An Integrative Approach to Diagram-based Collaborative Brainstorming: A Case Study

Diogo Azevedo[1], Benjamim Fonseca[1], Stephan Lukosch[2], Hugo Paredes[1], Jordan Janeiro[2] and Robert Owen Briggs[3]

[1]School of Science and Technology, University of Trás-os-Montes e Alto Douro, Vila Real, Portugal
[2]Systems Engineering Department, Technological University of Delft, Delft, Netherlands
[3]MIS Department, San Diego State University, San Diego, California, U.S.A.

Abstract. The need for computer supported collaboration has grown over the last years and made collaboration an important factor within organizations. This trend has resulted in the development of a variety of tools and technologies to support he various forms of collaboration. Many collaborative processes, e.g. strategy building, scenario analysis, root cause analysis and requirements engineering, requires various collaboration support tools. Within these synchronous collaborative applications to create, evaluate, elaborate, discuss, and revise graphical models, e.g. data flow, fishbone and brainstorming diagrams, play an important role. Currently, the necessary tools are not integrated and flexible enough to support such processes. In this paper, we present a case study done to a synchronous collaborative brainstorming diagram editor - CLSD that is integrated in a flexible group support system. Our approach goes beyond the current state of the art as we can be seamlessly integrated with other collaboration support tools such as text-based brainstorming or voting.

1 Introduction

Working practices had an important growth over the years, especially on group works - a group of people engaged in the execution of several objectives of a common task [1],[2]. Therefore such an effort should be helped by collaborative practices such as the Computer Supported Cooperative Work (CSCW), which improves the performance of a group in the execution of tasks, through group work supported by information and communication technologies. Groups can become even more productive when supported by Group Support Systems (GSS). It is decisive that GSS adopt techniques for the development of groupware applications that meet non-functional requirements (quality attributes) such as interoperability, integration, reliability and usability [3]. Collaborative graphical systems support a group of people concurrently editing graphical processes over the network. In the case of object-based graphical editing, the shared information space central is a unique scene of objects shared among other users, where previous approaches need to be applied and can be classi-

Azevedo D., Fonseca B., Lukosch S., Paredes H., Janeiro J. and Owen Briggs R..
An Integrative Approach to Diagram-based Collaborative Brainstorming: A Case Study.
DOI: 10.5220/0004106501100121
In *Proceedings of the 1st International Workshop on Interaction Design in Educational Environments* (IDEE-2012), pages 110-121
ISBN: 978-989-8565-17-4

fied into locking, serialization and multi-versioning [4]. In the locking approach adopted by systems such as Aspects [5], Ensemble [6] and GroupDraw [7] concurrency is restricted, and concurrent editing is allowed only if users are locking and editing different objects, and moreover responsiveness – the capability to answer on time – is affected due to delays for lock acquisition [4].

1.1 Motivation

Many collaborative processes, e.g. strategy building, scenario analysis, root cause analysis and requirements engineering, require various collaboration support tools. Within synchronous collaborative applications to create, evaluate, elaborate and revise graphical models by groups, e.g. data flow diagrams, work structure breakdowns and fishbone diagrams. Currently, there is lack of support on GSS for such processes. GSS must therefore offer users collaborative environments where they can interact [8], however many of these systems fail when providing the right tools for effective collaboration [9]. Analyzing how groups work and evolve is necessary when we consider the social dimension of the collaborative work [10]. With this paper, practitioners can gain the potential benefits of collaboration technology without having to take special training. This feature is only possible thankfully to the transparency and concurrency control of the ActionCenters, since former users will be simultaneously editing and deleting data (concurrency control), and latter the development and background should be invisible (transparency) to users [11].

1.2 Objectives

The Collaborative Line-and-Symbol Diagramming Component (CLSD Component) presented herein offers a collaborative environment to manage graphical models and thereby their related collaborative processes. To achieve such a collaborative environment we have been concerned with awareness that as claimed by [12] is defined as an understanding of others activities, which provides a context for your own activity. According to [13] group awareness information includes knowledge about who is on the collaborative environment, where they are working, what are they doing and their furthermore intentions [3]. Furthermore, the techniques and diagram types that could be used to support collaborative diagramming efforts were taken into consideration, and how the features and functions of a single-user differ from a multi-user diagramming tool in order to optimize the values that groups can create through collaborative diagramming. CLSD Component is integrated as a plug-in component within the Computer Assisted Collaboration Engineering platform (CACE), and thereby can be used in various different processes. CACE approach embeds collaboration expertise with collaboration technologies [14], so that participants can gain the same benefits without any special training [15].

