Empirical Development of a Heuristic Evaluation Methodology for Shared Workspace Groupware

Kevin Baker and Saul Greenberg
Department of Computer Science
University of Calgary
Calgary, Alberta, Canada T2N 1N4
{bakerkev,saul}@cpsc.ucalgary.ca

Carl Gutwin
Department of Computer Science
University of Saskatchewan
Saskatoon, Saskatchewan, Canada S7N 5A9
carl.gutwin@usask.ca

ABSTRACT
Good real time groupware products are hard to develop, in part because evaluating their support for basic teamwork activities is difficult and costly. To address this problem, we are developing discount evaluation methods that look for groupware-specific usability problems. In a previous paper, we detailed a new set of usability heuristics that evaluators can use to inspect shared workspace groupware to see how they support teamwork. We wanted to determine whether the new heuristics could be integrated into a low-cost methodology that parallels Nielsen's traditional heuristic evaluation (HE). To this end, we examined 27 evaluations of two shared workspace groupware systems and analysed the inspectors' relative performance and variability. Similar to Nielsen's findings for traditional HE, individual inspectors discovered about a fifth of the total known teamwork problems, and that there was only modest overlap in the problems they found. Groups of three to five inspectors would report about 40–60% of the total known teamwork problems. These results suggest that heuristic evaluation using our groupware heuristics can be an effective and efficient method for identifying teamwork problems in shared workspace groupware systems.

Keywords: Heuristic evaluation, groupware usability

INTRODUCTION
Commercial real-time distributed groupware is now readily available due to improvements in hardware, network connectivity, and the demands of increasingly distributed organizations. Yet with the exception of games and instant messaging, most real-time groupware is not widely used. One reason for this is that groupware has serious usability problems in how they support group work—collaborative systems are, at best, awkward to use [7].

The poor usability of current groupware results in part from the lack of practical and inexpensive groupware evaluation methodologies [6]. The CSCW community has yet to develop and validate techniques that make groupware evaluation cost-effective within typical software project constraints; most existing methods are too expensive and are rarely seen outside of research projects [15].

One way to address the paucity of groupware evaluation techniques is to adapt accepted low-cost "discount" methods developed for single-user software usability [10] e.g., usability observation, and inspection methods including heuristic evaluation and walkthroughs. Although these techniques have been extremely successful in improving the usability of traditional software, they cannot be applied unaltered in the groupware context.

The problem is that standard HCI methods focus on the *taskwork* aspects of an interactive system, but a main part of groupware usability is the support provided for *teamwork*—the 'work of working together.' Traditional methodologies will not uncover usability problems in teamwork support. For example, inspection methods have evaluators examine an interface for usability bugs according to a set of criteria [13], but these criteria do not assess the teamwork components necessary for effective collaboration in groupware.

The goals of the discount approach are still worth pursuing, but the techniques themselves require considerable adaptation. We are currently involved in several projects to do just this. Our initial focus is on methods for evaluating distributed real-time shared-workspace groupware—systems that allow remote collaborators to work together over a visual work surface. We have based our adapted techniques on a framework that reflects the ways that group work actually gets carried out in real-world visual workspaces. This framework sets out the *mechanics of collaboration*—the small-scale actions and interactions that group members must perform in order to get a task done in a collaborative fashion [8]. Using the mechanics, we have previously adapted methods from cognitive walkthrough [15] and simple inspection [17].

This paper concerns our adaptation of Nielsen's *heuristic evaluation methodology* (abbreviated HE) [9,11,12,14] to groupware. We previously introduced a new set of groupware heuristics based on the mechanics of collaboration (summarized in Table 1), and we determined that these heuristics can help evaluators identify usability problems specific to teamwork [2]. Now, we consider the problem of integrating these heuristics into an actual HE

CSCW'02, November 16–20, 2002, New Orleans, Louisiana, USA.

methodology. If the heuristics can form the basis of an inexpensive yet effective evaluation methodology, then there is considerable potential for using the method to improve the usability of commercial groupware systems.

In our analysis, we were particularly interested in whether different evaluators perform similarly using the heuristics, whether evaluator performance varies across separate evaluations, and most importantly, whether a small number of evaluators can find a large proportion of the teamwork support problems in a groupware system. We asked 27 evaluators to assess two shared-workspace groupware systems with the new heuristics, and we analysed their output in ways similar to Nielsen's original analyses of traditional HE [9,12,14]. Our results indicate that HE using our new groupware heuristics can be an effective and cost-effective tool for finding a certain class of groupware usability problems.

HEURISTIC EVALUATION

Heuristic evaluation is a widely accepted discount evaluation method for diagnosing potential usability problems in user interfaces [9,11,12,14]. HE has a small set of *inspectors*—usability experts acting as interface evaluators—visually inspect an interface and judge its compliance with *heuristics*: usability principles that describe common properties of usable interfaces. Heuristics help inspectors focus their attention on aspects of an interface that are often trouble spots, making detection of usability problems easier. These raw usability problems are then collected and, through a process called *results synthesis*, transformed into a cohesive set of problem reports that are passed onto developers [4].

