The Mechanics of Collaboration: Developing Low Cost Usability Evaluation Methods for Shared Workspaces

Carl Gutwin[1] and Saul Greenberg[2]
[1]*Department of Computer Science, University of Saskatchewan*
[2]*Department of Computer Science, University of Calgary*
gutwin@cs.usask.ca, saul@cpsc.ucalgary.ca

Abstract

We introduce a conceptual framework that articulates the mechanics of collaboration for shared-workspace groupware: the low level actions and interactions that must be carried out to complete a task in a shared manner. These include communication, coordination, planning, monitoring, assistance, and protection. The framework also includes three general measures of these mechanics: effectiveness, efficiency, and satisfaction. The underlying idea of the framework is that some usability problems in groupware systems are not inherently tied to the social context in which the system is used, but rather are a result of poor support for the basic activities of collaborative work in shared spaces. We believe that existing low-cost evaluation methods—heuristic evaluation, walkthroughs, user observations and questionnaires—can be modified to include this framework in a way that helps a groupware evaluator uncover these usability problems.

1. Introduction

With the increasing connectivity of the internet, the increasing power of the world-wide web, and the increasingly distributed nature of organizations, multi-user computer systems (groupware) are becoming increasingly common. Despite this growth, many groupware systems have serious usability problems. At best, working in groupware is awkward and frustrating compared to face-to-face collaboration.

There are many reasons for this poor usability. We have only sketchy knowledge of how people collaborate, and translating what we do know into effective designs is difficult. As well, the effort involved in assessing prototypes and final systems is onerous because there are no simple but effective evaluation techniques for groupware. In this paper, we will concentrate on this evaluation problem. In particular, we will propose a conceptual framework that will help us develop discount usability evaluation techniques that can be readily applied to the iterative development cycle (design, implementation, and evaluation) of groupware.

Traditionally, researchers and developers consider groupware evaluation a difficult problem, especially when compared to the relative ease of evaluating single-user systems. An oft-cited factor is that groupware acceptance is far more likely to be affected by social factors such as organizational culture, differences in personalities, and group dynamics (e.g. [7]). These combine to make the task of understanding group interaction a 'wicked problem' [5]. Consequently, traditional experimental and laboratory methods that remove the software from its context of use may obtain simplistic results that do not generalize well to real-world situations.

As an alternate to the laboratory, many groupware researchers now advocate the use ethnographic and sociologic methods that explicitly consider culture and context (e.g., [10,16]). While these methods have been successfully applied to real situations, they tend to be expensive and somewhat limited. They demand considerable time and evaluator experience. They work best at the beginning of design (to uncover and articulate existing work practices) and at the end (to evaluate how systems already deployed in the work setting are used). Because they are unsuited for rapid prototype evaluation, they are rarely appropriate for iterative design.

We agree that contextual considerations are extremely important. However, we also believe that there is pragmatic value in a complimentary perspective. Specifically, we claim some groupware usability problems are not strongly tied to social or organizational issues, but rather are caused by insufficient or mismatched support for the basic activities of collaboration. These activities, which we call the mechanics of collaboration, are the small-scale actions and interactions that group members must carry out in order to get a shared task done. Examples include communicating information, coordinating manipulations, or monitoring one another.

The activities that form these mechanics of collaboration are particularly important in shared-workspace groupware, where the group task involves objects, artifacts, and tools in a visual workspace. In

systems like this, if a group is unable to communicate effectively and efficiently about the task, or is unable to smoothly and easily coordinate their actions, performance and satisfaction are likely to suffer. The mechanics of collaboration are by and large separate from organizational politics or group dynamics, and usability problems in the mechanics can therefore be discovered and ironed out during iterative design. Although appropriate support for the mechanics of collaboration will not guarantee a system's suitability in the real world, failure to support them will almost certainly guarantee its demise.

In this paper we develop a conceptual framework defining groupware usability for shared-workspaces that is based on the mechanics of collaboration. We describe these mechanics, and propose that various single-user usability testing schemes can be applied to groupware by having them assess support for the mechanics. We believe that evaluating for these mechanics provides a middle ground between the brittleness of controlled experimentation and the expense of field techniques. While not complete on their own, they will provide a useful addition to the practitioner's toolbox.

