. Some of the related works avoid this handicap using markup languages (XML) in the definition of the tuple space and thereby, hiding implementation issues and promoting interoperability.

Acknowledgements

This research is partially supported by the R+D project TIN2004-08000-C03-02 of the Spanish MCYT.

References

1. Álvarez, J., Bañares, J.A., Muro-Medrano, P.R.: An Architectural Pattern to Extend the Interaction Model between Web-Services: The Location-Based Service Context. M.E. Orłowska et al. (Eds.): ICSOC 2003, LNCS 2910, pp. 271–286, 2003
2. Architecture Working Group: Recommended Practice for Architectural Description of Software-Intensive Systems. IEEE Std 1471 (2000)
3. Beaudouin-Lafon M.: Computer Supported Cooperative Work. Université Paris-Sud, France, John Wiley & Sons (1999)
4. Cabri, G., Leonardi, L., Zambonelli, F.: XML Dataspaces for Mobile Agent Coordination. In 15th ACM Symposium on Applied Computing, pages 181–188, 2000
5. Carriero, N., Gelernter, D.: Linda in Context. Communications of the ACM, Vol. 32, No. 4, 444-458, April 1989
6. Chung, G., Dewan, P.: Towards Dynamic Collaboration Architectures. Proc. ACM CSCW'04, 1-10
7. Ellis, C.A., Gibbs, S.J., Rein, G.L.: Groupware: Some Issues and Experiences. Communications of the ACM, Vol. 34, No. 1 (January 1991) 38-58
8. Garrido, J.L., Gea, M.: Modelling Dynamic Group Behaviors. In: Johnson, C. (ed.): Interactive Systems - Design, Specification and Verification. LNCS 2220. Springer (2001) 128-143
9. Garrido, J.L., Gea, M.: A Coloured Petri Net Formalisation for a UML-Based Notation Applied to Cooperative System Modelling. In: Forbrig, P. et al (ed.): Interactive Systems - Design, Specification and Verification. LNCS 2545. Springer (2002) 16-28
10. Garrido, J.L., Gea, M., Rodríguez, M.L.: Requirements Engineering in Cooperative Systems. Requirements Engineering for Sociotechnical Systems. IDEA GROUP, Inc.USA (2005)
11. Garrido, J.L., Padereswki, P., Rodríguez, M.L., Hornos, M.J., Noguera, M. A Software Architecture Intended to Design High Quality Groupware Applications. Proc. of the 4th International Workshop on System/Software Architectures (IWSSA'05), Las Vegas (USA), June 2005
12. Gelernter, D., Carriero, N.: Coordination Languages and Their Significance. Communications of the ACM, Vol. 35, No. 5, 96-107, February 1992
13. Grudin, J.: Groupware and Cooperative Work: Problems and Prospects. In Baecker, R.M. (ed.) Readings in Groupware and Computer Supported Cooperative Work, San Mateo, CA, Morgan Kaufman Publishers (1993) 97-105
14. Gutwin, C., Penner, R., Schneider, K. Group Awareness in Distributed Software Development. Proc. ACM CSCW'04, 72-81 (2004)

15. Jacobson, I., Booch, G., Rumbaugh, J.: *The Unified Software Development Process*. Addison-Wesley (1999)
16. Johanson, B. and Fox, A., "The Event Heap: A Coordination Infrastructure for Interactive Workspaces". In Proc. of the 4th IEEE Workshop on Mobile Computer Systems and Applications (WMCSA-2002) Callicoon, New York, USA (June, 2002)
17. Macaulay, L.A.: *Requirements Engineering*. Springer (1996)
18. Mehra, A., Grundy, J., Hosking, J.: Supporting Collaborative Software Design with a Plug-in, Web Services-based Architecture. Proc. of the ICSE 2004 Workshop on Directions in Software Engineering Environments, May 2004, Edingurgh, Scotland, IEE Press
19. Object Management Group: Unified Modelling Language (UML) 2.0 Superstructure Specification (OMG), August 2003. Ptc/03-08-02, 455-510
20. Papadopoulos G.A. and Arbab F.: Coordination models and languages. *Advances in Computers*, 46, Academic Press (1998)
21. Paternò, F.: *Model-based Design and Evaluation of Interactive Applications*. Springer-Verlag (2000)
22. Patterson, J., Day, M., Kucan, J.: Notification Servers for Synchronous Groupware. In Proc. of the ACM Conference on Computer-Supported Cooperative Work (CSCW'96) 122-129
23. Phillips, W.G.: *Architectures for Synchronous Groupware*. Technical Report 1999-425, Department of Computing and Information Science, Queen's University, May 1999
24. Roseman, M. and Greenberg, S.: Building Real Time Groupware with GroupKit, A Groupware Toolkit. *ACM Transactions on Computer Human Interaction* 3(1), (March 1996)
25. Schlichter J., Koch M., Burger M.: Workspace Awareness for Distributed Teams. In: W. Conen, G. Neumann (eds.), *Coordination Technology for Collaborative Applications*, Springer Verlag, Heidelberg (1998)
26. Shen, H., Sun, C.: "Flexible Notification for Collaborative Systems" In Proc. of ACM 2002 Conference on Computer Supported Cooperative Work (CSCW'02), New Orleans, USA (2002) 16-20
27. Sommerville, I.: *Software Engineering*. Addison-Wesley (7th ed.) (2004)
28. Sun Microsystems: *Jini Distributed Events Specification*. Version 1.0. Available on-line at <http://java.sun.com/products/jini/2.1/doc/specs/html/event-spec.html>
29. Sun Microsystems: *JavaSpaces Service Specification*. Version 2.2. Available on-line at <http://java.sun.com/products/jini/2.1/doc/specs/html/js-spec.html>
30. Tolksdorf, R., Glaubitz, D. XMLSpaces for Coordination in Web-based Systems. Tenth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 06 20 - 06, 2001
31. van der Veer, G., Lenting, B., Bergevoet B.: GTA: Groupware Task Analysis - Modelling Complexity. *Acta Psychologica*, 91 (1996) 297-322
32. Wyckoff, P., McLaughry, S., Lehman, T., and Ford D.: TSpaces. *IBM Systems Journal*, 37(3): 454-474, (1998)

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 assemble 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,

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 [Szyperki 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) [Szyperki 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.

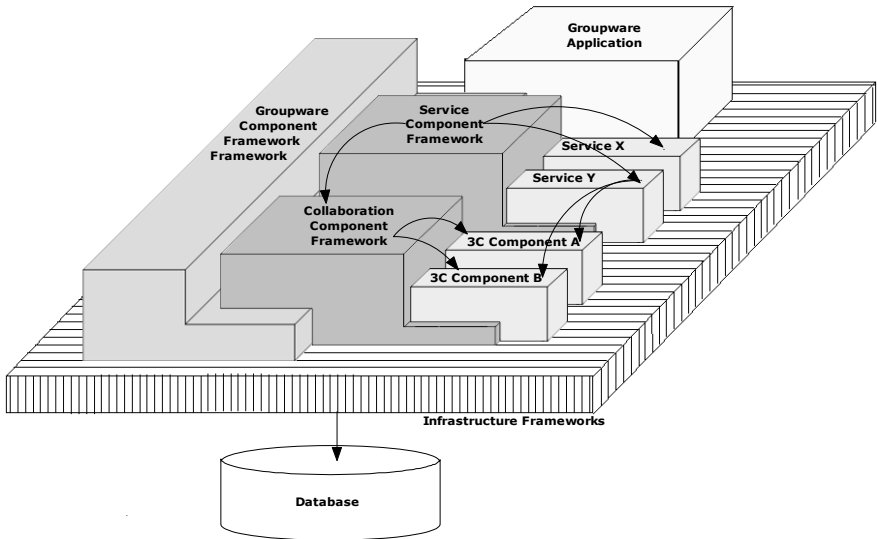


Fig. 1. The proposed architecture

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.

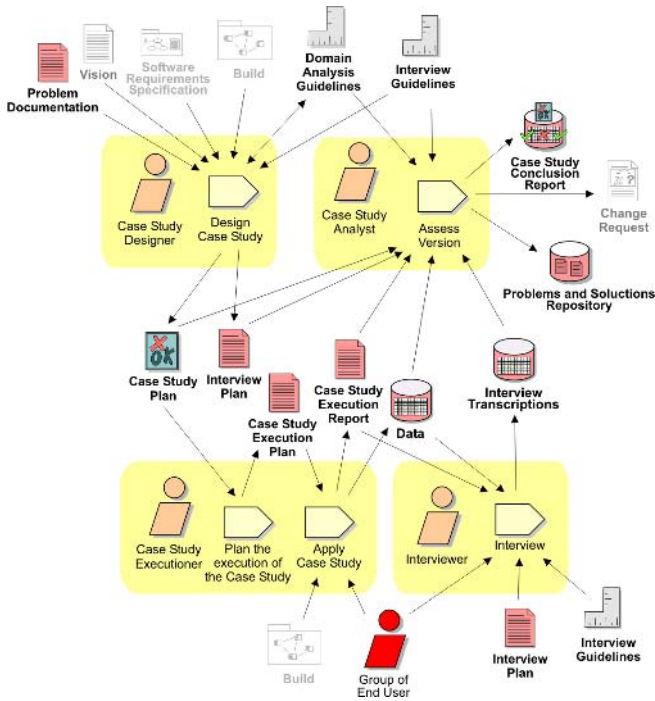


Fig. 2. Case Study activity of RUP-3C-Groupware (RUP's original artifacts are dimmed)

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

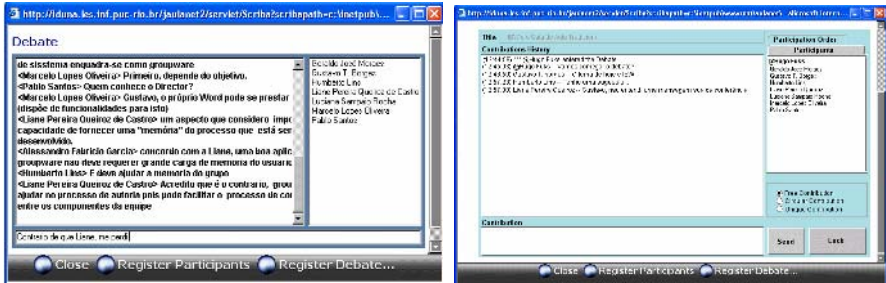


Fig. 3. Early Debate interface (left) and current Debate interface (right)

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 [Litu & 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.

Ontoolcole: An Ontology for the Semantic Search of CSCL Services

Guillermo Vega-Gorgojo¹, Miguel L. Bote-Lorenzo¹, Eduardo Gómez-Sánchez¹, Juan I. Asensio-Pérez¹, Yannis A. Dimitriadis¹, and Iván M. Jorrín-Abellán²

¹ School of Telecommunications Engineering,

² Faculty of Education

University of Valladolid, Camino del Cementerio s/n, 47011 Valladolid, Spain
{guiveg, migbot, edugom, juaase, yannis}@tel.uva.es, ivanjo@doe.uva.es
<http://gsic.tel.uva.es>

Abstract. Collaborative learning systems can benefit from service-oriented computing, allowing educators to integrate external tools, offered as services by software providers, in order to support the realization of collaborative learning situations. Since finding and selecting appropriate services is a challenging issue, the Ontoolcole ontology has been developed in order to support the semantic search of CSCL services intended for the use of educators. This paper presents significant enhancements made in Ontoolcole from a prior version. Namely, Ontoolcole incorporates an artifact module, a task-level coordination module and the description of static information resources, further improving the capabilities to describe complex CSCL tools such as stateful applications or decomposable group tasks. An experiment with real educators has been carried out to evaluate whether Ontoolcole can be employed by educators to search CSCL services. Evaluation results show that Ontoolcole's abstractions can fit educators' questions based on their real practice while retrieving useful tools for their educational needs.

1 Introduction

Computer Supported Collaborative Learning (CSCL) [1] is a discipline that promotes the use of Information and Communication Technologies within the context of collaborative learning. The so-called *collaborative learning systems* (CLSs) typically offer an environment with several tools that users may employ to accomplish some learning activities. With the increasing popularity of *service-oriented computing* [2], many CLSs, e.g. Gridcole [3], have embraced this paradigm allowing diverse organizations to offer software tools as services for the realization of a wide range of collaborative (besides individual) learning activities.

One desirable feature for CLSs is *tailorability* [4]. A tailorable application provides users with some means to modify its functionalities so as to better fit their needs [5]. In this sense, tailorable CLSs enable educators to easily integrate external tools in order to support the realization of new situations. Service-oriented computing can help the development of tailorable CLSs, since an educator can

tailor a service-oriented CLS to her setting by selecting adequate services among those offered by service providers [3]. Besides, service-oriented computing can promote *educational software reuse* since an available service can be employed to support different learning activities.

However, service-oriented computing introduces the challenge of *service discovery* since consumers need to find the most appropriate services for their current needs among the available ones. In the context of CSCL, educators are expected to perform the search of services in order to tailor a CLS to support their collaborative scenarios. These services are coarse-grained and directly consumable by learners and teachers through user interfaces, as opposed to other intermediary services primarily intended to be composed with others. Therefore, a service discovery facility should allow educators to query for their educational capabilities, rather than their low-level technical details. Moreover, educators should be able to query for their collaboration properties since they are critical to enable or not a collaborative scenario. Throughout this paper we will use the term *CSCL service* to refer to any service-based tool that may be discovered by educators and integrated in a CLS system. Note that, within this view, a CSCL service is not necessarily designed for learning nor it is required to be collaborative.

Concerning the type of searches, two very different types can be distinguished: syntactic and *semantic searches* (named navigational and research searches in [6], respectively). In the former type, the user provides the search engine with a combination of keywords, retrieving a set of documents based on the appearance of those keywords using Information Retrieval techniques [7]. In contrast, a semantic search is intended to gather information compliant with the semantics of the query. A search engine can achieve this by exploiting *ontologies* [8] that are used to explicitly formalize knowledge, enabling rich descriptions and robust information retrieval. We are interested in this latter type of searches for the discovery of CSCL services since they allow much more expressive queries and enhanced precision in service retrieval [9].

Current service discovery approaches are commonly accomplished using well-known registries. Service providers publish service metadata in registries while consumers query them to find appropriate services for their needs. However, standard registries, such as UDDI [10] in the popular Web Services architecture [11], only support syntactic searches. Moreover, just a restricted set of queries are possible, e.g. business entities and the services they provide in UDDI (similar to the Yellow Pages). *Semantic Web Service* (SWS) technology [12] promises to automate Web Service discovery allowing semantic searches, as well as automatic invocation and composition. OWL-S [13] is perhaps the most remarkable initiative for the semantic description of Web Services.

Our previous work [14] discussed these different approaches for the discovery of CSCL services and proposed an ontology of CSCL tools with the aim of allowing educators to perform semantic searches of service-based CSCL tools using learning concepts. Such an ontology describes tools through the supported tasks that can be performed by an actor (probably playing a role). In this

paper, we present an evolution of this ontology, named *Ontoolcole*, that introduces significant changes. Specifically, *Ontoolcole* incorporates an artifact module, a task-level coordination module and the description of static information resources. These changes dramatically improve the capabilities of *Ontoolcole* to describe complex CSCL tools (although simple tools can also be described in an easy way) such as stateful applications or decomposable group tasks. Besides, in order to evaluate if educators can benefit from using *Ontoolcole* for the semantic search of CSCL services, we have carried out an experiment with some practitioners posing questions from their real practice.

The rest of this document is organized as follows: section 2 presents an overview of *Ontoolcole* and discusses the limitations of a prior version. Section 3 depicts the new features introduced to overcome previous limitations and illustrates both the description of tools with *Ontoolcole* and the architecture of an *Ontoolcole*-powered service discovery system. Next, section 4 evaluates whether *Ontoolcole* can be employed by educators to search CSCL services. Finally, the main conclusions of the study are shown as well as current research work.

2 Ontoolcole: The Vision

Ontoolcole is an ontology for the description of collaborative learning tools, allowing the semantic search of CSCL services. This section first shows the design criteria that guided *Ontoolcole*. Then, a brief overview of *Ontoolcole* is presented, as well as the discussion of the limitations of the previous work [14].

2.1 Design Criteria

As discussed in [14], existing approaches for service discovery do not fit well into educators' needs for the search of CSCL services. Focusing on semantic searches, emerging frameworks for SWS, such as OWL-S, provide a generic Web Service description ontology that must be extended with a domain ontology to capture specific knowledge. However, service descriptions are limited to a low-level input/output transformation view. While this can be appropriate for business-to-consumer (B2C), many CSCL applications are interactive and have collaborative features that require other higher-level abstractions. This is the motivation for *Ontoolcole*, an ontology for the description of collaborative learning tools.

Ontoolcole aims at describing CSCL tools from their users' point of view, on the assumption that educators will search for service-based CSCL tools querying about their functionality. In this sense, *Ontoolcole* does not try to capture other aspects such as implementation details which concern mainly service providers. Decoupling the description of tools from technical details allows the use of *Ontoolcole* in a service discovery scenario, but it also offers new opportunities to the CSCL domain, such as a recommender system of learning tools for CSCL practitioners.

After concluding that the focus on tool descriptions should be put on supported functionality, a key design decision is to determine the level of detail, since

there is a strong trade-off between complexity in the description and utility for the search. A big, complex and powerful model may require too much effort to provide comprehensive tool descriptions and to query. On the other hand, a very simple model may result of little value for the final user. For instance, [15] provides a thorough conceptual model of groupware intended to compare and describe a wide range of groupware systems from their users' point of view. However, this model describes groupware at the object and the operation level, e.g. "*A drawing tool offers objects like straight lines, points, curved lines, [...]*". In a search scenario, supporting fine-grained queries about these aspects requires exhaustive tool descriptions, while in most cases it would be sufficient just to offer a higher level vision of their functionality. Thus, the decision in Ontoolcole has been to describe the tasks that can be performed with a tool (see section 2.2). Interestingly, a similar approach was taken in [16] for the search of bioinformatics services.

Besides, the search process is meant to be semi-automatic in order to allow the user to take the final decision in the service selection, reducing the required complexity of the model. In this sense, Ontoolcole tool descriptions do not need to be exhaustive but sufficient for an initial filtering from which the educator can then choose. Nevertheless, Ontoolcole has been constructed with the aim to be extensible and new extensions can leverage the automation of service discovery.

Finally, Ontoolcole has been formalized in OWL DL [17], a widespread and expressive language with definite semantics that allows the use of off-the-self Description Logic (DL) classifiers such as RacerPro [18]. A DL classifier can support the search of services by reasoning whether a service description of the set of available services is relevant to a query.

2.2 Ontoolcole Overview and Previous Limitations

An ontology specifies an explicit conceptualization of a domain that can be exploited by information systems such as a retrieval engine to organize information and direct the search processes [8]. In this sense, the Ontoolcole ontology specifies that a **Tool** supports one or more **Tasks** that are performed by an **Actor**, maybe playing a specific **Role**, as shown in Fig. 1. The realization of a **Task** may require an **Artifact** as an input or may produce an **Artifact** as an output. This simple schema is the basis of the description of tools in Ontoolcole and it has been employed with small variations in the CSCL/CSCW literature [19]. However, Ontoolcole extends this model offering five prototypical task types that can be further specialized, specifically **Perception**, **Construction**, **Communication**, **InformationManagement** and **Computation**. These terms are a reification of the uses that may be served by a tool [1]. This change is motivated in order to reflect the actor's point of view.

Each task type has its specific properties. A perception task, such as reading or hearing, can only be performed by a person or a group and require some artifact as input. Similarly, a construction task, such as writing or modeling, is performed by either a person or a group and produces some artifact as a result

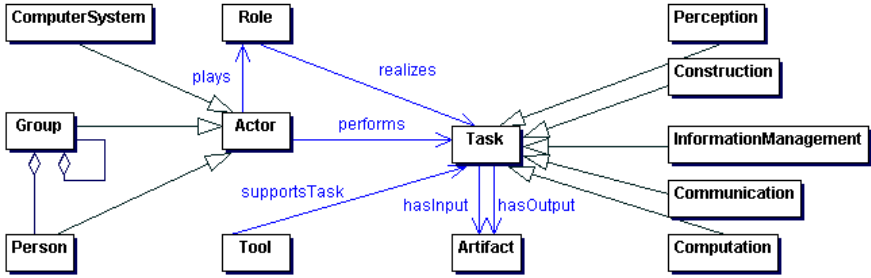


Fig. 1. Overview of the previous information model of Ontoolcole

of the task. A communication task is explicitly performed by a group exchanging messages of different types (text, graphics, audio, video or documents). In contrast, an information management task can be specialized to publishing, retrieving, searching, sending or deleting some artifact. Finally, a computational task, such as compilation or computer simulation, is always performed by a computer system transforming some inputs into some outputs. Collaboration work is naturally described in this model by specifying group tasks and the roles that participants may play during the realization of a task. Moreover, group tasks can be annotated in Ontoolcole as either synchronous or asynchronous.

Combining these elements, a wide range of tool types can be described. These tool definitions can be very generic, e.g. “communication tool: any tool that supports a communication task”, or more specific, e.g. “whiteboard: any tool that allows a group to draw synchronously an image”. In a similar way, tool instances can be described using Ontoolcole.

This brief description depicts Ontoolcole as in [14]. However, this preliminary version had three main limitations that were detected when using it to describe tool instances. First of all, it lacks an artifact model to accommodate the different products that may be used or created during the realization of a task. This is a critical aspect to differentiate the capabilities of a tool and the previous version did not offer a mechanism for this issue. For instance, many tools can support a writing task, but there are differences on the document types that can be edited (e.g. a questionnaire) and on the format types that a tool can support.

Second, while Ontoolcole seems appropriate to describe tools designed for single tasks (for example a simple text editor), there are many CSCL tools that embody a complex workflow of tasks (for example a questionnaire management tool) that cannot be described with this simple schema. Some kind of coordination model is required for the description of the dynamics aspects of a tool.

Third, some tools such as document repositories or bulletin boards keep some kind of archival storage that can be accessed during different task realizations. Thus, some mechanism is required to describe the static information resources that a tool can manage. Limiting the description of these capabilities precludes queries involving important features, such as the capability of storing text documents or supported format types of an artifact.

3 Ontoolcole: New Features and Operation

This section first depicts the new features introduced in Ontoolcole in order to overcome the three limitations depicted in section 2.2. Then, it illustrates how CSCL tools can be described with Ontoolcole. Next, the architecture of an Ontoolcole-powered semantic service discovery system is outlined.

3.1 New Features and Description of Tools

As discussed earlier, an important limitation of the previous version of Ontoolcole was the **lack of an artifact model**, which hindered the description of tools and reduced the capability of tool discrimination. This new model should help to describe the artifacts required or produced during a task realization.

The resulting artifact model is shown pictorially in Fig. 2. The main design criterion has been to separate the artifact type from the format type. The well-known MIME types [20] have been employed to define the artifact format types. Interestingly, metadata standards such as LOM [21] or Dublin Core Metadata Initiative [22] also separate the content from the format. In contrast, our approach makes heavy use of inheritance to structure the artifact types, and explicitly defines a vocabulary that can be shared both by providers to describe tools and by educators to submit their queries, leveraging the search process. Moreover, this artifact model further describes some elements (although not shown in Fig. 2). For instance, a **TextDocument** incorporates a property to specify the type of text document (generic, questionnaire, source program, etc.).

Besides, this artifact model is not intended to provide a very detailed artifact model to avoid increased complexity and low usability, as discussed in section 2.1. Moreover, new artifact types can be easily incorporated to allow the description of a wider range of tools since we are aware that domain-specific tools, such as network simulators, will require extensions to this model.

In our previous work, we detected the **need to specify the resources a tool can store**. To solve this, a new property has been defined that relates

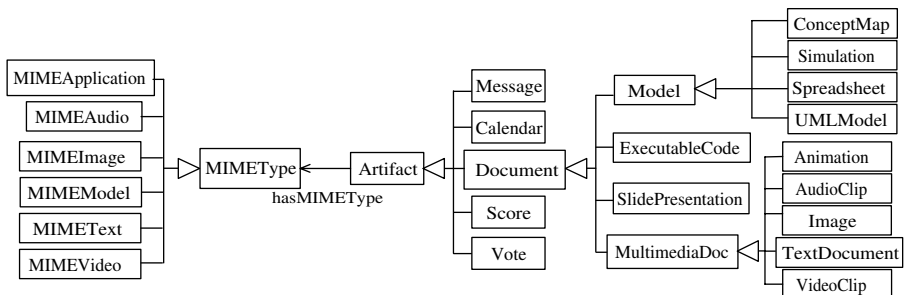


Fig. 2. The new artifact model for Ontoolcole represented as a UML class diagram. Classes on the right depict artifact types while classes on the left the well-known MIME types. A specific artifact instance, e.g. a **TextDocument**, should be related to a MIME type instance, e.g. **application/pdf**.

tools with the artifacts they can manage, exploiting the artifact model depicted above. With this simple approach it is possible to specify that an e-mail facility can store messages enclosing text and document files, for example.

Finally, the previous version did not allow the **description of complex workflows of tasks**. This is, perhaps, the most challenging limitation, and becomes quite important when describing asynchronous group scenarios, since different users may perform individual tasks that conform a whole complex task. Furthermore, the utility of this feature is not limited to such cases; for instance, participants engaged in a synchronous group drawing task may interchange text messages at the same time to coordinate their drawing. Thus, some mechanism is necessary to describe the concurrent realization of the drawing and communication tasks.

Fig. 3(a) outlines the coordination model developed for Ontoolcole. Basically, a **CompositeTask** has been defined as any task that is composed of exactly one **ControlConstruct** element. This latter element acts as a container of tasks or other control control constructs. Each control construct has different semantics: **Choice** implies that only one of the elements contained is performed; **Sequence** defines a time ordering; and **Split** serves to specify the concurrent execution of the elements contained. Combining these elements, it is possible to describe a wide range of complex workflows of tasks. Indeed, this coordination model is inspired in OWL-S Composite Processes [13], although adapted to a task context and using different OWL DL constructs, such as transitive composition properties, to aid in tool classification and query.

An example of the use of this coordination model is shown in Fig. 3(b), representing a possible scenario of an asynchronous group writing task. This description provides a high-level overview of the group task (top of Fig. 3(b)) that

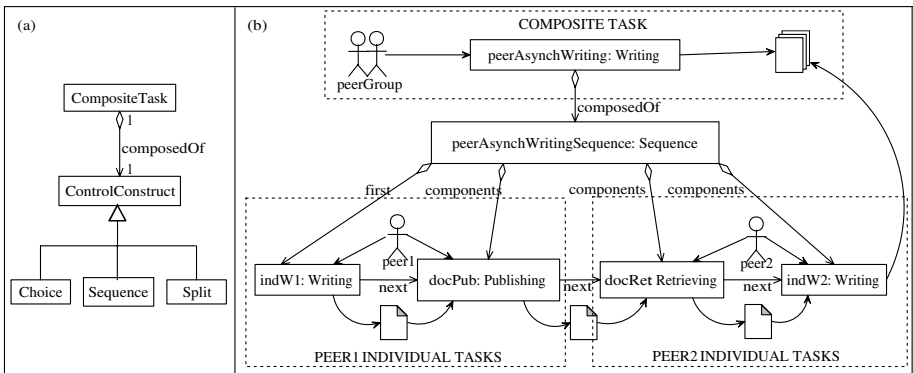


Fig. 3. Simplified view of Ontoolcole’s new coordination model (a). A **CompositeTask** is composed of a **ControlConstruct** element that includes other tasks or other control constructs. (b) illustrates the use of the coordination model for the description of an asynchronous group writing. A text document is produced as a result of a sequence of individual tasks performed by each of the members of the peer group.

is further refined with the description of the individual tasks performed by the members of the group (bottom of Fig. 3(b)). Remarkably, the sequence ordering is defined by the use of the `first` and `next` properties, pointing to the first and the subsequent tasks, respectively.

After overcoming the previous limitations, Ontoolcole has proven to be very versatile for the description of CSCL tools. Different instances of document repositories, drawing tools, whiteboards, collaborative text editors, multimedia players, questionnaire management tools, chats and concept map tools, among others, have been described with Ontoolcole. Exploiting Ontoolcole abstractions, tools are described by the depiction of their different scenarios of use. For instance, Fig. 3(b) forms the basis for the description of CoWeb [23], a popular asynchronous group text editor based on Wiki. Further aspects of CoWeb have been included such as group sizes, the storage of text documents and images, document formats as well as the description of other CoWeb supported scenarios: individual text edition and retrieval of stored documents.

We can draw out some lessons learned from the experience of describing tools with Ontoolcole. Since the emphasis of Ontoolcole is on the description of CSCL tools, it was necessary to provide the mechanisms for specifying their collaboration capabilities. This has been achieved by describing CSCL tools from the user's perspective, as discussed in section 2.1. In this sense, Ontoolcole is by no means limited to collaborative tools, since it can accommodate a broad range of individual applications, such as multimedia players or browsers. Besides, in order to simplify tool descriptions, only representative scenarios of use should be described. For instance, CoWeb can support many different variations of group writing but describing all the possible scenarios would only help to find matches to odd queries. Thus, a representative scenario can accommodate the majority of queries and can be used to offer a sample view of a tool when formatting query results to users. However, describing a tool requires a deep understanding of Ontoolcole in order to provide comprehensive descriptions. Furthermore, the popular ontology editor Protégé [24] has been employed for this issue and using Protégé needs some training for non-trivial usage. Thus, ongoing efforts should be carried out to develop an Ontoolcole-flavoured authoring system to allow providers to describe their own tools. The following further discusses this issue in the context of an Ontoolcole-powered service discovery infrastructure.

3.2 Using Ontoolcole for Semantic Service Discovery

As it has been shown in section 3.1, Ontoolcole can be exploited to provide semantic descriptions of CSCL tools. Such tools can be offered as services by external service providers and integrated in a service-oriented CLS as, for example, Gridcole [3]. Since Ontoolcole does not deal with implementation or technological details, it can be seamlessly employed to describe the functionality of services, enabling the semantic search of services using Ontoolcole abstractions.

In this sense, Ontoolcole can coexist with current service registries, such as UDDI [10]. An educator can query Ontoolcole and eventually select an appropriate tool. The entry found in Ontoolcole will point to one or more UDDI records

Table 1. Ontoolcole’s supported query types

| ID | Query Type | ID | Query Type |
|----------|--|-----------|-----------------------------|
| ToolType | Tool type | Storage | Allows artifact storage |
| Group | Supports or not group realization | GroupSize | Min/max group size |
| Time | Synch/asynch group task | TaskType | Supported task type |
| CompTask | Elements in a composed task | Ordering | Task ordering in a sequence |
| ArtType | Artifact type required as input/output | ArtMIME | MIME type of an artifact |

of registered services that implement the selected tool. Although registries store all the technical information (such as the WSDL interface) required to invoke a service, a CLS may impose additional constraints. For instance, it may restrict to those services adhering to the WSRP specification [25] to facilitate the end user interaction with a service. Note that integrating a service in a CLS is out of the scope of Ontoolcole.

Querying Ontoolcole semantic metadata requires a DL classifier such as *RacerPro* [18]. The classifier is fed both with the Ontoolcole ontology and the tool annotations to reason which ones comply to submitted queries. A schematic overview of the query types that Ontoolcole supports is shown in table 1. These query types map to Ontoolcole abstractions and can be combined to articulate specific query expressions in a language such as nRQL [18] to interrogate *RacerPro* for services. Hence, new features described in section 3.1 enabled query types *Storage*, *CompTask*, *Ordering*, *ArtType* and *ArtMIME*.

Concerning users, it is quite relevant to evaluate if Ontoolcole’s supported queries fit educators’ needs. This issue is tackled in section 4, depicting an experiment with educators. Besides, it is not feasible to assume that educators will learn a complex syntax such as nRQL to formulate their queries. Although this is more a human-computer interaction issue than an Ontoolcole’s limitation, it directly affects the usability of the system. To overcome this problem, we are currently working on *Ontoolsearch*, a visual query formulation facility for Ontoolcole. Using graphical interfaces to exploit ontology-based applications is a recent research topic with some initial solutions as [26]. In a similar way, a specialized service annotation facility for Ontoolcole would help service providers to generate semantic descriptions of their offered services. In the meantime, knowledge engineering experts can describe them using ontology editors as *Protégé*.

4 Ontoolcole: Evaluation

Sections 2 and 3 portrayed Ontoolcole and showed the architecture of an Ontoolcole-based service discovery infrastructure. Since Ontoolcole has been designed with the aim of facilitating to educators the search of CSCL services, it is necessary to evaluate if Ontoolcole accomplishes this goal. Thus, the emphasis should be on assessing whether educators’ real questions for services can be formulated with Ontoolcole. Besides, we want to assess if retrieved services satisfy users’ expectations for the questions they posed. To pursue these objectives, we

Table 2. User profiles of the participants in the experiment. ‘CS’ stands for ‘Computer Science’, ‘TM’ for ‘Telematics’, ‘ST’ for ‘Signal Theory’ and ‘ED’ for ‘Education’.

| Faculty Position | # of Users | Area of Knowledge | | | | CSCL Practitioner | | Ontoolcole Knowledge | | |
|------------------|------------|-------------------|----|----|----|-------------------|----|----------------------|------|----|
| | | CS | TM | ST | ED | Yes | No | Yes | Fair | No |
| Professor | 5 | - | 3 | - | 2 | 5 | - | 2 | 1 | 2 |
| Teacher | 8 | 2 | 3 | 1 | 2 | 4 | 4 | 2 | 1 | 5 |
| Grant holder | 1 | 1 | - | - | - | - | 1 | - | - | 1 |
| Student | 1 | - | 1 | - | - | - | 1 | 1 | - | - |
| Total | 15 | 3 | 7 | 1 | 4 | 9 | 6 | 5 | 2 | 8 |

have devised an experiment engaging several educators to pose non-restricted questions from their real practice.

Interestingly, some methodologies for ontology evaluation such as [27] propose the use of competency questions defined *a priori* and tested afterwards against an ontology to evaluate its compliance. In contrast, the proposed experiment involves real users to express their information needs based on their practice and without any restriction on the questions posed. Hence the goal of the former methodology is to evaluate whether an ontology can respond to a subset of predefined questions, while the aim of the devised experiment is to assess whether a specific ontology can fit the information needs of a community of users.

4.1 The Experimental Setup

In order to evaluate whether educators can search CSCL services using Ontoolcole we devised the following experiment. 15 education practitioners were recruited and demanded to formulate questions for tools in the context of their real practice. These abstract demands must be translated into well-formed queries to be submitted to an Ontoolcole-powered retrieval system. For this role we have employed a RacerPro system fed with a test knowledge base. Since the graphical query facility, Ontoolsearch, was not available at the time of the experiment, a human **mediator** has been in charge of translating user questions into nRQL queries. Translated queries have been paraphrased back to natural text and returned to participants in addition to the tools found by RacerPro. Nevertheless, the mediator hides the human-computer interaction issue while enabling the evaluation of the initial goals of the experiment. Finally, users have provided feedback about the translation quality and the utility of the found tools for their educational settings.

An overview of the profiles of the 15 participants in the experiment is shown in table 2. Most of them are professors or teachers either in Computer Science, Telematics, Signal Theory or Education. Remarkably, 9 out of 15 are practitioners in CSCL and 7 have some knowledge about Ontoolcole. It was expected that CSCL practitioners posed questions about CSCL settings, although they were not obliged to. Besides, evaluation results might be influenced by prior users’ knowledge about Ontoolcole. Section 4.2 further discusses these aspects.

Table 3. Overview of the context of the analysed questions

| Degree Studies | # of Questions | Setting | | | CSCL | |
|------------------------|----------------|--------------|---------|---------|------|----|
| | | Face-to-face | Distant | Blended | Yes | No |
| MSc Computer Science | 8 | 3 | 1 | 4 | 3 | 5 |
| MSc Telecommunications | 20 | 13 | 4 | 3 | 12 | 8 |
| PhD Telecommunications | 7 | 1 | 3 | 3 | 6 | 1 |
| MSc Education | 5 | 1 | 1 | 3 | 3 | 2 |
| Total | 40 | 18 | 9 | 13 | 24 | 16 |

In addition, a reasonable amount of educational tools should be described using Ontoolcole, in order to validate the retrieval part. For the experiment we prepared 21 tool descriptions, including text editors, browsers, document visualizers, authoring tools, document repositories, questionnaire management tools, chats, e-mail, drawing tools, spreadsheets and concept map tools. Since most of the tools have been employed in some courses following a CSCL methodology in our University, we expected they could be useful for questions referred to CSCL settings. However, the used dataset is too small to find adequate tools for very specific needs. In this sense, a bigger dataset would be valuable for a deeper analysis of the query results. This is part of future work.

Hence, in a complete scenario, after choosing a tool the user should select one of the available service implementations, as discussed in section 3.2. However, this step would add unnecessary complexity to the experiment reported in this paper, since it is not required for the evaluation goals considered here.

4.2 Evaluation Results

The participants in the experiment formulated 40 questions. An overview of the context of these questions is shown in table 3. They refer to courses on Computer Science, Telecommunications and Education as well as a doctoral course on Research Methodology. Besides, questions present a balanced mixture of face-to-face, distant and blended settings, while 24 out of 40 questions were conceived for the practice of CSCL. They refer to different activities such as UML modelling, signal/noise calculations, network simulations, design of didactic units, HTML authoring, group writing or document sharing among others. Some of them were very specific and used similar abstractions as those supported by Ontoolcole. For instance one participant asked for “*a tool for visualizing PDF documents*” and it was easily translated to nRQL without any information loss. However, in other cases questions were more abstractly defined and used different concepts as those modelled in Ontoolcole. For example, the question “*I’m searching a tool that supports working with your companion, although not only working, but requiring sharing information between them, not just dividing the work in two parts. Besides, it should allow seeing what your companion makes and support communicating with her in an easy way*” was paraphrased to “any tool that supports the construction of artifacts

Table 4. Summary of the evaluation results. Levels are: 5 (very good), 4 (good), 3 (somewhat good), 2 (somewhat bad), 1 (bad) and 0 (very bad). The first two rows distinguish questions referred or not to CSCL settings. Following three rows present the evaluation results according to the users’ knowledge of Ontoolcole. Final row shows the aggregated figures.

| Question Set | | Translation Understanding | Quality of the Translation | Found Tools (%) | Service Utility |
|------------------------|----|---------------------------|----------------------------|-----------------|-----------------|
| CSCL | 24 | 4.88 | 3.85 | 87% | 4.32 |
| Non-CSCL | 16 | 4.71 | 3.29 | 75% | 2.21 |
| Ontoolcole-aware: Yes | 16 | 4.94 | 3.28 | 100% | 3.02 |
| Ontoolcole-aware: Fair | 10 | 4.70 | 3.40 | 70% | 4.43 |
| Ontoolcole-aware: No | 14 | 4.75 | 4.33 | 71% | 3.80 |
| Total | 40 | 4.82 | 3.64 | 82% | 3.55 |

in group and, besides, supports the communication among group members” in the nRQL syntax. Hence the query does not include the awareness part of the question since it cannot be expressed with Ontoolcole.

Thus, the 40 questions were translated into nRQL queries and submitted to RacerPro. In some cases various queries were produced for a single question in order to disambiguate different possible interpretations. With the translated queries and the retrieved tools, a personalized response was sent to the participants of the experiment. Users’ feedback was collected concerning the quality of the translated queries and the usefulness of found tools. Table 4 summarizes the quantitative results.

Figures under column ‘Translation Understanding’ show that most participants understood very well the translated queries, indicating that Ontoolcole’s abstractions are comprehensible. Significantly, users ranked with an overall 3.64 (out of 5) the quality of the translations. Taking into account that users were not restricted on the questions they posed, this figure can be considered very positive. Interestingly, questions referred to CSCL settings were ranked a 17% better than non-CSCL. Since Ontoolcole emphasises the description of collaborative features, this difference is understandable. Curiously, users aware of Ontoolcole were more demanding than the rest (3.28 vs 4.33), although the mediator feels that some of the questions were thought as a challenge to Ontoolcole.

Besides the quantitative figures, participants provided valuable comments about translated queries that can help to explain the results in table 4. In this sense, some users formulated questions at an upper level than the one supported by Ontoolcole, using high-level abstractions such as “*group memory*” and “*pre-sential debates*”. In contrast, these were translated to “**stored documents that can be accessed by a group**” and “**synchronous communication**”. Users were aware of this ‘semantic gap’ although they considered that the essence of the questions was preserved in the queries. In other cases there were some misunderstandings due to natural language ambiguity. For instance, the mediator interpreted that a user demanded a programming environment when she required a tool for program designing (possibly a UML modelling application).