HE is popular with both researchers and industry. It is low cost in terms of time since it can be completed in a relatively short amount of time (i.e. a few hours). End-users are also not required; therefore, resources are inexpensive. Because heuristics are well documented (e.g., [12]), they are easy to learn and apply; even non-usability experts can use them with some success. It has a low cost to benefit ratio, where only 3-5 experienced inspectors are needed to identify ~75-80% of all usability problems [12].

HEURISTICS BASED ON THE MECHANICS OF COLLABORATION

We have developed groupware-specific usability heuristics that inspectors can use to carry out heuristic evaluations of teamwork support in groupware. We started this process by proposing five groupware heuristics [5] for evaluating general groupware environments based on the Locales Framework. We then narrowed our focus to shared visual workspaces, where we suggested a new set of heuristics [2] based on a theoretical framework called the *mechanics of collaboration* [8]. This framework was developed from an analysis of shared workspace usage and theory e.g., [3,7,18]. The mechanics describe the low level actions and interactions that small groups do if they are to complete a task effectively, such as communication, planning, monitoring, assistance, coordination, and protection. The underlying idea is that while some usability problems in groupware are tied to social or organizational issues in which the system has been deployed, others are a result of poor support for the basic collaborative activities in shared spaces. It is these activities that the framework articulates.

While the framework was developed with low-cost evaluation methods in mind, we had to adapt, restructure and augment it in order to rephrase it as a list of heuristics. Unlike single user heuristics which are somewhat independent—chosen by how well they identified 'standard' usability problems [11]—ours have the advantage that they are linked and interdependent as they collectively describe a partial framework of attributes of how people interact with shared visual workspaces. With these new heuristics, we believe that inspectors can evaluate how well groupware supports the ability of distributed people to communicate and collaborate with artifacts through an electronic shared visual medium.

A detailed description of these heuristics is found in [1,2], including the ways they are supported by standard groupware implementation practices, and including citations to the literature. For convenience, the heuristics are summarized in Table 1 but are not replicated in detail due to our space constraints.

EVALUATING WHETHER THE NEW HEURISTICS MAKE A METHODOLOGY

We wanted to determine whether the new groupware heuristics could be integrated into a low-cost evaluation methodology that parallels Nielsen's traditional heuristic evaluation. To this end, we examined 27 inspectors' relative performance and variability in identifying problems in two collaborative applications: GroupDraw and Groove. We want to emphasize that this paper is *not* a usability study of those systems; we do not report the actual problems found or make any statements about their design. Rather, this paper concerns *how well* evaluators could find usability bugs within them using the new heuristics.

Our methodology, terminology and analysis mirrors that used by Nielson [9,12,14] in his validation of the traditional HE process.

Participants

We recruited two categories of evaluators.

- *Novice evaluators* are groupware "novices" in that they lack substantive knowledge regarding CSCW principles but have reasonable HCI background. We recruited 16 computer science undergraduate students who were halfway through a second HCI course. We set this groupware evaluation as an assignment.
- *Regular specialists* were knowledgeable and experienced in both HCI and CSCW. We recruited 2 professors and 9 graduate students who had a history of research, applied work, and/or class work in CSCW and HCI. All were knowledgeable of groupware fundamentals.

Heuristic 1: Provide the means for intentional and appropriate verbal communication
The prevalent form of communication in most groups is verbal conversations. The mechanism by which we gather information from verbal exchanges has been coined *intentional communication* and is used to establish a common understanding of the task at hand. Intentional communication usually happens in one of three ways.

1. People talk explicitly about what they are doing and where they are working within a shared workspace.
2. People overhear others' conversations.
3. People listen to the running commentary that others tend to produce alongside their actions.

Heuristic 2: Provide the means for intentional and appropriate gestural communication
Explicit gestures are used alongside verbal exchanges to carry out intentional communication as a means to directly support the conversation and convey task information. Intentional gestural communication can take many forms. *Illustration* occurs when speech is acted out or emphasized, e.g., signifying distances with a gap between your hands. *Emblems* occur when actions replace words, such as a nod of the head indicating 'yes'. Deictic reference or *deixis* happens when people reference workspace objects with a combination of intentional gestures and voice communication, e.g., pointing to an object and saying "this one".