2. Shared-Workspace Groupware

Groupware allows people to work together across time and distance. An important class of groupware systems is that of applications that support distant collaboration though a shared workspace—a medium sized flat work surface where people collaborate by manipulating visible tools and task artifacts.

Common applications in this class include shared editors, drawing programs, and multi-player games. For example, a shared design application could allow engineers from several different sites to plan, view, and manipulate new parts for a manufacturing process. Similarly, a group troubleshooting system could allow expertise from many locations to be gathered together to investigate a problem and explore possible solutions (e.g. [19]).

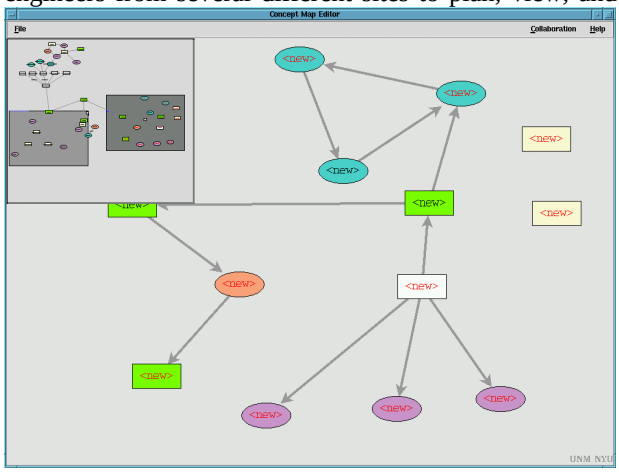

Figure 1. An example shared-workspace groupware system for creating and editing concept maps. A radar overview [8] is visible at top left.

The group tasks that happen in these systems tend to be of a few types. Based on McGrath's (1984) task circumplex, tasks in shared workspaces usually involve:

- creation of new artifacts,
- organization of existing artifacts,
- exploration of the space or of a set of artifacts,
- construction of larger objects from component pieces,
- the management of an autonomous system represented in the workspace.

3. The Mechanics of Collaboration

For a collaborative task to be accomplished in a shared workspace, a variety of activities must happen. First, the actual execution of the task must occur—words put on paper, objects placed in order, or parts fixed together to form a whole. This part of the task is no different for a group than it is for an individual, since the same actions still have to happen if the job is to get done. However, most group work involves another set of entirely different activities as well. If we consider task execution to be the *taskwork*, then this other set of activities is the *teamwork*—the work of working together—and a groupware system must support both taskwork and teamwork if it is to be truly usable.

Teamwork can be further divided into two areas: the social and affective elements that make up group dynamics, and the mechanics of collaboration. Although affective elements are important, we will not consider them further here. This leaves us with the mechanics—the things that groups have to do, over and above what an individual has to do, in order to carry out a task.

From our previous research on shared-workspaces (e.g., [9]) and from the literature (e.g., [1,20]), we have identified seven major activities that comprise the mechanics of collaboration.

Explicit communication. Group members intentionally provide each other with information, and verbal and written communication is a cornerstone of collaboration. In a visual workspace, however, the workspace and the artifacts themselves are crucial supports to explicit communication. In particular, people often use deictic references (e.g. "this one") in combination with pointing to an artifact.

Consequential communication. In addition to explicit communication, people also pick up considerable information that is unintentionally "given off" by others as they go about their activities. This is called consequential communication [17] and is also important

in smooth group operation. Two main types of consequential communication involve information given off by artifacts as they are manipulated by others (also called feedthrough – [3]), and information given off by the characteristic actions of a person's embodiment in the workspace.

Coordination of action. People organize their actions in a shared workspace so that they do not conflict with others. Shared resources and tools require that turns be taken, and some tasks require that actions happen in particular orders. In addition, people also learn to predict one anothers' actions and use those predictions to make the group more effective or efficient. Symptoms of poor coordination include people bumping into one another, duplicating actions that another person has just completed, or attempting to take shared resources at the same time.