1.3 Structure of the Paper

In the remaining of this paper, we define a set of concepts required within GSS and for Collaboration purpose. After that, we present the requirement analysis giving a scenario of collaborative processes and thereby the set of requirements. After that we take a closer look at the requirements that a GSS system has to fulfill, followed by the requirements that diagram-editors and the CLSD demand. Before concluding and pointing to future directions, we fully present a case and the related results of the study conducted to analysis the usability of the CLSD. More information about the CLSD can be found at [11].

2 Requirement Analysis

Here it is addressed the requirements that a GSS system has to fulfill to support the CLSD Component. Furthermore it addresses the requirements gathered from other existing diagram editors and therefore the selection of the features that better fit our collaborative diagram editor. Moreover, it addresses the functional requirements that the CLSD Component has to fulfill allowing collaboration engineers to configure synchronous collaborative applications that actually fit specific collaborative processes, such as strategy building, scenario analysis, root cause analysis and requirements engineering. The various processes of the system are identified, the multi-user approach is explained and the technical requirements are described.

2.1 GSS Requirements

To illustrate the requirements that a GSS system has to fulfill in order to support the CLSD Component and the interoperability needed between components we present a scenario of a collaborative strategy building process that uses collaborative diagramming and other collaborative applications, e.g. a text-based brainstorming. Two activities that can be considered in this scenario are: 1- a text-based brainstorming for strategy building (e.g. Outliner Component); 2- a diagram-based brainstorming (e.g. CLSD Component) to organize, connect and manage strategies based on the data gathered in the previous activity.

In the above scenario, we have to support collaboration engineers in designing collaborative processes (R1), such as strategy building, root cause analysis, and design suitable collaboration support (R2). For that the GSS needs to support the integration of components that support collaborative processes (R3), by allowing re-using of existing components [3] Furthermore, it must be able to share and exchange data efficiently (interoperability) between components (R4) [3],[16],[17], in order to reuse the data gathered for example from the first activity (text-based brainstorming) into the second activity (diagram-based brainstorming). Additionally, we do not know all the support that is needed so that the set of components must be extensible (R5) [3]) by software developers and an API (R6) to support them [18] should be provided. Finally, our scenario requires collaborative diagramming (R7), and for that we have

identified additional requirements.

The Action Centers therefore does not have any tools. As alternative, these tools are plugged into the Action Center as components to simply make them available in the runtime system [11]. So, the Action Center supports the design of collaborative applications (R1), and allows components (as our CLSD Component) to be assembled by Collaboration Engineers into the CACE editor (R2). These components have access to shared data (R4), are configurable (R3) and can be (re)-designed by other Collaboration Engineers (R5). They usually consist of a user interface for displaying data shared in a group, some input mechanism, and business logic. ActionCenters also fulfill the R5 since it is published under the BSD, permissive free software license, and is therefore open to anyone who wants to add new components [19].

Furthermore, the Action Center provides two JavaScript objects to manage data and their updates – ActionCenterListener, and an ActionCentersAPI (R6) that offers services to create and support the development of collaborative components. Additionally, the data is managed through dynamic communication channels using CometD[1] to a Universal Data Model (UDM) [15], to dynamically create and store arbitrary relational data. The UDM and the two JavaScript objects offer some mechanisms to manage contribution, such as *modifiedBy* to know who changed the data (R7.4), and *lockedBy* to edit-lock entities and their attributes to provide single-user editing (R7.5). A more detailed description of the system can be found on [15].

Action Center does not address all requirements needed for Collaborative Diagramming. For that purpose, we implemented our (R7) CLSD Component that consists of an XML wrapper and an implementation in JavaScript with Ext JS[2] and an extended library called Joint JS[3]. The JointJS library is used for to create diagrams that can be fully interactive for both implementing a diagramming tool (as our CLSD Component) as well as simply for publishing diagrams (R7.1, R7.2 and R7.3) [11]. The last requirements (R7.6 and R7.7) have been implemented in the XML wrapper, which is the CLSD Component, and was used the API provided by the ActionCenters as support to implement it.