Heuristic 3: Provide consequential communication of an individual's embodiment
A person's body interacting with a physical workspace is a continuous and immediate information source with many degrees of freedom. In these settings, bodily actions *unintentionally* "give off" awareness information about what's going on, who is in the workspace, where they are, and what they are doing. This visible activity of unintentional body language and actions is fundamental for creating and sustaining teamwork, and includes:

1. *Actions coupled with the workspace* include: gaze awareness (where someone is looking), seeing someone move towards an object, and hearing sounds as people go about their activities.
2. *Actions coupled to conversation* are the subtle cues picked up from our conversational partners that help us continually adjust our verbal behaviour. Cues may be visual (e.g., facial expressions), or verbal (e.g., intonation and pauses). These provide conversational awareness that helps us maintain a sense of what is happening in a conversation.

Heuristic 4: Provide consequential communication of shared artifacts (i.e. artifact feedthrough)
Consequential communication also involves information *unintentionally* given off by physical artifacts as they are manipulated. This information is called feedback when it informs the person manipulating the artifact, and *feedthrough* when it informs others who are watching. Seeing and hearing an artifact as it is being handled helps to determine what others are doing to it. Identifying the person manipulating the artifact helps to make sense of the action and to mediate interactions.

Heuristic 5: Provide Protection
Concurrent activity is common in shared workspaces, where people can act in parallel and simultaneously manipulate shared objects. Concurrent access of this nature is beneficial; however, it also introduces the potential for conflict. People should be protected from inadvertently interfering with work that others are doing, or altering or destroying work that others have done. To avoid conflicts, people naturally anticipate each other's actions and take action based on their predictions of what others will do in the future. Therefore, collaborators must be able to keep an eye on their own work, noticing what effects others' actions could have and taking actions to prevent certain activities. People also follow social protocols for mediating their interactions and minimizing conflicts. Of course, there are situations where conflict can occur (e.g., accidental interference). People are also capable of repairing the negative effects of conflicts and consider it part of the natural dialog.

Heuristic 6: Manage the transitions between tightly and loosely-coupled collaboration
Coupling is the degree to which people are working together. It is also the amount of work that one person can do before they require discussion, instruction, information, or consultation with another. People continually shift back and forth between *loosely*- and *tightly-coupled collaboration* where they move fluidly between individual and group work. To manage these transitions, people need to maintain awareness of others: where they are working and what they are doing. This allows people to recognize when tighter coupling could be appropriate e.g., when people see an opportunity to collaborate or assist others, when they need to plan their next activity, or when they have reached a stage in their task that requires another's involvement.

Heuristic 7: Support people with the coordination of their actions
An integral part of face-to-face collaboration is how group members mediate their interactions by taking turns and negotiating the sharing of the common workspace. People organize their actions to help avoid conflict and efficiently complete the task at hand. Coordinating actions involves making some tasks happen in the right order and at the right time while meeting the task's constraints. Within a shared workspace, coordination can be accomplished via explicit communication and the way objects are shared. At the fine-grained level, awareness helps coordinate people's actions as they work with shared objects e.g., awareness helps people work effectively even when collaborating over a very small workspace. On a larger scale, groups regularly reorganize their division of labour based on what the other participants are doing and have done, what they are still going to do, and what is left to do in the task.

Heuristic 8: Facilitate finding collaborators and establishing contact
Most meetings are *informal*: unscheduled, spontaneous or one-person initiated. In everyday life, these meetings are facilitated by physical proximity since co-located individuals can maintain awareness of who is around. People frequently come in contact with one another through casual interactions (e.g. bumping into people in hallways) and are able to initiate and conduct conversations effortlessly. While conversations may not be lengthy, much can occur such as coordinating actions and exchanging information. In electronic communities, the lack of physical proximity means that other mechanisms are necessary to support awareness and informal encounters.

Table 1. The groupware heuristics.