Planning. Some planning activities are too high-level to be considered mechanics of collaboration, but others happen repeatedly inside the shared workspace. For example, people divide and redivide the task as they go along, reserve areas of the workspace for their use, or consider various courses of action by simulating them in the workspace (e.g. indicating a path with a pointer before construction begins).

Monitoring. Many of the other mechanics of collaboration rely on the ability to monitor and gather information about others in the workspace. Much of this information is simply workspace awareness information [9]: who is in the workspace, where they are working, and what they are doing. In addition, there are situations where people monitor one another more explicitly. For example, in an apprenticeship situation, the expert must monitor the activities and whereabouts of the novice even if they are not always working in the same place.

Assistance. Group members provide help to one another when it is needed. Assistance may be opportunistic and informal, where the situation makes it easy for one person to help another, or it may be explicitly requested; either way, appropriate help requires that people understand what others are doing and where they are at in their tasks.

Protection. One danger in group work is that others may inadvertently alter or destroy work that you yourself have carried out. People must therefore keep an eye on their own work, noticing what effects others' actions could have and taking actions to prevent certain activity.

4. The Mechanics & Groupware Usability

The mechanics of collaboration, and the concept of teamwork more generally, allow us to state a definition of groupware usability that goes beyond what is normally included in single-user usability studies. We define groupware usability as:

> …the degree to which a groupware system supports the mechanics of collaboration for a particular set of users and a particular set of tasks [8].

This definition assumes that a groupware system is already usable from a single-user perspective, and concentrates specifically on usability aspects of group interaction. From this definition, we can now begin to consider ways of evaluating for the mechanics of groupware specifically, and for groupware usability in general. In particular, we believe that we can test for the mechanics of collaboration by examining if a group can perform them effectively, efficiently, and pleasantly (e.g. [14]). Each is described in turn below.

Effectiveness considers whether the activity was successfully completed, and the number and severity of errors made during that activity. A usable groupware system will not prevent the mechanics of collaboration from taking place, and will not cause group members to make undue errors in those activities.

Efficiency considers the resources (such as time or effort) required to carry out the activity. A good groupware system will allow the activities of collaboration to proceed with less time and effort than will a system with usability problems. Note that any measures of efficiency must be carefully focused on task activities, since groups often engage in off-task activities that are not detrimental to the overall shared work.

Satisfaction considers whether the group members are reasonably happy with the processes and outcomes of each of the activities of collaboration. Satisfaction will sometimes overlap with efficiency and effectiveness (that is, problems in the other areas are likely to reduce satisfaction).

Matching the seven mechanics of groupware against these three criteria measures provides a conceptual framework for evaluating groupware (see Table 1).

	Effectiveness	Efficiency	Satisfaction
Explicit communication			
Consequential communication			
Coordination of action			
Planning			
Monitoring			
Assistance			
Protection			

Table 1. The conceptual framework.

Using this framework, we will briefly revisit several discount usability techniques originally developed and successfully applied for evaluating singleware usability. Others are also pursuing the goal of low-cost evaluation techniques for groupware (e.g.[2]), and these efforts are

complementary. Our approach can be seen as a bottom-up method that originates from a fixed set of face-to-face collaborative behaviours rather from a top-down analysis of groupware features and characteristics.

With each technique, we will ask if it could be used to test if a group can perform a particular mechanical activity effectively, efficiently, and pleasantly. We caution that this discussion is quite preliminary, and we have not yet evaluated these evaluation methods.

5. Revisiting Discount Evaluation Methods

Because we are interested in techniques that can be done rapidly, we do not consider quantitative measures that require experimental methods or extensive analysis (e.g. performance measures or data-log analysis). Instead, we rely on interface inspection techniques such as heuristic evaluation and task-centered walkthroughs (e.g., [13]), observational methods (e.g., [4]) and subjective assessments by realistic participants (e.g. [15,18]). We describe each below, and assume that an evaluation of these interfaces from a single-user perspective (perhaps done earlier or in parallel) have already uncovered and repaired conventional usability problems.