Finally, non-CSCL questions were sometimes too specific, e.g. “*network simulators including the TCP protocol*”, requiring some kind of extension.

Concerning the tools found, there are revealing differences between questions referred or not to CSCL. As discussed in section 4.1, most of the tools in the test knowledge base are used in CSCL settings. This explains that users considered much more useful those tools found for CSCL-related questions, since they can accommodate a wide range of CSCL activities such as debates or collaborative group work. Besides, questions not referred to CSCL were sometimes very domain-specific, demanding network simulators, specialized computer graphics applications or very specific authoring tools for psycho-pedagogy, just to mention a few. Such questions matched no instance, or rather generic ones. However, the core of Ontoolcole (tasks, artifacts, etc.) fits these queries but extensions concerning specific artifact types (e.g. data-network model) would increase the precision of responses. In this sense, specialized extensions for Ontoolcole seem a feasible approach to exploit Ontoolcole in other non-CSCL domains.

To conclude the analysis, users suggested some enhancements to leverage Ontoolcole. Excluding specialized domain-extensions, CSCL users demanded the description of awareness, advanced commenting and meeting capabilities. Other required technological features such as supported OS or QoS parameters. In order to further analyse these aspects related to Ontoolcole extensibility, a final questionnaire was submitted to participants demanding them to rank Ontoolcole’s supported query types (see table 1) and 11 possible extensions extracted from users’ comments. Interestingly, although users did not know which query types Ontoolcole supported (the questionnaire included a disordered list of items), the top-five corresponded to existing `ToolType`, `TaskType`, `Group`, `Time` and `ArtType` query types in table 1. On the other hand, the bottom-five matched extensions suggested by users. Nevertheless, users ranked positively extensions for awareness and domain-labelling and we are currently considering the inclusion of some suggested new features in Ontoolcole.

4.3 Related Work

OWL-S describes service functionality in terms of the transformation produced by the service (represented by inputs and outputs) and the state change produced by the execution of the service (represented by preconditions and effects). In this sense, OWL-S allows the semantic description of service interfaces as description logic-based OWL classes in order to support automatic service invocation and composition. However, it is arguable that this kind of information is appropriate to search a service. On the one hand, it compels users to formulate questions at the interface level and in the experiment we carried out none of the questions was at this level of detail. On the other hand, the same service functionality can be offered with different interfaces utilizing other combinations of operations and message types, severely hindering the capability of finding an appropriate service. Our experiment shows that real questions proposed by education practitioners are very difficult to translate to OWL-S low-level interface-oriented

queries. Interestingly, these problems have been also reported for user oriented semantic service discovery in the very different domain of bioinformatics [28].

Finally, some comments can be made about syntactic and semantic searches. Traditional syntactic searches deal with textual information. However, some kind of metadata is required for the search of services since they are software components that cannot be directly queried. In this sense, a service registry such as UDDI defines an information model that allows syntactic queries about the elements contained in this model. In contrast, Ontoolcole ontology formally defines a vocabulary that describes conceptual elements about CSCL tools and the relations between these elements with definite semantics. Thus, a direct comparison between UDDI and Ontoolcole is not possible since the information contained in a UDDI record is very different from an Ontoolcole service description, although [14] discussed that the UDDI information model was not appropriate for the search of CSCL services. Furthermore, semantic searches supported by Ontoolcole were positively considered by users during the experiment, allowing very expressive queries using powerful OWL DL constructs such as concept inheritance or transitive properties.

5 Conclusions and Future Work

This paper presented significant enhancements made in Ontoolcole, an ontology of collaborative tools intended for the semantic search of CSCL services by educators. The artifact model developed defines an extensible structured depiction of the artifact types that can be employed during the execution of CSCL activities. The formatting information of the artifacts is described using the well-known MIME types. In addition, a mechanism has been built for specifying the resources a tool can store. Finally, a coordination model has been incorporated for the description of complex workflows of tasks. With these new features, a wide range of CSCL tools can be described, enabling the semantic search of services using Ontoolcole's abstractions.

In addition, we carried out an experiment with real users to evaluate whether Ontoolcole can be employed by educators to search CSCL services. 15 participants expressed their information needs for tools in the context of their real practice without any restriction on the questions they posed. A mediator translated those questions into queries and submitted them to an Ontoolcole-powered retrieval system. Participants ranked with an overall 3.64 out of 5 the quality of the translations, indicating that Ontoolcole has the semantic richness to support these queries. Significantly, questions referred to CSCL settings were ranked a 17% better due to Ontoolcole's emphasis on collaboration features. Moreover, the tools retrieved were rated with 4.32 out of 5 for the CSCL case, evidencing that users found useful tools for their educational needs.

Future work includes the development of Ontoolsearch, a visual query formulation facility for Ontoolcole. Ontoolsearch aims to substitute the mediator in the precedent experiment, allowing educators to autonomously search CSCL services. Besides, we are considering users' suggestions to produce a new version of

Ontoolcole. Then, a new experiment will be performed in order to assess whether educators can accomplish a complete service search scenario.

Acknowledgements

This work has been funded by Kaleidoscope network of excellence FP6-2002-IST-507838, Spanish Ministry of Education and Science project TSI2005-08225-C07-04 and Autonomous Government of Castilla and León, Spain, project VA009A05.

References

1. Koschmann, T.: Paradigm shift and instructional technology. In Koschmann, T., ed.: *CSCL: Theory and Practice of an emerging paradigm*. First edn. Lawrence Erlbaum, Mahwah, NJ, USA (1996) 1–23
2. Papazoglou, M., Georgakopoulos, D.: Service-oriented computing. *Communications of the ACM* **46**(10) (2003) 25–28
3. Bote-Lorenzo, M.L., Vaquero-González, L., Vega-Gorgojo, G., Dimitriadis, Y.A., Asensio-Pérez, J.I., Gómez-Sánchez, E., Hernández-Leo, D.: A tailorable collaborative learning system that combines OGSA grid services and IMS-LD scripting. In: *Proceedings of the Tenth International Workshop on Groupware: Design, Implementation, and Use (CRIWG 2004)*. LNCS 3198, San Carlos, Costa Rica, Springer-Verlag (2004) 305–321
4. Bote-Lorenzo, M.L., Hernández-Leo, D., Dimitriadis, Y.A., Asensio-Pérez, J.I., Gómez-Sánchez, E., Vega-Gorgojo, G., Vaquero-González, L.: Towards reusability and tailorability in collaborative learning systems using IMS-LD and grid services. *Advanced Technology for Learning* **1**(3) (2004) 129–138
5. Morch, A.: Three levels of end-user tailoring: customization, integration and extension. In: *Proceedings of the Third Decennial Aarhus Conference*, Aarhus, Denmark (1995) 41–45
6. Guha, R., McCook, R., Miller, E.: Semantic search. In: *Proceedings of the Twelfth International World Wide Web Conference (WWW2003)*, Budapest, Hungary (2003)
7. Baeza-Yates, R., Ribeiro-Neto, B.: *Modern Information Retrieval*. First edn. Addison-Wesley, Harlow, UK (1999)
8. Chandrasekaran, B., Josephson, J., Benjamins, V.: What are ontologies, and why do we need them? *IEEE Intelligent Systems* **14**(1) (1999) 72–81
9. Klein, M., Bernstein, A.: Toward high-precision service retrieval. *IEEE Internet Computing* **8**(1) (2004) 30–36
10. Organization for the Advancement of Structured Information Standards (OASIS): *Introduction to UDDI: Important features and functional concepts* (2004) URL: <http://uddi.org/pubs/uddi-tech-wp.pdf>, last visited April 2006.
11. Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., Weerawarana, S.: Unraveling the web services web. *IEEE Internet Computing* **6**(2) (2002) 86–93
12. Paolucci, M., Sycara, K.: Autonomous semantic web services. *IEEE Internet Computing* **7**(5) (2003) 34–41
13. Martin, D., et al.: *OWL-S: Semantic markup for web services (version 1.1)*. Technical report, DARPA Agent Markup Language Program (2004) URL: <http://www.daml.org/services/owl-s/1.1/overview/>, last visited April 2006.

14. Vega-Gorgojo, G., Bote-Lorenzo, M.L., Gómez-Sánchez, E., Dimitriadis, Y.A., Asensio-Pérez, J.I.: A semantic approach to discovering learning services in grid-based collaborative systems. *Future Generation Computer Systems (FGCS)* **22**(6) (2006) 709–719
15. Ellis, C., Wainer, J.: A conceptual model of groupware. In: *Proceedings of the 1994 ACM conference on Computer supported cooperative work (CSCW 1994)*, Chapel Hill, NC, USA (1994) 79–88
16. Wroe, C., Stevens, R., Goble, C., Roberts, A., Greenwood, M.: A suite of DAML+OIL ontologies to describe bioinformatics web services and data. *The International Journal of Cooperative Information Systems* **12**(2) (2003) 597–624
17. Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D., Patel-Schneider, P., Stein, L.: *OWL web ontology language reference* (2004) URL: <http://www.w3.org/TR/owl-ref/>, last visited April 2006.
18. Racer Systems GmbH & Co.KG: *RacerPro website* (2006) URL: <http://www.racer-systems.com/products/racerpro/index.phtml>, last visited April 2006.
19. van der Veer, G., van Welie, M.: Task-based groupware design: Putting theory into practice. In Boyarski, D., Kellog, A.W., eds.: *Proceedings of the ACM SIGCHI Conference on Designing Interactive Systems: Processes, Practices, Methods and Techniques (DIS-00)*, Nueva York, AZ, USA, ACM Press (2000) 326–337
20. Internet Assigned Numbers Authority (IANA): *MIME media types* (2006) URL: <http://www.iana.org/assignments/media-types/>, last visited April 2006.
21. IEEE Learning Technology Standards Committee: *IEEE standard for learning object metadata. Technical Report 1484.12.1-2002*, Computer Society/Learning Technology Standards Committee (2002)
22. DCMI Usage Board: *DCMI metadata terms* (2005) URL: <http://dublincore.org/documents/dcmi-terms/>, last visited April 2006.
23. Rick, J., Guzdial, M., Carroll, K., Hollaway-Attaway, L., Walker, B.: Collaborative learning at low cost: CoWeb use in english composition. In Stahl, G., ed.: *Proceedings of the Computer Supported Collaborative Learning Conference 2002 (CSCL 2002)*, Boulder, CO, USA, Lawrence Erlbaum Associates (2002)
24. Knublauch, H., Fergerson, R., Noy, N., Musen, M.: The Protégé OWL plugin: An open development environment for semantic web applications. In: *Proceedings of the Third International Semantic Web Conference (ISWC 2004)*. LNCS 3298, Hiroshima, Japan (2004) 229–243
25. Organization for the Advancement of Structured Information Standards (OASIS): *Web services for remote portlets specification* (2003) <http://www.oasis-open.org/committees/download.php/3343/oasis-200304-wsrp-specification-1.0.pdf>, last visited June 2006.
26. Catarci, T., Dongilli, P., Mascio, T.D., Franconi, E., Santucci, G., Tessaris, S.: An ontology based visual tool for query formulation support. In: *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI 2004)*, Valencia, Spain, IOS Press (2004)
27. Grüninger, M., Fox, M.: Methodology for the design and evaluation of ontologies. In Skuce, D., ed.: *Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing (IJCAI-95)*, Montreal, Canada (1995)
28. Lord, P., Alper, P., Wroe, C., Goble, C.: Feta: A light-weight architecture for user oriented semantic service discovery. In: *Proceedings of the Second European Semantic Web Conference (ESWC 2005)*. LNCS 3532, Heraklion, Greece, Springer-Verlag (2005) 17–31

CSCL, Anywhere and Anytime

Stephan Lukosch, Matthias Hellweg, and Martin Rasel

FernUniversität in Hagen

Department for Mathematics and Computer Science

58084 Hagen, Germany

{stephan.lukosch, matthias.hellweg, martin.rasel}@fernuni-hagen.de

Abstract. The FernUniversität in Hagen uses the web-based collaborative learning platform CURE to implement different collaborative learning scenarios such as seminars or lab courses. In these scenarios, students form groups and collaboratively solve tasks given by the teachers. Interviews with students that have used CURE showed major interest in using CURE nomadically without the need for permanent internet access. Nomadic use would allow students to work with CURE content at any time and place while maintaining the advantages of a shared, synchronized CSCL environment once they are online again. In this article, we describe which requirements we have identified for a nomadic use and how we extended CURE to fulfill these requirements.

1 Introduction

In recent years, the application of computer-based learning methods to classical lectures, seminars, or lab courses has been gaining in popularity. Though these methods cannot fully replace face-to-face learning, they represent a valuable enhancement, especially in view of continuing education or cooperative learning. In distance teaching, computer-based learning allows students all over the world to collaboratively acquire and apply new knowledge. Knowing that, the FernUniversität in Hagen developed a web-based collaborative learning platform named *CURE* [1] which has successfully been in use for over two years now.

The FernUniversität in Hagen is the only distance teaching university in Germany. Most of its students study at home, often as a part time student while being in a full-time employment. Thus, the independence in time and place is one of the major reasons for enrolling in a distance teaching university. Interviews with students that have used CURE during lectures, seminars, or lab courses revealed major interest in using CURE nomadically. Since CURE is web-based, spare time during train journeys or flights where no internet access is available cannot be used for reading or editing CURE content. Furthermore, the need for a permanent internet connection while working in the CURE environment, can lead to significant expenses for students studying at home using dial-up connections. Supporting a nomadic use of CURE would thereby significantly enhance the available learning possibilities.

In this article, we describe the requirements we have identified for a nomadic use and how we extended CURE to fulfill these requirements. The article is structured as follows: The next section introduces the collaborative learning platform

CURE which forms the basis for the work presented in this paper. Then, we discuss some scenarios that describe common use cases of CURE and demonstrate potential advantages of using CURE nomadically. Based on these scenarios, we determine the requirements for a nomadic use of CURE and then describe how we fulfill them. Finally, we compare our approach with the related work before concluding with a summary and an outlook on future work directions.

2 CURE in a Nutshell

CURE [1] is based on the metaphor of virtual rooms, that uses the room as a representation of a virtual place for collaboration. Room metaphors [2] [3] have been widely used to structure collaboration.

Figure 1 shows the abstractions that are offered by CURE. Users enter the cooperative learning environment via a virtual entry room named 'Hall'. Rooms can contain further subrooms, content in the form of so called pages, communication channels (e.g. chat, threaded mail) and users. The concept of virtual keys [4] is used to express the access rights a user holds for a certain room and the content of this room. Keys, for example, assign the right to enter a room, create sub rooms, edit pages, or to communicate within the room. Rooms with public keys are accessible to all registered users of the system.

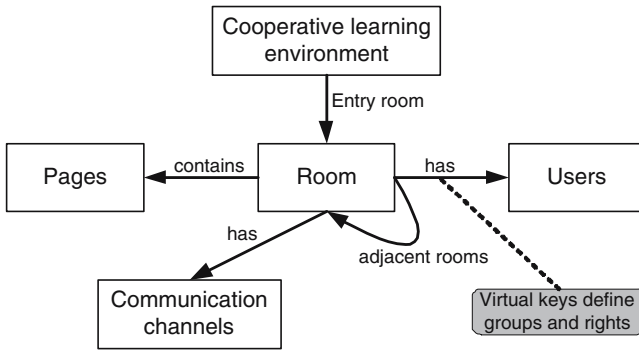


Fig. 1. CURE abstractions

When users enter a room, they can participate in collaborative activities and access the room's communication channels, i.e. by using the chat that is automatically established between all users in a room. They can also view the pages that are contained in the room. Users possessing suitable keys can freely edit the content of pages, with the changes being visible to all members in the room after uploading. Earlier versions of a page remain accessible to allow tracing of recent changes and support conflict resolution in case multiple users edit a page concurrently. Pages may either be directly edited using a simple Wiki-like syntax [5], or they may contain binary documents, e.g. JPEG images, Microsoft

Word documents etc. In particular, the syntax supports links to other pages, other rooms, external URLs or mail addresses. The server stores all artifacts to support collaborative access. Thus, when users leave the room, the content stays available, allowing them to come back later and continue their work on the room's pages. However, this implies that the CURE platform as such requires the user to be connected to the Internet to access content stored on the CURE server.

Seminar 1915 - Anwenderorientierte Groupware: Themenübersicht

Zu den Themen geben wir in diesem Semester nur grobe Schlagworte vor. Zusätzlich finden Sie zu jedem Thema eine URL von Google Scholar, einer Literaturodatenbank im WWW. Nutzen Sie diese URL, um sich einen ersten Überblick über den Stand der Forschung in Bezug zu den vorgegebenen Schlagworten zu verschaffen.

Die Bearbeitung der einzelnen Seminarthemen wird Gruppen zu 2 Personen erfolgen. Die Arbeit an Ausarbeitung und Vortrag sollte etwa gleichgewichtig auf die Mitglieder der Kleingruppe verteilt werden, der Vortrag soll gemeinsam präsentiert werden. Sie sollten vor der endgültigen Themenzuordnung den [Diskussionsraum](#) nutzen, um über Ihr Wunschthema zu diskutieren und einen Partner für die Bearbeitung dieses Themas zu finden. Tragen Sie sich nach erfolgreicher Suche in die unten stehende Tabelle ein.

Alle Themenwünsche wurden inzwischen bestätigt. Sie finden Ihren Arbeitsraum in folgender Tabelle.

| Bearbeiter 1 | Bearbeiter 2 | Titel | Arbeitsraum | Betreuer |
|-----------------------|------------------|--|---|-----------------|
| Ueli Gisiger | frei | "programming by example" groupware (Scholar) | Programming by Example | Till Schümmer |
| Peter Kilian | Marc Winkens | "participatory design" "design patterns" groupware (Scholar) | Participatory Design und Patterns | Till Schümmer |
| Christoph Hannebauer | Tonia Hannebauer | collaboration script groupware process analysis (Scholar) | Collaboration Scripts | Till Schümmer |
| Sarabdeep Singh-Pannu | frei | csw tailoring context (Scholar) | Tailoring | Stephan Lukosch |
| Gregor Sauerzapf | Tilo Behrmann | groupware "end user development" (Scholar) | End-user Development | Stephan Lukosch |

Zuletzt bearbeitet von Till Schümmer (08.02.06, 21:33)
additional contributors

CHAT-LOG

Martin Rasel (16:35:43) Hallo Stephan

Stephan Lukosch (16:35:56) Hallo Martin

Stephan Lukosch (16:36:16) Hast du dich schon für ein Thema entschieden?

Martin Rasel (16:36:36) Nein, ich überlege noch.

Fig. 2. A room in CURE

Figure 2 shows a typical room in CURE as it appears in the users web browser. The numbers in the figure refer to details explained in the following paragraphs. A room contains pages ① that can be edited by users having sufficient edit rights ②. CURE stores all versions of a page ③. It provides two room-based communication channels, i.e. a mail box ④ and a chat ⑤. Users can use the

room-based e-mail to send mails to the room. Users of the room that have sufficient communication rights will receive these messages.

By providing a plenary room, sharing and communication, a whole class or organization can be supported. By creating new rooms for sub-groups and connecting those to the classes' or organization's room, work and collaboration can be flexibly structured. Starting from the plenary room users can navigate to the connected sub-rooms ⑥.

For user coordination, CURE supports various types of awareness information:

1. A list of users with access to the room is available in the room's properties ⑦.
2. Small images represent those users currently present in the room ⑧.
3. If the chat is enabled in the room, users can directly start chatting to each other ⑤.
4. Users can trace who has previously edited the current page ⑨.
5. Daily reports automatically posted to all users of a room include all changes made since the last report was sent.

3 Requirements Analysis

In this section, we will determine the requirements for a nomadic use of CURE by describing two typical use cases and pointing out how nomadic use would improve the working process. The first use case describes a seminar conducted in CURE (see section 3.1) while the second describes the self-organization of an independent working team of students (section 3.2).

3.1 Seminar in CURE

Michael has enrolled in a virtual seminar and he just received the key to a room in CURE (like in figure 2) that the supervisors have created as a central meeting point for the participants. The room already contains a number of CURE pages that give a basic overview of the seminars subject and define the planned workflow and timetable. The general subject is split up into a number of sub-topics, each of which should be worked on by two students in collaboration. To reflect this, the main room contains a page with links to a number of sub-rooms, i.e. one sub-room for each sub-topic, and the students have been assigned private keys for their respective room. The supervisors have uploaded some PDF-documents to each sub-room that serve as a starting point for further investigations. In addition, a CURE page containing a template for the seminar thesis to be prepared by each group is provided. The students are supposed to use the sub-rooms for coordination and communication within their group. They are also encouraged to use the room for uploading drafts, intermediary results and the final seminar thesis. The main room should be used for communication between all participants of the seminar and as a central repository for CURE pages with information and links of general interest.

Michael would like to start by reading the provided material and then insert some first thoughts right into the thesis draft. Unfortunately, he soon has to leave for a job-related journey. Michael knows that he will have some hours of spare time on that journey, but he will not be able to access the CURE server during that time. Therefore, he settles for downloading the PDF documents to his notebook to have some reading material. During the journey, Michael has more time than expected, so he starts to work on the thesis by making notes in a simple text editor on this computer. However, since he does not have the provided draft with him, his work remains a coarse collection of thoughts. As Michael returns home, he connects to the Internet, enters his groups sub-room and notices that his colleague has already inserted some notes into the designated CURE page. Some of these overlap with Michael's work and he has a hard time copying and pasting his sketches to the common page without losing the existing text.

An application that allows a nomadic use of CURE could have supported Michael by enabling him to

- conveniently download some of the accessible CURE rooms together with the contained documents and wiki pages,
- view the downloaded pages offline just like they are presented within CURE,
- create and edit CURE pages, preferably with support for applying Wiki tags to structure and format the text, and
- synchronize local changes with the CURE server, preferably with support for convenient resolution of version conflicts.

3.2 Student Working Team in CURE

Christine wants to use CURE to collaboratively discuss the curriculum of the Bachelor of Computer Science with other students. She plans to create a room that contains sub-rooms for each class she has attended during her studies. These sub-rooms should be used to exchange and discuss lecture notes or other material provided by the students and collect documents for the preparation of written and oral exams. Therefore, she only wants to share the room with other students without giving access to the teachers. To allow browsing the collected material, the main room shall contain a page with links to the individual sub-rooms, which in turn shall each contain a table of contents linking to the rooms respective pages. She hopes that other students will use these rooms to retrieve documents, add material themselves, and discuss the contents via mail or chat.

Of course, this rather complex structure can be constructed using CURE. However, for each document or sub-room Christine wants to create, she has to interact with the server and wait for its response, which can become rather time consuming considering the scope of the project. Additionally, Christine will have to switch back and forth between rooms to update the table of contents or to create new sub-rooms, which can become confusing, in spite of the navigation tools CURE provides.

Again, an application that allows a nomadic use of CURE would have supported Christine, by, in addition to the previous scenario.

- offering a more intuitive and responsive interface than the web interface of CURE, e.g. by showing an overview of the projects structure at all times,
- supporting the convenient creation of rooms and pages as well as interconnecting links,
- allowing to upload selected rooms together with contained sub-rooms and documents, and
- respecting the access rights defined by virtual keys,

3.3 Summary

CURE as a web-based collaborative learning platform focuses on providing access using any web browser, with the only requirement being an existing Internet connection. While this ensures a maximum in flexibility, the user interface is limited due to the possibilities of the web browser. User input has to be sent to the server for processing before the result can be shown within the user interface. An independent local application on the users computer can improve the usability by reducing the response time. Furthermore, a local application can provide a more intuitive user interface compared to the possibilities available in web-based applications. However, the most important advantage of a local application that can synchronize user data with the central CURE-Server, and let the user view and edit CURE content locally, would be the independence from permanent Internet access. This would allow users to effectively use times of mobility and reduce online costs. Working and learning would become possible at any time and place while keeping the advantages of a shared CSCL environment.

Summarizing, the following requirements for a nomadic use of CURE must be met by a local application:

- R1:** A communication interface between the CURE server and a local client has to provide read and write access to the contents of CURE.
- R2:** CURE controls the access rights using virtual keys. These access rights must be respected by a local client. This includes authentication of the users.
- R3:** Users have to be able to individually select the content they want to access locally. The data has to be stored persistently on the users computer while maintaining the structure existing on the server (i.e. rooms, sub-rooms and contained pages).
- R4:** Editing of server based content should not be locked, to retain the collaborative nature of CURE pages. Therefore the local application has to provide synchronization of local data with the content stored on the CURE server. Support for the resolution of conflicting changes performed by multiple users on the same page has to be given.
- R5:** Compared to the web interface, an intuitive user interface has to simplify the creation and editing of content.

4 Approach

To address the identified requirements (R1-R5), we have integrated a communication interface, called *offlineCURE*-server, into the CURE-server. The

offlineCURE-server offers access to CURE data in parallel to the web interface. The local application, named *offlineCURE*-client, was implemented as a Java application to be independent from the operating system used by the users. The following sections will give a detailed description of how we addressed the above requirements (R1-R5).

4.1 Communication Interface (R1)

Figure 3 shows the overall architecture of CURE after being extended with *offlineCURE*. Without *offlineCURE* users can access CURE content using standard web browsers. The communication between a client and the server is established via HTTP and servlets running on the server respond to the clients requests. CURE content is stored in a database that is accessed by the servlets via the CURE kernel.

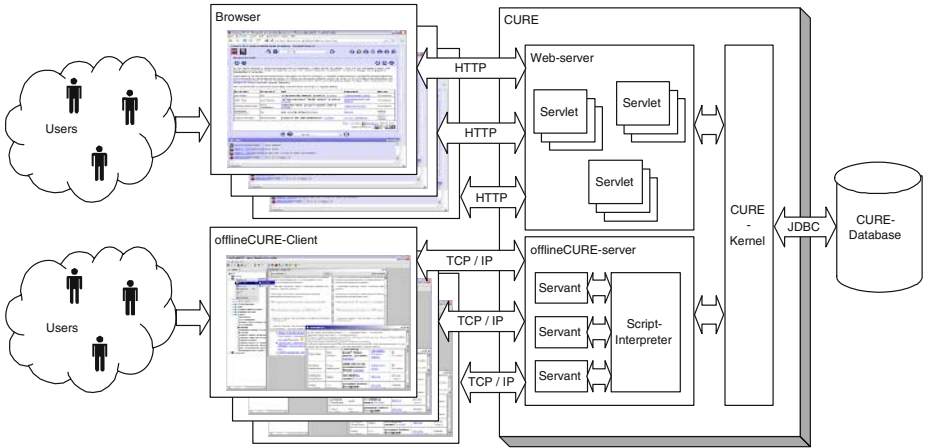


Fig. 3. *offlineCURE* architecture

The *offlineCURE*-client and the *offlineCURE*-server communicate via a TCP/IP connection. To ensure secure communication, it is possible to set up an SSL connection [6]. After the client established a connection to the *offlineCURE*-server, users have to authenticate themselves, before gaining access to the content. Alternatively, users have the chance to authenticate the server with a server-certificate, signed by a certification authority or a trust center.

Communication between the *offlineCURE*-server and the *offlineCURE*-client is handled using an intuitive and flexible script language. This language allows to access all necessary methods of the CURE kernel. Examples of basic commands are the download of rooms or pages, the upload of edited pages or rooms, the creation of new pages, rooms or keys, the synchronization of local content with server data, etc. Example script commands are:

- `get page * in room name "Hall";`
returns all pages in the room *Hall*.
- `create room name "Seminar" in room name "Hall";`
creates a new sub-room *Seminar* in the room *Hall*.
- `update page name "Welcome" in room name "Hall" -content="...";`
updates the page with the name *Welcome* in the room *Hall* with the specified content.
- `create key for room name "My Room" -rights=124 -freekey;`
creates a free key for the room *My Room* with the rights 124.

The servers response can have different output formats, with the standard format being XML. For higher security, the *offlineCURE*-server filters sensible data via a read-filter before sending responses to the client. When performing write-actions, like updating a page or a room, the content is filtered via a write-filter.

To allow a maximum of concurrency, the server uses a dedicated process for each client that receives and interprets script commands and generates the respective response. Database conflicts are avoided by making use of the object manager in the CURE kernel which is based on transactions. To improve concurrency, script commands are handed as atoms, i.e. if multiple commands are received from the client, each command is handled by one transaction. Furthermore, the *offlineCURE*-server distinguishes between read- or write-commands for the above-mentioned object manager of the cure kernel.

In addition to read and write access to CURE content, the *offlineCURE*-server supports user awareness by being able to send notifications, e.g. actions, messages and news, to each client via a separate communication-channel. This way, processes on the server-side or other applications can reach the users.

4.2 Respecting User Rights (R2)

In CURE, users can have different user rights. For each room users can access, they have virtual keys defining the rights within the respective room [4]. *offlineCURE* ensures these user rights by requiring user authentication and by checking the user rights before executing a script command .

Figure 4 shows a UML activity diagram describing the execution of a script command by the *offlineCURE*-server. After receiving one or more commands, the *offlineCURE*-server checks the syntax of the command to be executed. Since each script command is unambiguously related to a target room, the semantic part of the script interpreter can then check whether the user has sufficient rights to execute the command in the specified room. This guarantees that users can only execute commands via the communication interface that result in actions the user could also perform via the web interface. If a user does not have sufficient rights, the *offlineCURE*-server responds by sending an error code to the *offlineCURE*-client and stops the transaction. Otherwise, the *offlineCURE*-server consecutively executes the received commands via the CURE kernel and delivers its results.

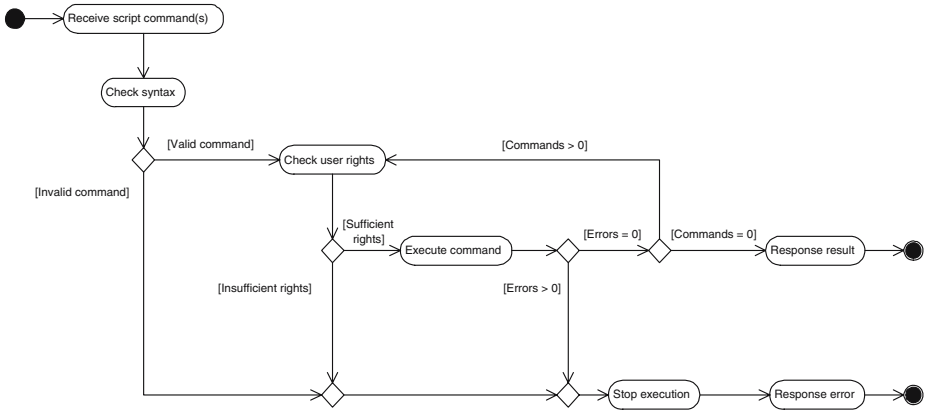


Fig. 4. *offlineCURE*-command execution

4.3 Selection of Local Data (R3)

Users enter CURE via a main room from where different sub-rooms can be accessed. Sub-rooms can again contain further sub-rooms and each room has its own content, i.e. wiki pages or binary pages. Thus, the content of CURE is hierarchically organized in a tree structure.

For the selection of data, the *offlineCURE*-client visualizes this tree and lets users mark the content they want to access nomadically, e.g. a room including all or part of the rooms content, one or multiple pages, or a complete branch of the tree. To reduce network communication, the *offlineCURE*-client only requests the information needed to build the currently shown depth of the tree. If a user selects deeper layers, the *offlineCURE*-clients recursively requests the corresponding information.

When the user has selected the desired content for nomadic use, the *offlineCURE*-client downloads the respective data and stores it locally. Relations between the replicated content objects is maintained, thus providing the same structure of rooms, sub-rooms and pages as on the server. Subsequently, the user can disconnect from the network and use the replicated data nomadically.

4.4 Synchronization of Local Data (R4)

Since *offlineCURE* is designed to enable the use of CURE content while being completely independent of network connections, the *offlineCURE*-client cannot directly propagate local modifications to the CURE server and other users. This may result in conflicts between the local version of a document and its version on the CURE-server. A way to avoid this, would be locking a document on the server whenever a user transfers it to the *offlineCURE*-client. However, this would prevent simultaneous, independent editing and thus contradict the collaborative nature of CURE. The possibility to edit a document would depend, e.g., on the author who has locked it [7].

Instead of using such a pessimistic approach, *offlineCURE* re-uses the optimistic approach of CURE and lets users work on separate logical versions of a document, and resolve conflicts when the content is uploaded to (synchronized with) the CURE-server. For this purpose, *offlineCURE* uses an optimistic replication strategy [8], i.e. the original content on the server is not locked and users still have the possibility to modify it using the web interface or edit local copies on their *offlineCURE*clients.

In CURE, version management uses timestamps that are assigned to CURE content when it is created. To detect conflicts between client and server data, the *offlineCURE*-client uses a combination of these server-based timestamps and client-based *modified bits* [8]. While the timestamps included in the local copy of content objects are never changed by the client, modified bits are set whenever a user changes a local document. Table 1 shows possible combinations of the server-based timestamp and the client-based modified bit. When users start a synchronization, the *offlineCURE*-client analyzes the local content to determine the necessary synchronization actions. The result of this analysis is visualized in a tree to enable the user to select which synchronization actions he wants to perform.

Table 1. Possible combinations of timestamp and modified bit and the corresponding synchronization action

| Server | Client | Synchronization action |
|--------|---------|--|
| t_1 | n/a | Content only available on the server; download to client |
| n/a | $t_1/+$ | Content only available on the client; upload to server |
| t_1 | $t_1/-$ | Content unmodified; no synchronization necessary |
| t_1 | $t_1/+$ | Content modified on the client; upload to server |
| t_2 | $t_1/-$ | Content modified on the server; download to client |
| t_2 | $t_1/+$ | Content modified on server and client; conflict resolution |

t_1, t_2 : Consecutive timestamps on the CURE-server

n/a: Document is not available at this site

- / +: Document was not changed / changed by an *offlineCURE*-client

In case of a conflict, the *offlineCURE*-client makes use of the CURE versioning approach. Figure 5 shows how CURE manages different versions of a document. A newly created document has the version $V1$. When a document is changed for the first time, the version number is changed to $V2$ (see figure 5 a). When this version is changed, it is assigned $V3$ (see figure 5 b). Figure 5 c) shows the result of another user editing $V2$ in parallel. In this case, CURE creates a parallel version $V4$ and immediately adds version $V5$ (see figure 5 d). The automatically created version $V5$ is called *merge-page* and contains the versions $V3$ as well as $V4$. In version $V5$, $V3$ and $V4$ are separated by text that points out the conflicts and requests the user to solve them.

CURE uses this versioning approach to manage conflicts that might occur when users modify the same version of a document via the web interface. This way, CURE prevents users from overwriting the changes other users have

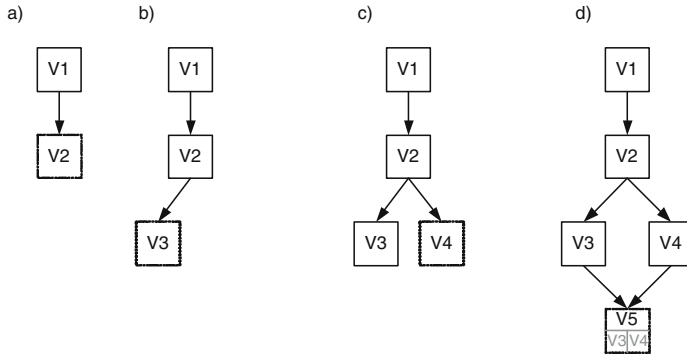


Fig. 5. Versioning in CURE

performed. Solving these conflicts via the web interface can be quite difficult. Therefore, the *offlineCURE*-client employs a different visualization for merging pages. If the client detects a version conflict during synchronization with the server, it allows the user to select the versions he wants to merge. The text of the selected versions is then compared blockwise and both versions are aligned in an editable form. Differing blocks are highlighted in the user interface and users can select which block they want to use in the merged version, or edit one version manually. When a user has solved a conflict, the merged document is uploaded to the server resulting in a new version.

4.5 User Interface (R5)

Figure 6 shows the user interface of *offlineCURE*. The left area of the user interface shows a tree-based overview of the content that is locally available. Compared to the web interface, this simplifies the navigation, since users always have an overview of the local content available. The level of detail can be tailored by expanding or collapsing individual tree branches. Context-sensitive menus allow users to perform actions in relation to the selected object, e.g. create a new sub-room in a room or delete a page. Additionally, the *offlineCURE* user interface offers static menus, keyboard shortcuts, and a toolbar with frequently used actions.

Figure 6 also shows two windows in which the user currently edits CURE pages. The upper left window contains the representation of a merge-page with two conflicting text versions. The lower right window shows a page containing a complex table. Both windows are split horizontally. The upper part is used for editing Wiki pages while the lower shows a real-time preview of the rendered page. The size of the edit and preview areas is variable, i.e. users can decide to display only the edit area while working on a long document, or use the preview area on its own for browsing CURE content offline. The real-time preview shows the page content like it would be rendered in the users browser when working with the CURE server. This also includes usable links to other

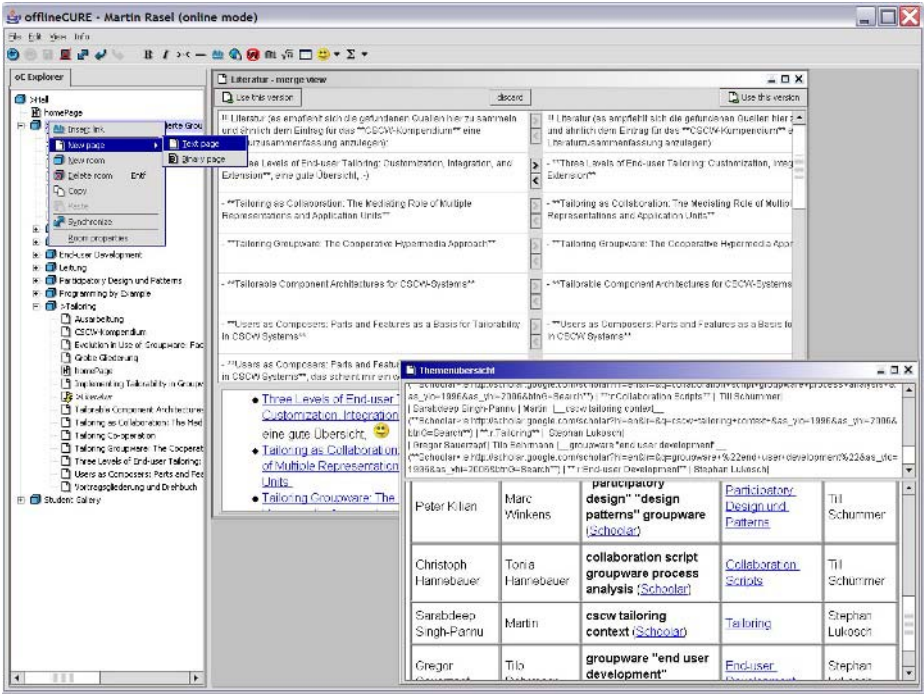


Fig. 6. offlineCURE user interface

documents, external URLs etc. Support for creating complex structures is given by indicating whether internal links on a page are valid, i.e. whether the linked content is available locally, and allowing users to conveniently create content via context sensitive menus attached to links. The toolbar provides support for incorporating Wiki-tags into the page text. Compared to the web interface, the *offlineCURE* user interface significantly improves the usability and workflow when editing CURE content.

5 Related Work

In this section, we compare *offlineCURE* with learning platforms or Wiki engines that offer a nomadic use. This excludes learning platforms like, e.g. *Moodle* [9] or *WebCT* [10], as these do not support nomadic use.

BSCW [11] and *BSCL* [12] are web-based collaboration platforms. *BSCW* allows to download documents, edit them locally, and then upload the modified document. *BSCWeasel* [13] allows to access a *BSCW*-server without using the web interface, but still requires a permanent network connection. *BSCW*, *BSCL*, and *BSCWeasel* do not support the synchronization of documents, when documents were modified by another user.

The learning platform *KOLUMBUS* [14] allows users to import and export content, but like BSCW, BSCL, and BSCWeasel it does not support the users when resolving conflicts that are due to parallel changes.

Blackboard Backpack [15] is a client-side software application that enables users to download content, e.g. course documents, announcements, calendar items, or tasks, from the *Blackboard* learning platform [16]. However, the content can only be read and annotated locally, i.e. local changes cannot be synchronized.