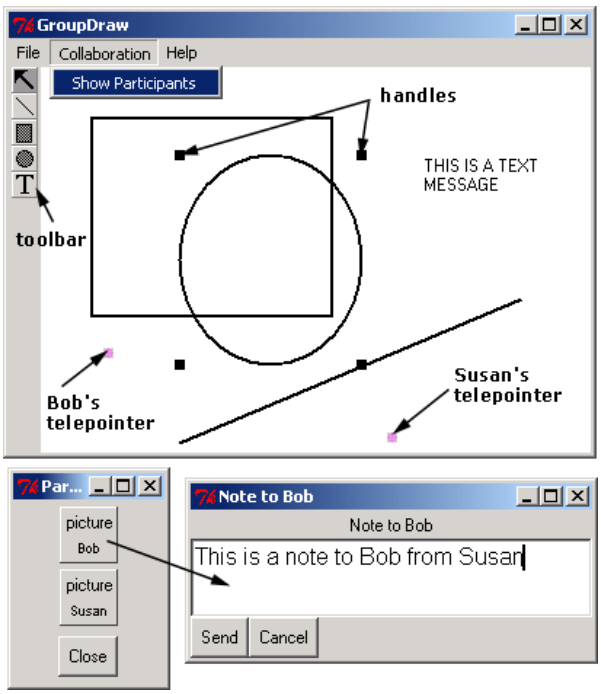

Figure 1. GroupDraw's shared workspace & notes system

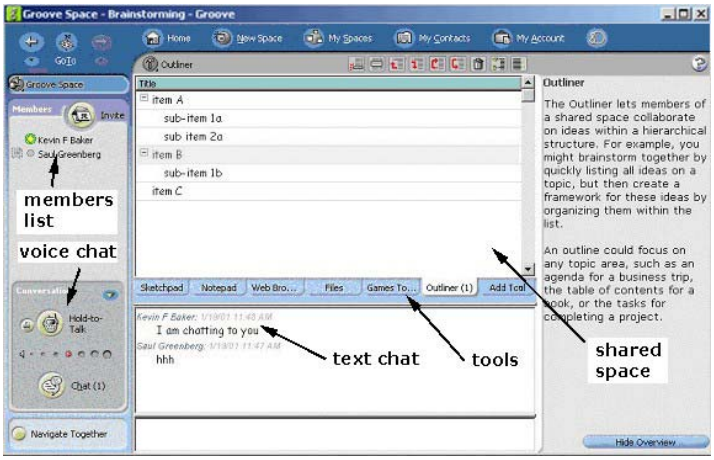

Figure 2. Groove and its outliner tool

Groupware systems

Participants evaluated two quite different shared visual workspaces contained in two real-time groupware systems: GroupDraw and Groove. These are briefly described here; a more thorough description is available in [1].

GroupDraw is an object-oriented 'toy' drawing program developed in our laboratory [16]; thus we were familiar with its functionality and had some a priori knowledge of its problems. Figure 1 illustrates a GroupDraw session, where two people are sketching a rudimentary drawing within the shared workspace. Participants can work simultaneously, where anyone can create, move, resize, and delete drawing objects at any time. As seen in the figure, GroupDraw supports multiple active cursors, displayed as small cross-hairs. Participants can communicate with one another by exchanging text notes through the notes subsystem. While GroupDraw has other features, we asked inspectors to evaluate only the shared workspace and the Notes functionality.

Groove (www.groove.net) is a commercially available professional product that provides a virtual space for real-time, small group interactions. Participants create shared spaces to communicate and collaborate with one another. Changes made to a shared space by one participant are automatically synchronized with all other computers. Figure 2 is a snapshot of a shared workspace called 'Groove Space'. The left side lists the space's members and their presence (Kevin is present in the space whereas Saul is not logged on). We see the Outliner tool that participants use to brainstorm and hierarchically organize ideas in real-time. Participants can switch tools via the Tools tab. Real-time communication is done via Groove's voice or chat facility in the bottom portion of the screen. While Groove contains many more tools and features, we asked inspectors to evaluate only the Outliner tool, the text chat and audio link. We had no prior experience with Groove and had no preconceived notion of its effectiveness as groupware.

Method for the heuristic evaluation

Training packet. In preparation for the training session, we first distributed and asked the inspectors to review a training packet (available in [1]) containing a detailed written description of the groupware heuristics.

Training session. Next, inspectors attended a one-hour training session on the groupware heuristics. This included a review of each heuristic, how to apply them during an evaluation, and real-time groupware examples that illustrated compliance and non-compliance with each heuristic. The training was to ensure that everybody had a good understanding of the principles behind each heuristic and how to use them. We also highlighted (but did not judge) the pertinent features of GroupDraw and Groove, and indicated what parts of the system they should inspect.

Evaluation Process. We gave inspectors only general instructions for conducting the evaluation, leaving them considerable freedom on how they performed it (which reflects real world practice). This is similar to Nielson's evaluations of traditional HE.

Each inspector decided when, where, and how they would perform the evaluation. They self-determined the length of time and amount of effort they would put into completing the evaluation. While inspectors could choose to work together (having another person at the other end of the groupware system is helpful to exercise its functionality), we asked them to minimize their discussions about the problems they saw.

Data collection. We gave inspectors a prepared stack of problem report forms that they could fill in, one per problem. For each problem, the inspectors recorded a description of the problem, the violated heuristic, a severity rating, and an (optional) solution to the problem. They judged a 'major' severity rating as one that represented a significant obstacle to effective collaboration, while a 'minor' rating is one that could be worked around by the participant.

ANALYSIS 1: CATEGORIZING FOUND PROBLEMS

In the first part of our analysis, we wanted to answer two questions.

1. Do the heuristics help inspectors find teamwork problems?
2. If we distill the lists of raw problems collected across all 27 inspectors into a single aggregated list of distinct total known teamwork problems, how do these two collections compare with each other?

Individual inspectors produced independent lists, each containing many problem reports. These reports cannot be compared directly because inspectors often identify the same problem with different terminology and/or different levels of abstraction. Thus we needed to analyze and transform the larger number of raw problem reports collected from individual inspectors into a form where they could be directly comparable with each other and meaningful to analyze. Using a method called *results synthesis* [4], we distilled the raw problem reports into a concise aggregated list of *known teamwork usability problems* for GroupDraw and Groove.