According to [3] there are GSS systems addressing some of the requirements described above, however for our approach we have chosen ActionCenters because it addresses all of the requirements and it fits with our purpose. Two parts form Action Centers: a CACE editor and a Process Support System (PSS). The CACE editor is a tool to design an effective work practice by defining the content and sequence of collaborative activities that are packaged into the PSS [15].

2.2 Requirements Gathered from other Existing Diagram Editors

The list of requirements is based on the analysis of other existing Diagram Software,

[1] The Dojo foundation. Cometd. More information can be found in http://cometd.org/, 2011.

[2] Ext JS is a javascript framework for developers. More information can be found in http://www.sencha.com/, 2011.

[3] Joint JS is a JavaScript library developed by David Durman, More information can be found in http://www.jointjs.com/, 2011.

such as Banxia[4], Smart Ideas[5] and Ext Designer[6]. In this case, the requirements address the interaction that Collaborative Diagramming has to provide to groups while they participate in collaborative environments. It must be possible for group members to insert, import (text-based) and manage ideas into a diagram-based format (R7.1), like our previous strategy building scenario. Then, ideas are diagram-based organized (clusters and colour manager) (R7.2) and connect arrows (connect ideas through arrows) (R7.3). Furthermore, group members can unintentionally provoke data conflicts between contributions and therefore it is required to provide remote field of vision - awareness with the scope (who has been doing what) of other members' activities [20], and data with their information and the resources that are nearby [21] (R7.4), and furthermore triggered locking mechanisms when updates occur (R7.5). Moreover, when changing from text to model based the CLSD Component should allow consensus building such as the organization of concepts even when their position is not defined (R7.6) (this can happen when data is imported from other components, such as the Outliner Component). Finally, the CLSD Component needs to implement a set of rules so that it would be possible to identify conflicting relations, such as arrows that will be connected at least at 2 concepts and also concepts that can change automatically from colour when moving from categories.

The features and requirements that the CLSD Component should support to create a collaborative modelling tool (R7), a scenario identifying the main and necessary functionalities of diagram editors was done. In this scenario the diagram editor should generate (blocks) concepts based on text (R7.1**)** (previously inserted on the database or at runtime), and furthermore it should converge on key concepts allowing users to merge sub-categories from main categories (R7.2). To converge on the key concepts the diagram editor needs first allow the connection (link) between concepts to (R7.3), which also leads to the organization of the diagram (R7.2) (model relations). For the CLSD Component have the necessary features and tools in order to provide awareness to other users in the collaborative environment, such as telepointers, remote field of vision and so on (R7.4). Furthermore, concurrency control is very common in collaborative scenarios since we have more than one user trying to manipulate the same data, which leads to the requirement of locking contributions upon their manipulation (R7.5). Moreover, in this scenario the diagram editor should provide consensus building on relations when moving from text to model (R7.6), as well as identify conflicting relations (R7.7) (an arrow will be connected at least at two concepts).

The requirements gathered from COMA, SmartIdeas and Banxia [11] were compared to the above scenario to validate the behavior of the scenario itself and to see if these requirements fit with it. This comparison was done to see if other diagram editors normally use the above scenario and if it is possible to implement it in collaborative diagram editors, such as the CLSD Component. After making such analysis and conclude that the above scenario match with the approach used in the diagram editors

[4] Banxia (Decision Explorer) is a proven tool for managing software issues. Structure and analyse of qualitative information. More information can be found in http://www.banxia.com/dexplore/, 2011.

[5] Smart Ideas concept-mapping software brings the power of visual learning to classrooms, through interactive white boards. More information can be found in http://smarttech.com/, 2011.

[6] Ext Gui Designer is a graphical user interface builder for web applications. Developed by Sierk Hoeksma. More information can be found in http://www.projectspace.nl/, 2011.

studied, the main features that the CLSD Component should provide and the requirements mentioned in the above scenario that must match with the requirements implemented in the CLSD Component where described.

2.3 Collaborative Line-and-Symbol Diagramming Requirements

The detailed analysis of diagram editors and their features revealed the most important requirements: add, edit and delete concepts (brainstorming concepts, ideas, words, blocks and so on); add and delete arrows to connect concepts; add, edit and delete notes; and auto save concepts, arrows, notes and diagrams every time any change is made. Furthermore other important requirements are: cluster concepts by displaying different colours for each category; export diagrams through XML files to be displayed out of the ActionCenters; add and delete telepointers (limited to 1 per user). Additionally, locking mechanisms, remote field of vision and telepointers extend the requirements of working in collaborative environments.