FirstClass [17] is a commercial groupware similar to CURE. But compared to CURE, FirstClass does not offer a versioning support. FirstClass allows users to lock documents when they want to modify it. While a document is locked, other users can only read it. Due to this, FirstClass does not fulfill our requirements concerning conflict resolution and synchronization.

eBag [18] offers a *digital schoolbag* which can contain images, videos, music, or text documents for learning at school. eBag allows pupils to move nomadically between stationary terminals, called *digital oases*, in classrooms, labs, libraries, etc. In these digital oases, pupils can access their documents. Out of these digital oases, pupils can access their digital schoolbag via a web browser. Though the developers call this offline use, the latter obviously requires a network connection. Therefore, eBag does not fulfill our requirements concerning nomadic use.

The collaborative learning platform *sTeam* [19] allows students to learn collaboratively without being connected to the learning platform. For that purpose, the students have to establish an ad-hoc network among themselves. Thus, sTeam allows to disconnect from the learning platform, but students have to meet at the same place and the same time for collaboration. Individual times of mobility cannot be used for learning.

Personal Wikis, e.g. *WikiWriter* [20], *EclipseWiki* [21] or *WikidPad* [22], allow to setup a local wiki engine and locally manage, create, and modify the content of this wiki. The content of these personal wikis is stored in a local database. Thus, nomadic working and learning is possible, but as the content cannot be shared, users cannot collaborate.

SimpleWikiEditMode [23] differs from the above personal wikis. It uses the text editor *Emacs* to access and modify the content of a remote wiki. However, SimpleWikiEditMode does not support the selection of local content. Additionally, it does not provide functions for identifying locally changed content or conflict resolution.

The *Wikipedia Editor* [24] is a plugin for the IDE *Eclipse*. With the help of this plugin, users can locally store and modify Wikipedia articles. Compared to our requirements, the plugin does not support conflict detection and resolution. During synchronization, articles that were changed locally are simply replaced with the current version in the Web.

To summarize, none of the above systems fulfills our requirements for nomadic collaborative learning. *offlineCURE* represents a significant step forward, as it allows students to learn while they are on the move. Students can synchronize their *offline* results with the learning platform and thereby integrate these in the collaborative learning process.

6 Conclusions

Interviews with students that have used CURE showed a major interest of using CURE nomadically. A nomadic use of CURE would allow students to use times of mobility for learning.

In this article, we have presented how we extended the web-based learning platform CURE with *offlineCURE* to allow nomadic learning. *offlineCURE* uses a communication interface in the CURE server to selectively download content that can be read and modified while disconnected from the network. *offlineCURE* respects the user rights that are represented as virtual keys in CURE. Users can download all content that are accessible for them. They can view and modify the local available content. Additionally, *offlineCURE* supports users to solve Conflicts that may result from concurrent modifications while being disconnected.

First experiences, while using *offlineCURE* have shown that all our requirements are satisfied. Users have reported that *offlineCURE* simplifies the organization of collaborative learning as the necessary content is available at any time and any place. Additionally, users have mentioned that the user interface of *offlineCURE* simplifies the creation and modification of content.

In future, we want to offer *offlineCURE* for the use on mobile devices such as PDAs. We also plan to broaden the use of *offlineCURE* and evaluate how the nomadic use of CURE effects the learning results of our students.

References

1. Haake, J.M., Schümmer, T., Haake, A., Bourimi, M., Landgraf, B.: Two-level tailoring support for cscl. In: Groupware: Design, Implementation, and Use. Proceedings of the 9th International Workshop (CRIWG 2003). LNCS 2806, Heidelberg, Springer (2003) 74–82
2. Greenberg, S., Roseman, M.: Using a room metaphor to ease transitions in groupware. In Ackermann, M., Pipek, V., Wulf, V., eds.: Sharing Expertise: Beyond Knowledge Management. MIT Press, Cambridge, MA, USA (2003) 203–256
3. Pfister, H.R., Schuckmann, C., Beck-Wilson, J., Wessner, M.: The metaphor of virtual rooms in the cooperative learning environment clear. In Streitz, N., Konomi, S., Burkhardt, H., eds.: Cooperative Buildings - Integrating Information, Organization and Architecture. Proceedings of CoBuild'98. Volume 1370 of LNCS., Heidelberg, Springer (1998) 107–113
4. Haake, J.M., Haake, A., Schümmer, T., Bourimi, M., Landgraf, B.: End-user controlled group formation and access rights management in a shared workspace system. In: CSCW '04: Proceedings of the 2004 ACM conference on Computer supported cooperative work, Chicago, Illinois, USA, ACM Press (2004) 554–563
5. Leuf, B., Cunningham, W.: The WIKI way. Addison-Wesley, Boston, MA, USA (2001)
6. Dierks, T., Allen, C.: The TLS protocol. Request for Comments 2246, IETF (1999)
7. Vitali, F., Durand, D.G.: Using versioning to support collaboration on the WWW. World Wide Web Journal 1(1) (1994) 37–50
8. Saito, Y., Shapiro, M.: Optimistic replication. ACM Computing Surveys 37(1) (2005) 42–81

9. Moodle: A free, open source course management system for online learning. <http://moodle.org/> (2006)
10. WebCT Inc.: Product homepage. <http://www.webct.com> (2006)
11. Appelt, W., Mambrey, P.: Experiences with the BSCW shared workspace system as the backbone of a virtual learning environment for students. In: Proceedings of ED-MEDIA99. (1999)
12. Stahl, G.: Groupware goes to school. In Haake, J.M., Pino, J.A., eds.: Groupware: Design, Implementation, and Use, 8th International Workshop, CRIWG 2002. LNCS 2440, La Serena, Chile, Springer-Verlag Berlin Heidelberg (2002) 7–24
13. BSCWeasel: Erweiterbarer Rich Client für das BSCW Groupware System. <http://www.bscweasel.de/> (2006)
14. Herrmann, T., Hoffmann, M., Jahnke, I., Kienle, A., Kunau, G., Loser, K.U., Menold, N.: Concepts for usable patterns of groupware applications, ACM Press (2003) 349–358
15. Blackboard Inc.: Blackboard backpack. <http://backpack.blackboard.com/> (2006)
16. Blackboard Inc.: Product homepage. <http://www.blackboard.com/> (2006)
17. FirstClass: Product homepage. <http://www.firstclass.com> (2006)
18. Brodersen, C., Christensen, B.G., Grønabæk, K., Dindler, C., Sundararajah, B.: eBag: a ubiquitous web infrastructure for nomadic learning. In: WWW '05: Proceedings of the 14th international conference on World Wide Web, Chiba, Japan, ACM Press (2005) 298–306
19. Eßmann, B., Hampel, T., Bopp, T.: A network component architecture for collaboration in mobile settings. In: ICEIS 2004, Sixth International Conference on Enterprise Information Systems, Porto, Portugal (2004) 337–343
20. WikiWriter: A stand-alone, shareware personalwiki for windows. <http://hytext.com/index.htm> (2006)
21. EclipseWiki: Eclipse Wiki Editor Plugin. <http://eclipsewiki.sourceforge.net/> (2006)
22. WikidPad: Product homepage. <http://www.jhorman.org/wikidPad/> (2006)
23. SimpleWikiEditMode: Product Homepage. <http://www.emacswiki.org/cgi-bin/wiki/SimpleWikiEditMode> (2006)
24. PLog4U: Eclipse Wikipedia Plugin. http://www.plog4u.de/index.php/Eclipse_Wikipedia_Plugin (2006)

Web Management of Citizens' Complaints and Suggestions*

V.M.R. Penichet, J.A. Gallud, M. Lozano, and M. Tobarra

LoUISE Research Group – Computer Science Research Institute (I3A), Castilla-La Mancha University, 02071 Albacete, Spain
jose.gallud@uclm.es

Abstract. People in modern cities like to participate and collaborate in local governments by means of suggestions and complaints submission. However, citizens are not used to work with administrative procedures; moreover many of them do not like to spend time in administrative queues. In this paper we show some empirical results obtained by the Complaints and Suggestions Web-Based Collaborative Procedure (CS-WCP) in its first year of service in the Albacete Town-Council (Spain). Before showing those results, the CS-WCP is described. Administrative procedures in town councils, intelligent agents, workflow processes and Web-based computing are mixed in the CS-WCP system. Notifications by means of e-mails and messages facilitate user-to-civil servant and system-to-user communication and collaboration.

1 Introduction

Quality of service inside public administration is undoubtedly an important issue to take into account. This is especially true in local administration. Local governments in advanced democracies are interested in the citizens' opinion about their management.

Politicians want people to participate in the government and researchers are looking for new ways to increase citizen participation [1]. The complaints and suggestion administrative procedure is one of the most common systems that allows people to be heard by their local government [8, 9].

A complaints and suggestion procedure affects different people and can be considered as a kind of computer supported collaborative work system (CSCW). Town councils are scenarios where CSCW systems could help to coordinate civil servants work, because the communication between each other is allowed. This is not only good for an internal use, but also for coordinating, communicating, and collaborating with citizens [2, 3, 6].

Citizens need to express what they think, and town councils need to know what their citizens think in order to improve. Lots of administrative procedures are processed every day in administrative units of the town councils, most of them initiated by citizens [4, 5 and 7].

* This work was partially supported by the Spanish CICYT project TIN2004-08000-C03-01 and the grant PCC05-005-1 from Junta de Comunidades de Castilla-La Mancha.

In this paper we show some empirical results of the Complaints and Suggestions Web-Based Collaborative Procedure (CS-WCP), an electronic administrative procedure which takes into account collaboration, communication that was presented in a previous work [12]. In this paper we describe some improvements of the application and report on its evaluation. We have carried out an evaluation analysis meant to verify the effectiveness of the existing application and the real users' satisfaction. To do that, we have collected real information during the last year. The conducted evaluation reports that the system improves the communication between citizens and the town council.

CS-WCP includes three intelligent agents supporting tasks that are processed in a semi-automatic manner. We say semi-automatic because these agents suggest what to do, and they could do it by themselves, but the last decision depends on the final responsible of the system.

The paper is organized in the following sections: section 2 describes the system and shows the workflow model. Section 3 shows how an intelligent agent can help in the reception of comments step. Section 4 describes the empirical results and finally the conclusions and future work are presented in section 5.

2 Description of the CS-WCP System

The objective of our CS-WCP system is to allow citizens to post their complaints or suggestions in an easy way, via web, and, at the same time, the system enforces public administration to solve the comments and to answer citizens in a finite time.

The complaints and suggestions procedure contributes to increase the participation of citizens in the city management and allows politicians to get a real feedback from people.

The system takes a comment (it may be a complaint or a suggestion) which is sent to the Town Council through the Web. It is processed, sometimes automatically, sometimes manually, and then, the comment is processed.

By means of workflow modeling the system is described in a comprehensible way to all the people involved in the development of the final system: civil servant, analysts and developers.

In particular, these blocks are: (a) *complaint or suggestion arrival*, (b) *validation of comments*, (c) *invalid comment workflow*, (d) *valid comment workflow*, and (e) *complaints control time*.

Any CSCW system has to define groups and roles played by people involved in the complaints and suggestions procedure. In CS-WCP we have four roles: *Citizen*, *Reception Responsible*, *Unit Responsible* and *General Administrator*.

A user in the system accessing to the Web without authentication, that is to say, with the default user, is considered to be a *Citizen*. Neither a user login nor a password is required to access the system as a *Citizen*. Complaints and suggestions could be sent through the system, but we have considered that a valid e-mail is essential for providing responses to the citizens. Any user with other role needs to be authenticated in the system.

A *Reception Responsible* user receives all the comments (complaints and suggestions) and he may personally answer the comments or assign them to *Unit Responsible*

users, assisted by the two intelligent agents, the *Unit Assignment* agent and the *Comment Classification* agent.

The *Unit Responsible* user is usually a civil servant in an administrative unit. Such a user only receives assigned comments from the *Reception Responsible* and he must answer in time. There is a final role in the system, the *General Administrator*. This user is in charge of creating, modifying and deleting users.

And on the other hand, a series of comment marks have been created so that users and administrators can follow the process of any comment: (1) *Kind*, an initial classification of the comments –might be a complaint or a suggestion; (2) *Received*, the comment has been received and saved in the system and may be processed; (3) *Invalid*, a rude, insulting, offensive or non constructive comment, which will not be accepted in the system; (4) *Analyzed and Valid*, if the content analyzed is accepted; (5) *Threshold*, when a timely warning threshold has been overcome; (6) *Timeout*, when a final time-based threshold has been exceeded; (7) *Assigned*, if the *Reception Responsible* has re-addressed the comment to a *Unit Responsible*; (8) *Answered*, for the case of a complaint that has been answered; and finally, (9) *Filed*, when the process is fully accomplished.

3 Validating Comments

Complaint or suggestion arrival comprises the time range from the moment when a user enters a comment in the system up to the logical bifurcation -complaint or suggestion-.

When a citizen wants to file a complaint or a suggestion (a comment, in general) by way of our CS-WCP, he must fill in an electronic form. And, some additional information is saved in an automatic manner: arrival date and hour of the incoming comment.

Users of the system are warned about acceptance conditions for their comments (rude, insulting, offensive, in general non constructive comments are not allowed) and they are also informed of the next steps following the current one.

It is important to have a correct e-mail for feed back and confirmation purposes. The process is started when a confirmation of a comment arrives to the system.

There are three tasks performed in parallel at this point: (1) a notification is sent to the *Reception Responsible*, (2) a new comment thread is saved, and (3) a gratefulness message including some information about the next steps is shown on the user's screen.

The *Reception Responsible* receives an e-mail with the new comment, but he has the whole information, even the notification in his intranet client too.

Depending on the kind of comment, complaint or suggestion, the workflow will take a different way, deriving to one or another task.

Once a comment is saved in the system, a set of tasks is performed depending on the kind of comment. If it is a complaint, a set of tasks will be performed, and a different set will be performed if it is a suggestion.

Anyway, an acknowledgement is always mailed to the user. Acknowledgements include the final date when the response should be answered (only for complaints). Comments at this point in the workflow process will be marked as *Received*

Comment. Afterwards, the *Reception Responsible* will analyze the comment in order to check if the content of the comment is appropriate, but the *Reception Responsible* is guided in his decision through the intelligent agent called the *Semi-automatic Garbage Content* agent. This agent behaves as a filtering agent [10, 13] and classifies comments as valid or invalid; then, a particular user of the system decides what to do with the comment, or it even can be automatically eliminated. For this purpose, the agent is fed by a vocabulary containing a full set of semantic terms related to unsound words. The agent automatically mines the comments to extract the number of words present in the unsound vocabulary database. The recommendation of the *Semi-automatic Garbage Content* agent is two-fold: valid comment, if the number of unacceptable terms in the comment is reasonably low or invalid comment, when the number of invalid terms overcomes a predefined score.

If the decision taken by the responsible person is finally that the comment is invalid, the comment is marked as an invalid comment, and it is separately saved (for future statistics purposes).

Spanish laws force public administrations to establish a limit time for any administrative procedure. That is, administrative procedures should be completed in a finite time. If a person of the public administration does not answer a question in time, obviously the system can not do much. Nevertheless the system helps the public workers by providing two control times.

A person who has to answer a complaint always can see how much time is left in the intranet of the CS-WCP system. He perfectly knows that the answers must be sent out before the final time. In order to provide an efficient aid, the system incorporates two thresholds: a *warning threshold* and a *final threshold*. The first one, the *warning threshold*, is always lower than the other one. When the procedure is near to finish without being answered, a new e-mail is sent to the person who must answer (*Reception Responsible* or *Unit Responsible*, warnings are only for people who should answer the complaint), alerting about the proximity of the final time. This complaint comment is stuck out in the intranet. Otherwise, if nobody answers a complaint after the *final threshold*, then this would be the worst situation and three parallel tasks would be performed: (1) to notify this fact to the *Reception Responsible*, (2) to mark the comment as a timeout comment, and (3) to show this information in the intranet of the *Reception Responsible*.

4 Empirical Results

The system is running from April 2005 and during these 12 months it has received 471 suggestions and complaints. The responsible of the system has not published the new service because this first year has been planned as a test period. With the CS-WCP system, the Albacete Town-Council wants to improve the citizen participation [11].

The CS-WCP system is not only a mailbox of suggestions. The Town-Council must provide an administrative response to the citizen who has sent a complaint or a suggestion.

One of the indirect benefits of the system is the inclusion of a simple filter by means of the e-mail validation. This technique has allowed avoiding a high amount of malicious emails sent to the system.

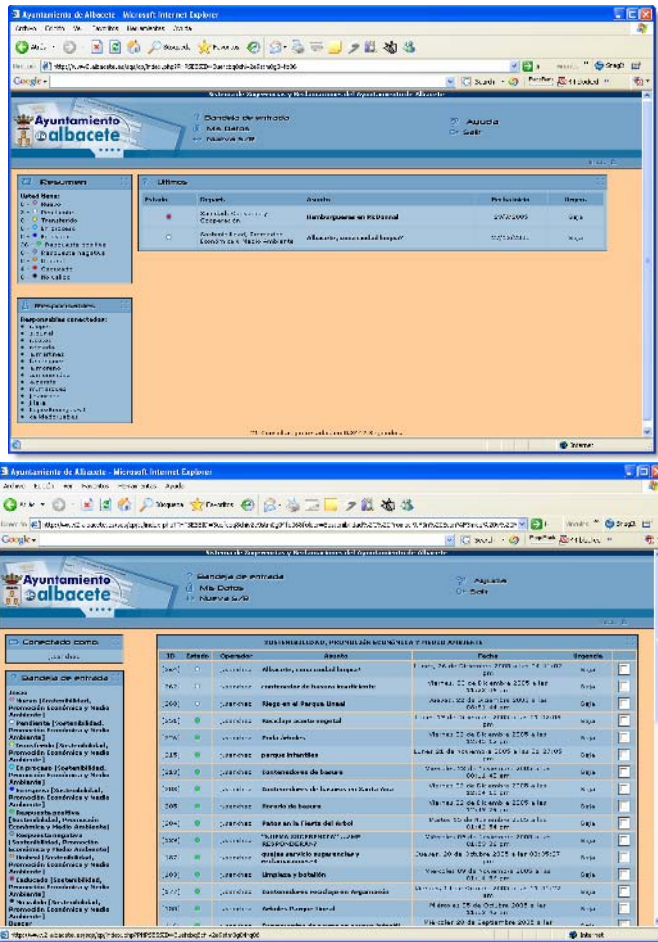


Fig. 1. Managing complaints and suggestions

Figure 1 shows two views of the administrative module of the CS-WCP. This module is used by civil-servant to follow the status of each complaint or suggestion. A color bullet indicates the status: new, pending, transferred, processing, waiting, positive response, negative response, out of date.

A responsible of an administrative area can check his own list of complaints and suggestions and he can manage the status of the item.

Figure 2 shows the form used by a responsible of an administrative area to answer a complaint or suggestion. The responsible can change the status and urgency of the item.

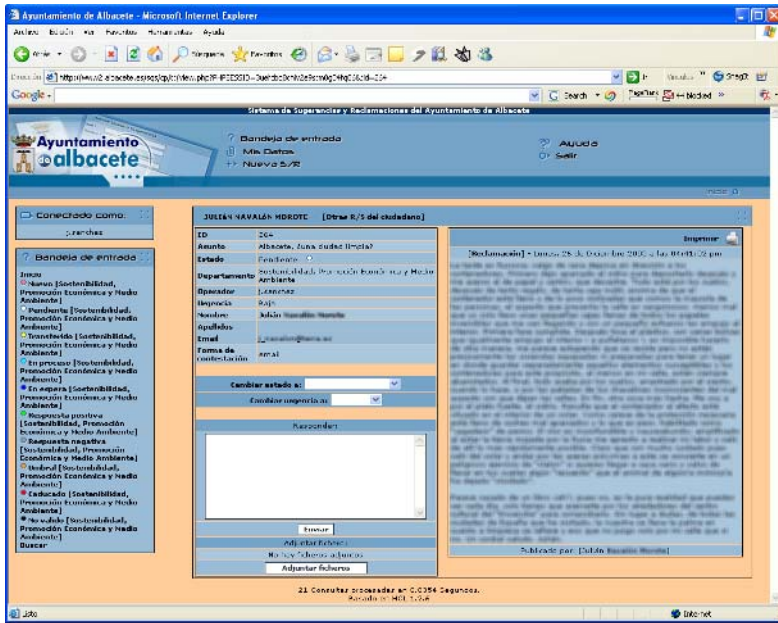


Fig. 2. A civil-servant uses this form to answer a complaint or suggestion

During this period of 12 months, over 75% of the suggestions and complaints arrived to the system via the Web. The civil-servant uses the CS-WCP in all cases to manage the different complaints and suggestions. The rest over 25% arrived by phone calls or by paper using the Citizen Office. It can be noted how citizens prefer to use the Web to communicate their suggestions or complaints.

Another interesting point is the channel selected by citizens to receive the administrative response:

- By letter: 33,3%
- E-mail: 63,91 %
- Fax: 0,21 %
- Phone: 2,55 %

As it can be observed, over 75% of citizens use the Web to put their complaint and suggestion but only over 63,91% of citizens prefer to receive the administrative response via e-mail.

The different complaints and suggestions can be organized in administrative areas depending on the subject. The areas involved in this first year of the system are the next ones:

- Mayor's Office: 13,16%
- Environment: 14,23%
- Quality: 10,83%
- Culture, Sports and Festivities: 12,31%
- Personal, Internal questions, Finances: 5,52%

- Woman, Equality, Participation: 1,91%
- Town Planning: 5,52%

The system has increased the quality of service of the local public administration thanks to the rapid processing of complaints and suggestions. Citizens feel that town-council hears what they have to say.

The system can be tested on <http://www.albacete.es> and it is available only in Spanish.

5 Conclusions and Future Work

An intelligent Web-based collaborative system to support the suggestions and complaints administrative procedure called CS-WCP has been presented. This is a research in progress paper that can contribute public services to increase their Quality of Service.. A good quantity of suggestions reveals the society degree of maturity. Modern public administrations need to hear the opinion of their citizens.

A town-council is a rich scenario for the deployment of CSCW systems because there are several groups and roles of people working together. Civil servants in a local administration are organized in functional groups that have to answer the suggestions and complaints from citizens.

Both complaints and suggestions should be managed by different groups inside the town-council in a collaborative way. A CSCW system can play an important role to help public administration reach a higher level of quality.

The main collaborative aspects managed in this system are the coordination between different civil servants to attend a complaint or suggestion and the communication between public administration and citizens.

The procedure has been modeled using a workflow system and moved from manual to semiautomatic due to the introduction of intelligent agents.

Future works include the deployment of the system and its evaluation using satisfaction questionnaires and usability metrics oriented to CSCW systems.

References

- [1] European Commission: eEurope 2005: An information society for all (An Action Plan to be presented in view of the Sevilla European Council, 21/22 June 2002). COM.2002 263, European Commission.
- [2] Greif, I.: Computer-Supported Cooperative Work: A Book of Readings. Morgan Kaufmann, San Mateo CA, 1988.
- [3] Grudin, J.: CSCW: History and Focus. University of California. IEEE Computer, 27, 5, 19-26. 1994.
- [4] Heeks, R.: Reinventing Government in the Information Age: International Practice in IT-enabled public sector reform, Routledge, New York, 2002 (second edition).
- [5] Holmes, D.: eGov: eBusiness Strategies for Government, Nicholas Brealey Publishing, London, 2001.

- [6] Johnson-Lenz, P. and Johnson-Lenz, T.: Consider the Groupware: Design and Group Process Impacts on Communication in the Electronic Medium. In Hiltz, S. and Kerr, E. (Ed.): Studies of Computer-Mediated Communications Systems: A Synthesis of the Findings, (Research Report, Band 16 Computerized Conferencing and Communications Center, New Jersey Institute of Technology: Newark, New Jersey, 1981.
- [7] Karacapilidis, N., Loukis, E., Dimopoulos, S.: A Web-based System for Supporting Structured Collaboration in the Public Sector. Proceedings of Third International Conference in E-Government, EGOV 2004; Zaragoza, Spain; 30th August to 3rd September, 2004
- [8] Korhonen, J., Pajunen, L., Puustjärvi, J.: Automatic composition of web service workflows using a semantic agent. IEEE/WIC International Conference on Web Intelligence (WI'03), pp. 566-569. 2003.
- [9] Lenihan, D. G.: Treating Information as a Public Resource. The Future of e-Governance Workshop, Syracuse University, 2002.
- [10] Lieberman, H.: Integrating user interface agents with conventional applications. Knowledge-Based Systems, volume 11, issue 1, pp. 15-24, 1998.
- [11] Masters, Z., Macintosh, A. and Smith, E., (2004); 'Young People and E-Democracy: Creating a Culture of Participation'; Proceedings of Third International Conference in E-Government, EGOV 2004; Zaragoza, Spain; 30th August to 3rd September, 2004
- [12] Ruiz V. M., Gallud J. A., Fernández-Caballero A., Lozano M. D., (2005); Complaints and suggestions web-based collaborative procedure. International Conference on E-Government, EGOV'05, pp. 336-342 (2005).
- [13] Sim, K.M.: Toward an ontology-enhanced information filtering agent. ACM SIGMOD, Volume 33 , Issue 1 (March 2004) , pp. 95-100, 2004.

Social Visualization Encouraging Participation in Online Communities

Lingling Sun¹ and Julita Vassileva^{2,*}

¹ Solutions AB TELUS Business Transformation, Edmonton, Canada

² Computer Science Department, University of Saskatchewan, Canada
gloriasun@hotmail.com, jiv@cs.usask.ca.

Abstract. In order to encourage users to participate more actively and bring more contributions to peer-to-peer (p2p) online communities, we propose to create a motivational community visualization based on the social comparison theory. This paper describes the design of static version and a dynamic version of this visualization developed in our lab, explains the advantages and the disadvantages of the static version and the reason why we decided to develop the dynamic version. This paper also gives a detailed evaluation on the dynamic version.

Keywords: participation, online communities, social visualization.

1 Introduction

The “cold-start” problem is well-known in most online communities, e.g. peer-to-peer (P2P) file-sharing networks, discussion forums, IRC systems, social networking or blogging systems. While some web-based online communities manage to attract users and grow enormously, others never reach the “critical mass” of active users needed to sustain the community and ensure that there are enough new things happening in the community, for example, new shared resources, posts, or blogs which can attract users to revisit the community. Often online communities created to serve a specific role in a certain geographic or institutional context remain unused. It seems like a “chicken and egg” problem: A community is only interesting if many people are participating and contributing; but to get users to contribute, you have to provide an interesting community first.

This paper proposes to motivate users to participate by visualizing the community and the levels of participation of all community members, hoping that the social visibility will enact social norms, stimulating users to engage in responsible and reciprocal behaviour. By participation we mean activities which benefit the community and demonstrate involvement in the community, like contributing materials, rating and commenting materials contributed by others, logging into the system and reading materials contributed by others.

The paper is organized as follows: in the next section, we give an overview of some of the work on motivation from the area of social psychology as well as other

* Corresponding author.

approaches to using visualization in online communities. In section 3, we describe our first approach to design the motivational visualization and the lessons learned from its brief deployment, as well as a second, improved version of the visualization. Section 4 presents the results of evaluation of the second version. These results and the implications as well as directions for future work are discussed in section 5.

2 Related Research

Most studies on human motivation have been done in the area of social psychology and for real communities. Leon Festinger found in his famous 1954 paper on social comparison [9], that humans tend to compare their achievements and actions with people who they think are similar to them in some way. For example, when a student wants to know if she is good at math, typically, she compares herself with the other students taking the same math class, rather than with her professor. However, when there is no suitable peer group, people will compare with almost anyone [9]. On the other side, when one knows that others will compare with him/her, one acts more responsibly. People normally want to be positively recognized in their community and are willing to make an effort to gain social reputation, providing the effort is affordable and worthwhile compared with the potential benefit of the reputation. Another interesting application of social comparison in e-collaboration, [18] studied short-time groups sharing ideas.

Asch's conformity study [1] had proved that people generally want to "fit in groups". The fitting behaviour at an interpersonal level happens, for example, when someone sees a friend doing or believing in something and starts believing or doing the same thing. Fitting, at a collective level happens, for example, when one sees trends in the behaviour of others, e.g. the style in dressing, and changes his/her own style of dressing to fit in, even if only for a particular occasion.

It seems that social conformity is a motivator for users in online communities too. Cosley et al.'s experiment [9] in the context of an online movie rating systems also proved that people generally want to "fit in" their group. In this experiment subjects were divided into four groups, A, B, C, and D. Each group was seated in a separate room so that they would not influence each other and were told to rate movies that they have seen before. The users in the first three groups A, B, and C were shown system-predicted ratings for each film they were rating and group D was not. The results showed that a significant proportion of the ratings given by the first three groups of participants correlated with the system-predicted values they were shown, which means that these users tried to conform to the predictions [4]. This experiment indicates that the designers of the software infrastructure of the online community can exploit the phenomenon of social comparison to influence user's behaviour. For example, if they are aware that most of the other users, similar to themselves contribute actively to a community, and that their contributions are lacking, they may be stimulated contribute more to the community [3]. For social comparison to take place, however, users have to be made aware of the behaviour of other users as well as of their own behaviour. Visualization has been used in online communities to create awareness about the other users and the things happening in the community. VisitorVille [16] visualizes websites as cities and visitors as passengers in busses.

VIUM [20] is designed to stimulate user reflection on their learning and displays learning concepts as a graph of texts in different fonts, colours and brightness to represent how much the user knows about each concept.

Social visualization approaches using different metaphors have been proposed to stimulate the activation of social norms in the online community. For example, the Babble System [8] visualizes a chat room as a pie with moving dots on the pie representing users to show which users are actively participating in the conversation (those, whose dots are close to the center) and which users are mostly listening (on the periphery). The Chat Circles [21] uses circles filled with different colors and texts representing the conversations. The Task Proxy [7] visualizes task groups in a company as differently-coloured cells in a honeycomb creating conditions for social comparison and therefore some social embarrassment on those groups who are lagging behind. This embarrassment would create social pressure and motivation to perform on par with others.

Erickson proposed a number of guidelines for designing social visualizations for online communities [6-8]. He makes an important distinction between “translucence” and “transparency”, emphasizing that the information showed in the visualization does not necessarily have to be very detailed and exact. In most of the cases, it is better just to give the user a general idea, and even in some cases a certain level of misrepresentation may be beneficial. Also, customization should be avoided; all users should see the same thing so that they can feel responsible for their actions, since they know that others see the same things as they and are aware of what they do [6].

There are also general design guidelines developed in the visualization community, which are aimed at reducing cognitive overload, using properties of human vision, and usability guidelines. The choice of metaphor is very important, since an appropriate metaphor is intuitive to use and doesn't require a complex legend for interpretation. Applying hierarchical structure [5, 18] and using composable layout and visual sets [15] are always helpful, when designing information-compact visualizations for large networks. Proper use of location and color contraction of visual components will successfully attract attention [12]. “Richly expressive information visualization is difficult to design and rarely found” [11], so it is always beneficial to make the visualization easily reusable in similar situations.

In the next section, we describe two versions of community visualization designed according to the guidelines mentioned above. The goal of the visualization is to stimulate social comparison and to motivate users to contribute resources to an academic paper-sharing online community called “Comtella”.

3 Two Visualization Designs for a File-Sharing Community

Comtella is a paper sharing online community developed in the Madmuc Lab at the University of Saskatchewan. It started in 2002 as a peer to peer, Gnutella-based file-sharing system (originating the name “Comtella”, which stands for “Community Gnutella”) [20]. It evolved through several versions, using different technologies, including a version in 2003-2004 where users shared paper URLs, not files and all servents resided on a server to ensure the simultaneous presence of nodes needed to allow sharing in a small community. Later (in 2004-2005), it was re-implemented as

an entirely centralized, web-based system for sharing URLs with a data-base on the server storing contributed URLs. Comtella was used to support students sharing academic papers in the MADMUC research lab and in several (5) undergraduate classes at the Computer Science Department at the University of Saskatchewan. The common feature in all the versions is that the community is relatively small (up to 40 users), closed, and that users share and search academic articles by the “areas of interest” or topics in the class (according to the weekly class schedule). The community visualization for Comtella was included from the very first Comtella implementation [3] and has evolved through two versions, a static and dynamic one. The next sections present the design of these two versions.

3.1 Static Design

The first version of the visualization was designed for the peer-to-peer version of Comtella. In this version, there were two important components of participation that were desirable for ensuring a sustainable community:

- sharing a lot of files (new or downloaded from others), which ensured redundancy and availability of resources necessary for the P2P search to work, and
- contributing a lot of new files, to increase the diversity of resources in available in the community.

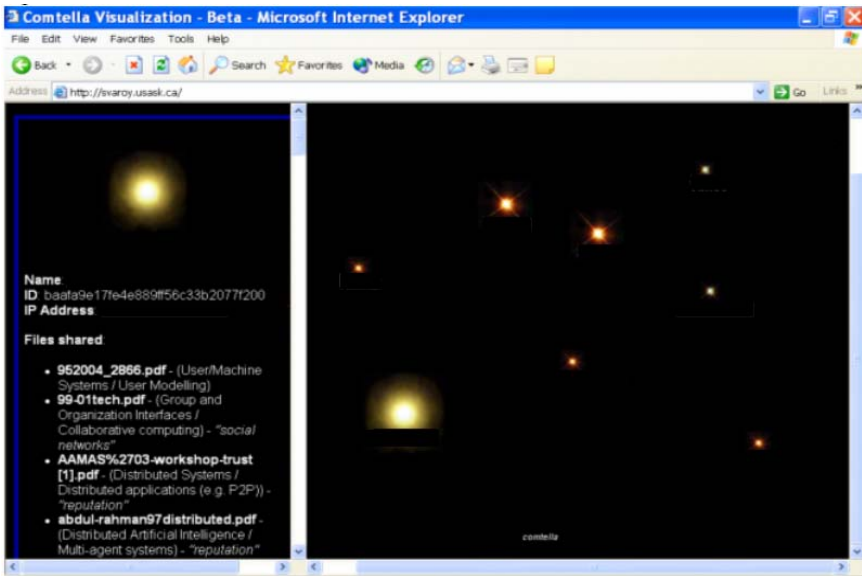


Fig. 1. Motivational visualization - static version (user names and IP address are blacked to preserve the privacy of the participants)

Therefore the visualization showed these two aspects of participation to encourage social comparison among users. It was designed a static webpage, which showed all users online at the moment. The same view was shown to every user, as

recommended by Erickson [6]. The visualization (shown in Figure 1), used the metaphor of a mid-summer night sky, first introduced in [2]. The size of a star was determined by the user's number of contributions (files shared in the Comtella community). Users who shared more than the average number of files per user in the community had larger stars, users who shared roughly the same as the average had medium-sized stars, and users who shared less than the average had small stars. The color of a star was either red or yellow. Red stars represented users who shared more new files than the number of files they downloaded from other users in the community, and yellow stars represented users who downloaded from others more than the number of new files they shared with the community. In this way people who enriched the shared resources pool with new files were recognized. The "sun", which displayed as a big yellow star, represents the "best user" among those who were currently online. The "best user" was the one that shared more than everyone else and has contributed new things to the community, rather than downloading from others.

We did not get a chance to carry out a systematic evaluation for this visualization. There were 16 users (graduate students in the MADMUC lab, as well as several computer science professors from the department and several other students). Approximately 10 of them were using it for more than one month. Most of the users accessed the visualization webpage at least once in each log-in session. About 200 unique papers were shared in the Comtella community, but mostly by one user. Through interviewing the users we obtained feedback about the visualization that revealed the following problems:

1. The visualization did not provide enough interactivity.
2. The graphical location of each star on the screen was random and meaningless, but users were trying to attach meaning to it.
3. The motivational effect was not very strong, though everyone found it a good idea in principle.
4. The users mostly had specialized interest in one topic, e.g. "distributed systems" and were forced to compare themselves with users with interests in different topics, e.g. "human computer interaction", whom they didn't necessarily know or want to compare with. It would have been better to see oneself as a "bright star" in the sub-community interested in a given topic.
5. The visualization was also misleading since it showed only the users who were online at the moment and calibrated the sizes of stars accordingly. In this way the best user at the moment, even if with relatively few contributions would appear as the Sun. However, this lead to inconsistency in time, since for example, a user who saw herself as large star one day, found that her star has shrunk when he logged in the next time, even though she had made several contributions in the meantime, simply because a bigger contributor happened to be logged at this time. This inconsistency was reported by users as unfair and actually discouraging contributions.
6. Because of visualizing only the users who were on-line at the moment, often users saw themselves as the only star on the night sky, which amplified a feeling of isolation and lack of community, instead of creating a feeling of co-presence.
7. The visualization was not self-explanatory. It needed a legend to explain the meaning of each size and colour of a star.

There was no evidence that the static visualization encouraged participation in any way. However, the users generally liked the idea and indicated in their feedback that an enhanced version of this visualization would be useful in quickly discovering what their colleagues were working on, easily finding out what the hot topics were, and raising the users' awareness of the online community. We re-designed the visualization by taking the user feedback into consideration and the resulting, dynamic design is presented in the next section.

3.2 Dynamic Design

The dynamic version of the visualization allows users to specify how they want to view the community by selecting different criteria, and the visualization is generated upon request with the latest data from the database. In this way, the users are provided with a simpler graphical language based only on the size and colour of the "stars", but these two dimensions get a different meaning depending on the criterion selected by the user. The following viewing criteria were introduced: view by topic, view by number of original contributions (default), view by number of shared files (also called total contributions), view by login-frequency, and view by status which can be "bronze", "silver" or "gold", depending on a summative measure of the total participation of the user in the system. Every user can use the same set of selection criteria to view the community, so everyone has an equal opportunity to see each possible customized view (somewhat coherent in theory with Erickson's guideline [6]). This allows social comparison to happen in different dimensions, depending on how the users may wish to define their "peers". The visualization is implemented as a graph-generating application written in Java embedded in the Comtella client interface.

The metaphor used in this design is the same (night sky with stars), but it is highly stylized – instead of stars, the visualization shows a group of nodes (i.e. circles) on a black background (see Figure 2). The static version of the visualization, taking the advantage of the web browser and HTML, was able to display pictures of real stars saved as JPG images. However, in the dynamic design the system has to draw every component of the visualization, depending on the selected view. It is easier and faster to draw primitive geometry elements such as a circle and a dot.

Unlike the static version, all users are shown, regardless on whether they are on- or off-line at the moment. This alleviates the problem with the inconsistent size of stars in the static version and creates a stronger feeling of "co-presence" in the system. Each user is represented by node, which is either filled or empty. A filled node represents a user who is currently online and an empty node an offline user.

The size of a node is defined by the contribution of the represented user in a selected topic under the selected criterion. Different views can be generated depending on the topic selected by the user from the pre-defined topic list (see Figure 2, the pull down menu on the top left corner shown enlarged in Figure 3). If the user does not make any selection the default topic is the "General View", showing the nodes with sizes reflecting and sorted by their original contributions for all the topics. The users are able to double-click on a node to see the list of papers shared by this user (the left black part of the window in Figure 2); a single click on any of these files opens up a comment-window (see Figure 4). When a user moves his/her mouse over a node in the visualization, a brief summary of the contributions made by the

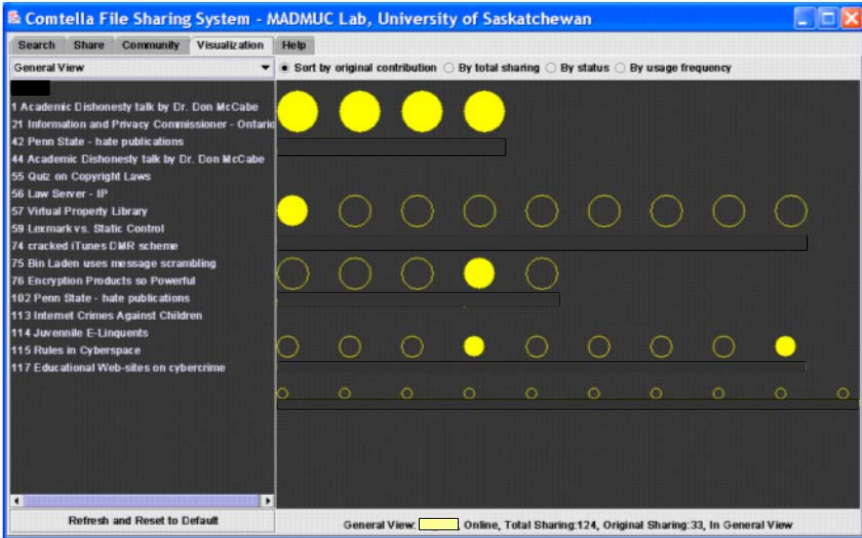


Fig. 2. The dynamic design of the visualization (user names hidden for privacy)

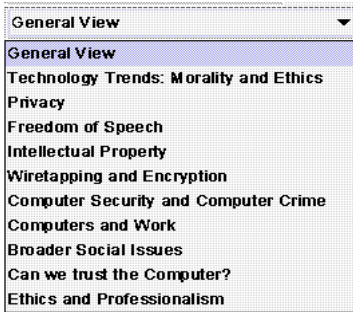


Fig. 3. Topic selection box

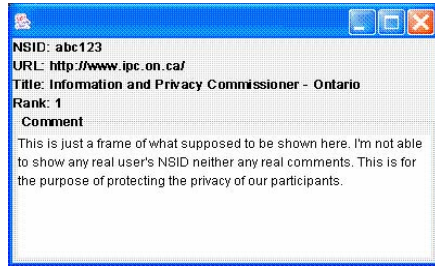


Fig. 4. A comment window

represented user appear in the bottom bar of the window, on the right side, below the visualization.