5.1 Heuristic evaluation

Heuristic evaluation is a widely-accepted discount evaluation method for diagnosing potential usability problems in user interfaces. It defines a particular interface inspection process where several evaluators examine an interface and judge its compliance with recognized usability principles called 'heuristics' [12]. Heuristics draw attention to usability problems often found in single user systems, such as how feedback is provided, how errors are minimized, how help is provided, and so on. Non-compliant aspects of the interface are captured as interface bug reports, where evaluators describe the problem, its severity, and perhaps even suggestions of how to fix it.

We can apply heuristic evaluation techniques to groupware usability by replacing the current set of heuristics by rephrasing those activities that comprise the mechanics of collaboration[1]. For example, groupware heuristics can now be statements such as "Provide the means for explicit communication" and "Allow people to monitor and gather information about others in the workspace". The inspector can then judge the interface by seeing if the means for groups to achieve a particular heuristic is available; if the means are there, the inspector can then ask if the group can use it effectively, efficiently, and with satisfaction.

The high-level nature of these heuristics as well as the small number of them (7) fits with Nielsen's (1994) view of how heuristics should be crafted. As with conventional heuristic evaluation, issues to consider include: how inspectors can be trained to the nuances of these heuristics; whether inspectors can use them effectively to uncover interface problems; how many inspectors are needed to discover the majority of interface problems, and whether these heuristics actually cover a large proportion of the usability problems typically found in groupware.

5.2 Walkthroughs

Another inspection technique is based on the notion of an interface walk-through. While there are many variations of how to perform a walkthrough, in all of them the inspector begins with a realistic and detailed task description, a description of the user, and an interface to evaluate. The inspector then 'walks through' the interface step by step by imagining each action the user would take while performing the task on the particular system. During each step the inspector asks a series of questions. In a task-centered walkthrough [11] the questions include:

1. Can you build a believable story that motivates the user's actions?
2. Can you rely on the user's expected knowledge and training about the system?

If the inspector believes that these questions cannot be answered satisfactorily, then a potential interface problem has been located. The inspector notes it, assumes it has been solved, and goes on to the next step.

Other variations of walkthroughs ask different questions, but almost all are concerned with how single users understand the system and use it to achieve a particular goal. In groupware, it may be possible to perform a walkthrough by articulating tasks that exercise the mechanics of groupware, and by asking questions related to the criteria. For example, a detailed task description would now include activities such as 'Saul and Carl are modifying an architectural floor plan. They examine the current floor plan, and discuss what needs to be changed. Carl makes the living room two meters larger. Saul sees this, and suggests that this would compromise the amount of cupboard space…' For each step in this process, the inspector would then ask:

1. Can the person/group perform the activity of groupware implied by this step effectively i.e., does the interface supply the means to do it?
2. Can it be performed efficiently i.e., is it believable that the person/group would go through the effort

[1] We have also developed another set of heuristics based on the Locales Framework that could be applied to groupware [6].

required by this interface to perform this step?

3. Can it be performed with satisfaction i.e., is it believable that the person/group would be motivated to do this step, and would they be happy with the outcome?

Of course, there are several issues. First, it is much easier to define a 'typical person' than it is to define a 'typical group'. The richness and variety of group interactions makes them much harder to typify. Second, we don't know if the task descriptions that drive the walkthrough can be expanded sufficiently to include not only taskwork, but to articulate teamwork as well. Finally, we expect that it will be harder for an inspector to answer the three questions above because of the difficulty of predicting particular group interactions.

5.3 Usability testing through observations

Observational user testing is done by observing how people perform particular tasks on a system in a laboratory setting [4]. The evaluator typically tries to observe where people have problems performing a task, and monitors people's talk to see where their conceptual model of the system is at odds with the actual system model. To get people to talk, the evaluator often asks them to 'talk-aloud' i.e., to say what they are doing as they are doing an action. Alternatively, the evaluator would have two people perform a task together, and monitor their speech for insights into what each was thinking.