Furthermore we compare the studied requirements with the requirements that our CLSD Component support. CLSD Component generate concepts based on text fetched from other existing components or from the database, or at runtime (R7.1); it is possible to connect concepts with arrows (R7.3) and differentiate them from categories and subcategories by highlighting their path (R7.2); is also allow users to former know who changed the data and latter to edit-lock entities and their attributes to provide single-user editing (R7.4) (R7.5), these features have been supported by the API provided by ActionCenters; the organization of the diagram (R7.6) is possible in the CLSD Component by dragging concepts and put them in the right position, however it is only possible to do it manually, so it is not possible to randomize automatically their position; finally it has a set of rules to identify conflicting relations (R7.7) (e.g. each arrows will have at least two concepts) and to maintain consensus building when moving from text (e.g. Outliner component) to model (CLSD Component) based.

3 CLSD

ActionCenters in combination with CLSD Component allows us to support various different processes that require different forms of collaboration. Considering a strategy building processes where data is gathered from a text-based brainstorming and stored into the UDM – Universal Data Model in Action Centers. The union of the text-based with the CLSD Component creates an Action Center, where the data, which is selected (identified) based on their relationship types and attributes by the ActionCenters, is forward fetched (import) from the UDM and loaded (insert) to the CLSD Component [11]. CLSD Component transforms it into a diagram-based format where group members can furthermore manage and organize data as collaborative processes. Each single user controls the selection and manipulation of data and until he or she is finished nobody else can have access to manipulate that specific data. For that purpose at each moment that concepts are being changed it shows a locking icon

and a scope of action (remote field of vision) of the user who is manipulating it. Moreover, one possible approach for our CLSD Component was to send Notifications to inform the users about the conflicts, but instead we decided to use locking mechanisms when updating contributions to the database, which stops possible conflicts between users and unnecessary notifications [11]. Users don't have advantages for knowing that a conflict has occurred, such unnecessary notifications could disturb their work and focus.

4 Case Study

A case study was conducted with 41 students of three distinct subjects and courses from the University of Trás-os-Montes e Alto Douro: BSc in Communication and Multimedia, BSc in Humans Rehabilitation and Accessibility Engineering, and PhD in Informatics. The aim of this study was to analysis the CLSD usability. Therefore a quantitative questionnaire named Computer System Usability based on [22], extended with a set of qualitative questions, was given to the students after they performed their tasks on the CLSD. The CLSD was used, in a learning perspective, to help the group of students to perform their current tasks.

The first subject where CLSD was used was Social and Cooperative Platforms from the BSc in Communication and Multimedia Degree and focuses on the use of social networks, multiuser 3D spaces and Web 2.0 developing platforms for communication and cooperation strategies. Eight groups of 4 to 5 students have used CLSD to record the general ideas (Figure 1) of their work and furthermore use it as a guide in the developing process of the purposed work.

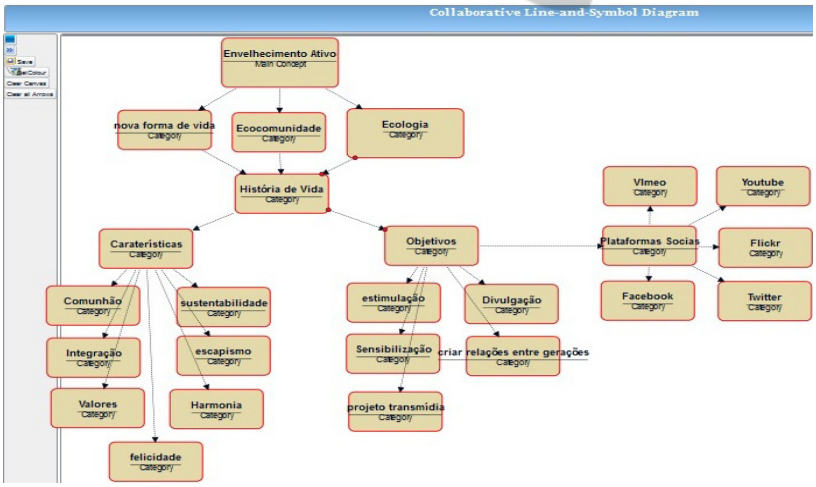

Fig. 1. Brainstorming made by a group of students upon the realization of a task (Portuguese).