Unlike the static version, here the nodes are not randomly located. Instead they are laid out hierarchically at four levels in the descending order of sizes. According to the guidelines [6, 8] for designing community visualizations, it is not necessary to visualize the exact difference in the numbers of articles shared by two different users. Instead, a certain amount of approximation is preferable, since it is harder for the users to discover small differences in the sizes visually. So instead of determining the size of a node by the exact number of contributions, we classified the contributions into four levels and users classified in the same level are represented by nodes of the same size. The following is the algorithm according to which the classification is done. If the total number of users is N :

$X = 10\% * N \rightarrow$ the number of nodes at Level One,
 $Y = 40\% * (N - X) = 36\% * N \rightarrow$ the number of nodes at Level Two,
 $Z = 50\% * (N - X - Y) = 27\% * N \rightarrow$ the number of nodes at Level Three,
 $R = 27\%N \rightarrow$ the number of nodes at Level Four.

An obvious alternative to this approach is to have all four classes of equal size, i.e. to divide the total number of users into equally-sized groups. This method will not generate noticeable gaps between different levels and therefore may reduce the motivational effect. To be more specific, if there is the same number of users at the highest level as there is at other levels, then the best contributors will not be able to feel the exclusiveness of their high social status as much as they would, if they see themselves as some of the few at the exclusive top level. Similarly, the users who barely make any contributions to the community may not feel so urged to participate more if the majority of users do not contribute. There would be a stronger motivation to comply, if there are not many people at the bottom level and they are among the few. For this reason, there should not be too many people at either the top level or the bottom level. However, the top level should not appear too hard or impossible to reach, therefore its size cannot be too small; otherwise, it will be limited to only one or two users. Similarly, the size of the lowest level also can not be too small; otherwise, users at this level may feel too discouraged and may give up using the system. By not combining the two middle levels we also hope for maximizing the motivational effect; it would be harder to persuade people to improve, if they see that most of the others are just like them. Moreover, merging the second and the third levels will create a super-sized middle level which will make the top and the bottom levels appear too small and exclusive.

In the next section we present the evaluation of the social visualization which was carried out in a undergraduate computer science class.

4 Evaluation of the Dynamic Visualization in Comtella

Comtella was used to support students in CMPT 490 to share web links to class-related articles and do their course-work. CMPT 490 (consequently renumbered to CMPT 409) was a 4th year computer science class on social impact of information technology offered by the Computer Science Department at the University of Saskatchewan. The list of the areas of interest corresponded to the weekly topics discussed in the class, as shown in Figure 3. The experiment lasted three months, or thirteen weeks, and covered ten different topics. Each topic was discussed for one week except the sixth topic, “computer security and computer crime” which was discussed for two weeks plus an extra week in between (the reading-week break); thus this topic ran over three weeks. The last week was entirely dedicated to team-project presentations, and there was no particular topic in Comtella for this week.

The students were first given a Comtella client with interface excluding the visualization. In the middle of the term, after the sixth topic (i.e. after the reading week break) they were given a new client with interface including the visualization. System usage data was collected from each participant and compared before and after the introduction of the visualization. The effect of the visualization was evaluated by four participation metrics: (1) the total number of shared articles for each topic, (2)

the number of original (new) shared articles on each topic, (3) the number of comments given on the shared articles, and (4) the number of ratings given on the shared articles. Also a qualitative study of the user experience was conducted through an online questionnaire which the students voluntarily filled after the end of the class. The questions were trying to find out if the students felt that they belonged to a community and if the visualization had encouraged them to compare themselves with others, to compete, and to what degree it motivated them to participate.

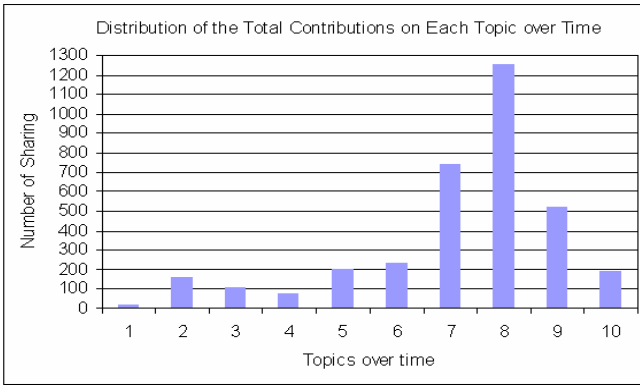


Fig. 5. Distribution of the Total Contributions on Each Topic

Figure 5 shows the distribution of the total contributions made by all users over the 10 topics. It shows that the number of total contributions after introducing the visualization is significantly higher than the number contributed before the visualization was introduced. In particular, the contributions for topic 7 are about 3 times higher than for each of the previous topics. Similarly, Figure 6 shows the distribution of the original contributions made by all users on each topic throughout the experiment. Topic 7 represents the contribution in the first week after applying the new interface including the social visualization. Even though the number of

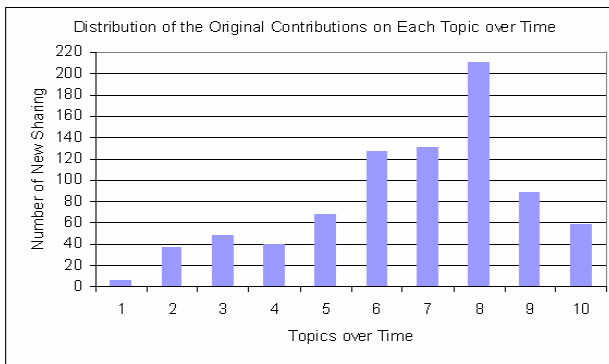


Fig. 6. Distribution of the Original Contributions on Each Topic

contributions for topic 7 is not much larger than for those for Topic 6, one should keep in mind that topic 6 was “Computer Crime and Security”, which traditionally attracts most interest in the class and is therefore discussed over two weeks (plus one week of break between them), so the students had plenty of time and intrinsic motivation to bring new contributions.

The marked increase in contributions (also in ratings and comments) in the second half of the term is obvious from Figure 7. Row marked “before 7” in this figure refers to the sum of the data from topics one to six, and “after 7” refers to the sum of the data in all topics after and including topic 7. Figure 7 shows that the visualization had a strong positive effect on the students’ sharing behaviour. Yet, towards the end of the term, the number of contributions dropped down close to the levels before introducing the visualization. We are not sure what the reason for this decline in participation was: there are many factors that play a role in a real classroom experiment, such as the inherent interests of students in certain topics, the amount of the coursework (most class projects are due by the end of the term, including the class project for CMPT 490, which limits the amount of time the students have to spend searching and contributing papers to Comtella). Also the professor’s promotion of the system in the class, may have played a role. The novelty effect is well known in the area of Human-Computer Interaction and may account for the initial interest in the students to use the system with the new interface, which died off as users got familiar with the visualization. An alternative explanation for the drop in participation towards the end of the course may be that the “score” had been settled already anyway. Participants had been using the tool for a number of weeks and everyone knew which fellow students were the ‘high-achievers’. It was not possible to join this ‘elite’ group with a week’s effort. This combined with increasing stress to finish the semester may have caused a drop in motivation and hence in participation.

| | Total Contribution | | Original Contribution | | Comments | | Ratings | |
|----------|--------------------|--------|-----------------------|--------|----------|--------|---------|--------|
| | number | % | number | % | number | % | number | % |
| All | 3526 | 100% | 821 | 100% | 888 | 100% | 578 | 100% |
| before 7 | 803 | 22.77% | 331 | 40.32% | 176 | 19.82% | 73 | 12.63% |
| topic 7 | 745 | 21.13% | 131 | 15.96% | 162 | 18.24% | 112 | 19.38% |
| after 7 | 2723 | 77.23% | 490 | 59.68% | 712 | 80.18% | 505 | 87.37% |

Fig. 7. Participation Data

Did those users that accessed the visualization more often contribute more than those who didn’t access the visualization? Figure 8 maps the original contributions made by each user against the number of times s/he accessed the visualization (the original contributions view). Each point represents a user, where the x-coordinate of the point shows the number of times she accessed the visualization and the y-coordinate shows the number of original contributions made by this user. We can see that most points are scattered around the diagonal but there are also 5 outliers.

Similarly, Figure 9 shows the total contribution made by a user against the number of times s/he accessed the visualization (the total contributions view). The correlation coefficient for the data in Figure 8 is 0.66, while the correlation coefficient of data in Figure 9 is 0.34. This means that the visualization had a stronger effect on encouraging original contributions.

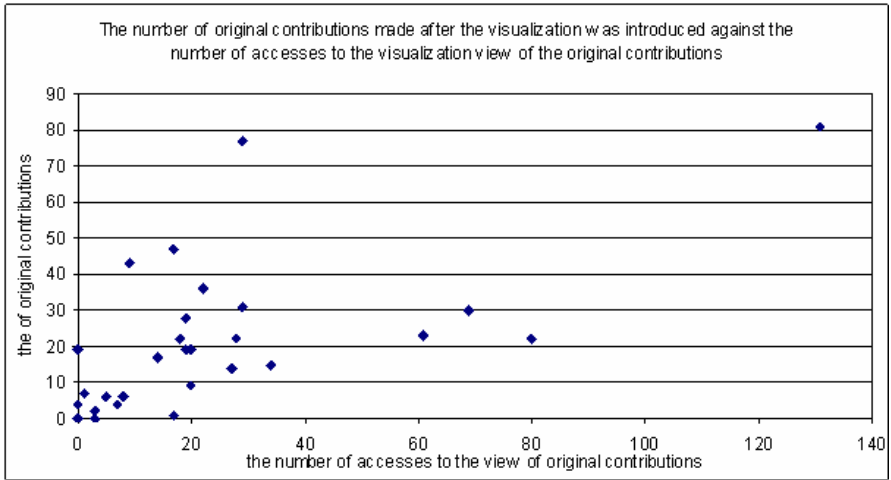


Fig. 8. Original Contributions against Usage of the Visualization

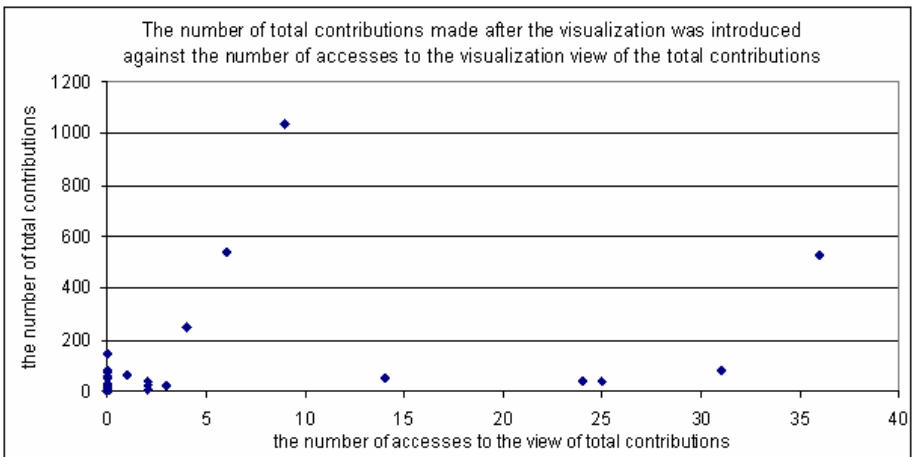


Fig. 9. Total Contributions against Usage of the Visualization

The usage data also showed that users did not want to make the effort to do extra selection on the sorting criteria (i.e. to select different views). They usually left the sorting criteria as default (“sorted by original contribution”) and this indicates why the visualization encouraged more original contributions than anything else. None of the users ever selected on “sorted by usage frequency”, which indicates this criterion was irrelevant for the users.

For the qualitative study, all students were invited to fill in a questionnaire and received a 3% bonus towards their total mark in the class as a reward for the effort. To prevent any influence on the evaluations that the students would give, the students received the same benefit regardless if they participated in the experiment (signed the consent form) or if they used the system. The students who didn’t sign consent exited

the questionnaire immediately and no data was stored in the system about them apart from the fact that they have logged in to fill it, so that they could get the 3% bonus. The class instructor had no access to the students' data about the experiment and didn't know who used the system and who did not, she wouldn't know who answered the questionnaire and who did not. The answers were anonymized immediately, as well as all the participation data from the database, to mitigate eventual influence on the answers. The questions regarding the visualization together with the results (what percentage of the 35 students chose each answer) are listed below:

1. What would your reaction be if you saw yourself as one of the smallest nodes in the visualization?
 - a. Take immediate action: share more links to make your node larger. (35% chose this option)
 - b. Think of sharing more links, but later. (19%)
 - c. Feel unhappy, but do nothing. (10%)
 - d. Feel that the system is unfair, so it doesn't make sense to contribute. (10%)
 - e. Do not care, so will do nothing. (16%)
 - f. Other, please specify: (10%)

In the optional comments given for this question, a couple of students expressed concern about the quality of the shared articles as they noticed that not all the users with bigger nodes were sharing useful resources.

2. If you saw yourself as one of the largest nodes in general, what would you do?
 - a. Feel proud of your status and try to contribute even more. (42% chose this option)
 - b. Feel proud, but at the same time, in some sense 'exploited'. The others are not bringing in so much, so I will stop or decrease my contributions. (6%)
 - c. Feel worried, you may be raising the bar too high and others may hate you or you may be perceived as an 'overachiever' by the others. (6%)
 - d. Feel nothing, since it is not important to me. (19%)
 - e. Other, please specify: (26%)

In the specifications given, some students said that if they were already one of the biggest stars, they would only share more links if they had time and in this case they would focus on the quality of their contributions. One student stated that if she saw him/her self as one of the biggest stars she would feel it was too easy to reach a high contribution level and the evaluation system needed to be improved.

3. If their answers to the above two questions are affected by whether the topic is of interest to them?
 1. a. Yes (55%)
 2. b. No (32%)
 3. c. I don't know. (13%)

This suggests that the users' competitive attitude doesn't depend much on whether a topic is of interest or not. It is important to notice that users replied in the context of a class where topics are changed based on a weekly schedule and are pre-selected. Perhaps the answers would have been different in an interest-based community e.g. a research group (as the system in with the static visualization described in Section 3.1), where users have long term personal interests.

4. What would you like to know about other users?

| | -2 | -1 | 0 | 1 | 2 |
|--------------------------------------|--------|--------|--------|--------|--------|
| who is online | 13% | 16% | 26% | 23% | 22% |
| how much others contributed | 3% | 6% | 13% | 35% | 43% |
| who downloaded from me | 3.23% | 9.68% | 12.90% | 35.48% | 38.71% |
| who I downloaded from | 3.23% | 19.35% | 19.35% | 38.71% | 19.35% |
| who gave similar ratings as me | 3.23% | 9.68% | 22.58% | 41.94% | 22.58% |
| am I a freeloader or an contributor? | 16.13% | 16.13% | 12.90% | 35.48% | 19.35% |

To sum up, about 77% of the subjects wanted to know how much others contributed and 55% were interested in knowing if others see them as freeloaders or as active contributors. This indicated there is great potential for motivating subjects to contribute more through this prototype visualization.

5 Discussion

The experimental results and the user feedback discussed previously showed that the motivational visualization effectively increased the participants' awareness of their Comtella community and encouraged social comparison as a result both the total contributions and the original contributions went up significantly, and participants gave more comments and ratings. The effect is more obvious on the original contribution than it is on the total contribution.

It should be acknowledged that it is hard to control the "noise" in real world experiment during a deployment of a system in class, such as the users' inherent interests in different topics and the currency and relevancy of the topics in the real world. It can be argued that the increase in participations may be caused by the subjects being more interested in topics discussed in the latter half of the term when the visualization was introduced. It can be also a result of getting used to the interface and developed a level of comfort with using the tool. It can also be argued that the topics discussed latter in the term were "hotter" and it was easier to find relevant articles on the web. However, according to the class instructor the opposite is true: the most interesting topics for students and richest of materials on the web are "Privacy", "Freedom of Speech", "Intellectual Property", "Wiretapping and Encryption" and "Computer Security and Computer Crime", which were discussed in the earlier half of the class. The topics discussed latter after the new interface was introduced, "Computers and Work", "Broader Social Issues" are too broad and often fail to attract student's interest. For the last two topics "Can we Trust the Computer", and "Ethics and Professionalism", it is harder to find online materials, since they are of more professional interest than general.

One clear conclusion was that user-customizable views are hardly needed, since users tend not to use them; most users only used the default view. This seems to support the first guideline in design visualization by Erickson (2003). Our main motivation for providing the different views was to simplify the complex graphical language (expressed in the legend) necessary to decode the meaning of the

visualization, based on our experience from the first version of the visualization. In retrospect, we could have not visualized unnecessary information and still provided a single view.

One general observation that was made in the experiment was that as the quantity of contributions increased, their quality somewhat deteriorated. This may be because the visualization showed only the quantity of the articles shared by each user regardless of the quality i.e. the visualization did not encourage comparison among the users with respect to the quality of their contributions. Several users found ways to game the system and exaggerate their nodes in order to gain higher status and visibility. Motivating social comparison in the quality of the contributions, comments, and ratings is an important future direction of research. There exists a lot of research on reputation mechanisms that can be helpful in this endeavor.

Motivating active users to continue their contributions or even increase them is another problem. In the questionnaire, one student indicated that if one was already the best contributor and was visualized as the largest node, there was no motivation for him/her to continue contributing. Another one indicated that she would then try to submit higher quality contributions. Probably the visualization has to take into account both the quality and quantity of user contributions and change dynamically the view to stimulate social comparison depending on what is needed most by the community – just any papers (if there is still very small number of shared resources) or high quality papers (if there is an abundance of resources).

From a technical point of view, a major issue in the design of the visualization is the clustering algorithm, used to classify users into four contribution levels based. Ideally, the algorithm should find a significant difference between the marginal cases on both sides of a boundary between two clusters. More specifically, if the list of nodes is sorted by the original contributions, there could be the case that the last node at the top level only shares one or two files more than the first node at the second level, which may share 10 more files than the second node at this level. A better algorithm should find reasonable gaps between contributions of users to classify these users into different levels. A compromise between desired sizes and sharper boundaries would be a direction to explore in the future.

One should not forget that the ultimate purpose of the Comtella system (apart from being an experimental tool for testing motivational approaches for user participation) was to facilitate students in finding and reading fresh materials related to the topics of their studies or research. One student commented that she didn't like the competition for bringing resources, since people were caring more about finding resources than about reading them, and *this* should be the main goal in the system. It seems that reading articles found by others is also an important way of participation, even if it is "invisible" from the viewpoint of the community, since it does not contribute to enlarging the pool of shared resources. It is important to remember that the resources are only valuable for the community, if they are *read* by the community members. Otherwise, participation becomes a game with no higher, in this case educational, purpose. Therefore, we should try to find ways of encouraging people to participate even as a "silent audience", if not as active contributors.

Acknowledgement. This work is supported by the second author' NSERC Discovery Grant.

References

- 1 S. E. Asch. (1951) Effects of Group Pressure upon the Modification and Distortion of Judgments. In *Groups, Leadership, and Men*, pages 177-190.
- 2 C. Alexander, S. Ishikawa, M. Silverstein, with M. Jacobson, I. F.-King, and S. Angel. (1977). *A Pattern Language: Towns, Buildings, Constructions*. New York: Oxford University Press.
- 3 H. Bretzke and J. Vassileva. (2003). Motivating Cooperation on Peer to Peer Networks. *Proceedings User Modeling UM '03, LNAI 2702*, pp.218-227.
- 4 D. Cosley, S. K. Lam, I. Albert, J. A. Konstan, J. Riedl. *Is Seeing Believing? How Recommender Interfaces Affect Users' Opinions*. CHI 2003, April 5 – 10, 2003, Ft. Lauderdale, Florida, USA.
- 5 S. G. Eick and G. J. Wills. (1993). Navigating Large Networks with Hierarchies. *IEEE Visualization '93 Conference*, pages 204-210.
- 6 T. Erickson. Designing Visualizations of Social Activity: Six Claims. *ACM CHI Proceedings* April 5 – 10, 2003 Ft. Lauderdale Florida, USA.
- 7 T. Erickson, W. Huang, C. Danis, and W. A. Kellogg. A Social Proxy for Distributed Tasks: Design and Evaluation of a Working Prototype. *ACM Proceedings CHI 2004*, April 24-29, 2004, Vienna, Austria.
- 8 T. Erickson and W. A. Kellogg. Social Translucence: Using Minimalist Visualizations of Social Activity to Support Collective Interaction. In *Designing Information Spaces: The Social Navigation Approach* (eds. K. Hook, D. Benyon, A. Munroe), Springer-Verlag: London, 2003, pp. 17-41.
- 9 L. Festinger. A Theory of Social Comparison Processes. (1954). *Human Relations*, 7, 117-140.
- 10 Free Peers, Inc. (<http://www.freepeers.com/>), BearShare (<http://www.bearshare.com/>).
- 11 M. C. Humphrey. Creating Reusable Visualizations with the Relational Visualization Notation. 11th *IEEE Visualization 2000 Conference (VIS 2000)*. October 08 – 13, 2000. Salt Lake City, UT.
- 12 V. A. F. Lamme. Why Visual Attention and Awareness are Different. *TRENDS in Cognitive Sciences* Vol. 7 No. 1 January 2003.
- 13 LimeWire. *The Gnutella Protocol Specification v0.4 – Document Revision 1.2.*: http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf
- 14 MetaMachine. *eDonkey2000 and Overnet* (<http://www.edonkey2000.com/>).
- 15 T. Pattison, R. Vernik, and M. Phillips. Information Visualization Using Composable Layouts and Visual Sets. *Conferences in Research and Practice in Information Technology*, Vol. 9. Commonwealth of Australia, 2001.
- 16 R. Savage VisitorVille <http://www.visitorville.com/meet-the-mayor.html>, <http://www.visitorville.com>.
- 17 Sharman Networks Ltd. KaZaA (<http://www.kazaa.com/>) Media Desktop, 2001.
- 18 M.M. Shepherd, R.O. Briggs, B.A. Reinig, J. Yen, and J.F., Jr. Nunamaker, (1996). Invoking Social Comparison to Improve Electronic Brainstorming: Beyond Anonymity. *Journal of MIS*. 12(3), 1996, pp. 155-170.
- 19 T. C. Sprenger, R. Brunella, and M. H. Gross. H-BLOB: A Hierarchical Visual Clustering Method Using Implicit Surfaces. 11th *IEEE Visualization 2000 Conference (VIS 2000)* October 08 – 13, 2000. Salt Lake City, UT.
- 20 J. Uther and J. Kay. VIUM, a Web-Based Visualization of Large User Models. *Proceedings User Modeling, UM'03, LNAI 2702*, pp. 198-202.
- 21 J. Vassileva. Motivating Participation in Peer to Peer Communities. *Proceedings of the Workshop on Emerging Societies of Agents. ESAW'02*, Madrid, Spain, September 2002. Available on line at <http://www.ai.univie.ac.at/~%7Epaolo/conf/esaw02/preproc/E0029.pdf>.
- 22 F. B. Viegas and J. S. Donath. Chat Circles. *Proceedings of the SIG CHI conference on Human factors in computing systems: the CHI is the limit*, 1999.

Analyzing the Roles of PDA in Meeting Scenarios

Gustavo Zurita¹, Pedro Antunes², Luís Carriço², Felipe Baytelman¹,
Marco Sá², and Nelson Baloian³

¹ Department of Information System and Management of the Economy and Businesses School,
University of Chile, Diagonal Paraguay 257, Santiago, Chile
gnzurita@facea.uchile.cl, felipe@baytex.net

² Department of Informatics of the Faculty of Sciences of the University of Lisboa,
Bloco C6, Campo Grande, 1700 Lisboa, Portugal
{paa, lmc, marcosa}@di.fc.ul.pt

³ Department of Computer Science of the Engineering School of the University of Chile,
Blanco Encalada 2120, Santiago, Chile
nbaloian@dcc.uchile.cl

Abstract. This paper proposes a conceptual model standardizing the meeting information structures underlying several scenarios of PDA use in meetings. The paper characterizes the memory and process components necessary to support XML-based interoperability between meeting systems. The scenarios, information model and architecture were validated through their adoption in three applications, developed by different teams and covering quite different domains. The applications, encompassing several meeting scenarios and adopting multifaceted device combinations, demonstrate the high level of interoperability supported by the proposed conceptual model.

1 Introduction

PDA have been recently regarded as powerful CSCW devices, combining several well-known characteristics such as autonomy, mobility, pervasiveness, small form factor and unobtrusiveness with shared information support. One good example is the mediation between healthcare personnel in hospital environments, where mobility and flexibility are paramount [1].

PDA may also assume a fundamental role in the meeting environment. PDA represent an opportunity to turn meetings more fluid, simplifying the way people bring information into and out of meetings, and serving as a dissemination tool for meeting-related information throughout the organization [2]. Another important role is revolutionizing the role of technology in meetings, which has always been problematic, since people handle meeting information in very subtle ways. Moreover, meeting processes are governed by complex procedures, which in many circumstances require an expert facilitator, who may benefit from PDA support [3].

The complexity associated to meetings has always challenged information technology [4]. Meetings may be distributed in time and space, posing significant restrictions to shared context awareness. They also bring together people with very distinct abilities, making it difficult to specify the interaction requirements. Many

times people are forced to plan the meeting process in advance, while other times such advance planning is impossible (e.g., emergency management), making technology configuration and management highly contextual dependent. To complicate even further these matters, meeting activities must be constantly adapted to the groups' varying perceptions of problems and goals.

In this complex scenario, in order to make significant advances in the diffusion of PDA in meetings, we must start by disentangling such complexity. In this paper we analyze the different elements that make up a meeting and identify the relevant relationships between these elements and the PDA functionality. Our contributions to the state of the art are the following:

- Characterize several representative meeting scenarios where PDA may be used to best advantage as meeting tools, either because they support information management, simplify the group process or increase contextual awareness.
- Model the information structure required by the application scenarios. This includes standardizing the memory and process elements belonging to the meeting information system.
- Propose and validate a generic architecture for the above information structure.

The paper is organized as follows. First, we present the research context. The following section is dedicated to describe the meeting scenarios. Next, we present the information systems and architectural perspectives of meetings. We then present three applications where the proposed architecture has been implemented and used. Finally, we present our research conclusions.

2 Research Context

Even before PDA became popular, researchers explored meeting support with Personal Computers (PC). Known as EMS (Electronic Meeting Systems), these solutions offer procedures and mechanisms aimed at achieving effective and productive asynchronous and face-to-face meetings [5]. PDA attracted the attention of EMS researchers mostly because of their increased user-interface abilities, including freehand input [6], remote shared view control [7], ubiquitous note taking [2, 8] and mobility [9]. Some of these studies also experimented the integration of PDA with large shared displays in the group context, in a configuration known as Single Display Groupware (SDG) [7, 10, 11].

In this promising research line we highlight the development of several systems: NotePals [12], RoamWare [9], Pebbles [7], Notable [8] and ShareNote [10]. NotePals and ShareNote allow creating personal notes in PDA, publicizing notes when people meet and recording the common notes when people leave meetings. While NotePals supports asynchronous uploading/downloading notes from a shared repository, ShareNote also supports synchronous use. Notable is also focused on annotations, building on the Post-It metaphor to integrate notes taken in meetings with other documents. RoamWare is mostly focused on supporting mobile face-to-face meetings, including those held in such places as corridors, relying on wireless technology to detect the group members and support information sharing. The Pebbles project allows PDA to be used in the meeting context as if they were PC mice and keyboards.

We observe that although many of these systems identify some common meeting information components (e.g. personal and meeting notes) and processes, they do not aim to standardize, integrate or support the interoperability between these information and processes components, as proposed in this research.

3 Meeting Scenarios

From our point of view, the standardization of meeting information structures should be anchored on meeting scenarios, providing rich descriptions of the system functionality and use in terms of goals, inputs, process and outputs in varied contexts (see Tables 1 and 2). As we many times observed in our experiments, the following scenarios are not mutually exclusive, but rather characterize several primary conditions driving the system use. Sometimes it is even possible to identify the occurrence of secondary conditions, which may as well be characterized in scenarios.

Ritual meeting. In the ritual meeting the most important is socializing and communicating [13]. The reasons behind this type of meeting are varied and include, for instance, team building and establishing norms in forming groups [14]. The type of participation is mixed and it should not be expected that everyone is holding a PDA or knowledgeable about meeting technology.

The meeting process is usually simple. Considering the lack of preparation and the heterogeneity of the participation, this type of meeting requires a human facilitator who is responsible for conducting the meeting process [3]. We expect that a reduced amount of information leaves the meeting, since outcomes are mostly intangible (satisfaction, sense of belonging, knowledge about others, etc.)

The PDA usage in this scenario is most probably sporadic and restricted to few users. This type of meeting could therefore be anchored in a SDG, so that the facilitator may lead the group activities. This scenario enables face-to-face meeting participants to collaborate via a shared computer display and simultaneously use PDA as input devices or remote commanders [15].

Deliberate meeting. This type of meeting requires advance preparation, either because the potential failure has severe organizational impact, or the problem and process are complex. The deliberate meeting is mostly related to problem solving and decision making. The meeting participants are carefully selected and previously informed about the meeting agenda. Therefore, this scenario integrates various asynchronous activities accomplished prior to meetings, such as agenda preparation, document sharing and preliminary discussion. This scenario also emphasizes post-meeting information, delivering the necessary context for subsequent activities. A shared repository is necessary to maintain such an integrated perspective. The participants may use their PDA to interact with the repository, updating information synchronously during the meeting or asynchronously after the meeting.

Finally, we note deliberate meetings usually require highly structured processes, organizing and regulating the participants' interventions. PDA may support these structured processes, like voting or decision-making.

Meetings ecosystem. This scenario is associated to an ill-defined or unexpected reality. The most significant difference to the deliberate meeting is that advance planning is compromised and it is up to the group to adopt the best strategy to achieve the intended goals (which may also be compromised [16]).

The meeting may then be regarded as an aggregate of sub-meetings with different goals, a behavior that has been observed in collaboratories [17]. PDA support this organized chaos: setting up sub-groups; defining tasks and sub-tasks; and supporting information exchange and awareness while moving through different contexts.

The shared repository and SDG may be required only in certain cases. There may exist no shared repository when the participants only share notes with the sub-groups where they are collaborating, but do not need to integrate and share this information. On the other hand, when the meeting outcomes are necessary in other meetings, they can be shared through the repository. As pointed out by [17], this scenario could use a SDG, allowing the participants to obtain situation awareness and use collaborative tools to enhance the meeting experience (e.g., sharing small text messages or voting). While the SDG provides situation awareness, PDA may be used to quickly interact with these features.

Creative/design meeting. This type of meeting is mostly focused on ideas generation. The focus on creativity may be supported with the brainstorming technique [18], while design builds upon creativity with discussion, assessment, decision-making and planning. PDA may support the design process mostly through collaborative sketching. For instance, a group of architects may work jointly on a sketch at a construction site [19].

The most natural data-entry mode for design is the stylus, because it reproduces the pen-and-paper mental model [20]. Sketching affords the visual symbols and spatial relationships necessary to express ideas in a rapid and efficient way [21]. This type of activity is also called brainsketching [20]. The ability to immerse the design activities in the physical context affords the meeting participants to immediately explore the proposed solutions, thus enhancing creativity and productivity. Note that a shared repository is dispensable in this scenario, because in most situations input is not necessary and the outcome does not require further work by the group.

Ad-hoc meeting. Most meetings are ad-hoc: unscheduled, spontaneous, lacking an agenda, and with an opportunistic selection of participants [22]. This type of meeting happens anywhere and anytime. During an ad-hoc meeting, the participants tend to mix individual and collective activities. In this situation, a PDA may be used individually, but the synchronization with other PDA offers the group an overall perception of the work being carried out.

This scenario emphasizes the opportunities of PDA to overcome several restrictions imposed by the environment, e.g. the lack of a whiteboard, table, paper, etc. Furthermore, PDA may automatically obtain information about the meeting location and other PDA in the vicinity, thus preserving the meeting context.

Once the ad-hoc meeting has ended, it has produced various types of outcomes, consisting on private and public data such as agreements, to-do lists, deadlines and schedules. PDA may serve to upload this data into the shared repository, either

immediately or later on, depending on the available connectivity. This functionality is fundamental to construct a coherent view of what happened in the meeting.

Learning meeting. Learning meetings are very different from other types of meetings, as they are more focused on the use of the technology to enhance the learning experience [23]. In this scenario teachers may interactively present a lecture through the system or involve students in problem solving activities.

According to [23], the degree of anonymity supported by PDA in this scenario helps reducing evaluation apprehension by allowing group members to submit their ideas without having to speak up in front of the group; and parallel communication aids reducing domination, since more persons may express their ideas at the same time. This scenario also emphasizes the PDA support to process structures, which may help reducing coordination problems by keeping the group focused on the agenda. This scenario includes a wide spectrum of activities, including the generation and organization of ideas, group analysis, decision making, group writing and action planning.

Table 1. Meeting scenarios and associated functionality

| Meeting | Major goals | Meeting Input | Meeting process | Meeting Output |
|--------------------|--|--|---|---|
| Ritual | -Social interaction | None | Simple, conducted by the facilitator | Group formation, intangible information |
| Deliberate | -Problem solving -Decision making -Information integration | Attached documents, agenda, attendees list | Highly structured, conducted by the facilitator | Report and other formal meeting elements |
| Meetings ecosystem | Unknown problem solving | Ill defined agenda | Ill defined | Organized chaos |
| Creative/design | -Brainstorming -Brainsketching -Collaborative design | On-site gathered material (e.g. building snapshot) | Unstructured with free collaboration | New ideas and sketches |
| Ad-hoc | -Opportunistic decision-making | None | Simple | -Report -Individual notes |
| Learning | -Brainstorming and brainsketching -Collaborative writing -Problem solving -Developing social skills | Class planning, pedagogical practices and materials (e.g. reading documents) | Structured, conducted by the teacher | Pedagogical achievements: new ideas and solutions (individual or group writing) |

Table 2. Meeting scenarios and associated components

| Meeting | PDA | SDG | Repository |
|--------------------|---|---|------------------------------------|
| Ritual | -Input device -Manage SDG | Necessary to focus the participants attention | None |
| Deliberate | -Manage meeting data -Manage meeting process -Manage SDG | -Necessary to focus the participants attention -Manage meeting process | Context awareness |
| Meetings ecosystem | -Synchronize context -Publish progress -Move data across groups | Necessary for situation awareness | None |
| Creative/design | -Input device | None | None |
| Ad-hoc | -Substitute SDG -Share notes | None | Overall perception of outcomes |
| Learning | -Input device -Share notes -Brainstorming and voting | Desirable to focus the participants | Pre- and post- meeting information |

4 Information Systems Perspective

Our main goal in this section is to characterize meetings from an information systems perspective. This goal is relatively difficult because of the high degree of informality associated to several meeting scenarios. Nevertheless, the following characterization in terms of meeting memory and meeting process codifies and integrates our knowledge about electronic meetings.

Meeting memory. The meeting memory is an organized persistent storage of the information produced or manipulated in relation to meetings. We identify three fundamental meeting memory elements (Fig. 1): agenda, meeting data, and meeting report [2]. The agenda is a critical element to successfully manage meetings, since meetings tend to crystallize their actions around it [24], and is mandatory in planned meetings. The prototypical agenda has two different types of information: a list of topics or goals that the group must deal with; or a series of steps that the group should execute to accomplish their goals.

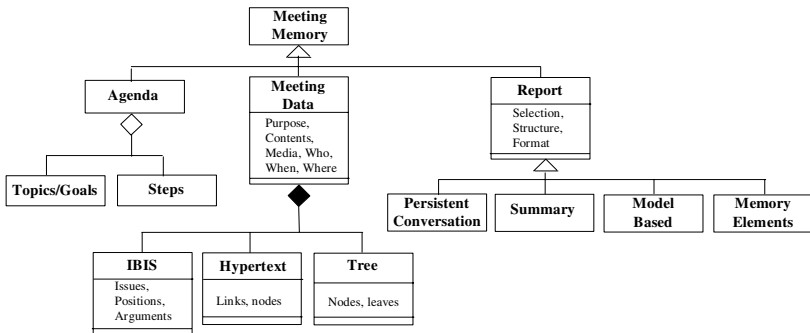


Fig. 1. Memory elements

The meeting data concerns all the data distilled during meetings. The major attributes associated to the meeting data are [25]: purpose; contents; media used; who is involved in producing the data item; when was the data item produced and where should the data item be produced or used. Several information models are adopted to structure meeting data. Two very common models are hypertext and IBIS [26]. The case studies discussed in this paper utilize trees (Nomad and LightMeet) and the hypertext model (JointTS).

The meeting report aggregates the tangible outcomes of meetings and is characterized by the particular selection, structure and format of the report items. We identified four different report types associated to electronic meetings:

- Persistent conversation – Transcripts of all the information exchanged in meetings, most often produced by automatic transcription tools.
- Summary of the outcomes – Referencing only the pieces of information considered most important, like voting results. These summaries are in general generated by humans.

- Model based information structures – When the underlying information model is applied to automatically produce meeting reports.
- A collection of group memory elements generated and structured during meetings – This includes selected elements like action plans and calendaring information, commonly produced in meetings.

Meeting Process. The meeting process structures the activities executed by the meeting participants (Fig. 2). The nature of these activities changes as the participants move forward towards their goals. Therefore, we typify activities in accordance to that progression and consider three increasing levels of detail:

- Level 1 – The meeting as a whole, i.e. one single activity;
- Level 2 – The partition of the meeting process as a sequence of activities; and the decomposition of these activities in sub-activities;
- Level 3 – The fragmentation of the meeting in an intricate collection of elementary individual interventions.

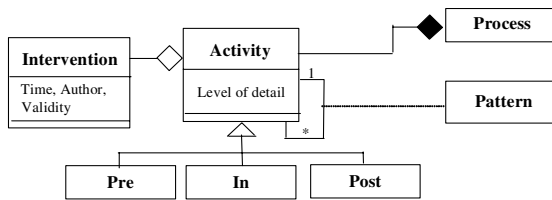


Fig. 2. Process elements

The level 1 characterizes the meeting process while maintaining the perspective of the whole. This allows typifying meetings in genres [25], e.g., briefings, progress report meetings, staff meetings and management meetings.

The second level focus on the decision structure. This follows a logical view over decision making that is recurrent in literature [27], where the goal is divided in partial goals that can be accomplished in a systematic way. Furthermore, this level also regards meeting processes as decomposable in multiple levels of detail, with goals and sub-goals. For instance, [4] propose a decomposition with several basic patterns like diverge, converge, organize, elaborate, abstract and evaluate.

Finally, in level 3 the meeting process is characterized according to the flows of individual interventions produced by the participants. The generic attributes associated to these interventions are:

- Time – The moment when the intervention is produced. Based on this attribute, we can characterize meetings as synchronous or asynchronous.
- Author – The person that produces an intervention may be identified or not. This factor can have an important role in the process results. Several researchers have reported the positive effects of anonymity in the interaction process (e.g. [28]).
- Validity –The validity corresponds to the time during which the intervention can be accessible. The validity has repercussions on the meeting memory.