Groupware evaluation would be similar, with the exception that the evaluators would be trained to observe and analyze the collaboration through a set of criteria based on the conceptual framework presented in Table 1. This could be done either on-the-fly (perhaps noted on a coding sheet), or after the fact (i.e., through video analysis using video annotation software). For example, the criteria for the first row of Table 1 (explicit communication/effectiveness, efficiency, satisfaction) can be partially restated as:

Intelligibility and interpretability of spoken-written-gestural communication:

a) Did the system make it difficult to hear/read what others were saying/writing?
b) Did the system make it difficult to understand what others were saying/writing?
c) Did the system make it difficult to gesture and refer to items in the workspace?
d) Did the system make it difficult to see and understand what others were pointing to?

Of course, there are several issues with this method. First is the difficulty of acquiring suitable people for observations: it is harder to schedule, and harder to predict the expected group interaction. If the software expects a certain level of intimacy between collaborators, then the subject pool may be quite small. Second is the difficulty of interpreting a scene based on these questions. While people may be having problems, groups are remarkably resilient at adapting their interactions to succeed in even awkward collaborative situations.

5.4 User questionnaires

Several researchers advocate evaluation through questionnaires that are filled in by the people using the system (e.g. [15,18]). This could be done after a usability observation (as described above), where the participants would complete the questionnaire. The evaluators could then conduct a group interview to follow up both their observations and the questionnaire answers, perhaps including discussions of possible solutions to the problems.

Similar to our other methods, we would create a groupware questionnaire by organizing it around the seven activities of collaboration. These include questions that consider effectiveness, efficiency, and satisfaction in each area. Since the data is largely subjective, observations or answers should be used as indications that problems may exist in a particular area rather than definitive assessments; agreement between multiple participants or observers, of course, is a stronger indication of a problem.

The questionnaire would be similar to the ones used by evaluators (Section 5.3) but rephrased to be answered from the person's experiences. For example, the criteria for the first row of Table 1 can be partially restated as:

Intelligibility and interpretability of spoken-written-gestural communication

a) It was easy to hear/read what other people were saying/writing
b) It was easy to understand what others were saying/writing
c) It was easy to gesture and refer to items in the workspace
d) It was easy to see and understand what others were pointing to

Issues here are similar to those discussed in Section 5.3. Additional issues are that we have to pay particular attention to the wording of these questions (both to make them understandable and to reduce any implicit bias), and that we cannot afford too many questions as this may make people reluctant to answer them.

6. Conclusions

In this paper, we introduced a conceptual framework for developing discount usability evaluation techniques that can be applied to shared-workspace groupware. The

framework is based on support for the mechanics of collaboration: the low level actions and interactions that must be carried out to complete a task in a shared manner. These include communication, coordination, planning, monitoring, assistance, and protection. The framework also includes gross measures of these mechanics: effectiveness, efficiency, and satisfaction. The underlying idea of the framework is that some usability problems in groupware systems are not inherently tied to the social context in which the system is used, but rather are a result of poor support for the basic activities of collaborative work in shared spaces. We believe evaluation schemes based on this framework will occupy a middle ground between brittle experimental techniques and time-consuming field techniques, where they will provide the kind of formative information valuable in an iterative groupware development process.

This is initial work. We have performed only limited testing of how well particular evaluation methods can be adopted to the framework in Table 1. In particular, we have drafted a set of detailed questions to drive user observations and user questionnaires, available from www.cs.usask.ca/projects/hci/. Early indications are that the scheme does provide people with a framework for considering issues that they would have otherwise missed. However, the questionnaires are still being revised, particularly to reduce the number of questions and to improve the clarity of the questions.

As our work matures, we plan to evaluate the various groupware evaluation methods in two ways. First, a particular testing scheme will be applied to a system with known groupware usability problems; afterwards, we will analyze how well evaluators using the scheme could uncover particular problems. In this approach, the testing scheme could also be compared with other evaluation techniques; if a scheme can identify problems that other methods cannot, then the scheme has some value. Second, we will use the various schemes in the context for which it was intended—iterative design of groupware. We will observe its use on a realistic development project, and determine whether the development team finds the scheme to be useful in improving the usability of the final product.