The second subject making use of the CLSD was Telematics Applications for Inclusion from the Humans Rehabilitation and Accessibility Engineering Degree and focuses on the problem solving of the accessibility gaps at UTAD. Therefore, a group

of 13 students have made a brainstorming session where they first have identified and discussed the existing accessibility problems. Furthermore they have made other discussion at the same brainstorming session but now with the purpose of gathering solutions and link them to the current problems.

The PhD students of Informatics made a brainstorming session to identify the problems of a specific PhD research. Therefore the CLSD was used after a brief presentation of the research topic and the set of students have presented their perspective and scientific approach for the current problem.

In terms of achievements of the current tasks 80,48% of the students had agree that they could effectively complete their work using CLSD, where 78,046% were able to complete their work quickly, and 65,85% could efficiently complete the current work. Though 73,167% of the students believe that they became productive quickly using the CLSD. Such achievements and productivity may be related to the simplicity of CLSD, because 90,24% of the students claim that it was easy to learn how to use CLSD. This information can be found in more detail in Table 1.

Table 1. Computer system usability questionnaire [22] results.

Computer System Usability	
Overall, I am satisfied with how easy it is to use CLSD	75,608%
It was simple to use CLSD	63,41%
I can effectively complete my work using CLSD	80,48%
I am able to complete my work quickly using CLSD	78,046%
I am able to efficiently complete my work using CLSD	65,85%
I feel comfortable using CLSD	78,047%
It was easy to learn to use CLSD	90,24%
I believe I became productive quickly using CLSD	73,167%
The CLSD gives error messages that clearly tell me how to fix problems	29,265%
Whenever I make a mistake using the CLSD, I recover easily and quickly	41,464%
The information provided with CLSD is clear	65,855%
It is easy to find the information I needed	82,929%
The information provided for CLSD is easy to understand	80,5%
The information is effective in helping me complete the tasks and scenarios	75,606%
The organization of information on CLSD screens is clear	82,93%
The interface of CLSD is pleasant	82,927
I like using the interface of CLSD	60,967%
CLSD has all the functions and capabilities I expect it to have	43,901%
Overall, I am satisfied with CLSD	78,047%

The students from the first subject were able to efficiently complete the proposed work and move to the next activity. The second subject have gathered the accessibility problems at UTAD and then related them with proven solutions to their current problems. The PhD students have done one brainstorming session were all their feedback, related to a mobile applications PhD topic, was given to help improving the research objectives and issues of the current research.

5 Results

The quantitative and qualitative questionnaire delivered to students during the case study has generated several results about the usability of the CLSD and the related features, such as: ease of use, interface and organization of CLSD, the effectiveness and efficiency to complete the proposed work, the productivity gained, usability issues and so on. The results of the quantitative questionnaire based on [22] are described and presented by percentage of the most relevant results. These results can be found at Table 1. Furthermore, it was done a more specific analysis to the quantitative questionnaire by crossing qualitative data, such as the degree, experience using collaborative tools and example, age of the users, and so on.

The most used collaborative tools were Facebook with 51% and twitter with 44% and the daily intention of the students upon the use of collaborative tools are related with entertainment, work and research especially for the younger students, which ages converge between 20 and 22 (Figure 2). This last result goes beyond the extreme growth of social networks in the business, teaching and learning fields. However the majority of the students had answer that the purpose of using collaboration tools was for work and research, when it was asked which collaboration technologies they most used they have answered facebook and twitter, so the lack of information and knowledge of the existing collaboration technologies in the work field is notorious. There are collaboration technologies that could be integrated into social networks in order to call students attention and extend the use of such tools. This could be a good approach to be put in practice and at the same time have a bigger impact in the collaboration field. However, for now the CLSD will just be supported by Action-Centers. At first view and with the first set of analysis from the questionnaires, it had revealed that the CLSD is on the right path to fulfill the demands of the users. However some issues still need to be taken care to allow a better interaction and solving problems for users. Furthermore, the features that the CLSD embrace will be extended with the feedback provided by the students and a deeper analysis will be done to the questionnaires.