Assuming a complementary view of the meeting process, the process activities can also be classified according to what is designated as the meeting lifecycle. A detailed analysis of meetings allows verifying that the meeting lifecycle consists of three stages: (1) the pre-meeting stage, considering activities that have to be executed before the meeting; (2) the in-meeting stage, considering activities accomplished during the meeting; and (3) the post-meeting stage, considering activities that may be required afterwards.

5 Architectural Perspective

We now describe the generic architecture enabling the features and topologies suggested by the meeting scenarios and information system components described above. In this architecture, illustrated in Fig. 3, any device (including PDA, repository and SDG) can establish network connections with other devices using either automatic detection or direct IP connection. Users interact with the PDA and SDG through pen-based gestures, mouse and keyboard. Once an individual interaction requires meeting management, the action itself is encoded in XML and delivered to all other devices, including the shared repository if persistent storage is necessary.

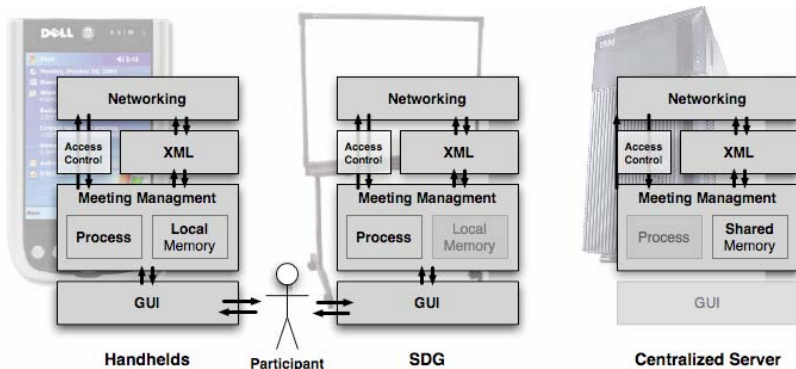


Fig. 3. System framework configured for PDAs, SDG or a Centralized Server

Of course, this generic architecture may be implemented in slight different ways. For instance, one of the applications described below uses a message server directly connected to the SDG and communicating with every PDA through point-to-point channels. When a user interacts with the PDA, e.g. to write an idea, the PDA sends an encoded XML message to the server, which sends the message to the other PDA and SDG; and applies locally, on the SDG, the corresponding interaction. When the users interact directly with the SDG, XML messages are distributed from the server to the PDA. We implemented other instances of this framework in Java and .NET platforms and tested them in several meeting scenarios, as described in the next section.

6 Applications

6.1 Nomad

Nomad is based on the proposed system architecture to support creative/design, had-hoc and learning meetings using PDA. This emphasizes the ability to generate and share meeting notes using sketches and writing. Fig. 4 shows a screenshot of Nomad during a typical design meeting.

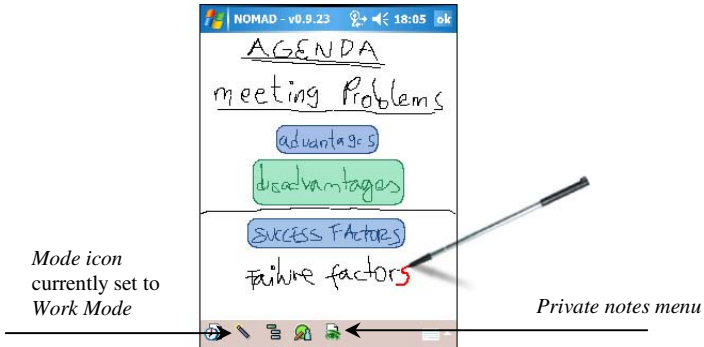


Fig. 4. Nomad screenshot

Annotations and sketches are produced from freehand inputs with the PDA stylus. Nomad recognizes special gestures, which trigger certain information management functionalities. This is done after the user raises the stylus tip, when the system tries to match the stylus movement with a list of pre-defined gestures. Two examples are the select and cut functions illustrated in Fig. 5. The select function utilizes either the single click gesture or a more complex gesture designated double lasso, which consists in double surrounding an area with a closed shape. The cut function utilizes a cross gesture to remove strokes from the screen.

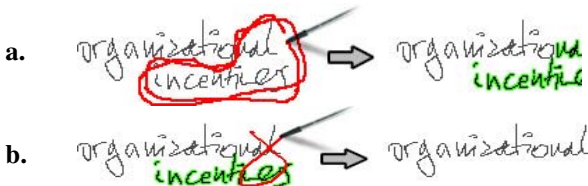


Fig. 5. Double lasso gesture for complex selecting and cross gestures for cutting

Since the PDA workspace is very restricted, Nomad organizes meeting information in a tree structure, using hierarchical pages. The pages are associated to parent nodes, usually representing a label for the page. To create a page, the user must write or draw a title and then surround the title with a partial rectangle, as shown in Fig. 6. The system recognizes this gesture as a node creation and associates a new page to the selected node.



Fig. 6. Surrounding title to create node

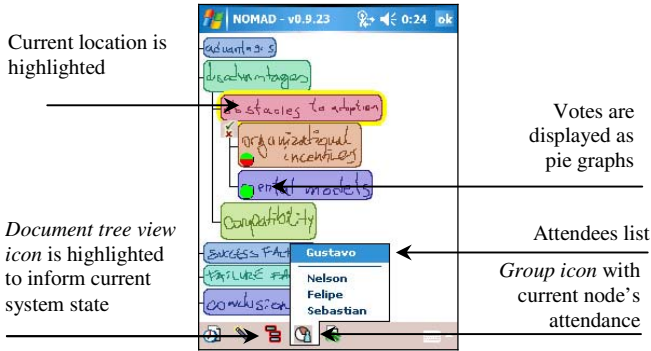


Fig. 7. Overview window

As the information tree may become very large, Nomad includes an overview window allowing the user to move around and zoom in and out using predefined gestures (Fig. 7). Clicking on the overview leads the users to the corresponding nodes.

The overview also gives awareness on who is participating in the meeting or working on a node (see Fig. 7). Another functionality allows users to vote for or against nodes. This functionality requires a designated person, the meeting facilitator, to request the participants to vote on one or several designated nodes. Then, the other participants vote using predefined gestures, as shown in Fig. 8. The results are represented as pie charts, where the green portion represents positive votes, red represents negative votes and black represents users who have not voted.

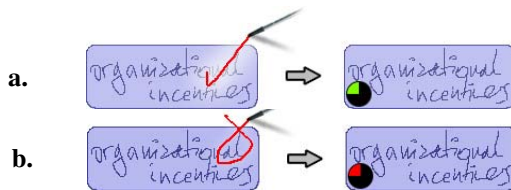


Fig. 8. Voting gestures (note also the voting results)

The Nomad system operates in the synchronous mode, relying on peer-to-peer detection and communication to share meeting information. The system may support a shared repository and SDG, although no dedicated components have been developed yet. The whole information tree is represented in XML, and may be stored and retrieved in that format. Furthermore, Nomad allows users to work individually, generating local XML files and, when necessary or convenient, import local files into a meeting. The system provides two special import options: one where the individual

and meeting trees are merged, another where the individual tree becomes a new version of the meeting tree.

Nomad can also be used for supporting collaborative learning activities, especially for collaborative problem solving and discussions. Students may exchange their ideas and solutions with the help of sketches. The teacher may also benefit from on-line assessment of the students' work, making this a motivating scenario.

6.2 LightMeet

LightMeet is a meeting support system implementing one variation of the architecture described in this paper, having the SDG, centralized repository and PDA components; and using direct IP connections to exchange XML messages. LightMeet supports face-to-face meetings centered on the SDG, allowing participants to manage private meeting information using PDA. Because of this focus on the SDG, LightMeet is mostly adequate to ritual and deliberate meetings, and some specific instances of the learning scenario where the presence of the SDG may be beneficial.

LightMeet adopts a radical approach to the information system: all meeting data is organized in a tree. This not only includes the typical meeting data, such as proposed solutions, comments and priorities; but also the agenda and report, which are treated by the system as "special" nodes. Our fundamental reasons for adopting this radical approach was that we aimed to develop a very simple mental model of electronic meetings, so that any participant not familiarized with them could nevertheless participate and interact with the system.

As shown in Fig. 9, the SDG allows the meeting participants to create and manage the meeting information using the "drag & drop" and "explorer" metaphors that are now very common in many technological devices. Observe that the participants may have tabs which display specific portions of the tree, such as, for instance, the agenda. The admin tab is dedicated to manage the connections to PDA.

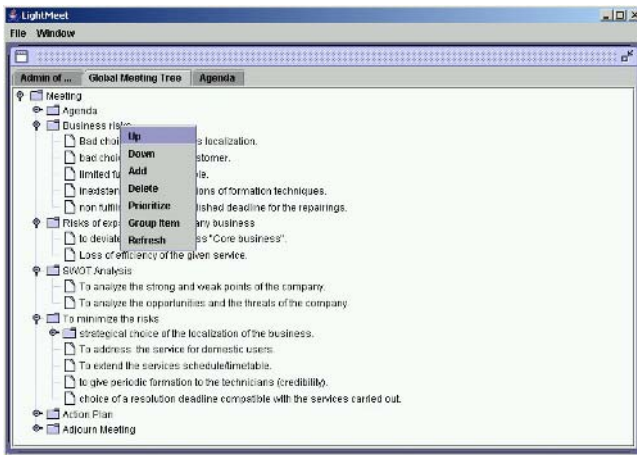


Fig. 9. LightMeet SDG component

Each PDA connected to the SDG operates as a dynamic cursor over the tree available in the SDG, i.e. one single node and its immediate sub-nodes are displayed in the PDA (Fig. 10). When the user moves around the tree, the PDA sends a message to the SDG requesting the corresponding node and sub-nodes. The participants may freely create, delete, move and modify the nodes. Since the agenda and report are special nodes (e.g. they cannot be deleted), the PDA users have shortcuts to easily move there. The PDA also have private spaces where users can privately edit nodes. There is also support for publishing these private nodes in the SDG.

The SDG adopts an optimistic approach to concurrency control, allowing the participants to freely manipulate the nodes, and relying on the face-to-face interactions to resolve any occurring problems. As we observed in our experiments, the information management is sometimes chaotic but the meeting participants can easily define a social protocol. Sometimes, when there is a definite need to control the information management, the interaction is restricted to the SDG.

The LightMeet system was evaluated in a laboratory experiment with two groups of users with different levels of proficiency with computers. One group had five participants with low computer skills, including five persons with degrees in different fields, such as economics, management and pre-kinder. The other group had six participants highly proficient with computers, mostly with degrees in informatics and mathematics.

The experiments were conducted with short briefings about the meeting technology, followed by face-to-face electronic meetings, and concluded with a questionnaire with 25 questions based on SUMI (Software Usability Measurement Inventory). The meetings involved the discussion of the risks of underpinning a home-based business, using pre-defined agendas resembling SWAT analysis.

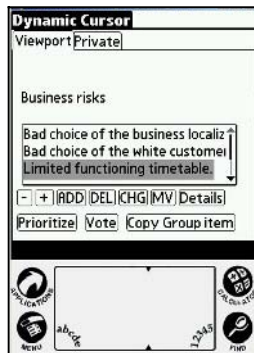


Fig. 10. LightMeet PDA component

The obtained results indicate that the affective criteria (user friendliness and emotional reaction) were the most positively evaluated, followed by the ease of learning, control, and efficiency. The most negative criteria concerned the system utility. More details about the system and evaluation results can be found in [29].

6.3 JoinTS

The major goal of the JoinTS (Joint psychological Therapy Support [30]) system is exploring PDA support to psychotherapy. The psychotherapy processes occur in several scenarios, two of them currently covered by JoinTS: individual and group meetings.

Individual Meetings. Individual psychotherapy meetings are complex and demanding. They require numerous activities performed by a pair of participants, constituted by a therapist and a patient. The therapy meetings are fundamentally centered on face-to-face interactions, held in the therapist's office, but are also inherently related to individual activities performed in between meetings [31].

On the therapist's side, we account for the preparation and follow-up of face-to-face meetings, while on the patient's side we include multiple activities prescribed by the therapist, such as responding to questionnaires, planning daily life and registering thoughts throughout the day. JoinTS supports all this information management, including the customization of questionnaires and forms, form filling, note taking during face-to-face meetings, registering thoughts, visualizing and analyzing the patient's accomplishments (Fig. 11).

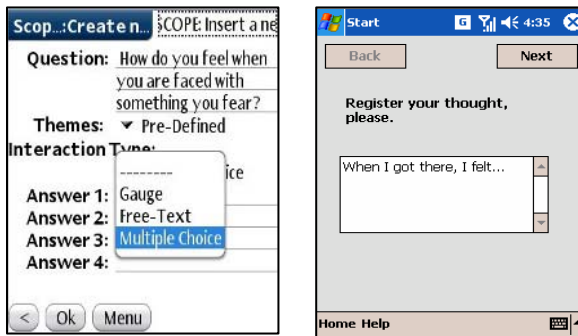


Fig. 11. Screenshots of two JoinTS tools developed for different PDA. **Left:** The therapist's questionnaire customization tool developed for Palm. **Right:** The therapy tool developed for Pocket PC.

The initial therapy steps may be classified as ritual meetings, where the therapist discusses and sets up the stage for addressing the patient's problems. Most of these activities are conversation-based and collaborative. After the main problems are defined, the therapist adopts a scenario more similar to deliberate meetings, since therapy meetings become more carefully planned and follow a strict agenda. Homework is given to the patient and the results are analyzed and discussed in subsequent meetings.

The role of PDA supporting this process occurs in multiple ways: (1) supporting the therapist's note taking during meetings, without obstructing the face-to-face interactions; (2) supporting the patient's individual tasks in between sessions; (3) supporting the exchange of questionnaires between the therapist and patient; and (4) functioning as a SDG, focusing the patient's attention to specific issues presented by

the therapist during sessions. We specifically emphasize the important role of PDA assisting the patient's individual tasks in between sessions, presenting hints and suggestions whenever an abnormal behavior is detected, according to rules previously specified by the therapist.

Group sessions. Individual therapy is frequently complemented with group therapy. Here, meetings involve several patients sharing the same pathology. Although the main activities are similar to those described for individual therapy, goals and procedures diverge for each scenario. Generally, questionnaires are filled cooperatively and thought registration is many times subject to consensus. Overall, every activity requires the intervention and participation of all patients, always guided by the therapist. On occasions, therapists work in pairs to accommodate the various parallel tasks that have to be accomplished during meetings. Consequently, two groups emerge. The first one is composed by the therapists, who exchange specific information between them, whilst the second group is composed by patients.

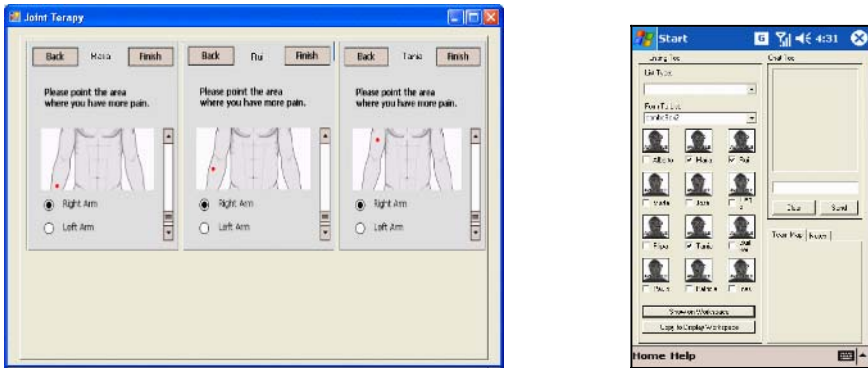


Fig. 12. **Left:** Screenshot of the JoinTS SDG, showing the selected patient's PDA. **Right:** The list of patients is selected on the therapists' PDA. You may also observe the chat space allowing therapists to share notes.

As with the individual therapy, initial meetings focus on the adaptation to the procedures and group. These ritual meetings are then followed by deliberate meetings, with more clear objectives and schedules. Therapists control the topics and subjects addressed during these meetings, communicating simultaneously with several patients. This scenario requires a SDG to focus all the participants on the objectives and facilitate collaboration. PDA support the various underlying activities, including control of the SDG (see Fig. 12). Also, in this scenario, the JoinTS system uses a shared repository to preserve the therapists' annotations and support reviewing patient records and questionnaires. The therapists, using their PDA, may retrieve relevant information from the repository and publish it in the SDG.

Another possibility that has been explored is supporting private communication between the two therapists using PDA (see Fig. 12). This subgroup is particularly

important when critical issues emerge. In many cases, resolving these critical issues becomes a major goal, turning the therapists particularly active. Then, a meeting ecosystem scenario emerges. The communication between therapists is considered uncomfortable for patients in normal meetings but, using PDA, these private conversations are less conspicuous.

Considering the JoinTS architecture, the system utilizes wireless communication and relies on a centralized server to synchronize and control the information flows between the participants, shared repository and SDG. There is one communication channel available for sending information from PDA to the central repository and then either to the SDG or the therapists' PDA. There is a second channel available for receiving information directly from the therapists' devices, using multicast.

The shared repository preserves all the information exchanged by the system as well as descriptions of the meeting activities. Each intervention is stored with its validity, author and time. A log for each participant is generated, as well as a log for the whole meeting. Therapists may trace the patients and groups' evolution from meeting to meeting. All this information is preserved and exchanged in XML format.

7 Conclusions

In this paper we provide an integrated perspective over electronic meeting systems and how users may utilize PDA in meeting environments, highlighting the physical and information architectures. Regarding the physical system, we identify three major system components: PDA, SDG and shared repository. Considering the information system architecture, we identify several information components related with the meeting memory and the meeting process.

The fundamental implications drawn from this research result from the opportunity to make electronic meeting systems interoperable to a level that has not been achieved before, allowing different devices (in our case, PDA, SDG, repository) to exchange, share and manipulate meeting-related information in an integrated way. Such level of interoperability is possible because of two fundamental reasons: (1) we support information exchange using the XML standard, thus allowing very different devices to plug in the meeting system; and (2) we standardized the meeting information structure around common memory and process components, such as agenda, report, and other meeting data structures, like trees.

This research emerged from three independent research groups, working in different fields and developing their own prototypes, but nevertheless sharing a common understanding about the fundamental nature of meetings, the important roles that PDA may assume in meetings and the requirements to make interoperable meeting systems. We strongly believe the proposed architecture benefits from such varied and complementary perspectives, as well as parallel development efforts.

Acknowledgements. This paper was partially supported by Fondecyt 1050601, DI-Universidad de Chile Nro. I2 04/01-2, and the Portuguese Foundation for Science and technology, Project POSI/EIA/62473/2004.

References

- [1] Muñoz, M., Rodriguez, M., Favela, J., Martinez-Garcia, A., González, V.: Context-Aware Mobile Communication in Hospitals. *Computer* **36** (2003) 38-46
- [2] Costa, C., Antunes, P., Dias, J.: Ems/PDA: Connecting Meetings with People in Organisations. Proceedings of the 24th Information Systems Research Seminar in Scandinavia, IRIS 24, Ulvik in Hardanger, Norway (2001)
- [3] Antunes, P., Ho, T.: The Design of a GDSS Meeting Preparation Tool. *Group Decision and Negotiation* **10** (2001) 5-25
- [4] Briggs, R., Vreede, G., Nunamaker, J.: Collaboration Engineering with Thinklets to Pursue Sustained Success with Group Support Systems. *Journal of Management Information Systems* **19** (2003) 31-64
- [5] Jessup, L., Valacich, J. (eds.): *Group Support Systems*. Macmillan Publishing Company, New York (1993)
- [6] Davis, R., Lin, J., Brotherton, J., Landay, J., Price, M., Schilit, B.: *A Framework for Sharing Handwritten Notes*. ACM Press, San Francisco, California (1998)
- [7] Myers, B., Stiel, H., Gargiulo, R.: Collaboration Using Multiple Pdas Connected to a Pc. Proceedings of the ACM Conference on Computer Supported Cooperative Work. ACM Press, Seattle, WA (1998) 285-294
- [8] Baldonado, M., Cousins, S., Gwizdka, J., Paepcke, A.: Notable: At the Intersection of Annotations and Handheld Technology. *Lecture Notes in Computer Science*, Vol. 1927. Springer-Verlag, Heidelberg (2000) 100-113
- [9] Wiberg, M.: Roamware: An Integrated Architecture for Seamless Interaction in between Mobile Meetings. Proceedings of the 2001 International ACM SIGGROUP Conference on Supporting Group Work. ACM Press, Boulder, Colorado (2001) 288-297
- [10] Greenberg, S., Boyle, M., Laberge, J.: Pdas and Shared Public Displays: Making Personal Information Public, and Public Information Personal. *Personal Technologies* (1999)
- [11] Stewart, J., Bederson, B., Druin, A.: Single Display Groupware: A Model for Co-Present Collaboration. Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit. ACM Press, Pittsburgh, Pennsylvania (1999) 286-293
- [12] Davis, R., Landay, J., Chen, V., Huang, J., Lee, R., Li, F., Lin, J., Morrey III, C., Schleimer, B., Price, M., Schilit, B.: Notepals: Lightweight Note Sharing by the Group, for the Group. Proceedings of the CHI 99 Conference on Human Factors in Computing Systems. ACM Press, Pittsburg (1999) 338-345
- [13] The 3M Meeting Management Team: *Mastering Meetings*. McGraw-Hill, Inc., New York (1994)
- [14] Webne-Behrman, H.: *The Practice of Facilitation*. Quorum Books, Westport, Connecticut (1998)
- [15] Myers, B.: The Pebbles Project: Using Pcs and Hand-Held Computers Together; Demonstration Extended Abstract. Adjunct Proceedings CHI'2000: Human Factors in Computing Systems. ACM Press, The Hague, The Netherlands (2000) 14-15
- [16] Rosenhead, J. (ed.): *Rational Analysis for a Problematic World*. Jonh Wiley & Sons, Chichester, England (1989)
- [17] Mark, G.: Extreme Collaboration. *Communications of the ACM* **45** (2002) 89-93
- [18] Osborn, A.: *Applied Imagination*. Charles Scribner's Sons, New York (1963)
- [19] May, A., Mitchell, V.: Opportunities and Challenges for Location Aware Computing in the Construction Industry. Proceedings of Mobile HCI (2005) 255-258

- [20] Van der Lugt, R.: Functions of Sketching in Design Idea Generation Meetings. Proceedings of the 4th conference on creativity and cognition. ACM Press, Loughborough, UK (2002) 72-79
- [21] Forbus, K., Ferguson, R., Usher, J.: Towards a Computational Model of Sketching. IUI '01: Proceedings of the 6th international conference on Intelligent user interfaces, Santa Fe, New Mexico (2001) 77-83
- [22] Romano, N., Nunamaker, J.: Meeting Analysis: Findings from Research and Practice. Proceedings of the 34th Hawaii International Conference on Systems Science, Hawaii (2001)
- [23] Tyran, G., Sherpherd, M.: Collaborative Technology in the Classroom: A Review of the GSS Research and a Research Framework. *Information Technology and Management* **2** (2001) 395-418
- [24] Niederman, F., Volkema, R.: Influence of Agenda Creation and Use on Meeting Activities and Outcomes: Report on Initial Results. Proceedings of the 1996 Conference on ACM SIGCPR/SIGMIS Conference, Denver, Colorado (1996) 192-205
- [25] Antunes, P., Costa, C., Pino, J.: The Use of Genre Analysis in the Design of Electronic Meeting Systems. *Information Research* **11** (2006)
- [26] Conklin, J., Begeman, M.: Gibis: A Hypertext Tool for Exploratory Policy Discussion. *ACM Transactions on Office Information Systems* **6** (1988) 303-331
- [27] Ho, T., Antunes, P.: Developing a Tool to Assist Electronic Facilitation of Decision-Making Groups. Fifth International Workshop on Groupware, CRIWG '99. IEEE CS Press, Cancun, Mexico (1999) 243-252
- [28] Connolly, T., Jessup, L., Valacich, J.: Effects of Anonymity and Evaluative Tone on Idea Generation in Computer-Mediated Groups. *Management Science* **36** (1990) 689-703
- [29] Pereira, L.: Electronic Meetings with PDA. Department of Informatics. Faculty of Sciences of the University of Lisboa (2006)
- [30] Carriço, L., Sá, M.: Hand-Held Psychotherapy Artifacts. Proceedings of the 11th Human Computer Interaction International Conference, Las Vegas, USA (2006)
- [31] Carriço, L., Sá, M., Antunes, P.: Mobile Devices for Active Psychotherapy. Proceedings of the 8th Int. Conference on Enterprise Information Systems, Paphos, Cyprus (2006)

Supporting the Management of Multiple Activities in Mobile Collaborative Working Environments

Jesus Camacho¹, Jesus Favela¹, and Victor M. Gonzalez²

¹Departamento de Ciencias de la Computacion, CICESE, Ensenada, Mexico
{camacho, favela}@cicese.mx

²School of Informatics, University of Manchester, United Kingdom
vmgonz@manchester.ac.uk

Abstract. Many modern working environments are characterized by the need to manage multiple activities simultaneously. This is the case for instance of hospital work, which also demands a high degree of mobility and collaboration among specialists. These working conditions have motivated us to design and implement mobileSJ, a mobile information management tool based on the concept of working spheres. The tool allows users to gather information related to a working sphere, including documents, contacts and pending tasks. The tool assists users when switching between tasks, facilitates the sharing of activity related information with colleagues, as well as the synchronization of information among multiple devices, including handheld computers and public displays.

1 Introduction

Modern work environments require professionals to constantly switch among different activities. Within the context of office work, previous studies have shown that the engagement in each activity can be rather brief, averaging just a few minutes [3, 4]. Commonly the transition between activities is not simple because it requires important context switching not just of mental states but also switching at the level of retrieving physical or digital representations of resources. In order to preserve the status of an activity and facilitate context retrieval, people often organize their workspaces or seed “marks” on it [6, 7, 11]. This process can be complex when multiple activities are managed simultaneously, and when multiple resources are associated to each activity.

Performing an activity usually implies the use of a diversity of information resources such as documents, notes, agendas, calendars, or diagrams. The need to remember the location and gather all the information resources related to an activity is likely to involve certain effort and sometimes results on cognitive overload for the user [12]. Furthermore, as some researchers have pointed out, current computer operating systems make the invocation of such resources in the digital realm (bring them to the focus of attention) a problematic task [5, 12].

Although handling resources and managing multiple activities can be problematic for many types of information work contexts, a particularly challenging context is the one experienced by medical workers. Hospitals are dynamic and intensive work environments where people have multiple activities and responsibilities, and cope with

frequent contingencies that require them to constantly adjust and readjust their actions [1]. In addition, hospital workers are highly mobile and experience a high degree of collaboration and coordination with colleagues. The activities of most hospital workers clearly are not tied to a desktop or a specific location because they need to move to locate colleagues, take care of patients, and access information and other resources distributed in space [1, 10]. This phenomenon identified as *local mobility* [2] requires the user to change his “work place” constantly and even suddenly. Besides, the specialized nature of medical work makes the treatment and care of patients an inherently collaborative effort among specialized medical workers who have to be in constant communication with each other to be able to perform their activities.

In this paper we describe the design of an application to support mobile workers in managing their multiple activities and collaborations. The rest of this paper is organized in the following way. In section 2 we briefly explain our approach to the concept of working sphere and its instantiation in the “Sphere Juggler” application, which served as the basis for the tool mobileSJ presented here. In section 3 we discuss mobile worker’s need to manage multiple activities by focusing on the work of a particular kind of medical workers: medical interns. In section 4 we describe the support application mobileSJ, its functionality and architecture. Finally in section 5 we present conclusions and directions for future work.

2 Working Spheres

The concept of “working spheres” [4] was introduced as a proposal to conceive the way in which people organize and execute their work activities. A working sphere has been defined as “a set of interrelated tasks, which share a common motive, involve the interaction with a particular constellation of people, use ensembles of resources and have their own individual time framework” [4]. As a concept, a working sphere refers to a particular way to abstract human work from the perspective of those executing it, and more important, as a way to represent those efforts that transcend mere actions (e.g. a phone call). From a system perspective, a working sphere could be supported by implementing a repository where the resources and the applications concerning to each sphere can be stored and easily recovered whenever necessary.

The application “Sphere Juggler” [9] implements the concept of working sphere and creates its computational representation: an E-sphere. This application allows the user to manage their multiple activities and their information and contextual resources in a centralized way. The user can define E-spheres for each of his activities and associate to them information resources, contacts relevant to the activity, emails related to the activity and pending issues to help the prospective memory. When a user switches between E-spheres, each E-sphere is enabled to quickly gather and retrieve its own workspace state (windows positions, status and overlay order) and context information like opened documents, idle time, etc. in a silent manner. Thus, the application hides the resources and state of the previous sphere and shows the ones related to the recently restored activity. This helps the user with the necessary context switching when changing activities.

3 Characterizing Mobile Workers with Multiple Activities: The Case of Medical Interns

To illustrate the need to support the management of multiple activities in a mobile and intensively collaborative working environment such as a hospital, we consider the work of a medical intern. We base our analysis on the results of a workplace study conducted in the internal medicine area of a mid-size teaching hospital [8].

The daily routine and typical duties of an intern can be summarized as follows:

Medical interns meet at 7 am with the physician in charge of the area at the internal medicine office, where they briefly discuss the night's events described by the intern who worked the night shift. After the discussion, the interns gather information related to the patients assigned to them and place it in each of the patients' bedrooms. They walk down to the laboratory to gather laboratory results of the patients, and attach them to the medical record. Later, the interns meet at the internal medicine office and for one or two hours they listen to a colleague's assessment of a particular medical issue or interesting clinical case. After that, they go to the bed wards where along with the physician they conduct the ward round. During the round they discuss each patient's clinical case consulting the patient's medical record and laboratory tests. Finally, once the medical interns finish the round, the rest of the shift is spent mostly doing paperwork in the internal medicine office.

Medical interns are considered physicians-in-training; they provide the most hours of patient care in the unit and are constantly moving throughout the hospital. Interns are responsible for the care of five or six patients. One of their main responsibilities is to create clinical histories whenever a new patient arrives in the hospital. They are also responsible for providing care and follow-up on patients during their stay in the hospital. Other tasks for which medical interns are responsible have a more collaborative nature, for instance, they participate in ward rounds with attending physicians and in meetings where clinical cases are discussed. Finally, since they are still students they have to prepare presentations and reports for their courses.

We describe in more detail how the demands of medical intern's work shape the requirements for a system supporting the management of their working spheres.

3.1 Management of Multiple Activities and Resources

Given that medical interns have to cope with multiple activities, which are often fragmented by interruptions, and which require them to gather and consult a great amount of information resources, the application must provide users with mechanisms to easily manage their activities and their associated resources. Most of the medical intern's activities are centered on their patients and their courses, which can be considered their main working spheres. These working spheres are not just their own, they are often shared, as is the case of a patient whose care is the responsibility of the intern, but also of the attending physician and responsible nurse. In addition, the responsibility is transferred to other interns as they change shifts. When the intern returns the next day, she needs to get updated on the current state and main events related to the patients for which she is responsible.

3.2 User Mobility

Although medical interns use computers, a system based solely on a desktop application will not provide interns with adequate support since they are constantly moving around the hospital. One of our previous studies revealed that medical interns spend at least 40% of their work shift away from a base location, such as an office or medical station [8]. Thus, the application should be designed to be used in both desktop computers and mobile-devices such as PDAs or smartphones. The desktop-based version and the mobile version have to be independent but must have the ability to synchronize their contents. Thus, the application should provide mechanisms for the synchronization of activities and resources, taking into account the differences in the capacities of the devices, both in terms of memory as well as screen size.

3.3 High Degree of Collaboration and Communication

Because of its complex nature, hospital work demands close coordination and collaboration among different specialists. In order to support these collaborative interactions, the application must include ways for the users to share activities and information resources among heterogeneous devices, such as between two PDAs when two colleagues encounter each other in a hallway. The application should allow as well the visualization of spheres and resources in semi-public displays to facilitate the collaboration among colleagues. Besides, the application must provide users with simple mechanisms to communicate with colleagues while the user is on the move.

4 The mobileSJ Application

mobileSJ was designed and implemented to assist mobile users in the management of their multiple activities and collaborations. The application has its origin in a previous design called SphereJuggler that runs on desktop computers and does not support shared activities and collaborations [9]. In contrast, mobileSJ runs on PDAs and SmartPhones and includes a set of capabilities not present in Sphere Juggler, like sharing of activities and resources, as well as ways to communicate with colleagues while in the move through either SMS messages or phone calls. This application is completely independent but at the same time offers the ability to communicate and synchronize with the desktop SphereJuggler.

We illustrate the functionality offered by mobileSJ by considering a typical work shift of a medical intern, as described in section 3.

4.1 Support for the Management of Activities

The medical intern has several patients assigned to him, as well as a few school projects. He uses SphereJuggler in his PC and mobileSJ in his PDA. Overtime, he has created an e-sphere for each of his activities and associated information resources, contacts information and pending issues to them. He might also search specialized medical digital libraries for information relevant to his activities and associate this new information (URLs, PDF documents, etc.) to the corresponding e-spheres.

When the intern has to go to the hospital, he connects his PDA to the PC, which launches a synchronization application. The application automatically transfers the new resources created in the PDA to the PC. The interface of the synchronization application shown on Figure 1a, allows the user to select those files that he wants copied from the PC to the PDA or deleted from it. This is done in consideration of the limited storage capacity of the PDA. To assist the user in deciding which files to transfer, the size of each file is indicated, as well as the remaining space in the PDA. Alternatively the user can synchronize his spheres over the internet through a Web service application called SincroServer thus making his spheres and resources available wherever he has Internet access (his workplace or school). When the intern arrives at the hospital, he no longer needs to locate and gather documents related to each of his patients. He can access those documents directly from the e-spheres in mobileSJ.

4.2 Support for Mobility

During the ward round, when the group arrives with a patient assigned to the medical intern, the intern uses mobileSJ to select the sphere that represents that patient, and the application displays the information resources, contacts and reminders of pending issues associated to the patient (Fig. 1b). If the intern needs to consult any of these resources, he just selects it and the resource is opened in the appropriate application (i.e. web browser, PDF reader, etc.). This is done without the need for the user to specify the resource location or the application required to open it. The application shows all the information resources contained in each sphere either if they are physically in the PDA or not (indicating the latter with a special icon). When the user needs a resource that is not in the PDA he can download it easily from the SincroServer application. Furthermore, through SincroServer the user can synchronize any specific sphere he needs (or all of them) with the server while he is moving around the hospital, selecting the information resources he wants to download (through a wireless internet connection). During the ward round, the intern might create additional resources related to his activities (personal or medical notes, photos, link to a web page, etc.), and automatically associate them to the current activity (Fig. 1c).

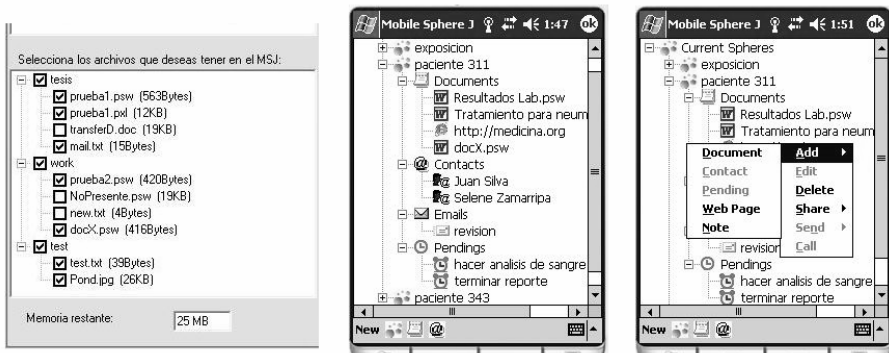


Fig 1. (a) Synchronization app. (b) E-spheres and their resources. (c) Adding new resources.

When the medical intern finishes the round, he moves to the physicians' office to write the medical notes. He connects his PDA to the PC and the synchronization application merges his e-spheres and transfers the new resources he has created or changed to the desktop computer. Once in the desktop computer, he can comfortably edit the notes he took in the PDA to complete the medical note, consulting information resources related to the patient from Sphere Juggler when required.

4.3 Support for Collaboration

As he moves around the hospital to perform his work, the medical intern needs to consult with the attending physician, nurses and other interns. To discuss a clinical case, he can use mobileSJ either by sharing a working sphere with a colleague or by using the application to navigate the resources related to a working sphere on a public display. To work with a colleague on a shared sphere, the intern selects the e-sphere, and then the resources that the user wants to transfer to either a public display, or the PDA of the co-worker (Fig. 2a). This is done to ensure that the application won't transfer resources considered private by the user. When the public display receives the sphere and its resources, it automatically displays its information resources. In this way, the colleagues can comfortably review the case in the public display. When they finish the interaction, the medical intern can transfer the updated sphere to his colleague's PDA, making it immediately available to him from his own mobileSJ client.

In addition to these collaborative services, mobileSJ allows users to send email or SMS messages, and when working in a smart phone, initiate a telephone call. Contacts can be associated to e-spheres as any other resource. When working on that e-sphere, the user can select the contact he wants to communicate with and the system shows him the various mechanisms available to establish the communication (Fig. 2b). This is done directly from the mobileSJ interface without the need to introduce a telephone number or open external applications. The user just needs to select the contact and the communication option and the system does the rest.

4.4 Implementation of mobileSJ

mobileSJ runs on PDAs and smartphones with the WindowsCE OS. The application is implemented in five modules on the PDA (Figure 3). The mSJ module interacts with the modules in the desktop computer (sincroSJ) and the Web Server (SincroServer). The sincroSJ module is used to synchronize spheres between the desktop computer and a mobile device through an ActiveSync connection. SincroServer can be used to synchronize the spheres of a device with the spheres of the server through the Internet. In the PDA, the mSJ module is the interface through which the user interacts with the system, allowing him to create and manipulate e-spheres and their associated resources. An additional module allows users to transfer spheres and resources between different devices like PCs, PDAs and public displays. This module detects all registered devices in the proximity and presents them as options for the user to decide to which device he wants to transfer the sphere/resource. The Communication module is responsible for handling communication with contacts related to their activities.

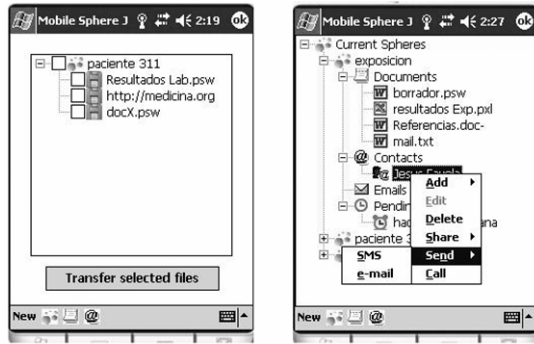


Fig. 2. (a) User selection of resources to be shared or transferred. (b) Communication options.

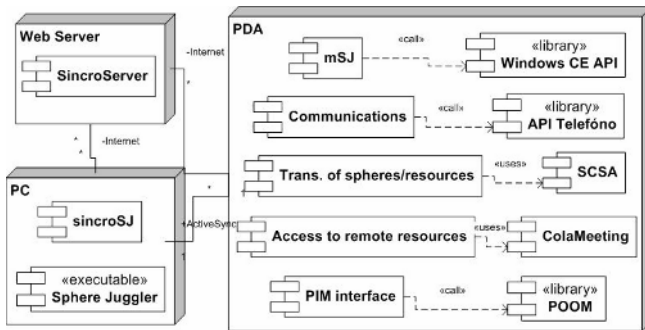


Fig. 3. mobileSJ architecture. Seven modules deployed in a PDA, a PC and a Web Server

This includes the ability to send emails, SMS messages and phone calls. This module interacts directly with the phone API of the device. A fourth module allows the user to visualize files stored in the desktop computer. This is convenient for large files that the user doesn't want stored in the PDA, or those for which there is no corresponding application in the PDA. The user can remotely manipulate the application from the PDA. Finally, the application includes a module that handles associations between contacts and pending issues in the PIM (personal information management) tool of the PDA and mobileSJ, allowing the user to seamlessly and consistently work with both applications. The user can add contacts to a sphere that were previously defined in the PIM tool, or add new elements to the PIM tool directly from the mobileSJ interface and at the same time associate them to a sphere.