Analysis Method

Step 1. If an inspector listed multiple problems in a single problem report, we separated them into one problem per report.

Step 2. We classified each problem into one of the following categories:

- *Raw teamwork problem*: a usability problem that can be categorized according to one of the groupware heuristics;
- *Out of scope*: a problem that is not categorized according to the groupware heuristics;
- *False positive*: a 'problem' that turned out not to be a usability problem at all.

Out of scope and false positives were counted and culled out of the set.

Step 3. We grouped together obvious duplicate problems and treated them as a single entity. If different inspectors classified the duplicate problem under different heuristic labels, we relabeled it with the best matching heuristic.

Step 4. We then grouped the teamwork usability problems according to the eight groupware heuristics.

Step 5. Within each heuristic group, we grouped together similar problems. This is somewhat difficult, as inspectors describe problems in their own unique way; it is not always immediately apparent that multiple inspectors are in fact addressing the same problem [4]. Inspectors may use different terminology for the same situation. They may identify different symptoms for the same fundamental problem, or describe them at quite different levels of abstraction. For example, two inspectors reported these two teamwork problems about Groove:

1. The system does not support consequential communication of an individual's embodiment.
2. I don't have a clue where somebody is within the shared space and what they are doing.

At a high level, both problems address the same issue; however, the level of abstraction is different. The first describes the problem's root cause (its complete lack of compliance with heuristic 3), while the second provides the symptoms or consequences for not supporting embodiment. Still, they are obviously related, so we would group these together as a single teamwork usability problem. In practice, we described each grouping as a single problem in terms of its symptoms rather than its root cause.

Step 6. Based on the ratings and argumentation of the inspectors, we rated the final teamwork usability problems as major or minor.

Results

Table 2 breaks down the inspectors' original problem reports into our three categories. The 'Raw teamwork problems', 'Out of scope' and 'False positives' results are those at the end of step 2 (duplicates are not removed). The 'Total known teamwork problems' are the consolidated problems that are the outcome of the entire results synthesis process after Step 6. We caution that the total known problems are not necessarily the complete set of teamwork problems for GroupDraw and Groove, for it is impossible to know for sure whether every single usability problem has been uncovered.

System	Raw teamwork problems	Out of scope	False positives	Total known teamwork problems
GroupDraw	321	15	14	62
Groove	331	21	15	43

Table 2: Breakdown of problem reports

Adding together the three first columns in Table 2, we see that inspectors recorded a significant number of problem reports for GroupDraw and Groove—over three hundred for each. Relatively few of these reports contained out of scope problems or false positives. After consolidation, the 321 raw teamwork problems for GroupDraw and 331 for Groove were distilled into 62 unique and distinct problems for GroupDraw and 43 for Groove (the actual problems and their consolidation are catalogued in [1]).

Discussion

Do the heuristics help inspectors find teamwork problems related to the support of real-time collaboration within a shared workspace? The answer is a clear 'yes' as indicated by the significant number of teamwork problems for GroupDraw and Groove. The small number of 'Out of scope' and 'False positives' indicate that inspectors were focused on collecting groupware usability problems.

How do the 27 different lists of raw teamwork-based problem reports translate into a final list of consolidated teamwork problems? From Table 2, we see 62:321 (GroupDraw) and 43:331 (Groove) final to initial problems. Thus the ratio of consolidated problems to original raw problem reports is about ~1:6 i.e., there is quite a bit of redundancy in the problems found by inspectors. This is in

keeping with traditional outcomes of results synthesis in HE [4].

ANALYSIS 2: INSPECTOR PERFORMANCE

We can conclude so far that a large group of inspectors using the groupware heuristics will identify a significant number of teamwork problems. We also see that there is much redundancy in the problems found, for the reduced final problem list is a sixth of the initial list. This leads to the following important hypothesis: only a few inspectors are needed to find a good number of teamwork problems. If this hypothesis is true, then we would have identified a low-cost groupware evaluation methodology.

We can recast this hypothesis into several fine-grained questions that we can answer by analyzing our data.

1. Can we rely on a single inspector to uncover many problems? How well do individual evaluators perform?
2. Are inspectors consistent i.e., will an inspector who is 'good' at identifying teamwork problems in one system also be 'good' at identifying problems in another?
3. Are problems equally likely to be found by most inspectors, or are some problems only found by a few?
4. Are problems that are generally hard to identify only found by those inspectors who uncover many problems, or can 'weaker' inspectors who find relatively few problems find them as well?
5. How many inspectors do we need to use if we are to uncover a good number of problems i.e., what is the cost to benefit ratio in terms of inspectors used *vs* problems found?
6. How do inspectors perform in terms of uncovering major *vs* minor problems?