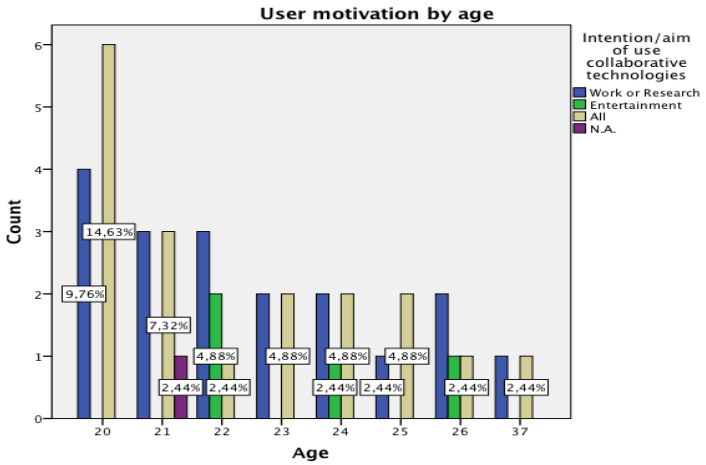

Fig. 2. Intention of using collaborative technologies.

6 Final Remarks

In this paper, we presented a Collaborative Line-and-Symbol Diagramming Component – CLSD Component assembled in a CACE editor and supported by ActionCenters. A collaboration support tool that consists of a XML wrapper and an implementation for creating diagrams that can be fully interactive for both implementing a diagram-based brainstorming session to manage collaborative processes as well as simply for publishing diagrams. Furthermore, we summarize the purposed requirements that CLSD Component address:

- (R7) CLSD Component allows the creation and elaboration of a brainstorming session (collaborative modeling);
- (R7.1) Group members can generate concepts based on: insert, import and fetch data from other components, into a diagram-based format through a collaborative environment provided by the CLSD Component;
- (R7.2) Converge on key concepts: merge sub-categories from main categories, by highlighting their path; Organization of the resulting diagram: the concepts and linking arrows can be dragged around the canvas area, which provides the necessary feedback to help users organizing/structuring the resulting diagram. Other feature implemented that also allows the structuring of the resulting diagram is the colour manager;
- (R7.3) Connection between concepts (arrows): users can connect concepts through arrows allowing the creation of a hierarchical diagram;
- (R7.4) Context awareness: the CLSD Component provides the necessary awareness, in order to make users aware of the scope area that other users are working on, therefore he supports remote field of vision, telepointers and so on;
- (R7.5) Locking mechanisms: using the provided API from ActionCenters the CLSD Component locks concepts upon the manipulation of them;
- (R7.6) Consensus building: transition from text to model based has different levels because they can be: text inserted at runtime, text inserted in other component that must be fetched by the CLSD Component, and text that is already in the database. The CLSD Component has been implemented taken into consideration these transitions levels, which provides a transparent environment to users (they do not need to know from what source the text comes from, they just need know who create it, and for that the awareness tools have been implemented);
- (R7.7) Identifying conflicting relations: set of rules that forbid the wrong use of the provided tools, such as arrows that must have at least two concepts connected, and so on;

The results from the conducted study case are still being under analysis and therefore a more extensive analyses will be done that will include also the analysis of videos from 8 recorded sessions. We want to use the results to improve the flexibility and usability [23] of our component, and furthermore see the exchange of data between components when changing from a text-based brainstorming to a diagram-based brainstorming. Furthermore, it will be compared traditional text-based approaches with diagram-based approaches. Finally, a new feature will be developed allowing the adding of new diagram types to the CLSD Component, which will allow users to choose the diagram type that better fits their interests.