5 Conclusions

Working environments, such as hospitals, are characterized by the need to manage multiple activities simultaneously, constant local mobility, frequent interruptions, and intense collaboration and communication. These conditions impose important demands on users that need to frequently switch between tasks, contributing to a decrease

in efficiency and becoming a source of errors and mishaps. We have designed an implemented a mobile task management tool called mobileSJ, which allows users to associate digital resources to a working sphere, navigate thru them, share spheres and resources with colleagues, and communicate with them trough various means. As the next step of this work, we plan to evaluate the usability of mobileSJ by replicating in a laboratory conditions similar to those experienced by potential users. Once evaluated, we plan to conduct a short-term trail study where medical interns will use the mobileSJ to support their work.

References

1. Bardram, J.E. and C. Bossen. Moving to get aHead: Local Mobility and Collaborative Work. In Proc. of *ACM ECSCW'03*. Helsinki, Finland (2003), pp. 355-374.
2. Bellotti, V. and S. Bly. Walking Away from the Desktop Computer: Distributed Collaboration and Mobility in a Product Design Team. In Proc. of *ACM CSCW'96*. Boston, MA (1996), pp. 209-218.
3. Czerwinski, M., E. Horvitz and S. Wilhite. A Diary Study of Task Switching and Interruptions. *CHI 2004*, ACM Press, Vienna, Austria (2004), pp. 175-182.
4. González, V. and G. Mark. Constant, Constant, Multi-tasking Craziness: Managing Multiple Working Spheres. *CHI 2004*, ACM Press, Vienna, Austria (2004), pp. 113-120.
5. Kaptelinin, V. UMEA: Translating Interaction Histories into Project Context. *CHI 2003*, (2003). Ft. Lauderdale, Florida, pp. 353-360.
6. Kirsh David. The Context of Work. *Human Computer Interaction* (2001) 16: pp. 305-322
7. Malone, T. How do People Organize Their Desks? Implications for the Design of Office Information Systems. *ACM Trans. on Office Information Systems* (1983) 1(1): pp. 99-112
8. Moran, E., Tentori, M., V. Gonzalez, Favela, J., Martinez-Garcia, A. Mobility in Hospital Work: Towards a Pervasive Computing Environment. Accepted for publication in *Intl. Journal of Electronic Healthcare* (2006)
9. Morteo, R., Gonzalez, V., Favela, J. and Mark, G. Sphere Juggler: fast context retrieval in support of working spheres. *ENC 2004*, IEEE Press (2004), pp. 361-367
10. Munoz, M., M. Rodriguez, J. Favela, V. Gonzalez, and A. Martinez-Garcia. Context-aware mobile communication in hospitals. *IEEE Computer* (2003) 36(8), pp. 60-67
11. Rouncefield, M., J. A. Hughes, T. Rodden and S. Viller. Working with Constant Interruption: CSCW and the Small Office. *ACM CSCW 94*, Chapel Hill, N.C. (1994), pp. 275-286
12. Volda, S., Mynatt, E. D., MacIntyre, B., y Corso, G. M. Integrating virtual and physical context to support knowledge workers. *IEEE Pervasive Computing*, (2002) 1(3): pp.73-79

Seamless Interaction Among Heterogeneous Devices in Support for Co-located Collaboration

Antoine Markarian, Jesus Favela, Monica Tentori, and Luis A. Castro

Department of Computer Science, CICESE Research Center, Ensenada, México
amarkarian@gmail.com, {favela, mtentori, quiroa}@cicese.mx

Abstract. In some working environments users experience a high level of mobility while requiring collaborating and coordinating their activities with colleagues involving the exchange and analysis of documents distributed in space or time. Medical workers stand out among others by the demands imposed by hospital work. These new forms of interaction pose new challenges for the design of pervasive computing environments aimed at seamlessly integrating heterogeneous devices. Based on workplace studies conducted in a hospital, we designed and implemented a mobile collaborative system aimed at supporting co-located collaboration, proximity-based application-sharing, and the remote control of heterogeneous devices. The results of a preliminary evaluation show that users perceive the services provided by the application to be useful and efficient, even though the manipulation of the remote display through the PDA was less efficient than with the keyboard and mouse.

Keywords: Remote control, proximity-based application sharing, heterogeneous devices, hospital work, co-located collaboration, local mobility.

1 Introduction

Efficiency-oriented activities in several working environments require people to collaborate in close proximity in order to achieve their goals. Medical workers in the hospital environment, sales and stock people at retail stores, design engineers at software companies, among others, share a common set of issues regarding the need to move around the premises to locate colleagues and access information and resources. These working conditions, where people move between buildings or rooms in a local environment, correspond to what has been referred to as local mobility [2]. For example, design engineers, engaged in frequent trips away from their desk to consult with others, obtain and transfer information, and update themselves about the status of their projects. In part, those observations have directed researcher's attention towards the aspects of local mobility and collaboration confirming its relevance as a substantial component in the daily lives of professionals [9,10].

Research in CSCW and Ubiquitous Computing has focused on assessing the impact of technology on mobile collaborative environments. In particular, the increasing availability of non-desktop computing devices (everything from small handhelds to large wall displays) and ubiquitous wireless networks are enabling researchers to explore many novel interaction paradigms aimed at seamlessly

integrating heterogeneous devices to support co-located collaboration. The Peebles project [15] explores different uses of PDAs allowing users to remotely control the keyboard and the mouse input of a personal computer. Dahlberg and Sanneblad introduce an interaction style for mobile devices called "Proximity Based Notification", which attempts to combine elements of "status-event" systems by indicating the proximity of devices or people and transferring data, such as emails and notes, between them [5]. Roth presents a mobile application based on "mobility patterns" which allows users to share data between devices by using the synchronization pattern and, control a PC by using a remote proxy pattern [17]. Miller, developed a remote application controller which allows a group of individuals sharing a display space and also, controlling the applications running in such space [11]. This pervasive use of technology raises important questions for how users manage interactions between co-located collaborators and the technology.

A domain of work that represents a fruitful area to study mobile workers and the collaboration experienced by them is that of hospital settings. The medical activities in hospitals are characterized by the need for coordination and collaboration among specialists with different areas of expertise, an intense information exchange, the integration of data from many devices or artifacts and the mobility of hospital staff, patients, documents and equipment. Nurses and doctors have to move from place to place to conduct their work and to collaborate with colleagues using information resources (e.g. patient's medical records) which often are transported to different locations. Thus, the hospital can be seen as an information space and it is by "navigating" this space that hospital staff can get the information required to perform their work effectively [3]. Such rich characteristics have made hospital the focus of attention for many developments in mobile and ubiquitous technology [1].

It is also within this area of application that our own design and research efforts have been directed. In particular, we have aimed to understand and derive adequate ways to support hospital work with technologies such as digital whiteboards and PDAs [6, 14]. Furthermore, the complex characteristics of a hospital environment motivate us to place special emphasis in the integration of heterogeneous devices in support of co-located collaboration and local mobility. In this paper, we introduce a mobile collaborative application inspired by workplace studies conducted in a public hospital aimed at supporting the seamless interaction among heterogeneous devices. We illustrate how co-located collaboration and impromptu face-to-face interactions can be supported by allowing a group of hospital workers to remotely control heterogeneous devices with their handhelds and establish proximity-based application-sharing. In addition, we present the results of a preliminary evaluation discussing that the services provided by the application are useful and efficient facilitating the co-located collaboration.

The rest of this paper is organized as follows. In Section 2 we briefly describe the results of a workplace study performed to understand the mobility and collaboration experienced by hospital workers. In addition, this section presents two scenarios derived from the study that guided the design of the system. In Section 3 we explain the architecture and design of a mobile collaborative system, as well as, some of its technical challenges. Section 4 presents the results of a preliminary evaluation of the system. Finally, section 5 presents the conclusions and directions for future work.

2 Understanding Mobility and Collaboration in Hospital Work

The characteristics of hospital work call for a new computing paradigm in which collaboration support is of particular interest. Because of this, we decided to understand how collaboration is experienced by hospital workers. This will help us ground our design by translating current work practices into those to be supported by appropriate collaborative tools.

2.1 Collaboration and Mobility in Hospital Work

We conducted a study aimed at revealing how much time hospital workers spend in different activities; how much they move, where they move to and why; with whom they collaborate more often and the artifacts they use in support of their work [13]. This study was conducted for a period of five weeks, where two medical interns, two head nurses and two physicians were shadowed for two complete working shifts and interviewed by a couple of researchers. The main contribution of this study relies in the characterization of mobile work and the information usage practices that hospital workers engage in.

We found that individuals spent, on average, 59.92% of their time in their base location while the rest is spent on-the-move (40.08%). Even though it is clear that hospital staff spends as much as half of their working shifts on-the-move; they do spend a considerable amount of time at what we have referred to as their base station. They need to do this in order to fill administrative forms, write medical notes or analyze medical evidence. Mobile computing devices, such as PDAs, are not appropriate for many of these tasks, such as, writing relatively long documents or reading a medical article. Thus, the need to provide the environment with heterogeneous computing devices, ranging from handheld computers that can be used to capture and access limited amounts of information, to PCs that can be used at fixed sites for longer periods of time, and finally, semi-public displays located at convenient places, that can be used to share and discuss information with colleagues. Sharing and controlling these devices should be seamless, so as not to interrupt the natural execution of the task at hand.

Also, we found that hospital workers spent 77.28% of their time performing activities which require co-located collaboration. For example, hospital workers spent, on average, 15.35% of their time evaluating clinical cases to assess the health condition of a patient. This activity often requires from hospital workers examining clinical evidence, consulting reference material and discussing patient diagnoses with colleagues. Thus, hospital workers require the simultaneous sharing of patients' medical records or medical results (e.g. X-Ray images) to enrich a discussion. We observed that those activities are often performed on-the-move. Research in CSCW has been conducted towards supporting this type of interaction in office environments through computer-mediated communication [8, 9, 18]. These solutions are clearly not appropriate for a hospital environment, where according to our data; workers spend only a fraction of their work shift in front of a computer, and more than twice this amount of time away from their base station. Indeed, what is required is support for impromptu face-to-face encounters.

These characteristics have suggested us to envision a mobile collaborative system that supports co-located collaboration with the ability of detecting the presence of other devices in the vicinity, as well as, allowing sharing information between heterogeneous devices, remotely monitoring other computers, and sharing handheld applications.

2.2 Scenarios

We decided to use scenarios as a way to frame our understanding of hospital work practices and also to project our vision of how their work could be augmented with impromptu face-to-face collaborative tools. We next describe two scenarios:

Scenario 1: Remote control of heterogeneous devices.

This scenario illustrates how the system can be used to remotely control another device.

Everyday, at the internal medicine office, the medical interns meet with the attending physician to discuss the status of their patients. They help each other by discussing the diagnosis as well as future treatments for the patients. The physicians decide to discuss the case of the patient in bed 226 who is not responding to the treatment as expected. They show on a public display the information related to the patient such as an X-Ray image and the medical record. While the discussion takes place Dr. Diaz controls the public display from his PDA scrolling through the medical record and highlighting some information relevant to the patient's diagnosis, by using the remote control tool. Also, Juan, the medical intern in charge of the patient, wants to contrast the highlighted sections by pointing with a telepointer an area of the X-Ray image. Finally, Dr. Diaz needs to share a recent article, stored in his computer office, which he considers relevant to enrich the discussion. By changing the device controlled, he accesses his computer office retrieving this article.

Scenario 2: Proximity-based application-sharing.

This scenario illustrates how the system can be used to establish application-sharing by proximity among different mobile devices.

Everyday, at the nursing station, the medical interns meet Dr. Diaz, the attending physician, to jointly conduct the ward round. By moving from patient to patient in their bedrooms, they discuss each patient's clinical case assessing their health condition. They start with the patient in bed 220 consulting his medical record and laboratory tests. Using the application-sharing tool Dr. Diaz, synchronizes his PDA with those of the medical interns to share the information consulted while discussing the clinical case. In this system, the devices in the vicinity of Dr. Diaz are used to select the target devices to establish application sharing by proximity. Once the users are sharing the application, Dr. Diaz draws a box to zoom a specific area of the patient's medical record standing out information relevant to the patient's diagnosis.

Scenarios such as these were generated to bridge the gap between current medical practices and an idealized mobile collaboration medical environment. They address current sources of misshapes or look to simplify complex tasks through the use of mobile computing technology.

3 The Mobile Collaborative System

Co-located collaboration can be supported by allowing a group of hospital workers to remotely control a public display with their handhelds as noted in scenario 1. In this case Dr. Diaz uses his PDA to control the mouse of the public display. The application is capable of handling concurrent controllers but only one at a time. The user that has control of the floor is able to move the cursor as well as type on the device being controlled, while the others are only able to point at the screen using a telepointer, as shown in the scenario. Figure 1 illustrates how hospital workers could collaborate through a large display using their PDAs.



Fig. 1. Remote control of a large display (a) A physician interacts with a public display while a colleague remotely interacts with the public display from his PDA (b) A close up of the PDA's remote control application

Impromptu face-to-face interaction can be supported by allowing a group of hospital workers to establish proximity-based application-sharing among different mobile devices as described in scenario 2. In this case the source device would ask Dr. Diaz handheld for permission to share its screen. When granted, a screen capturing thread is launched and the screen image is sent to the medical interns' handhelds. In this case, there is no hierarchical meeting, which happens while sharing a public display assigning control and telepointers. Figure 2 shows a screenshot of the application-sharing functionality.

Figure 3 shows the architecture of the component-based system that uses a client-server architecture as a basis for its implementation. Wireless connectivity between servers and mobile clients is achieved through 802.11b access points. The system includes four agents, two on the source device and two on the target device. We implemented two versions of these agents: one for PDAs using Windows Mobile on the top of mSALSA and another for desktop (PC, laptops, public displays) using Windows XP on the top of SALSA class framework [16].



Fig. 2. Proximity-based application sharing functionality (a) The MiniView tool provides a small depiction of the information shared (b) The directional pad used to control de speed of the cursor (c) Options of the application sharing application

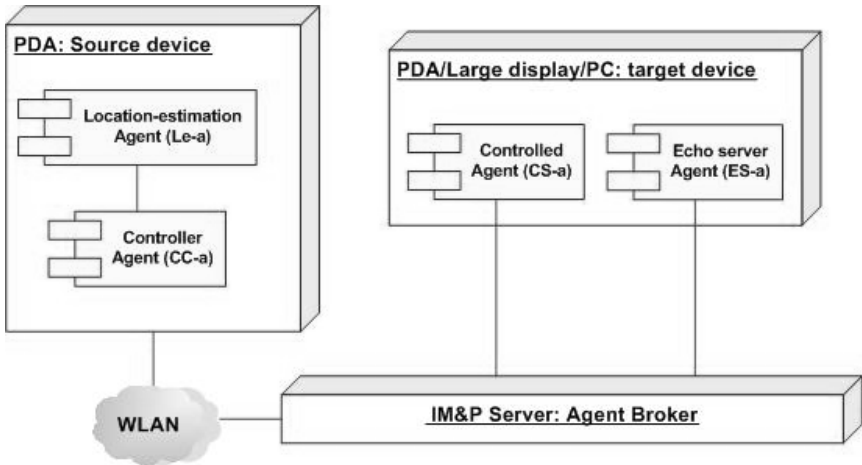


Fig. 3. Architecture of the mobile collaborative system

In the architecture the *agent Broker* handles communication between agents through XML (eXtensible Markup Language) messages storing the state of agents and notifying their changes to other agents subscribed to them. The *Location Estimation agent (LE-a)* determines the approximate location of devices within a hospital. Through this component the proximity based functionality is provided. The *Controller agent (CC-a)* that resides in the source device is used to establish the proximity-based application sharing and remote control of computing devices working in pair with the *Echo server agent (ES-a)* and *Controlled agent (CS-a)* that reside in the target device. We next describe the functionality of the agents with more detail.

3.1 The Location-Estimation Agent (LE-a)

The aim of this agent is to determine a set of mobile devices in the vicinity in order for the system to achieve a proximity-based functionality. This agent resides in the PDA and is used to estimate the position of devices with a considerable accuracy [4]. Radiofrequency (RF) signals received by the mobile device from at least three access points are measured to obtain their signal strength. A trained back propagation neural network is embedded within the component and it is used to estimate the approximate location of the users. Neural networks can learn from training examples and map input sequences (signal strength) to output sequences (2D coordinates). We make use, in addition to signal strengths, of the neighborhood in physical space of the location to be estimated to train the neural network. However, when estimating the location in real-time, the network takes the estimation of the previous time instant as an extra input, which is considered to be a neighbor location. In this way, the network takes into account information from the past to reduce the error which is diminished by almost 55% when using this approach. This component was trained and tested with data from the internal medicine area of the hospital. Several paths followed by medical staff were covered in order to test our approach.

3.2 The Controller Agent (CC-a)

This agent resides on the PDA commanded by user input. These inputs are processed and relevant XML messages are sent using instant messaging. The controller is used to request and then revoke the duo mouse/keyboard and telepointers. When the stylus is touching the screen, a thread routinely wakes up after a certain amount of time has passed to sample the cursor position. This architecture allows the sending of messages only when necessary (i.e. when the stylus is moving on the screen). In addition to the processing of the movement, the user can tap on the screen to generate a click. The tap needs to be fast enough (i.e. the time between the down and up states of the stylus is less than a certain amount of time) to be interpreted as a click. The *Telepointers* that the users can control are identified by an ID and a particular color. This identification number will allow the *Controlled agent* to handle different pointers.

3.3 The Controlled Agent (CS-a)

This agent resides in the target device, interpreting messages from controllers to allow movements and mouse/telepointer assignation. Whenever a new message arrives, this agent translates it into mouse or pointer movements. This agent also assigns pointers to different controllers storing them in a collection indexed by the ID of the *Telepointer*. The Telepointer is drawn in the screen by calling the native API within Java.

3.4 The Echo Server Agent (ES-a)

This agent operates on the target device and launches a worker thread for each client connection as illustrated in figure 4. Once certain conditions are met, the server broadcasts a full resolution image of the screen to all the clients. Then, it is up to the clients to process the image according to what the user wants to see. Doing the

processing of the image, there is a tradeoff between the delay of sending the image among shared devices and the time a user is willing to wait viewing different positions of the image. Indeed, we chose which trade off was the least disrupting for the user: the delay before screen changes are echoed on the PDA or a short time to display a different position of the screen. The first idea would underpin that the server, instead of sending a full resolution image, would only send the view the user wants. Thus, each time the user wants to change his view the server has to send the new view. The second idea needs more time to send the full resolution image but once it is sent, it is faster to change between views as the program just need to redraw the screen from the same source image but with different boundaries. It seems that this last solution is more efficient as the screen is not very likely to frequently change radically.

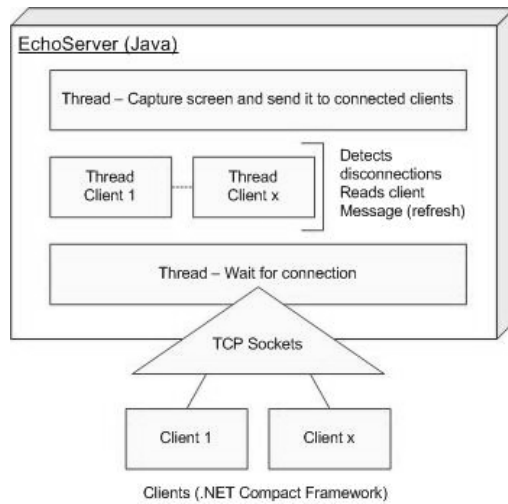


Fig. 4. The echo server functionality

3.5 Technical Challenges

As we said, the functionality should not be intrusive by any means. Thus, the control application has been integrated deeply with the screen echoing. This imposes several technical and design challenges that had to be overcome to be able to run as smooth as possible. We next describe how we deal with those challenges.

Dealing with the limitations of the mobile device.

To reduce the use of PDA computing time to receive and process the distant screen image and optimize network exchanges, we decided to send an image to the PDA only when it is needed. To achieve this, each time a new screen is captured, it is compared, by calculating the Mean Square Error (MSE), with the last image sent. If the MSEs' difference is high enough, the new image is sent. This automatic update has been seen as sufficiently efficient in the evaluation. However, to improve the user

experience a manual update is possible upon user request, this is why the PDA has been given the ability to send data to the server. Obviously, to make the aforementioned automatic update efficient and useful, the PDA should compute the image (resize and adapt) only when a new image is available. Finally, we have to consider the limitations of memory and speed that the development on mobile devices triggers. For this, we perform as much work as possible on the devices more powerful such as a PC or the public display.

Dealing with different aspect ratios.

Sharing information among heterogeneous devices raises presentation challenges related to the amount of information we could be able to display in devices with different screen sizes. We have to get as much valuable information displayed on a 57 inches public display adapting it for displaying in a 3.5 inches PDA, where the width over height ratio is indeed $\frac{4}{3}$ for a screen but $\frac{3}{4}$ for a PDA. Many ideas have emerged as a possible answer to the aforementioned challenges. As an example the ratio problem could be solved by rotating the PDA. The size issue would be answered by zoom in and zoom out feature. To cope with this, we built up several solutions and tried to assess all of them using the following parameters: intuitive user manipulations, efficiency (number of user manipulations) and best use of PDA available screen space. Thus, the user should be able to focus on the area he or she wants in a minimum number of simple manipulations.

Dealing with breaking connectivity.

An additional issue we have to take into account is the fact that due to the wireless nature of the connection, connectivity breaks are very likely to happen. This is why all requests from the PDA have to be acknowledged by the host computer to take effect. Once the acknowledgement is received, the program reacts accordingly on the PDA. In this way the user cannot be blocked in a state that does not correspond to its status on the remote computer. If the acknowledgment gets lost the user simply repeats the command. To understand this, Figure 5 shows the message exchanges required to get and revoke a telepointers. The acknowledgment ensures that the state of the user is in phase with its actual state on the server (i.e. it really has the telepointer)¹.

Dealing with thread affinity.

One of the problems that appeared due the chosen development framework is what is known as thread affinity. Thread affinity means that their properties and methods can be called only by code running on the same thread that created the interface. Threads needing to call other threads code happen many times in our application. It is the case when the thread that manages the socket and receives images wants to update the view once the image has been received. The thread that created and that manages the view is the main thread, calling any of its graphical properties from another thread would crash the program.

¹ The exchanges are the same for the mouse but additional checks would have been done to decide whether the user can get the mouse or not.

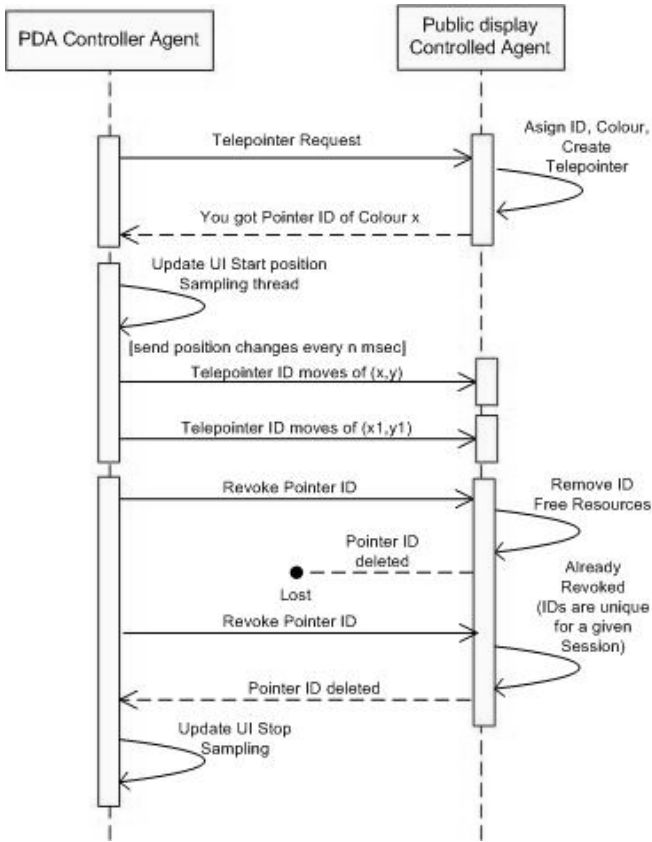


Fig. 5. Message exchange to get a telepointer

A work around is to use an Invoke method which runs a delegate on the thread that originally created the control, allowing the delegate to safely interact with the control. However, the framework we used supports only the ability to delegate methods that are handling events. Therefore we had to write an event-handler delegated to the User Interface thread. The socket thread raises a notifying event when socket operations complete and passes any associated data with the operating system.

4 Preliminary Evaluation

We conducted a preliminary evaluation aimed at identifying the weaknesses of the design and obtaining user feedback about the quality of the services provided by the application, as well as, to asses if the approach supports collaboration resulting in enhanced creativity. For this, four desirable properties were identified as relevant for evaluating the system: (1) effectiveness of shared control and pointers in a meeting

situation, (2) pertinence of the control sharing mechanism, (3) seamlessness of the application and (4) the precision and speed of the application experienced by users.

The subjects of the study were six graduate students, some of them having already used a PDA and others not. These subjects were selected because they often move around engaging in co-located interactions to conduct their work and consequently they seemed ideal for the purposes of our study. Each informant was introduced to the characteristics of the study, its purposes, goals, and was asked to participate voluntarily. Although the users and the environment with whom and where the evaluation was conducted, are not related to hospitals, the characteristics of our informants present similar requirements regarding to collaboration and mobility to those experienced by hospital workers. Thus, we think that the results obtained through this evaluation are not restricted to graduate students and could be easily extended and applied to hospital environments.

4.1 Procedure

The evaluation was conducted in a meeting room with a public display and several personal computers. Before we started the evaluation we gave each user a PDA, simulating indeed, a real co-located meeting environment saturated with heterogeneous devices. We next describe the evaluation session and the measurements used to analyze the data gathered through this session.

Evaluation session.

The evaluation required two subjects to collaborate in performing three different tasks as Figure 6 illustrates. Thus, we randomly grouped our subjects in three pairs conducting an evaluation session with each group including the following activities:

First, voice-recorded instructions were listened by the users describing the application and explaining how to use its functionalities.

Then, users had to complete three tasks used for testing different aspects of the application. The first task was completed individually requiring each user to check his email and fill a survey with/without the echo functionality. This task served also for learning purposes and they were given up to eight minutes to complete it. The second task required each group to collaborate by creating a picture slideshow accompanied with a soundtrack with the aim of making a promotional clip for a city. In this task, each subject played a particular role: as a sound designer, choosing one song among ten that were available; or as a graphic designer choosing eight pictures from the thirty pictures available. This task had to be completed in 15 to 20 minutes allowing subjects to freely use the application as they preferred. They could use the echo or not, browse pictures with the mouse functionality or with the keyboard emulation and use the telepointer and exchange control whenever they want. Two versions of this task were executed one using the PDA for creating the promotional clip and the other one using a mouse in collaboration with a colleague and individually. Finally, the third task was conceived to estimate the precision and the speed of the application. The user had to click a red spot in synchronization with the tick of a metronome set at a very low speed. The priority was to follow the metronome instead of the precision.

After the tasks were completed, users' impressions were collected through a questionnaire with 7 point Likert-scale assertions, which included topics such as their

satisfaction with the modality in which they worked, their understanding of the application and what they would like to see in future versions, as well as, a section for free comments.



Fig. 6. Two groups of subjects controlling the public display with their PDAs during the evaluation session

Measurements.

We wanted to capture qualitative data and quantitative measurements of the users' inputs. Therefore, we videotaped the evaluation session capturing the users' interaction with the PDA and the verbal communication exchanges they were having while performing each task. This gave us a good qualitative set of information regarding the users' experience with the services provided by the application.

In addition, the screen of the public display was captured using the Morae software [12] which registers all user inputs for statistics and analysis. During the tasks, all user inputs were recorded: the number, exact position and window, on which the clicks were performed, which keys were pressed on and where.

4.2 Results and Discussion

Here we present some of the results obtained through the video tape, the users' inputs captured and the questionnaire regarding the users' perception of ease and usefulness of the system, as well as, the precision and speed movement of the application.

Users' perception of ease of use and usefulness.

Subjects were positive about the application, in the Likert-scale assertions as well as in the free comments sections. The main advantages the subjects highlighted were all related to the collaborative aptitudes (pointer and control services) provided by the application.

Table 1 shows the answers to some assertions of the questionnaire concerning the overall application. As the results show the most preferred service was the echo functionality with an average of 6.6 (close to '*strongly agree*'), followed by the control functionality with 6.5. Despite of this, users slightly agree regarding the

efficiency of the application, with an average of 5.2; however, the whole application seemed attractive to users with an average grade of 6.5.

Table 1. Perception of easy of use and usefulness

| <i>Question</i> | <i>Average</i> <i>(1: Strongly disagree and 7:Strongly agree)</i> |
|---|--|
| The control functionality is useful | 6.5 |
| The echo functionality is useful | 6.6 |
| The functionalities are sufficiently integrated | 5.6 |
| The application is efficient | 5.2 |
| The whole application is attractive | 6.5 |

We provided a list of seven improvements and we asked the subjects to choose three of them which, in their opinion, need to be attended with higher priority (table 2). All users agreed that the application must provide better precision regarding to the point and movements of the control functionality; however the users agreed that the application provides adequate right-click ability. Also, half of the subjects recognized that a click and hold ability must be incorporated into the application.

Table 2. Improvements for the application

| <i>Improvement</i> | <i>Choices</i> |
|---|----------------|
| A less jerky movement | 5 |
| Improved speed | 2 |
| Higher precision | 6 |
| Right click ability | 1 |
| Click and hold ability (to move a window for example) | 3 |
| New pointer shape | 0 |

However, even though the overall application was well rated in the questionnaires, it seems that moving the pointer on the screen using the PDA is still a bit tedious. Thus we decided to conduct a specific analysis looking closer the precision and the speed movements.

Precision and speed movement.

First, a time analysis was conducted focusing on how much time each team invested in creating a complete slideshow (task 2). We found that it took more time for subjects collaborating to complete the task, spending, on average, 17:20 min. More than double the amount of time spent by the subjects working alone with the PDA, who spent on average, 7:14 min., and a single individual working with a mouse who took only 4:30 min. to complete the task. To better understand this we need to look in

more detail at what happened during the creation of the slideshow. Thus, we analyzed the clicks done during these activities.

Figure 7 shows the click density, which represents the number of clicks done within a given interval of time (depending on the total length of the experiment). From these graphs we can notice that the number of clicks performed by a group (117) is higher than that of the single individual who clicked 64 times. This partially explains why it takes much longer for a group to perform the task. Now, analyzing where they clicked, the pairs reviewed more than 20 pictures (from 20 to 24) whereas the person alone reviewed just between 12 and 14 pictures. Identically, they also spent more time choosing and listening to the songs. Thus, the additional time spent by the group was to a large extent devoted to the analysis of more alternatives.

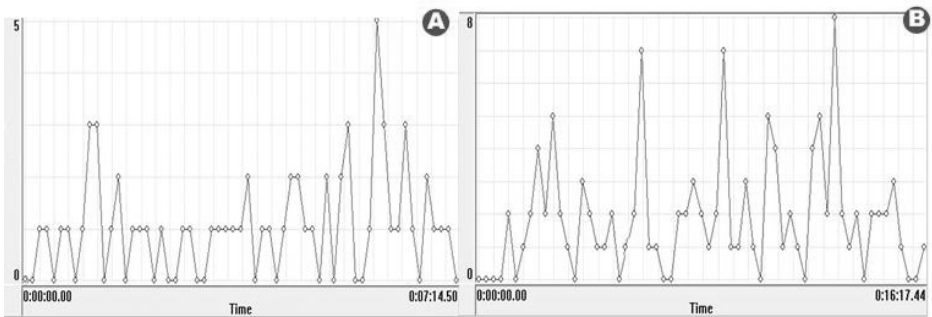


Fig. 7. Click density of (a) a single person using the PDA (b) two subjects using PDAs

We also conducted a precision analysis to evaluate the imprecision radius by examining the clicks done while the users performed the third task. From this, we can conclude that the subjects were unable to follow the metronome (even after lowering the speed). Reviewing the record of the second task showed some imprecision in click locations which didn't seem problematic though. In addition, it has been reported that the precision was barely adequate (4.2/7). We think that the problem only arises when speed and precision are necessary. To improve precision (and a less erratic movement), one needs to regularly adjust the speed of the cursor (via the slider, better precision slower speed of movements) which requires additional actions by the user.

Another critical parameter for the control application is the speed of the cursor movements. The questionnaire relates that this issue was not handled very well by the application (3.33/7). The problem is not related to the fact that the cursor is too slow, the cursor movement is perceived to be too jerky (it jumps from one position to another). A shorter sampling time may solve the problem.

The overall application was perceived to be useful and efficient, despite that moving a pointer on a remote screen using the PDA led to some precision problems and decreased efficiency. Users seemed to require additional experience with the tool as we observed that they improved their precision and speed towards the end of the exercise.

5 Conclusions

Research in CSCW has pointed out several challenges regarding how novel technology could enhance co-located collaboration in mobile environments. Different working environments demand considerable coordination and communication from the professionals that work in such settings while navigating the premises. Hospitals stand out as environments where several specialists are involved in the treatment of a patient, requiring frequent information exchanges; triggering essential coordination and collaboration issues within a highly mobile environment. Those conditions make hospitals convenient settings for deployment of mobile and ubiquitous computing technology.

In this paper, we present a set of services available through a handheld computer to support mobile workers' collaboration in local mobility settings. We propose to assist hospital workers' co-located interactions by integrating heterogeneous devices such as public displays and handhelds. We designed and implemented a system that creates collaborative environments on-the-move, around a public display or with other mobile devices. To provide this functionality, the system allows the collaborative control and sharing of the display of another computer in the vicinity. Our attention focused on offering wireless concurrent control and pointing capabilities with a pleasant user experience. The results of the preliminary evaluation of the system show that the users experienced some difficulty moving the cursor of the remote screen, yet, they found the application to be useful in facilitating co-located collaborations. With the lessons learned in this evaluation, we plan to conduct a second evaluation with medical staff with the aim of assessing how the application would allow users working together in an efficient and creative way during their everyday practices.

References

1. Bærbak, H. and J.E. Bardram.: Supporting Human Activities—Exploring Activity-Centered Computing. In Proc. of Ubicomp. Goteborg, Sweden. Springer-Verlag (2002) 107-116
2. Bellotti, V. and Bly, S.: Walking Away from the Desktop Computer: Distributed Collaboration and Mobility in a Product Design Team. In Proc. of the CSCW. Boston, MA. ACM Press (1996) 209-218
3. Bossen, C.:The Parameters of Common Information Spaces: the Heterogeneity of Cooperative Work at a Hospital Ward. In Proc. of CSCW. ACM Press (2002) 176-185
4. Castro, L. and J. Favela.: Continuous Tracking of User Location in WLANs Using Recurrent Neural Networks. In Proc. of the ENC. Puebla, Mexico. IEEE Press. (2005) 174-181
5. Dahlberg, P. and Sanneblad, J.. The use of Bluetooth enabled PDAs. In Proceedings of IRIS-23. Uddevalla, Sweden. (2000) 150-161
6. Favela, J., Rodriguez, M., Preciado, A and Gonzalez, V.: Integrating context-aware public displays into a mobile hospital information system. IEEE Transactions on Information Technology in Biomedicine. Vol. 8 No.3 (2004) 279- 286
7. Dahlbom, B. and Ljungberg, F.: Mobile informatics. Scandinavian Journal of Information Systems, Vol. 10, No. 2 (1998) 227-234.

8. Isaacs, E., Tang, J.C. and Morris, T.: Piazza: A Desktop Environment Supporting Impromptu and Planned Interactions. In Proc. of CSCW. ACM Press (1996) 315-324
9. Kraut, R., Fish, R., Root, R. and Chalfonte, B.: Informal communication in organizations: form, function and technology. In S. Oskamp & S. Spacapan (Eds), People reactions to technology in factories, offices and aerospace. The Claremont Symposium on Applied Social Psychology. Sage Publications. (1990) 145-199
10. Kristoffersen, S. and Rodden, T.: Working by Walking Around: Requirements of flexible interaction management in video-supported collaborative work. In Proc. of Human Computer Interaction. Springer Verlag (1996) 315-329
11. Miller, J.R.: The remote application controller. In Computers and Graphics. Vol. 27. No. 4 (2003) 605–615
12. Morae software. Available at: <http://www.techsmith.com/morae.asp>
13. Moran, E., M. Tentori, V.M. González, A.I. Martinez-Garcia, and J. Favela: Mobility in Hospital Work: Towards a Pervasive Computing Hospital Environment. To appear in the Int. J. of Electronic Healthcare
14. Munoz, M.A., Rodriguez, M., Favela, J., Martinez-Garcia, A.I. and Gonzalez, V.: Context-aware mobile communication in hospitals. IEEE Computer. Vol. 36. No. 9 (2003) 38-46.
15. Myers, B. A.: Using Handhelds for Wireless Remote Control of PCs and Appliances. Interacting with Computers. Vol. 17. No. 3 (2005) 251-264
16. Rodriguez, M., Favela, J., Preciado, A. and Vizcaino, A.: Agent-based Ambient Intelligence for Healthcare". AI Communications. Vol. 8. No. 3 (2005) 201 – 216
17. Roth, J.: Patterns of Mobile Interaction. In Personal and Ubiquitous Computing, Vol. 6, No. 4 (2002) 282-289
18. Whittaker, S.: Rethinking video as a technology for interpersonal communications: theory and design implications. Int. J. Hum.-Comput. Stud. Vol. 42. No. 5 (1995) 501-529

Predicting User Interest Region for Collaborative Graphics Design Systems in Ubiquitous Environment

Jiajun Bu¹, Bo Jiang², Chun Chen¹, and Jianxv Yang¹

¹ College of Computer Science, Zhejiang University, Hangzhou, P.R. China

² College of Computer and Information Engineering,
Zhejiang Gongshang University, Hangzhou, P.R. China

{bjj, nancybjjiang, chenc}@zju.edu.cn

yangjianxv@yahoo.com.cn

Abstract. The fast expansion of wireless networks and mobile devices enables portable devices to join collaborative graphics design conveniently. The limitation of display size and computational power of these embedded devices makes it hard for mobile users to browse large pattern that renewed in real time efficiently. We present a novel user interest region prediction algorithm to forecast user's intention in the near future. Related experiment was carried out to test the effectiveness of the algorithm. Results show that the algorithm can well predict mobile user's interest regions. Based on the prediction, only sub-patterns and operations that might be interested to user are issued to the embedded sites. User study results indicate that the proposed approach is effective and the feasibility of the collaborative graphics design system in ubiquitous environment is enhanced.

1 Introduction

Collaborative graphics design systems (GDS) [1, 2, 3] play an active role in assisting several end users to fulfill the corporate graphics editing tasks. With the rapid development of mobile computing and embedded technology, devices that support GDS can be both the traditional desktop PC and portable PDA which connects to collaborative sessions via mobile networks. Equipped with such collaborative devices, mobile cooperators may take part in collaborative pattern design to view or edit the shared graphics documents at any time and place conveniently.

As one of the fundamental traits of GDS, "what you see is what I see" (WYSIWIS) was presented to focus and coordinate collaborators attentions on what seems to be a shared visual workspace or document. In CSCW research the concept of providing different views on shared information is also known by the term "relaxed WYSIWIS" as introduced by Stefik et al. [4]. Relaxed WYSIWIS or "what you see is what I can see" (WYSIWICS) support flexible (also referred to as tailorable or customizable) views for multi-user collaborative editing applications. WYSIWICS also well supports mutual awareness of GDS for embedded devices with limited screen size. However, in collaborative graphics design especially for those large graphics documents, transmitting each drawing operation or the whole pattern with high resolution to mobile embedded sites for consistency maintenance may put a heavy burden both on the

narrow bandwidth networks and the embedded devices. Therefore, learning and predicting users' interest for viewing in GDS and sending what users may like to see might be an effective way to promote the usability of GDS and enhance the global collaborative efficiency.

In recent years, some work has been done to develop user interest model to assist users to browse large images on PDA. In [5, 6], the authors proposed an attention model based image adaptation approach. Based on an automatically extracted image attention model, a number of features to aid or automate common image browsing tasks are designed. The method to detect user interest maps and extract user attention objects from the image browsing log is proposed in [7]. Contrast based image attention analysis by using fuzzy growing [8] is presented to detect attention area. The approaches are deemed can represent users' actual image viewing interest. However, above research only extract user attention objects or regions for static images that stored in PDA beforehand. To our knowledge, there is a lack of reports of detecting and predicting users' interest region on the collaborative workspace in which the drawing objects are dynamic changed.