To answer these questions, we replicate Nielsen's analysis methodology [9,12, 14] on our data. In general, we scored inspectors by matching the problems that each individual inspector uncovered in their heuristic evaluations against the final compiled list of known teamwork usability problems. Where reasonable, we compare our results with Nielsen's analysis of traditional HE.

Average Performance of Individual Evaluators

Can we rely on a single inspector to uncover many problems? We answer this by examining the number of teamwork problems found by each inspector. Table 3 breaks down the average performance of individual evaluators, categorized according to the groupware systems (GroupDraw *vs* Groove) and the inspector type (novice *vs* regular), as well as combined results. Figure 3 graphs all this data, where it compares the proportion of inspectors and the proportion of the total number of teamwork problems they found. We also overlay Nielsen's data on traditional HE collected from 77 inspectors of the Mantel system and 34 inspectors of the Savings system.

Results. On average, novice inspectors found ~24% of the total known usability problems and regular specialists found ~19%. As a comparison, Nielsen's novice inspectors and regular specialists uncovered an average of 22% and 35% of known usability problems [9].

The tables and graphs show quite a difference between individual performances; the best novice evaluator uncovered ~41% of all the known usability problems whereas the worst found only ~11%. Similarly, the best regular specialist found ~44% of the problems and ~9% for

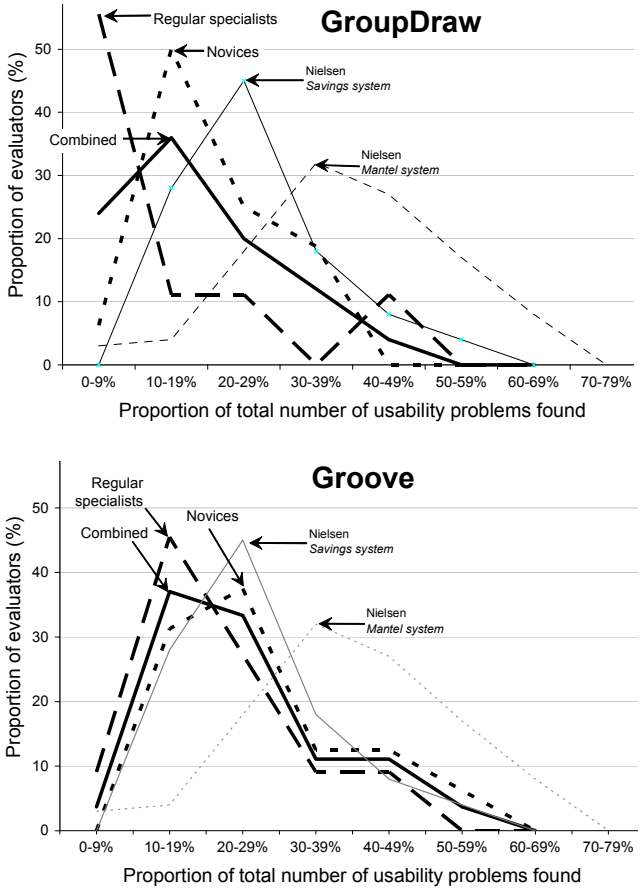

Figure 3. Distributions of the proportion of teamwork problems found by evaluators

System	Inspectors	Total known teamwork problems	Best evaluator (%)	Worst evaluator (%)	Average problems found (%)	Standard deviation
Group Draw	16 novice	62	33.9	8.1	20.3	7.5
	9 regular*	62	40.3	1.6	14.0	12.3
	All	62	40.3	1.6	18.0	10.0
Groove	16 novice	43	53.5	16.3	27.9	11.0
	11 regular	43	46.5	9.3	22.2	9.5
	All	43	53.5	9.3	25.6	10.8
Both	All novice	105	41.0	11.4	24.1	10.3
	All regular	105	43.9	8.6	18.5	11.9
	All	105	43.9	8.6	21.9	10.2

**two regular evaluators did not do the GroupDraw evaluation*

Table 3. Individual differences in evaluator's ability to find usability problems

the worst. Figure 3 provides a clearer picture of the actual (roughly normal) performance distribution: the majority of the evaluators do about average. Only a few do quite well, and only a few do quite poorly.

Our groupware inspectors do well. However, their performance is somewhat on the lower end of Nielsen's evaluators for traditional HE, as seen by his overlaid data on Figure 3 as well as the data he reports in four additional system evaluations [9,14].

Discussion. Although using one inspector is clearly better than using none, we cannot rely on a single inspector uncovering many of the teamwork problems (as only about 1/5th to ¼ are found).