Acknowledgments. We gratefully acknowledge the support of the National Institute of Standards and Technology (NIST), Microsoft Research, and the Natural Sciences and Engineering Research Council (NSERC).

[1] Clark, H. (1996) *Using Language*. Cambridge University Press, Cambridge.

[2] Drury, J., Damianos, L., Fanderclai, T., Kurtz, J., Hirschman, L., and Linton, F. (1999) Methodology for Evaluation of Collaborative Systems. Technical Report (version 4.0), The Evaluation Working Group of the DARPA Intelligent Collaboration and Visualization Program.

[3] Dix, A., Finlay, J., Abowd, G., and Beale, R. (1993) *Human-Computer Interaction*, Prentice Hall.

[4] Dumas, J.S. and Redish, J.C. (1993) *A Practical Guide to Usability Testing.* Ablex, 1993.

[5] Fitzpatrick, G. (1998) *The Locales Framework: Understanding and Designing for Cooperative Work.* PhD Thesis, Department of Computer Science and Electrical Engineering, The University of Queensland

[6] Greenberg, S., Fitzpatrick, G., Gutwin, C. and Kaplan, S. (1999). Adapting the Locales Framework for Heuristic Evaluation of Groupware. *Proceedings of OZCHI'99 Australian Conference on Computer Human Interaction*, Wagga Wagga, NSW Australia, November 28-30.

[7] Grudin, J. (1990) Groupware and cooperative work: Problems and prospects. In *The Art of Human Computer Interface Design*, B. Laurel, Ed. Addison-Wesley, 171-185.

[8] Gutwin, C. and Greenberg, S. (1999). The Effects of Workspace Awareness Support on the Usability of Real-Time Distributed Groupware. *ACM Transactions on Computer-Human Interaction (TOCHI)* 6 (3), 243-281.

[9] Gutwin, C. and Greenberg, S. (1999) A Framework of Awareness for Small Groups in Shared-Workspace Groupware. Research Report 99-2, Computer Science Department, University of Saskatchewan.

[10] Hughes, J., King, V., Rodden, T. and Andersen, H. (1994) Moving out of the control room: Ethnography in system design. In *Proc CSCW'94*, p429439, ACM Press.

[11] Lewis & Reiman (1993) *Task Centered User Interface Design*. From ftp.cs.ucolorado.edu/pub/cs/distribs/clewis/

[12] Nielsen, J. (1994) Heuristic Evaluation. In Nielsen and Mack (1994)

[13] Nielsen, J. and Mack, R. (editors) (1994) *Usability Inspection Methods.* Wiley and Sons.

[14] Olson, J. S., Olson, G. M., Storrosten, M., and Carter, M. (1992). How a group-editor changes the character of a design meeting as well as its outcome, Proc. CSCW'92, Toronto, Ontario, 1992, 91-98.

[15] Ravden, S. and Johnson, G. (1989) Evaluating Usability of Human-Computer Interfaces. John Wiley & Sons.

[16] Rogers, Y. and Bellotti, V. (1997) Grounding blue-sky research: How can ethnography help? *Interactions*, 4(3), 58-63.

[17] Segal, L. (1995) Designing Team Workstations: The Choreography of Teamwork, in *Local Applications of the Ecological Approach to Human-Machine Systems*, P. Hancock, J. Flach, J. Caird and K. Vicente ed., 392-415, Lawrence Erlbaum, Hillsdale, NJ.

[18] Shneiderman, B. (1997) Designing the User Interface. Addison-Wesley, 1997.

[19] Steves, M., and Knutilla, A. (1999) Collaboration Technologies for Global Manufacturing, Proceedings of ASME Symposium on Manufacturing Logistics in a Global Economy, Nashville TN.

[20] Tang, J. (1991) Findings from Observational Studies of Collaborative Work, *International Journal of Man-Machine Studies*, 34(2), 143-160.