Due to the limited display size and computational power, PDA can hardly pre-download the large patterns with high resolution and process all the collaborators' editing operations for consistency maintenance. The above limitation hampers mobile users to join the collaborative graphics editing efficiently.

The contributions of this paper include:

- We propose an effective algorithm that can predict users' interest region in collaborative graphics editing process dynamically.
- Based on user interest region prediction, those regions that the user may be interested in are pre-downloaded to the PDA and only the remote collaborator's operations that related to the interest regions are issued to mobile sites. Thus, the usability of CGD in ubiquitous environment is greatly enhanced.

The rest of the paper is organized in four parts. We first describe the traits of user interest region in CGD and the necessity of user interest region prediction. Second, we present the user interest prediction algorithm. Then, an experiment is carried out to test the effectiveness of the algorithm. Finally, we conclude the paper.

2 User Interest Region in CGD

While browsing graphics users might be interested in certain regions that are notable or prominent. In reality, if objects and their surroundings are of varying contrast, the objects would be the attention area for people. As it is shown in Figure 1(a), users' intention will focus on the blue flower definitely. However, besides the attribute of human visual perception like sensitiveness to contrast, user's attention observed has intensive relationship with the real-time editing operations that issued by his collaborators in CGD. In most cases, user is inclined to trace the variation of the graphics on the canvas. In Figure 1(b), while the designer draws the leaves, the viewer's intention might follow the editing process.

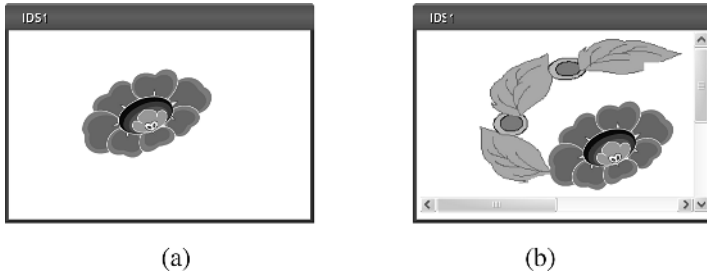


Fig. 1. Example of user interest region. (a) User's intention focuses on the flower. (b) Viewer's intention moves from the flower to the leaves that drew on the top left corner of the flower.

As it can be seen from Fig. 1, user interest region may be changed dynamically during the collaborative graphics editing procedure. Therefore, predicting users' future intention and pre-download some part of a shared document in that region or transmit user interest-oriented drawing operations to the screen-limited handheld devices in real-time collaborative design might be a brand new method to promote the effectiveness of CGD in ubiquitous environment.

3 User Interest Region Prediction

Since users' interest region can mainly be reflected by the collaborative operators' drawing focus region, that is, the editor's intention. An indirect prediction algorithm, which we called user focus prediction algorithm, is proposed. An Extend Prediction Set algorithm is adopted to predict the future focus region of designers. The algorithm predicts user's focus area by extending current focus region.

3.1 Description of Context

Definition 1: Focus Context. The context of user's focus can be described as a quadruple Focus Context:

$FocusConU_k$: the owner of the focus;

$FocusConU_k^{orientation}$: the focus orientation of user k;

$FocusConU_k^{region}$: the focus region of user k;

$FocusConU_k^{time}$: the focus time of user k;

3.2 Prediction Algorithm

The algorithm of focus region prediction is illustrated as follows:

Step 1: Chip the original pattern.

The pattern is divided into $M \times N$ small square grids, which is denoted as $W[M][N]$.

Step 2: Initialization.

Initialize the focus region of user $FocusConU_k^{region}$ and the corresponding focus time $FocusConU_k^{time}$. Take user's first clicked square as the starting one. Take the starting one together with the other eight squares surrounding it as the stub, the current tested region and also an initialized $FocusConU_k^{region}$, while the $FocusConU_k^{time}$ started its time simultaneously. The corresponding information about the owner $FocusConU_k$ can also be fetched and recorded in the system.

To illustrate the processing, we take set $\{R_0, R_1, R_2, R_3, \dots, R_8\}$ as example, as shown in Fig.2.(a).

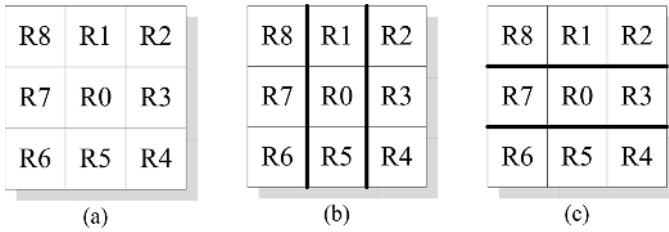


Fig. 2. (a) The current tested region. R_0 is the starting square. R_0 together with $\{ R_1, R_2, R_3, \dots, R_8\}$ is the stub. (b) Stub is chipped from left to right, leaving three horizontal groups. (c) Stub is chipped from top to bottom, leaving three vertical groups.

Step 3: Calculate the intensity.

Gathering user k 's operating information in the 3×3 squares and processing it, finally determining the focus orientation $FocusConU_k^{orientation}$ according to operating intensity calculated.

Definition 2: Operating Intensity

$$I_{R_i}^{U_k} = \sum_{j=1}^n N_j^{R_i} \alpha_j, j \in [1, n], n \in [1, 2, 3, \dots]$$

$I_{R_i}^{U_k}$ is user U_k 's operating intensity in region R_i ;

$N_j^{R_i}$ is user's operating number in region R_i at time slot j , where operating number is the click number and time slot is shown in Fig.3;

n , depends on system, is the number of time slots. In our system, we have $n = 5$, that is, we take the latest 5 time slots as the calculated units such that, the information in the focus region is fetched whenever user has any operation on it in 5 time slots;

α_j is power value timed to each item, where $\alpha_j = c\alpha_{j+1}, (c > 1)$, here we have $c = 3/2$. α_j decreases in descending order of importance in negative orientation of coordinate.

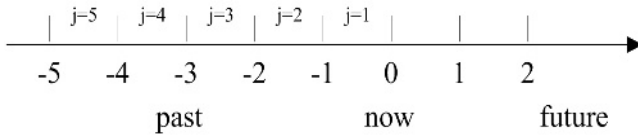


Fig. 3. Time slots. The abscissa denotes time which is partitioned into many slots. The subscript 0 denotes now, the left of 0 indicates past and the right of 0 is future.

Step 4: Abstract base set and determine orientation.

Since we want to predict the user’s focus by extending user’s current focus set, we would start this extending work from a basic set which is extracted from the stub.

(Step 4a): Firstly we group the stub into three horizontal and vertical components respectively, as shown in fig.2.(b)(c). Each component is a basic set candidate.

Calculate each ratio of intensity of component to the total intensity.

$$\left\{ \begin{array}{l} Ratio_{horiz}^{left} = \frac{I_{R_8} + I_{R_7} + I_{R_6}}{SumI_R} \\ Ratio_{horiz}^{center} = \frac{I_{R_1} + I_{R_0} + I_{R_5}}{SumI_R} \\ Ratio_{horiz}^{right} = \frac{I_{R_2} + I_{R_3} + I_{R_4}}{SumI_R} \end{array} \right., \quad \left\{ \begin{array}{l} Ratio_{vert}^{top} = \frac{I_{R_8} + I_{R_1} + I_{R_2}}{SumI_R} \\ Ratio_{vert}^{center} = \frac{I_{R_7} + I_{R_0} + I_{R_3}}{SumI_R} \\ Ratio_{vert}^{bottom} = \frac{I_{R_6} + I_{R_5} + I_{R_4}}{SumI_R} \end{array} \right.$$

Where, $SumI_R$ is the total intensity; $Ratio_{horiz}^{left}$ is the ratio of sum of intensity of left component to the total intensity, the others means similar;

We choose the largest one of the six ratios, denote it as $Ratio_{bar}^{biggest}$.

(Step 4b): Secondly, we choose a corner stub as a component which is the basic set candidate. The corner is composed of four regions. The four regions are either top-left the stub, either bottom-left it, either top-right it or bottom-right it.

Calculate each ratio of intensity of component to the total intensity.

$$\left\{ \begin{array}{l} Ratio_{corner}^{topleft} = \frac{I_{R_8} + I_{R_1} + I_{R_7} + I_{R_0}}{SumI_R} \\ Ratio_{corner}^{bottomleft} = \frac{I_{R_7} + I_{R_0} + I_{R_6} + I_{R_5}}{SumI_R} \end{array} \right., \quad \left\{ \begin{array}{l} Ratio_{corner}^{topright} = \frac{I_{R_1} + I_{R_2} + I_{R_0} + I_{R_3}}{SumI_R} \\ Ratio_{corner}^{bottomright} = \frac{I_{R_0} + I_{R_3} + I_{R_5} + I_{R_4}}{SumI_R} \end{array} \right.$$

We choose the largest one of the four ratios, denote it as $Ratio_{corner}^{biggest}$.

(Step 4c): Choose the basic set.

$$basic\ set\ is\ a \begin{cases} bar & \text{if } [Ratio_{bar}^{biggest} \div (4/9)] \leq [Ratio_{corner}^{biggest} \div (1/3)] \\ corner & \text{if } [Ratio_{bar}^{biggest} \div (4/9)] > [Ratio_{corner}^{biggest} \div (1/3)] \end{cases}$$

If the basic set is a bar, we will choose it from the six components indicated in Step 4a. On the other hand, if the basic set is a corner, we will choose it from the four

components indicated in Step 4b. To illustrate, if the $Ratio_{horiz}^{left} \div (1/3)$ is the largest one, R_8, R_7, R_6 is chosen as the basic set.

(Step 4d): Determine the orientation $FocusConU_k^{orientation}$.

If the basic set is a bar, the orientation may be one of right, left, top and bottom, or stay still. For instance, $\{R_8, R_7, R_6\}$ is the basic set, that is, the $Ratio_{horiz}^{left}$ is chosen, the orientation is left. If either $Ratio_{horiz}^{center}$ or $Ratio_{vert}^{center}$ is chosen, the stub stay still, no changes are needed.

If the basic set is a corner, the orientation may be one of top-left, bottom-left, top-right and bottom-right. For example, $Ratio_{corner}^{topright}$ is chosen, $\{R_1, R_2, R_0, R_3\}$ is the basic set, the orientation is top-right.

Step 5: Extend the focus region.

If the stub stay still, no orientation is chosen, there is no need to extend current focus set.

(Step 5a): If one of right, left, top and bottom orientations is chosen, the system will extend three regions along with the one of the four orientations respectively.

(Step 5b): If one of top-left, bottom-left, top-right and bottom-right orientations is chosen, the system will extend five regions along with the one of the four orientations respectively. To illustrate:

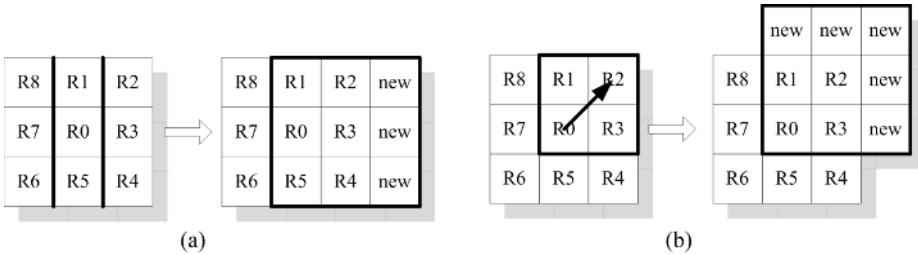


Fig. 4. (a) Right orientation is chosen, and three new squares that adjacent to $\{R_2, R_3, R_4\}$ are selected as user’s focus. The three new squares together with $\{R_0, R_1, R_2, R_3, R_4, R_5\}$ compose the new stub, a new region to test. Component $\{R_6, R_7, R_8\}$ is also of the user’s focus, but has not been a part of stub. If any one of $\{R_6, R_7, R_8\}$ has not been used for a period of time, system will destroy it from the user’s focus. (b) Top-right orientation is chosen, and five new squares that adjacent to the top-right corner are selected as user’s focus. The five new squares together with $\{R_0, R_1, R_2, R_3\}$ compose the new stub.

4 Effectiveness of the Prediction Algorithm

To test the effectiveness of the algorithm, an experiment was carried out. Ten volunteers (5 male, 5 female) from local University were invited to our lab. The experimenters were divided into 5 groups (2 persons for each group). For each couple, one person drew on desktop PC and the other viewed the shared pattern on Microsoft Pocket PC emulator. The desktop used in our experiment was Dell PC running our collaborative design system, using a 17-inch monitor set to 1024x768 resolution,

512M memory and 2.4G CPU. While each couple was drawing on PCs and viewing on the emulators, the prediction algorithm predicted the future focus region. The length of the small grid's edge is set to twentieth of the width or the height of the canvas. The experiment lasted half an hour for each group.

As for designers who edit their graphics documents, the focus prediction accuracy can be calculated as follows:

$$\text{Focus Prediction Accuracy} = \frac{\text{Total Correct Predictions}}{\text{Total Predictions}} \times 100\%$$

Where, if designers do click on the $\text{FocusConU}_k^{\text{region}}$ that is predicted, the prediction is deemed as a Correct Prediction.

PDA users' interest region prediction accuracy depends on users' evaluation. They grade the accuracy according to the proportion between the predicted focus region and the region that they really like to view.

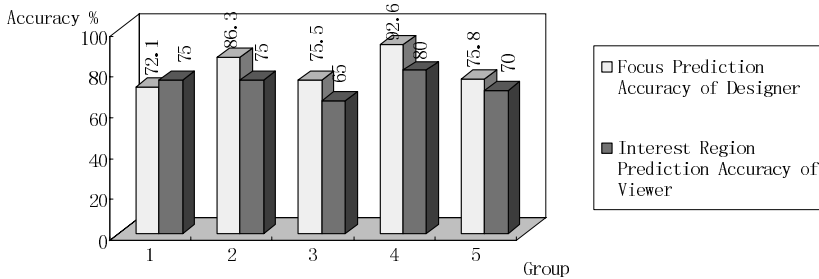


Fig. 5. Accuracy of focus prediction for designers on PC and accuracy of interest region prediction of viewers on PDA

As it is shown in Fig.5, the average accuracy for focus prediction is about 80.5%, while for interest region prediction the average accuracy is approximately 73%. From the experiment, it is obvious that the focus prediction algorithm might represent users' intension in most cases. Moreover, as the accuracy of PDA user's interest region prediction is close to that of PC user's focus prediction, the algorithm can work well for predicting PDA users' viewing interest. And also, users viewed shared patterns on PDA found that it was convenient to participate in collaborative graphics design with the function of user interest prediction.

5 Conclusions

Extracting and predicting mobile user's viewing interest region and push the related design information and collaborative editing operations to the resource-deficient embedded sites is proved to be an efficient way to provide good awareness to the mobile collaborators. In this paper, algorithms that used to predict the user interest region for portable devices in Internet-based real-time GDS are presented. The study shows that the algorithms we proposed can forecast mobile user's intention and provide mobile users with awareness information on collaborative workspace exactly what they want.

By introducing the user interest region prediction, the usability and the efficiency of the GDS in ubiquitous environment is improved dramatically. In future work, we will study on more sophisticated prediction scheme to improve the accuracy of the prediction.

Acknowledgements

This paper is supported by National Natural Science Foundation of China (60573176) and Zhejiang Provincial Natural Science Foundation of China under Grant No. Z603231. Thanks to the volunteers for taking part in the experiments.

References

1. Bo Jiang, Jiajun Bu, Chen Chen, Jianxv Yang: Remote Control Point Motion Prediction in Internet-Based Real-Time Collaborative Graphics Editing Systems. In: Proceedings of 11th International Conference on Groupware (2005) 137-144
2. X. Wang, et al.: Achieving Undo in Bitmap-Based Collaborative Graphics Editing Systems. In: Proceedings of 2002 ACM Conference on Computer Supported Cooperative Work (CSCW'02), November 16-20. New Orleans, Louisiana, USA (2002) 68-76
3. C. Sun and D. Chen: Consistency Maintenance in Real-Time Collaborative Graphics Editing Systems. *ACM Transactions on Computer-Human Interaction*, 9,1 (2002) 1-41
4. Stefik, M., Bobrow, D.G., Foster, G., Lanning, S. and Tatar, D.: WYSIWIS revised: Early experiences with multiuser interfaces. *ACM Transactions on Office Information Systems*, 5(2), April. (1987) 147-167
5. Chen, L.Q., Xie, X., Fan, X., Ma, W.Y., Zhang, H.J., and Zhou, H.Q.: A Visual Attention Model for Adapting Images on Small Displays. *ACM Multimedia Systems Journal*, Vol. 9, No. 4, Oct. (2003) 353-364
6. Liu, H., Xie, X., Ma, W.Y., and Zhang, H.J.: Automatic browsing of large pictures on mobile devices. *ACM Multimedia 2003*, Berkeley, CA, USA, Nov. (2003) 148-155
7. Yu-Fei Ma, Hong-Jiang Zhang: Contrast-based Image Attention Analysis by Using Fuzzy Growing. In: Proceedings of MM'03, November 2-8, Berkeley, California, USA (2003) 374-381
8. Xing Xie, et al.: Learning User Interest for Image Browsing on Small-form-factor Devices. In: Proceedings of CHI 2005, Portland, Oregon, USA (2005) 671-680

A Conceptual and Methodological Framework for Modeling Interactive Groupware Applications

A.I. Molina, M.A. Redondo, and M. Ortega

Department of Information Technologies and Systems.

Computer Science and Engineering Faculty.

Castilla – La Mancha University.

Paseo de la Universidad, 4. 13071 – Ciudad Real. Spain

{AnaIsabel.Molina, Miguel.Redondo, Manuel.Ortega}@uclm.es

Abstract. In the last years the production of systems supporting learning and work in-group has been high. However, the design and development of this kind of systems is difficult, especially due to the multidisciplinary involved. We propose a design and development process of the presentation layer. This process is based on the use of several models for representing collaborative and interactive aspects of this kind of systems. In this process several techniques and notations are used. In this paper we introduce our methodological approach and the conceptual framework on which our proposal is based.

Keywords: conceptual framework, groupware design, interaction design.

1 Introduction

The use of applications framed inside the paradigm of *Computer Supported Collaborative Work* (CSCW) is increasing, mainly due to the widespread diffusion of applications based on the Web. On the other hand, User Interface (UI) is one of the elements that is acquiring greater attention on these days. CSCW systems development is not a trivial task due to the multidisciplinary of such systems. Problems generated in this kind of applications come mainly from three areas: the social nature of these systems, problems in the field of distributed systems and problems related to Software Engineering. Cooperative behaviors modeling support or shared information workspaces are becoming requirements to take into account when developing these systems. Studying the existing alternatives for CSCW systems modeling [1, 2, 3, 4] we have noticed certain deficiencies in modeling the collaborative aspects, particularly, proposals that combine group work applications aspects and interactive aspects. In the same way, those that are especially dedicated to the modeling of the interactive aspects of the applications do not usually give support for modeling collaborative aspects.

These problems confirm and justify the lack of a *conceptual and methodological framework* supported by a coherent *set of notations* for designing interactive and collaborative tools. We have defined a notation called CIAN (*Collaborative Interactive Applications Notation*). This notation is a simplification of another notation for *workflow* modeling, called APM (*Action Port Model*), proposed by Carlsen [5]. This notation has been enriched to support a differentiated modeling of cooperative and

collaborative tasks, while it has been simplified in some aspects (to characterize a task just a task identification, the roles involved and the objects manipulated are included). There are differences between cooperation and collaboration, pointed out by Dillenbourg [6], which must be considered. These differences affect the division of tasks, the roles participation in the tasks and the obtained product as a result in a joint activity. CIAN might be used in a methodological framework for designing groupwork systems.

Some theoretical models for the development of *groupware applications* have been proposed in the literature: the *Group Task Analysis* framework (GTA) [7], the ontology proposed by Barros for the design of CSCL (*Computer Supported Cooperative Learning*) applications [9], based on the Activity Theory [7]. A survey of some of the main cooperative models and theory proposals can be found in [10, 11]. However, there are no proposals that allow combining concepts relative to the design of groupware systems and their more interactive part.

We believe that the employment of an ontology is of great utility as the basis for the definition of our methodological proposal. By means of the definition of a *conceptual framework* we can clarify the concepts that will be managed and modeled, as well as the relationships that exist among them. This conceptual framework can be defined by means of an ontology. This ontology will serve as basis for the definition of the modeling techniques that can be used in each of the phases of our methodological approach. It captures the concepts whose semantics is interesting to represent by means of the group of views and notations that can be used in the various stages of our proposal. This conceptual framework must include concepts relative to interactive and collaborative aspects.

In this paper we introduce our approach methodology to develop the presentation layer of a collaborative system, describe the conceptual framework in which it is based and point out the relationship between them. For this, the paper is organized in the following way: section 2 introduces our methodological approach for designing interactive groupwork applications, presenting a brief explanation of its stages and the aspects that can be specified in each. Section 3 explains the conceptual framework on which our methodological framework is based. Also, some aspects of the CIAN notation are described in this section. Finally, the conclusions obtained are explained.

2 Methodological Framework

In this section we present the stages in our methodological approach, named *CIAM (Collaborative Interactive Applications Methodology)*. This methodology is based on a conceptual framework, which is described in section 3. Our proposal implies adopting different viewpoints for creating models of this kind of systems. The first stages undertake a group-centered modeling, going on in subsequent stages to a process-centered modeling (cooperative, collaborative or coordination process), approaching, as we go deeper into the abstraction level, a more user-centered modeling, in which interactive tasks are modeled, that is, a dialog between an individual user and the application. Two first modeling approaches describe the context [8] in which the interactive model is created, and serve as starting point for the last one. In this way, collaborative aspects (groups, process) and interactive (individual) modeling

problems are tackled jointly. These framework acts as a guide for designers to create conceptual specifications (models) of the main aspects that define the presentation layer in CSCW systems. Specified information in each stage serves as a basis for modeling in the following stage. This information is extended, related or specified in a more detailed way in the next stage in the process. The stages in this proposal (see figure 1) and their objective are enumerated as follows and, in the next section, the conceptual framework proposed is explained.

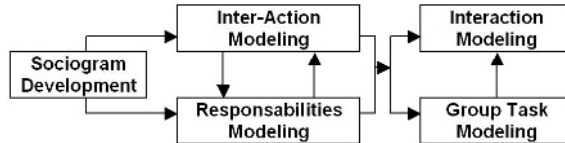


Fig. 1. CIAM methodological proposal stages

- **Sociogram Development.** In this phase, the organization structure is modeled, as well as the relationship between its members. Organization Members are in one of those categories: *roles, actors, software agents*; or in aforementioned associations, forming *groups* or *work teams*, consisting of several roles. The elements in those diagrams can be interconnected by means of three kinds of basic relationships (*inheritance, performance* and *association*).
- **Inter-Action Modeling.** In this phase, the main tasks (or processes) that define group work in the previously defined organization are described. For each process, 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*. Processes will be interconnected by means of several kinds of relationships that, in several cases, can be interpreted as dependences.
- **Responsibilities Modeling.** In this phase, attention is payed to the individual perspective of each organization member (role), adding to their shared responsibilities the ones which are exclusive for them. We can see that the specified information in this phase is supplemented with the previous one. It is necessary that both models be consistent to each other.
- **Group Tasks Modeling.** In this 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 modeled in a differentiated way: **(a) Cooperative Tasks** are specified by means of the so-called *responsibilities decomposition graph*, in which subtasks make up the group task, so that, in a lower abstraction level only an *individual task* must appear. **(b) Collaborative Tasks** modeling includes specification of the roles involved, as well as the data model objects manipulated by the work team (that is, the *shared context* specification). Shared context is defined as the set of *objects* that are visible to the users set, as well as the *actions* that can be executed on them. Once the objects that make up the *shared context* have been decided, it is necessary to fragment this information in 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).

- **Interaction Modeling.** In the last phase interactive aspects of the application are modeled. An *interaction model* for each individual task detected in the diverse phases of the gradual refinement process is created. A *interactive tasks decomposition tree* in CTT [9] is developed. 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 modeling [10].

3 Conceptual Foundations of the Methodological Framework

In this section our conceptual framework is presented. This framework is used as basis in our methodological proposal, with the aim of capturing the semantics and the main concepts used in the process of development of the presentation layer in CSCW systems. The proposed metamodel must include concepts for describing all the various perspectives to be considered for the definition and construction on this kind of environments (global view of the work flow, organizational perspective, data view, interaction perspective). This ontology can be splitted in several parts, one for each viewpoint that is modeled in each of the phases that compose our methodological approach. The overall metamodel proposed as basis of our methodological approach is shown in figure 2. We use the *Unified Modeling Language* (UML) to capture the main concepts in our model. We can identify four views that are presented in this section. Next, the views that compose our conceptual framework are explained:

- **The organizational view.** This view includes the concepts of *role* and *actor*. An *actor* can be specialized into *Group* and *Software Agent*. A *Group* is composed of several actors. An actor can play several roles and a role can be played by several actors. A *Work team* is formed by a set of roles that work in a combined way in the realization of a certain group task. A *role* can include subroles. An organization is formed by the group of roles that must be supported by the groupware tool to be designed. All these concepts are used in the *sociogram development* phase, and must be represented by the models created in this methodological stage. In figure 2.A we can see the organizational view of the metamodel. Figure 3 shows the visual aspect that a *sociogram* represented in CIAN notation presents. We will not go deeply into the description of the diagram on figure 3 because it is not the main objective of this paper. A detailed description of the CIAN notation and examples of its application can be found in [11].
- **The inter-action view.** This is the sub-ontology that allows representing the *inter-action model*. The models generated in the *inter-action modeling* phase specify the main tasks and processes in which the organizational *roles* that the CSCW system to be designed will use are involved. In this model not only the workflow between tasks is included, but also the context in which the group works are carried out is shown graphically, by means of the *roles* and the *objects* of the domain model involved. The root concept of this view is the *task*. A task is an activity performed by actors to reach a certain goal. Complex tasks can be decomposed into smaller subtasks. The *tasks*, which can be of *individual*, *cooperative* or *collaborative* nature, are related to each other by different kinds of

relationships: *time relationships*, *data dependence* and *notification*. In figure 2.B we can see the *inter-action* view of the metamodel and figure 4 shows an example of *inter-action* model in CIAN notation.

Figure 4 shows the *Unstructured Decision-Making Process*, also called *Brainstorming*. The *inter-action model* allows specifying the complete operation of the

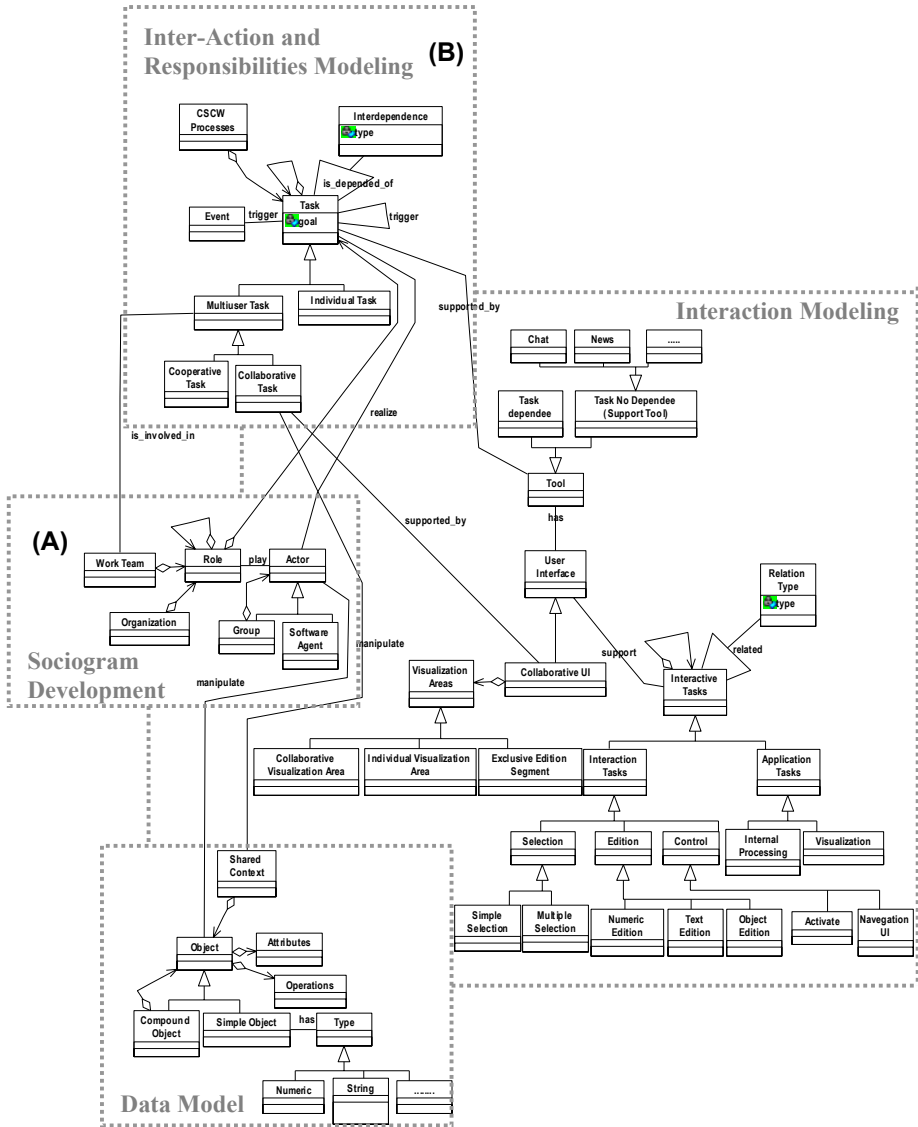


Fig. 2. The overall conceptual model for interactive groupware systems development

group process that can be cooperative, collaborative or mixed. This diagram is represented by means of a graph whose nodes are the states and the arcs are the transitions among states. Each state of the diagram is represented by means of a node in form of rounded rectangle that contains three parts with the following information: (a) The head of the state includes the *task name* (on the right) and its type (on the left). To indicate the type we use the icons displayed in the table shown in figure 4.a. (b) In the left-hand lower corner the roles involved in the execution of this task are enumerated. (c) In the right-hand lower corner the objects manipulated by the task are shown, with *access modifiers* (*L* stands for *Readable*, *E* for *Writable* and/or *C*) at task level.

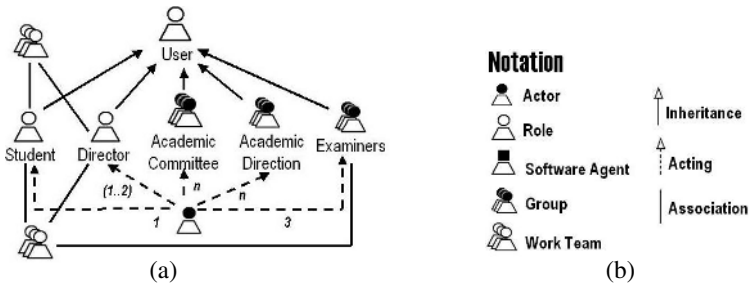


Fig. 3. An example of *sociogram* (a) represented in CIAN Notation (b)

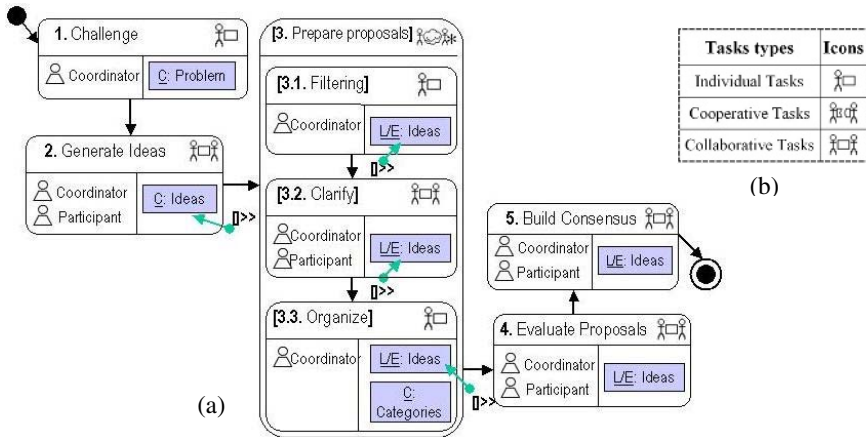


Fig. 4. An example of an *inter-action model* in CIAN notation. This model represents a *brainstorming* process. Types of tasks handled by notations used in the methodology stages (a).




The CIAN notation also allows representing abstract tasks, that is, group work tasks that can be decomposed into others in a lower level of abstraction and of different kinds. It also supports the specification of optional tasks (their names appear in brackets, []).

- **The data view.** In this view we include the data manipulated in the process. *Objects* are entities that are manipulated to perform *tasks*. An *object* has *attributes*

and *operations*. Furthermore, objects may be in a type hierarchy and can also be contained in other objects. The data view, in our proposal, is specified by means of the UML notation (using a class diagram that represents the data manipulated in the collaborative process).

- **The interaction view.** In this view we specify the interactive aspects of the groupware applications, that is, the issues related with the user interface. The root concept of this view is the *interactive task*. An interactive task can be classified into two types: *Application Tasks* and *Interaction Tasks*. *Application tasks* are tasks completely executed by the application (that is, visualize information or an internal processing). An *interaction tasks* are tasks performed by the user interacting with the system. We consider three interaction techniques as basic: selection, edition and control. The *User Interface (UI)* supports *Interactive Tasks*. A *collaborative user interface* is a specialization of UI. In a collaborative user interface we can identify three main visualization areas: *collaborative visualization area*, *individual visualization area* and *exclusive edition segment*. It can happen that a model does not include an area of individual visualization. This indicates that we are in a situation in which all the members that collaborate can see exactly the same objects.

Table 1. Icons for representing visualization and access features to the shared context

| Icon | Definition |
|---|--|
|  | Area of the shared context for collaborative visualization |
|  | Area of the shared context for individual visualization |
|  | Segment of the shared context for access of exclusive modification |

To model the interaction view, a notation exists broadly diffused in the community of the Computer Human-Interaction. This language is CTT [9], which we have already mentioned previously. Using CTT, the models built have a hierarchical structure, which allows representing several levels of abstraction. We have enriched this notation with three new icons that represents the three visualization areas aforementioned (see table 1). These icons are used as roots of the subtrees in the interaction tree in CTT notation separately: (a) the subtree that represents the interaction with the shared context that is common for all the members of a group involved in a group task (*collaborative visualization*), (b) the interaction of individual nature for each member (*individual visualization*) and (c) the subtree that specifies the dialog with the area of the shared context that only can be accessed by one member of the group at a time. Using CTT we can reach high levels of detail in the interaction model. This facilitates obtaining the final design of the user interfaces. Using our extension of CTT we can identify additional information about the areas that compose the collaborative user interface. Thus, this extension has a higher-level semantics which organizes and expresses in a better way specific aspects of collaborative applications.

4 Conclusions

The user interface of collaborative applications is an important aspect in order to support effective work in-group. Model-based design is an extended technique in the user interface development process. Reviewing approaches that deal with the modeling and development of user interfaces supporting collaborative tasks, we have detected that there is not a proposal that links interactive and collaborative characteristics. We have introduced a methodological approach for solving this lack. This proposal consists of a set of phases, in each of which models are created in a specific notation. With the aim of creating a coherent set of notations we have based them on a conceptual framework, that is, an ontology. We have used the conceptual framework definition to clear the concepts and the relationships among these. This ontology has served as basis for our methodological approach to define and design interactive groupware systems.

Acknowledgements

This work has been supported by the Castilla – La Mancha University and the Junta de Comunidades de Castilla – La Mancha in the project GAMTest (PCI-05-005).

References

- [1] J. L. Garrido, "AMENITIES: Una metodología para el desarrollo de sistemas cooperativos basada en modelos de comportamiento y tareas," in Dpto. LSI. U. de Granada, 2003.
- [2] R. Hourizi, P. Johnson, A. Bruseberg, and I. Solodilova, "Modelling Collaborative Work in UML". 21th European Conf. on Human Decision Making and Control, Glasgow, 2002.
- [3] F. Paternò, C. Santoro, and S. Tahmassebi, "Formal model for cooperative tasks: Concepts and an application for en-route air traffic control". DSV-IS '98, Abingdon, 1998.
- [4] M. van Welie and G. C. van der Veer, "Groupware Task Analysis," in Handbook Of Cognitive Task Design, E. Hollnagel, Ed. LEA., NJ, 2003, pp. 447-476.
- [5] S. Carlsen, "Action Port Model: A Mixed Paradigm Conceptual Workflow Modeling Language". 3er IFCIS COOPIS 1998.
- [6] P. Dillenbourg, M. Baker, A. Blaye, and C. O'Malley, The Evolution of Research on Collaborative Learning, vol. Learning in humans and machines. Towards an interdisciplinary learning science. London, 1995.
- [7] B. A. Nardi, Context and Consciousness. Activity Theory and HCI, 1996.
- [8] H. Beyer and K. Holtzblatt, Contextual Design: Defining Customer-Centered Systems. San Francisco: Morgan Kaufmann Publishers, 1998.
- [9] F. Paternò, C. Mancini, and Meniconi., "ConcurTaskTree: A diagrammatic notation for specifying task models". Interact'97, Sydney, 1997.
- [10] A. I. Molina, M. A. Redondo, and M. Ortega, "CIAM: Una Aproximación Metodológica para el desarrollo de Interfaces de Usuario en aplicaciones groupware". INTERACCION 2006, Puertollano (Spain), 2006.
- [11] A. I. Molina, M. A. Redondo , and M. Ortega, "A methodological approach for designing collaborative and interactive applications: a practical example," Software - Practice and Experience (SPE) Journal, 2006.

Collaborative Design and Tailoring of Web Based Learning Environments in CURE

Mohamed Bourimi

FernUniversitaet in Hagen

Department for Mathematics and Computer Science - Cooperative Systems

58084 Hagen, Germany

mohamed.bourimi@fernuni-hagen.de

Abstract. The majority of existing web based learning platforms does not offer a flexible way to design learning environments. They support only a fixed or restricted view on the execution of learning processes. Furthermore, the adaptation and tailoring of learning environments by end-users is rarely supporting different learning scenarios and processes. Both, design and tailoring are usually supported as activities of individual users. In order to support collaborative design and tailoring, a collaborative process was developed and implemented prototypically in the collaborative learning platform CURE. The process supports collection and categorisation of existing designs as templates in a shared central repository, providing searching and rating mechanisms as well as awareness facilities to facilitate reuse of templates and to contribute to the extraction of *best practices*. First experiences show the advantages and expansion potentials of this approach. The applicability of the process and its portability to other platforms is discussed.

Keywords: Collaborative design, collaborative tailoring, CSCL environments, CURE, knowledge sharing, shared artefacts, tailorability

1 Introduction

Web based learning platforms are more and more used worldwide at many universities, educational institutions and large organisations in their learning and further education programs. The use of such platforms is playing a primary role at universities for distance learning and distributed institutions. Instructors and teaching staff build up their learning environments using available possibilities of the respective platform with the aim to reduce redundancy and time scalability problems and learning location dependencies as well as communication problems. This results in setting up shared spaces or learning rooms and making them available to the students. In such workspaces, learning materials, communication means and coordination facilities are made available according to the organisation's and learner's needs to perform learning activities.

The support of the intended learning scenario is one of the main requirements. Performing a learning scenario in a learning platform is reflected in the organisation and configuration of the workspaces, contents and tools of this platform. Existing

platforms mostly lack flexibility with regard to design and tailoring support of the learning environments, as they have fixed or restricted views with respect to organisation and realization of learning activities. As a typical example for this, the class oriented view of many commercial learning platforms has to be mentioned. Multiple forms of learning, such as seminars and different forms of practical and lab courses are not supported sufficiently. In this paper, three important needs are introduced and presented, which must be addressed for a flexible adaptation of learning environments: End-user tailorability before and during the learning process, support for simple synchronous collaborative design, and facilitation of identifying and reusing best practices.