These findings are in line with those found in traditional HE, where one evaluator typically uncovers only a modest number of problems compared to the total found problems. However, our results also show that inspector performance with the groupware heuristics is somewhat less than inspectors doing traditional HE. Possible reasons for this lesser performance include:

1. Inspectors may find our current version of the heuristics more difficult to learn and apply when compared to normal heuristics.
2. Inspectors (even our regular specialists) did not have anywhere near the same exposure and personal experiences with groupware as compared to traditional inspectors' exposure to single user systems.

One unexpected outcome visible in Table 3 and Figure 3 was that regular specialists found on average less problems than the novice inspectors. We will defer our discussion of why this occurred until the end of this paper. For now, we will continue to break down the performance between the two types of inspectors but we will not speculate on why this difference exists.

Consistency of individual inspector performance

Are inspectors consistent in their performance? That is, is an individual inspector's ability to uncover problems with one system consistent with his or her performance when evaluating another system? If it is, then perhaps we could identify a few 'good' inspectors and use them instead of many 'poor' inspectors.

To answer this, we correlated and plotted the proportion of the total teamwork problems found by the each evaluator for each system.

Results. Figure 4 plots the results as a scatterplot, and also draws the best-fit linear regression line. The correlation coefficient was r^2=0.63.

Discussion. Inspectors are somewhat consistent in their performance across systems, as suggested by the modest correlation of r^2=0.63. That is, if we divide the total number of inspectors into thirds, the individuals making up the top third in terms of performance on GroupDraw are roughly the same top third performers for Groove. The same can be said with the bottom two thirds of inspectors. However, inspectors still exhibit large variability. Thus while some people are better than others at doing heuristic evaluation of user interfaces, their performance cannot be guaranteed from one system to the next.

Our correlation results are somewhat stronger than Nielsen and Molich's [14] analysis of traditional HE, where they report only a weak correlation (r^2=0.33).

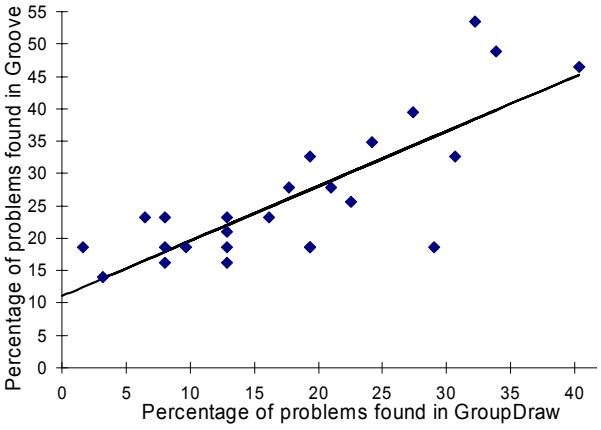

Figure 4: Scatterplot of the proportion of usability problems found by the same inspector in both systems

Proportion of inspectors who found each problem

Are problems equally likely to be found by most inspectors, or are some problems only found by a few? To answer this, we counted the number of inspectors who found each teamwork problem. Figure 5 plots the results for both systems. The x-axis sorts each problem from 'hard to find' (where few inspectors found them) to 'easy to find' (where many inspectors found them). The y-axis shows the proportion of inspectors who actually found each problem.

Results. In both plots, we see tremendous variation in the proportion of inspectors who found particular problems. At the extremes, the 'hard to find' problems were found by only 4% of the inspectors, while the 'easy to find' were

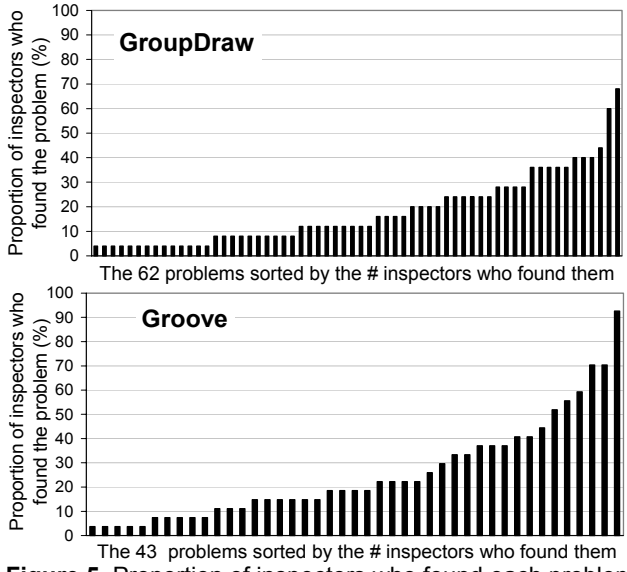

Figure 5. Proportion of inspectors who found each problem

noticed by 68% (for GroupDraw) and 93% (for Groove) of the inspectors.

Discussion. Teamwork usability problems differ considerably in how many inspectors are likely to find them, which implies that some problems are harder to find than others. Thus we should expect that some problems will go unnoticed during a HE of an interface, particularly if few inspectors are used. There is nothing unusual about this result: in one system, Nielsen reports that less than 15% of inspectors (the novices and regular specialists combined) doing traditional HE found the 'hardest' problems, while more than 70% of them found the 'easiest' ones [9].