References

1. Rowley, A. and Dollimore (2006): Secure Group Communication for Groupware Applications, ACM International Conference Proceedings Series: Volume 204, Proceedings of the annual research conference of the South African institute of computer scientists and information technologists on IT research in developing countries, pp. 198-205.
2. Frost Sullivan (2007): 'Meetings around the world: The Impact of Collaboration on Business Performance'.
3. Pelegrina, Ana B., Rodríguez-Dominguez, C., Rodríguez, María Luisa, Benghazi, Kawtar, and Garrido, José Luis (2010): 'Integrating Groupware Applications into Shared Workspaces', RCIS, pp. 557-568.
4. Ignat, C., and Moira C. Norrie (2006): 'Draw-Together: Graphical Editor for Collaborative Drawing', CSCW 06 Proceedings of the 2006 20th anniversary conference on computer supported cooperative work, ACM, pp. 269-278, ISBN: 1595932496
5. Von Biel (1991), V. Groupware Grows Up. MacUser, June 1991, 207-211.
6. Newman-Wolfe, R. E., Webb M., and Montes, M. (1992): 'Implicit locking in the Ensemble concurrent object-oriented graphics editor'. Proceedings of CSCW, Toronto, Canada, pp. 265-272.
7. Greenberg, S., Roseman, M., Webster, D. and Bohnet, R. (1992): 'Issues and experiences designing and implementing two group drawing tools'. Proceedings of the 25th Annual Hawaii Intl. Conference on the System Science, Kuwaii, Hawaii, January 1992, 138-150.
8. Duque R., Manuel Noguera, Crescencio Bravo, José Luis Garrido and Maria Luisa Rodríguez (2009): 'Construction of interaction observation systems for collaboration analysis in groupware applications', Advances in Engineering Software 40(12): pp. 1242-1250.
9. Grudin J. (1994): 'Computer-supported cooperative work: history and focus', IEEE Comput, pp. 19-26.
10. Grudin J., (1988): 'Why applications fail: problems in design and evaluation of organization or organizational interfaces', Proceedings of the ACM conference on computer-supported cooperative work, pp. 85-93.
11. Azevedo, Diogo, Janeiro, Jordan, Lukosch, Stephan, Briggs R. O. and Fonseca, Benjamim (2011): 'An integrative approach to diagram-based collaborative brainstorming', Proceedings of the ECSCW 2011 Workshop on Collaborative usage and development of models and visualizations, scientific publication.
12. Dourish, P. and Bellotti, V. (1992): 'Awareness and coordination in shared workspaces', Conference proceedings on Computer-supported cooperative work, volume 0, pp. 107-114.
13. Gutwin, C., Schneider, K., Paquette, D., Penner, R. (2004b): 'Supporting Group Awareness in Distributed Software Development', Engineering Human Computer Interaction and Interactive System, vol. 3425, pp. 383-397.
14. Briggs, R. O., Kolfschoten, Gwendolyn L., Vreede, Gert-jan, Albrecht, C., and Lukosch, S. (2010). 'Facilitator in a Box: Computer Assisted Collaboration Engineering and Process Support Systems for Rapid Development of Collaborative Applications for High-Value Tasks' Information Systems, pp. 1-10.
15. Mametjanov, A., Kjeldgaard, D., Pettepier, T., Albrecht, C., Lukosch, S., and Briggs, R. O. (2011): 'ARCADE: Action-centered Rapid Collaborative Application Development and Execution', Hawaii International Conference on Systems Sciences, volume 0, pp. 1-10.
16. Hofte, H., ter, Maurice A. W. Houtsma, Hermen J. van der Lugt, (1995): 'CSCW Infrastructure Research at TRC', ACM SIGOIS Bulletin, vol. 15.
17. Simone, C., Mark, G. and Giubbilei, D., (1999): 'Interoperability as a means of articulation work', WACC '99: Proceedings of the international joint conference on Work activities coordination and collaboration, pp. 39-48.

18. Riehle D., (2000): 'Framework Design: a Role Modeling Approach Dissertation', ETH Zurich.
19. Buttler, T., Jordan Janeiro, Stephan Lukosch, and Briggs R. O. (2011): 'Beyond GSS: Fitting Collaboration Technology to a Given Work Practice', Collaboration and Technology – 17th International Conference, CRIWG'11, Paraty, Brazil, October 2011.
20. Dix, A., Finlay J., Abowd G., and Beale R. (1993): 'Human-Computer Interaction', Prentice Hall.
21. Segal L., (1995): 'Designing Team Workstations: The Choreography of Teamwork', Local Applications of the Ecological Approach to Human-Machine Systems, pp. 392-415.
22. Lewis, J. R. (1995); 'Computer Usability Satisfaction Questionnaire' IBM Psychometric Evaluation and Instructions for Use. International Journal of Human-Computer Interaction, 7:1m 57-58.
23. Holzinger, A. (2005): 'Usability engineering methods for software developers', (C. Stephanidis Ed.) Communication of the ACM vol. 48(1), pp. 71-74 ACM. Retrieved from http://portal.acm.org/citation.cfm?id=1039539.1039541