In this paper, an approach is introduced which tries to fulfil the above mentioned needs when supporting collaborative design and tailoring of learning environments. Section 2 provides a problem and requirements analysis. Section 3 discusses the state-of-the-art, while Section 4 gives a short introduction of the learning platform CURE. Section 5 presents our approach with a practical scenario. Section 6 explains the architecture and implementation of the prototype. Section 7 reports on initial evaluation results. Section 8 concludes the paper with a summary and future work.

2 Requirements Analysis

Flexible adaptation and tailoring of learning environments pose three major needs to the designers and users of learning platforms:

- N1. **End-user tailorability before and during the learning process:** Learning scenarios have a high variability with respect to their individual design and organisation. As an example, the learning activities of the same course can be performed in several ways within the same learning environment. This high variability is often not taken into account, especially when a combination of special learning scenarios (e.g., a course integrating group work and individual learning phases as well as assignments and peer-review) should be performed, which requires a dynamic adaptation of the environment during the learning process by end-users (e.g., creating ad-hoc group workspaces, or discussion forums for peer-review). Many courses at the FernUniversität use a blended learning approach. Such courses possess a high portion of self-organized teamwork. E.g., in different phases of a lab course, different groups are formed, through both instructors and students. The collaboration among these groups takes place in different areas with different participants, settings and organisations of activities. Our experiences show that the interactions within a group can in many cases only be specified by the group itself. End-users must therefore be enabled to build up their own environments and to tailor them according to their needs, in a flexible way before as well as during the learning process.
- N2. **Support for simple synchronous collaborative design:** An analysis of the design activities (primarily in web based learning platforms) points out that several interactions are needed in the corresponding graphical user interfaces (GUI) in order to carry out a single design task. When these GUIs are tedious

to use, end-users can feel overloaded due to complex designs and configurations. If the designed features offered by the learning platform cause a considerable load, this feeling increases. Experiences show that documentation and trainings are already needed for elementary functions to motivate and guarantee a successful use of the platform [29]. Additionally, our experiences show that the need for support of further forms of collaboration becomes substantial at the design and tailoring level. This support must cover synchronous aspects of distributed collaboration, too. The faculties and departments of the FernUniversität in Hagen are distributed over the city. Furthermore, collaboration takes place with external employees and mentors, which are distributed across different cities. Since, in our case, the collaborative learning platform CURE is used in several courses of different disciplines, an online support program for the users is needed when designing the activities in their learning environments. Furthermore, teaching staff of the same department (possibly at the same location) sometimes need the possibility to create or adapt learning environments synchronously, as such design and tailoring decisions must be carried out in order to respect the agreements of co-teachers and the given deadlines. These examples show that synchronous facilities are needed for the design and organization of a learning environment, which may only requires little effort and time.

- N3. Facilitation of identifying and reusing best practices:** Existing learning platforms are not directly confronted with the needs mentioned above, because, on the one hand, they mainly offer predefined templates (e.g. course or classroom templates) and, on the other hand, they do not support flexible design and tailoring by (non-privileged) end-users. The reuse of existing organisations and configurations is not always in the foreground either. The collaborative construction and manipulation of designs focuses predominantly on (learning) content. Thus, identifying best practices and extracting use-patterns from previous teaching and learning activities within a given learning environment becomes a tedious task for interested users and communities. Therefore, mechanisms for facilitating and encouraging such activities, under consideration of inexperienced and unskilled users, are also needed.

Flexible and collaborative design and tailoring, with participation of the end-users while organising and performing learning scenarios, is the key to fulfil the needs N1-N3. As a starting point, a web based learning platform is presupposed whose base services can be called through a programming interface (API). A domain model represents the learning environments running in the learning platform. Calls from this API lead to model changes in the respective platform and consequently result in a manipulation of the learning environments. To support the collaborative design and tailoring of learning environments the following three requirements groups are identified:

- R1.** Organising and performing different learning scenarios shall not be limited. End-users (teaching staff and students with sufficient rights) must be flexibly and interactively allowed to change their environments (an instance in use of their course designs and configurations). Changed course designs and

configurations must be executable and must change the behaviour of the underlying platform in a permanent manner, and on demand. The mechanisms provided for this purpose shall be simple to use with as low as possible effort.

- R2. Synchronous and asynchronous cooperation shall also be supported. It shall be possible for the user to create new designs synchronously or asynchronously either with other users or individually. The adaptation of existing course designs and configurations shall be supported, too. Essentially, the support of synchronous cooperation represents a fast and efficient way to reach consensus between the users involved by cooperative design and tailoring activities. Furthermore, synchronous and asynchronous cooperation and communication contributes to the exchange of expertise and know-how and provide new possibilities for online support.
- R3. Resulting course designs and configurations, seen from the didactical perspective, represent a valuable source of knowledge that could be shared among the users. Providing categorisation, preview, import and export as well as searching facilities on course designs and configurations promote cooperation and knowledge exchange. The possibility for the change and expansion of this knowledge shall be given to support reusability and organizational improvement processes. Use of ranking, communication and awareness facilities shall motivate the users and facilitate the formation of a *community of practice* [1].

3 State-of-the-Art

Tailoring is investigated in many works as key design requirement of collaborative systems [20][24][25]. Indeed, this term is differently conceived and taken into account in those systems. First definitions emerged in these researches have to be classified in the CSCW¹ and HCI² fields, where tailoring is described as the human and technical way to adapt a system during its use [11]. Tailoring is carried out in the first place by end-users with the aim to reach better support on the system side in the respective task domain. This can be carried out collaboratively between end-users [29] and includes developer [23]. Furthermore the cooperation can take place at different levels and with different refinements [22][29]. Seven mechanisms were suggested in [15] to better support collaborative tailoring. The interesting aspects in the context of the approach described here were taken into account in the requirements analysis in the last section (e.g. sharing artefacts among users, providing awareness information about these artefacts). In the following, collaborative tailoring is closely considered by learning platforms with regard to design and tailoring support of learning environments with consideration of the considered requirements.

Commercial learning platforms like WebCT [27], BlackBoard [4] and LearningSpace (LS) [19] pursue an oriented classroom view on the organization of the learning environment and provide templates for the course design. Therefore,

¹ Computer Supported and Collaborative Work.

² Human Computer Interaction.

different learning forms, which do not correspond to this view (like lab courses), are not supported sufficiently. If one takes into account this case where only privileged users (e.g. instructors or course administrators) may be able to adapt the learning environment, then fulfilling the requirement R1 at such platforms becomes difficult, namely by assigning additional rights to the end-users. Mechanisms for importing and exporting course configurations are available with different stage. So LS makes for instance a selective choice possible with preview support. However, these are intended only for the individual use. A cooperation focus cannot be recognized so that the requirements in R3 are partially fulfilled and R2 are not fulfilled at all.

Free available learning platforms such as BSCL [2] and Moodle [21] also allow manipulation of their learning environments only by privileged users. While BSCL uses shared workspaces to support learning activities, Moodle provides course rooms. In contrast to BSCL, Moodle supports a selective importing and exporting of course configurations and provides predefined course templates. Further customizations of the learning environments in Moodle could occur by the activation of pluggable modules (e.g. chat). Support of collaborative design and tailoring within the Moodle community is provided via moderated course exchange areas. Two basic principles have to be taken into account: First, allowed participants could only be Moodle-users, who offer at least three own contributions (in form of course activities) and, secondly, renounce thereby to all copy rights. A non moderated variant is the course exchange area [21], where it is possible to exchange course activities and rank them. Searching for categories and key words is also possible. Though, these mechanisms require for our aims further improvements. Thus, course descriptions and a preview function are not available. In addition, course activities are provided as archives which presuppose a running Moodle instance and privileged permissions to import them. Synchronous aspects also remain unconsidered so that a similar coverage of the requirements R1-R3 arises for this category like the case of commercial platforms.

Another interesting learning platform is GridCole [5] although the web based aspect is only partly covered. The access web server is mainly used for the authentication of the users, provision of common documents and for the download of the client software. The latest one is a thick-client software which allows users to join running collaborative learning activities on the server. The basic idea of GridCole is based on the support of learning scenarios that are described in IMS LD [12] and meant to provide environments in order to reference and perform different services (e.g. voting or chat services) that are distributed in the network. An IMS LD script describes the learning method in form of one learning flow specified by the instructor, who determines participants and configures the required resources manually. Similarly to the case of the learning platform L³ [28], the environments cannot be changed dynamically (at runtime). Also, learning environments are available only during the learning phases episodes. Thus, entering such environment before the beginning or after the end of a learning activity is not possible. In addition, learning activities can only be initiated by a teaching staff.

Symba [3], a web based framework designed to support collective activities in a learning context allows for synchronous design and tailoring of the learning environments by end-users. Thereby, the first concern is to make students explicitly work on their organisation in a learning context by describing it as a plan composed of tasks. After having defined a given task (description, individual or collective

nature, needed tools etc.) the platform generates an adequate learning environment to perform it. Tailoring by (re-)definition can take place during the learning process, too. However, those environments are only available as long as the required phases (a set of tasks) of the corresponding learning contexts (plans) are performed. A collection of best practices could take place, in our opinion, on the level of such plans and task definitions if a representation form is provided to make them persistent. We have no information if this has happened.

In summary, none of the platforms mentioned above covers all requirements. The requirements R1 and R3 are only partially fulfilled, while collaborative design and tailoring remain mostly unconsidered. Course templates are often available in form of (zipped) archives and are intended more for the individual use. The synchronous aspects of the requirement R2 is not fulfilled within all examined platforms except Symba. Approaches which focus on collaborative tailoring can be found in the CSCW field (e.g. [29],[15]). However, these were rarely used in the CSCL context.

4 CURE in a Nutshell

CURE is a collaborative learning platform used at the FernUniversitaet in Hagen to support collaborative learning [10]. CURE uses the room metaphor [7] in order to model shared workspaces for groups. Virtual keys determine access permissions and allowed interactions in the room [8]. Figure 1 shows the conceptual design of the CURE learning platform.

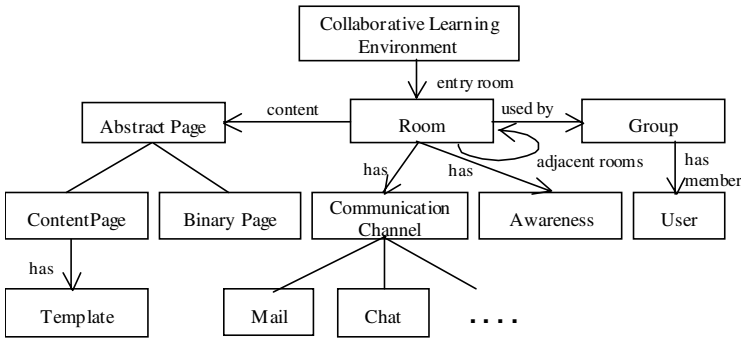


Fig. 1. Conceptual design of CURE

Users who have keys to a certain room can cooperate and work between each other and therefore form a group. A room in CURE disposes of synchronous and asynchronous communication channels (chat, threaded mailbox). Furthermore, rooms provide learning materials for group members in the form of pages. These can be regular content pages containing text or those pages, which include binary data such as word or pdf documents. For the creation and asynchronous edition of content pages, a wiki editor [16] is used. Changes edited and stored with the help of this editor are immediately visible to other users. A versioning system helps by the resolution of conflicts and managing of previous pages versions. Interaction rights are

defined, as mentioned above, by the keys of the user. Hence for example some participants may not be able to use the communication means or could only view the pages but not edit them.

Since the contents are permanently stored within the room, users can work asynchronously. Furthermore synchronous and asynchronous awareness information is provided, which makes cooperation and coordination easier. So for instance users can see who is actually in the room. Daily mail reports on changes and actions of other group members are an example of an asynchronous awareness.

Structuring learning environments is carried out by connecting rooms. A special room, called Hall, represents the central access point to the learning portal with its learning environments. From the Hall, adjacent rooms can be recursively reached. The navigation between the rooms is supported via a global room list and a graphical room maps. Both representations are generated and shown for every user depending on his keys (e.g. users can see only rooms to which they have access). In addition, every room possesses a navigation bar (a drop down menu) which shows its parent and its children. Furthermore, users can support the navigation creating links in content pages, using simple wiki syntax. Links to other rooms and other pages are also supported. Clicking on a link, results in moving to the corresponding room or page. In the case of a room, a particular page (the welcome page of this room) will be displayed.

Users who disposes about sufficient rights, like creating adjacent rooms, passing on or copying their keys, and editing the content, can therefore at any time adapt the environments according to their needs (also during the learning process) as well as initiate different kinds of group formation (cf. to this the explanations of the topic tailoring [9] and end-user controlled group formation [8]). The feasibility of the requirement with respect to better support of end-user tailoring in R1 will be fulfilled by choosing CURE as a target platform.

5 Approach

To meet the requirement categories (R1-R3) identified in Section 2, the approach introduced here consists of a process for collaborative design and tailoring of learning environments, and its support through an extended collaborative learning platform. Each step of the proposed process is supported by one or several components of the extended collaborative learning platform. In the following, we will first present the proposed process, followed by the description of the architecture of the proposed extended collaborative learning environment and its major components.

5.1 A Process for Collaborative Design and Tailoring of Learning Environments

Figure 2 shows our process that suggests different steps (nodes of the diagram) of collaborative design and tailoring of learning environments. The transitions (arrows in the diagram) between the steps are triggered by an end-user's need in a given use situation (e.g. an adaptation or tailoring need) or his/her intention to reach certain goals (e.g. sharing designs or reflecting on them with others). A specific collaborative

design and tailoring of a learning environment is performed through a sequence of steps according to Figure 2. The usual begin of a tailoring is the normal use of a learning environment in a given learning platform (A), in our case CURE. If the need to tailor the environment arises, a tailoring and use step occurs based on the tailoring capabilities of that environment (B). Interested parties with sufficient access rights can decide to extract a template from a running learning environment instance (C) or they can create a new template (D). A given template can be manipulated, e.g., in order to address the needs of a specific learning situation, or of a different learning scenario (E). This manipulation can take place individually or in a cooperative manner. Once the design and tailoring of a template is done, the designer(s) can apply the resulting design templates in their learning environments (F), e.g., to create new learning environments or to modify existing ones. In addition, designers may make design templates public (G). This encourages interested users (i.e. members of CoPs) to exchange knowledge collaboratively through activities (such discussions, annotations etc.) on shared design templates (H) and to reuse them if they want, e.g., for creating another template (E) or for creating/modifying learning environments (F).

In our approach, we propose to distribute these process steps across three different environments: the *learning platform* addressing steps A and B, a *collaborative design and tailoring environment* addressing steps C, D, E, and F, and a *community portal* addressing steps G and H. This separation of concerns is useful in order to allow (1) design and tailoring without directly affecting running learning environments, (2) flexible involvement of third-party users (e.g., experts or newcomers needing training) who do not possess access rights to the real learning environment, and (3) discussion, sharing, and reuse of templates across learning platforms.

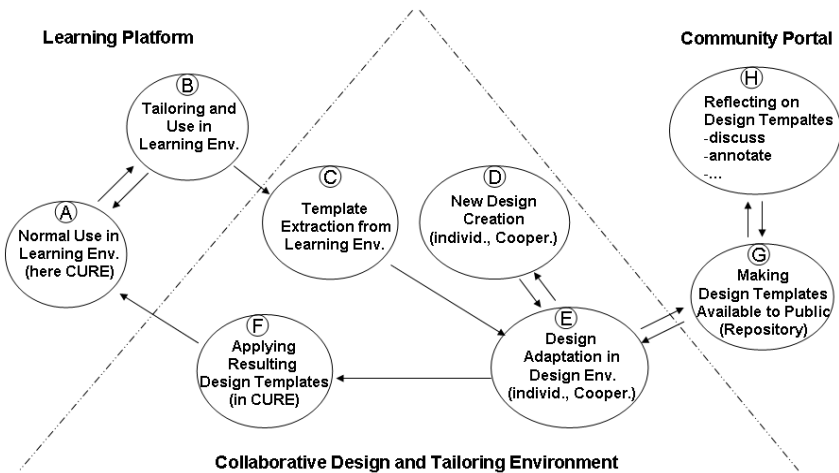


Fig. 2. A process for collaborative design and tailoring of learning environments

Since our process does not prescribe neither a certain begin nor a specific sequence of steps diverse combinations of steps may occur. Thus, it is more adequate to explain how the process solves the problem of collaborative design and tailoring of learning

environment by demonstrating its use with the help of a practical scenario. For this purpose, the following section first defines the architecture of an environment supporting the process, and then explains how the process steps and transitions are supported by the major components of the architecture of our extended collaborative platform showing thereby how a given requirement from Section 2 is fulfilled. Additional functionalities not addressed explicitly by the process, such as means to store changes locally or community-wide as well as awareness and communication support that ease the accomplishment of such steps, are mentioned, too.

5.2 A Collaborative Learning Platform Supporting Collaborative Design and Tailoring of Learning Environments

The architecture of the environment supporting the process for collaborative design and tailoring proposed in the last section is depicted in Figure 3 as a block diagram. Since CURE and the collaborative tailoring portal (community portal) are realized as web applications, each made available through its own web server (Tomcat), the end-users can access them using a typical web browser. The collaborative design and tailoring environment (see Figure 2) is provided by the cooperative design editor, a thick client application. Being able to communicate with CURE and the community portal (bidirectional arrow), the design editor acts as a mediator between them, allowing the transfer of course design templates in both directions. Such templates can be stored client-side using a local repository and in a community repository at the server-side, while CURE learning environments remains in a separate database. Figure 4 shows these major components and their interaction.

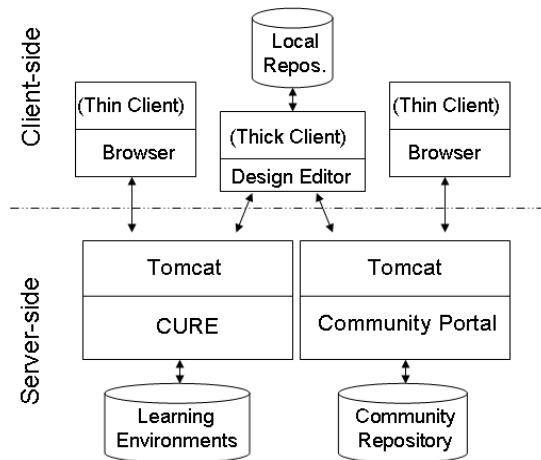


Fig. 3. The architecture extended collaborative platform

The CURE servers contain learning environments while the design editor is mainly used for their creation and manipulation as well as for the configuration of course structures. In our case, the process steps “normal use in learning environment” (A) and “tailoring and use in the learning environment” (B) from Figure 2 are supported

by the different CURE servers. The steps “template extraction from learning environments” (C), “create new template” (D), “design manipulation” (E) and “applying resulting design templates” (F) are supported by the design editor. Making the collaborative design editor a separate component creates two benefits: users can get more flexible graphical support, which cannot be easily realized with existing web technologies (R1), and reuse of course designs and structures from different servers. The design editor uses a local repository for storing local sessions. In addition, the editor allows for synchronous and asynchronous design sessions.

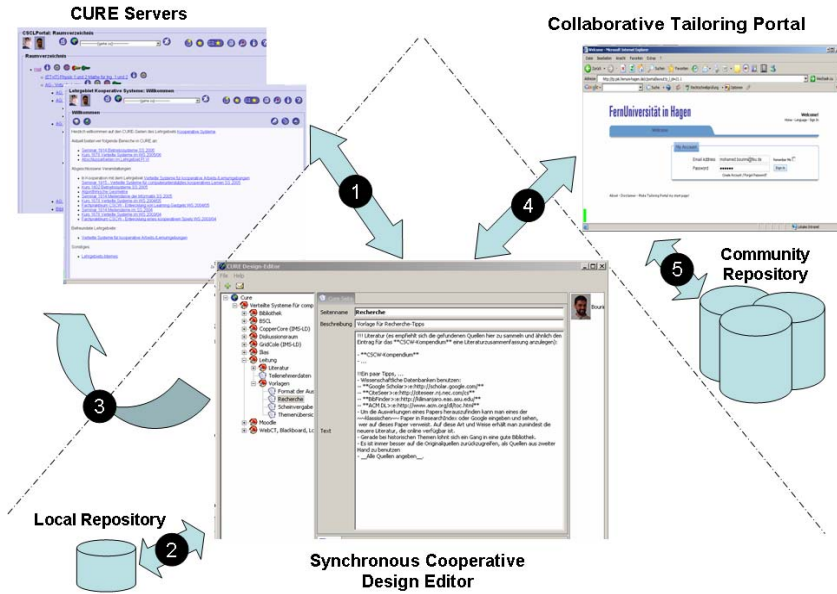


Fig. 4. Components of the approach and their collaboration

To encourage collaborative design and tailoring at the course template level, a collaborative tailoring portal is used. The portal is responsible for supporting the process steps “making design templates available to public” (G) and “reflect on design templates” (H). Due to the decoupling from the learning platform (here CURE), the emerging community is not limited to the use of the respective CURE server. The portal is primarily responsible for the exchange of course design templates, which are categorized and versioned in a community repository that can be accessed by interested users. In the first place, support for reflection on course designs for individual categories (courses, seminars, labs, etc.) is provided. Therefore, the portal offers various functionalities, e.g., forums, blogs, rating and publish & subscribe mechanisms. Moreover, the client software for the cooperative editor is to be downloaded from there. (R3)

The designer can create new designs or adapt existing ones using the design editor. Importing existing course structures from CURE is possible utilizing the extraction or import functions (Step C) (cf. Figure 4 - 1). Thus, only those rooms to which the

designer has sufficient rights are made available for import. In this step, the room and content structure is duplicated in the design editor while user objects and access permissions are copied on demand. For example, it makes no sense if new participants are assigned to a running course. To prevent loss of modifications, the editor saves the manipulated course structures locally in a repository (cf. Figure 4 - 2). Applying changes to a learning environment is done by the “export to CURE” function (cf. Figure 4 - 3). The designer is asked to choose a parent room in CURE as a target room, where the template is applied (i.e., the structure defined in the template is appended) (step E). Since the design editor and tailoring portal are two environments, exchange of templates must be supported through corresponding import and export functions facilitating the transitions between steps E and G (cf. Figure 4 - 4). Sufficient rights are needed in the portal. Moreover, portal users can rate and comment on course designs, which have been exported from the design editor to the portal. This data, and more information (e.g., usage statistics), are also stored as metadata of those designs in the community repository (cf. Figure 4 - 5). (R3)

5.3 A Sample Usage Scenario

In our scenario, an instructor wants to design a new learning environment for a particular course at the beginning of the semester. This instructor begins his work starting the design editor. Figure 5 shows the category view of the design editor, which is displayed at the start.



Fig. 5. Category view

The instructor has access to the global category view of the portal (Figure 5 - (a)) or the local repository (b). The global category view offers available shared design templates in the collaborative tailoring portal while the local view provides locally stored templates. The instructor wants to use a design template for a seminar from his colleague, who reported positive experiences (e.g., contribution in forums, entry in a blog etc.). From the figure, we can see that he/she is looking at the first version (c) of the seminar "Verteilte Systeme für computerunterstütztes kooperatives Arbeiten". To help him/her making a decision to use this template, the information pane (d), the properties (e), comments (f), and usage statistics (g) of the selected template version

are displayed, which are based on the metadata of the respective template. Such information can only be viewed in the design editor, and it is not available for local templates. Metadata are only specified during the export into the portal. Clicking on the load button (h) switches to the design view, which is depicted in Figure 6. Now, the manipulation of this template in the design editor is possible. If the metadata displayed are not sufficient to make a decision, the preview function (i) can be used. This function presents the course structure in a separate pane as a tree (similar to pane (a) of Figure 6).

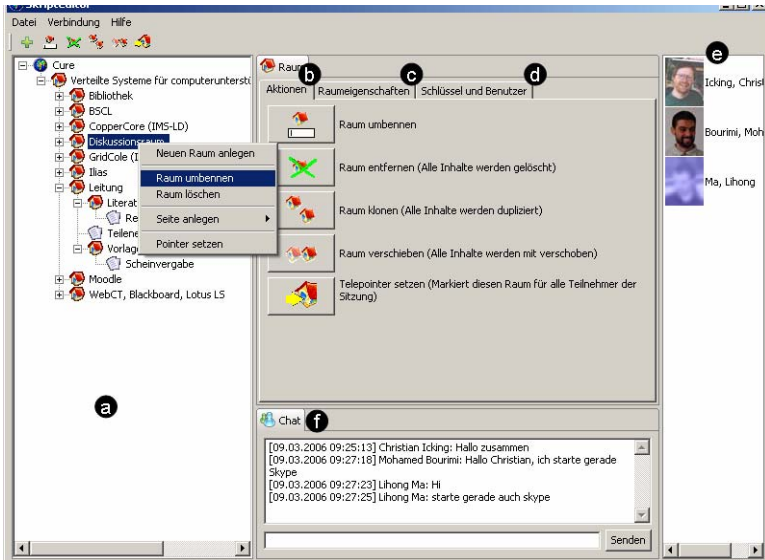


Fig. 6. Design view of the synchronous design editor

In Figure 6, three instructors (Icking, Bourimi and Ma) work jointly on a course template together that was imported from the portal (e.g., as described in the previous step). The collaboration is needed because the three instructors will supervise the course together. Thus, consensus about design concerns will avoid future changes and therefore not disturb the learning phases. After the coordinator (here Icking, shown at the top of the user list (e)) registered an active session in the portal, the other instructors were able to join the session. The structure of the course template is represented as a tree (Figure 6 – (a)), any modifications will be shared immediately. The participants of a synchronous design session are shown in the editor (e) and can communicate using the chat (f). Alternatively, an audio conference system like Skype can be used. To ease coordination, tele-pointers can be used by all participants. Portal users are aware of all active design sessions. Thus, interested people can join a session. (R2)

Since the cooperative editor allows only one participant to modify the course design at a time, other participants can only request the control from him. The new user in control can manipulate the design directly by drag & drop or with the help of function tabs in the middle pane (e.g. b+c+d in figure 6 for manipulation of rooms). Furthermore, the editing user must explicitly commit the changes (export functions).

The semantics of the drag & drop manipulations depends on the kind of artefacts. For instance, dragging a room and dropping it on another corresponds to the moving function. Dropping a room on a content page will add a link to the room to the content of the page. In addition, a link to the content page will be added to the welcome page of the room. (R1)

Once the intended course organisation is reached, the user applies the course design template to a target room in a given CURE server. If the template does not include specification of users and access permissions, these must be specified at this time by the user.

6 Implementation

To validate the presented approach in this paper, a prototype was implemented based on open source java-based frameworks. As mentioned before, the prototype consists of three different main components (see Figure 3). The implementation of the synchronous editor was carried out as an eclipse application [6] from which a native application can be generated any time for the most popular operating systems. The editor's replication functionality was realized integrating the JGroups framework [13]. Synchronous aspects of design sessions are therefore largely covered by the started editor instances. Therefore the performance depends only on the individual started clients and not other components.

The collaborative Portal and the editor access a common jabber server [26] which provides chat and awareness services. The portal itself is an adapted version of a rich functional Liferay portal [17]. This provides message boards, Blogs, Wiki editors and many other useful tools and functionalities, in the form of Portlets conformable to the JSR-168 specification [14]. Portlets are small functional units that can be easily aggregated and integrated with each other to build the functionality of the view and controller layer. The business logic and data persistence layers of the community portal were extended [18] implementing a set of new components using existing functionalities of the Liferay portal and providing new ones needed for our approach, while export and import interfaces use XML structures for the exchange of the data between the prototype components. An evolutionary development procedure was pursued. Thus, successive functional expansions were easy to be carried out.

7 Experiences

We initially tested the proposed approach with instructors from the department of computer science and some developers of the prototype. Thus, our test users were already familiar with using the CURE platform (without the proposed extensions and the proposed tailoring process). In the following, we report initial feedback and experience with the proposed process and its support in the prototype.

Regarding the prototype system, we observed general usability as well as the extent to which the prototype supports the proposed process. Our test users judged graphical manipulations of learning environments via an editor very positive. Users had no trouble in using the editor. Concepts like drag & drop functions or the

representation of the room structure, as trees, were intuitive for the involved testers. Those skills are mostly gained from previous dealing with computers. The analogies to the CURE platform and the semantics of the manipulations had merely to be communicated.

The users were very pleased with the possibility of carrying out local experiments because this considerably lowers the inhibition threshold for design experiments. The import and export functions were primary used by the developers for the extraction of the available course structures for the purpose of evaluation. Instructors were able to easily adapt their learning environments (individually and with others) for their future courses.

Since the winter semester 2003/04 CURE is used at the FernUniversität. More than 45 courses are currently available. These are at present investigated with the aim to extract good practices. First analyses show, for instance, that instructors have used the concepts offered by CURE differently. We hope to find structural and usage patterns that consistently lead to good learning performance and which therefore are candidates to reuse and for training new teachers.

Unfortunately, cooperation aspects on the level of the portal could not be exhaustingly evaluated because of the size of the tester community. However, first feedback indicates that a better support of awareness may be necessary to improve the motivation and participation.

8 Conclusions and Discussion

In this paper, an approach was presented to better support collaborative design and tailoring of (web based) learning environments. The main idea is to support a collaborative design and tailoring process consisting of cooperative activities. The need for such cooperation arises from the analysis of realistic scenarios. The support of graphical manipulations (here with the help of a design editor) and the collection of results of those design sessions (here in repositories) form an adequate base for this approach. Extending a tailorable learning platform (CURE) with support for performing the collaborative tailoring process enables end-users to reuse previous designs and to adapt those to their needs. End-users are enabled to collaboratively adapt and tailor their learning environments themselves. The provision of mechanisms for rating and evaluating collected designs (here through the portal) facilitates the extraction of good practices.

This contribution goes beyond the state-of-the-Art. None of the learning platforms mentioned before, except Symba, supports synchronous cooperation during design and tailoring activities. However, Symba's use remains very special and the learning environments are no more available after the end of the corresponding learning process (see Section 3). Available mechanisms for importing and exporting course designs aim at individual use. Thus, such functions presuppose privileged rights and running installations of the respective platform. These limitations have been overcome in the approach presented in this paper.

In addition, most platforms have a restricted view on the execution of learning processes and do not sufficiently support end-user tailoring. Our initial experiences indicate the applicability of our approach. Flexible graphical means and synchronous

work on shared artefacts (designs) bring many advantages. The cognitive gap of the users to the platform is reduced and exchange of expertise is encouraged.

The proposed process for design and tailoring of learning environments is a first step for sharing course designs across learning platforms. The proposed separation between design and tailoring activities, community support, and learning platform supports such generalisation across learning platforms. For instance, the community portal can be used to encourage building and collaborative (inter-)acting of CoPs interested in sharing knowledge at different levels of design and tailoring of learning environments (i.e., IMS LD scripts) or in other domains (i.e., learning content object). Adding a new learning platform to the proposed architecture requires the use of an implementation of the extract (C) and apply (F) steps based on the API of the new learning platform, so that the design and tailoring environment can import and export design templates from and to that platform. More powerful possibilities of a platform-independent collaborative design and tailoring can be reached if in addition to such implementations common or standardised representations of design templates are used among different platforms. The only problem we predict will consist of the adaptation of those course templates to specifics of the learning environments of the new platform, i.e., access permissions and user assignments that underlie mostly different access policies. However, the design and tailoring environment could be adapted in such situations to support such access policies or such adaptation activities could take place in the learning platform after a successful export.

Future works will focus primarily on community-related concerns (i.e. extending the testers community, motivating contributions etc.) and to respond to emerging needs for these purposes. A better support for the proposed process through the tools and potential extensions based on users' feedback as well as the extraction of best practices are also important.

Acknowledgments

Special thanks to Jörg M. Haake for the deep discussions that contributed positively to this work. Thanks are also due to Jaroslav Sos and Matthias Schwitzgebel for their contributions to the implementation of the prototype.

References

1. Andriessen, J.H.E., Huis in't Veld, M., Soekijad, M.: Communities of practice for knowledge sharing. In Andriessen, J.H.E., Fahlbruch, B., eds.: *How to Manage Experience Sharing*. Elsevier, Amsterdam (2004) 173–194
2. Appelt, W.: BSCL-basic support for collaborative learning. In Bode, A., Desel, J., Rathmeyer, S., Wessner, M., eds.: *Proceedings of DELFI 2003: 1. E-Learning Fachtagung Informatik. GI-Edition Lecture Notes in Computer Science, Germany, GI (2003) 120–128*
3. Betbeder, M., Tchounikine, P.: Symba, a tailorable framework to support collective activities in a learning context. In: *Groupware: Design, Implementation, and Use. Proceedings of the 9th International Workshop (CRIWG 2003). LNCS 2806, Springer (2003) 90–98*
4. Blackboard Inc.: <http://www.blackboard.com/products/as/index.htm> (2006)

5. Bote-Lorenzo, M.L., Vaquero- González, L.M., Vega-Gorgojo, G., Dimitriadis, Y., Asensio- Pérez, J.I., Gómez-Sánchez, E., Hernández -Leo, D.: A tailorable collaborative learning system that combines ogsa grid services and ims-ld scripting. In: *Groupware: Design, Implementation, and Use. Proceedings of the X International Workshop on Groupware (CRIWG 2004)*. LNCS 3198, Springer (2004) 305–321
6. Eclipse.org: <http://eclipse.org> (2006)
7. Greenberg, S., Roseman, M.: Using a Room Metaphor to Ease Transitions in Groupware. In Ackermann, M., Pipek, V. and Wulf, V. (Ed.): *Beyond Knowledge Management: Sharing Expertise*, MIT Press: Cambridge, MA, 2003.
8. Haake, J.M., Haake, A., Schümmer, T., Bourimi, M., Landgraf, B.: End-user controlled group formation and access rights management in a shared workspace system. In *CSCW'04: Proceedings of the 2004 ACM conference on Computer supported cooperative work*, ACM Press (2004) 554-563
9. Haake, J.M., Schümmer, T., Haake, A., Bourimi, M., Landgraf, B.: Two-level tailoring support for CSCL. In: *Groupware: Design, Implementation, and Use. Proceedings of the 9th International Workshop (CRIWG 2003)*. LNCS 2806, Heidelberg, Springer (2003) 74–82
10. Haake, J.M., Schümmer, T., Haake, A., Bourimi, M., Landgraf, B.: Supporting flexible collaborative distance learning in the cure platform. In: *Proceedings of the Hawaii International Conference On System Sciences (HICSS-37)*, IEEE Press (2004)
11. Henderson, A.: Tailoring Mechanisms in Three Research Technologies. In: *Proceedings of GROUP*. (1997)
12. IMS Golbal Learning Consortium, Inc.: <http://www.imsglobal.org/learningdesign> (2006)
13. JavaGroups: <http://www.jgroups.org/javagroupsnew/docs/index.html> (2006)
14. JSR-168 Spec.: <http://www.jcp.org/aboutJava/communityprocess/final/jsr168/> (2006)
15. Kahler, H.: Supporting Collaborative Tailoring. PhD thesis, Roskilde University (2001)
16. Leuf, B., Cunningham, W.: *The Wiki Way*. Addison Wessley, Longman, 2001.
17. Liferay Portal: <http://www.liferay.com/web/guest/home> (2006)
18. Liferay Portal – Developer Zone: <http://www.liferay.com/web/guest/devzone> (2006)
19. Lotus Software: <http://www-128.ibm.com/developerworks/lotus/products/elearning/> (2006)
20. Maaß, S.: Software-Ergonomie. Benutzer- und –aufgabenorientierte Systemgestaltung. In *Informatik Spektrum* 16(4) (1993) 191-205.
21. Moodle: <http://moodle.org/> (2006)
22. Mørch, A.: Three levels of end-user tailoring: customization, integration, and extention. MIT Press (1997) 51-76
23. Mørch, A.I., Mehandjiev, N.D.: Tailoring as collaboration: The mediating role of multiple representations and applicationunits. *Comput. Supported Coop. Work* 9(1) (2000) 75–100
24. Mørch, A., Stiemering, O., Wulf, V.: Tailorable groupware (workshop): issues, methods, and architectures. In: *GROUP*. (1997) 4-7
25. Oberquelle, H.: Menschengerechte groupware - software-ergonomische gestaltung und partizipative umsetzung. In Hartmann, A., Herrmann, T., Rohde, M., Wulf, V., eds.: *Menschengerechte Groupware*. Volume 42 of *Berichte des German Chapter of the ACM.*, Teubner (1994) 31–50
26. Shchepin, A.: ejabberd, a free and open source distributed fault-tolerant jabber/xmpp server. <http://ejabberd.jabber.ru/> (2006)
27. WebCT Inc.: <http://www.webct.com> (2006)
28. Wessner, M., Haake, J.M., Tietze, D.A.: An infrastructure for collaborative lifelong learning. In: *HICSS*. (2002)
29. Wulf, V.: "Let's see your search-tool!" - collaborative use of tailored artifacts in groupware. In: *Proc. Of the international ACM SIGGROUP conference on supporting group work*, ACM Press (1999) 50-59

Author Index

- Anguita, Rocío 155
Anido-Rifón, Luis 187
Antunes, Pedro 62, 364
Asensio-Pérez, Juan I. 310
Avouris, Nikolaos 140
- Baloian, Nelson 179, 364
Baytelman, Felipe 364
Bote-Lorenzo, Miguel L. 310
Bourimi, Mohamed 421
Bratitsis, Tharrenos 54
Brownholtz, Beth 270
Bu, Jiajun 405
- Caeiro-Rodríguez, Manuel 187
Camacho, Jesus 381
Capponi, Maria Francisca 1
Carrapatoso, Eurico 246
Carriço, Luís 364
Carte, Traci 12
Castro, Luis A. 389
Chalon, René 163
Chen, Chun 405
Chidambaram, Laku 12
Claude, Francisco 179
Cole, Cassie 12
Collazos, César A. 102, 219
- David, Bertrand 163
Delotte, Olivier 163
de Souza, Jano Moreira 78
de Vreede, Gert-Jan 38
Dimitracopoulou, Angelique 54
Dimitriadis, Yannis A. 155, 310
- Favela, Jesus 381, 389
Feltz, Fernand 94
Ferreira, Antonio 62
Fonseca, Benjamim 246
Fuks, Hugo 302
- Gallud, José A. 341
Garfield, Monica 12
Garrido, José Luis 286
- Gea, Miguel 286
Gerosa, Marco Aurélio 302
Geyer, Werner 270
Gobert, Xavier 94
Gómez-Sánchez, Eduardo 310
González, María Paula 102
González, Miguel 286
Gonzalez, Victor M. 381
Granollers, Toni 102
Grasse, Thomas 203
Guicking, Axel 203
- Harrer, Andreas 118
Hellweg, Matthias 326
Hicks, Lindsey 12
Hurtado, María V. 286
- Imbert, Matthieu 163
- Jiang, Bo 405
Jorrín-Abellán, Iván M. 310
- Kahrimanis, Georgios 140
Konow, Roberto 179
- Lagos, Maria Ester 1
Llamas-Nistal, Martín 187
Lozano, María 341
Lukosch, Stephan 326
- Malzahn, Nils 118
Marcos, José Antonio 155
Margaritis, Meletis 140
Markarian, Antoine 389
Martínez, Alejandra 155
Masserey, Guillaume 163
Matsumoto, Mitsuji 179
Molina, Ana Isabel 413
Morán, Alberto L. 22
- Neyem, Andrés 228
Noguera, Manuel 286
Noirhomme, Monique 94
Noteboom, Cherie 38
Nussbaum, Miguel 1

- Ochoa, Sergio F. 228
Ortega, Manuel 413
Otjacques, Benoît 94
- Payne, Matt 38
Penichet, Victor M.R. 341
Pereira de Lucena, Carlos José 302
Perez, Carmen 22
Pimentel, Mariano 302
Pino, José A. 62, 228
- Rasel, Martin 326
Redmiles, David F. 270
Redondo, Miguel Ángel 413
Rodríguez, Marcela 22
Roth, Benedikt 118
- Sá, Marco 364
Sarmiento, Johann 132
Sendín, Montserrat 219
Silva Filho, Roberto S. 270
Stahl, Gerry 132
- Steinhauser, Lucas 38
Sun, Lingling 349
- Tarmizi, Halbana 38
Tentori, Monica 389
Tobarra, Manuel 341
Trausan-Matu, Stefan 132
- Vassileva, Julita 349
Vega-Gorgojo, Guillermo 310
Vivacqua, Adriana S. 78
- Wan, Jian 262
- Xu, Xianghua 262
- Yang, Jianxv 405
- Zhang, Chi 38
Zigurs, Ilze 38
Zurita, Gustavo 364