Ranking of individual performance

Are problems that are generally hard to identify only found by those 'good' inspectors who tend to uncover many problems, or can 'weaker' inspectors find them as well?

To help answer this question, we plotted our data in Figure 6 as follows. First, each column corresponds to one inspector and the problems he or she found (marked in black). Inspectors are sorted from left to right according to the number of problems each had found i.e., the 'worse' inspector on the left found few problems, while the 'best' inspector on the right found many problems. Second, each row represents a usability problem and identifies the inspectors who found it. Problems are sorted from bottom to top by how 'hard' they were to uncover i.e., the 'easiest' problems on the bottom are found by many inspectors, while the 'hardest' problems on the top are found by few inspectors. Filled squares represent that the evaluator assigned to the column found the problem assigned to that row.

Results. Our graphs and results look very similar to those found in Nielsen's analysis of traditional HE [9,12]. This graphic tells us several things.

1. While there is some overlap, inspectors tend to find different usability problems. This is indicated by the considerable differences between the columns.
2. 'Good' inspectors may sometimes overlook 'easy to find' problems, as indicated by the unmarked squares in the lower right quadrants.
3. 'Weak' inspectors may sometimes discover 'hard to find' problems, as indicated by the marked squares in the upper left quadrants.

Discussion. Again, it seems that relying on a single inspector to do the groupware evaluation is far from optimal. Using more than one evaluator is likely better, both because there is only modest overlap in the problems they find, and because even 'good' evaluators can overlook problems that are rated as 'easy to find'. We don't have to use only good inspectors, for even weaker inspectors who find few problems can uncover difficult problems overlooked by others. Consequently, we argue that multiple inspectors will provide better overall performance than using a single 'best' inspector i.e., the combined wisdom of multiple evaluators is likely better than that of the best evaluator.

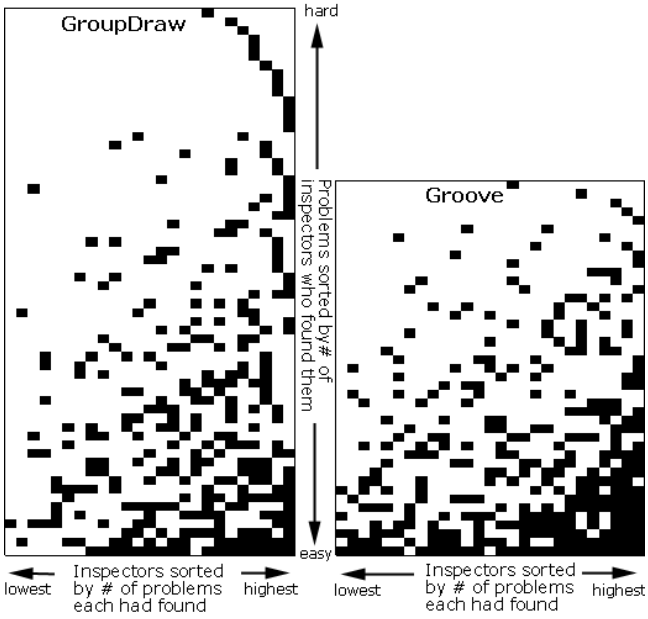

Figure 6: Problems found by each type of evaluator for both systems

Performance of aggregates of inspectors

How many inspectors do we need to use if we are to uncover a good number of problems i.e., what is the cost to benefit ratio in terms of inspectors used *vs* problems found?

The naïve approach is to use as many inspectors as possible; however, this will compromise the methodology's discount status due to the increased cost and preparation time. Ideally, just a few inspectors are needed to find many problems, which would mean that the HE will have a good cost/benefit ratio [14].

To measure this cost/benefit, we formed aggregates of inspectors of varying numbers, where we took representative samples of novice inspectors and regular specialists. For each aggregate size, we then counted the average proportion of problems they found. A given usability problem was considered 'found' if at least one member of the group recorded it.

Results. Figure 7 graphs the average proportion of problems uncovered by each size of aggregate for both GroupDraw and Groove. For comparative purposes, we also superimpose Nielsen's results for 31 novice evaluators and 19 regular specialists evaluating one interface [9].

The shape of all lines indicates that initial performance increases rapidly as the number of inspectors increase, but then more slowly afterwards. For example, while one inspector will find an average of 20-24% of the problems in both systems, three inspectors will find between 40-50% of the problems, and five will find about 50-60%. However, quite a few more inspectors are required to do much better than that—finding 75% of all problems require nine evaluators for GroupDraw and seven for Groove.

Discussion. Three to five inspectors will find 40-60% of the total known teamwork problems. This is a reasonable optimum that trades off the cost of employing multiple

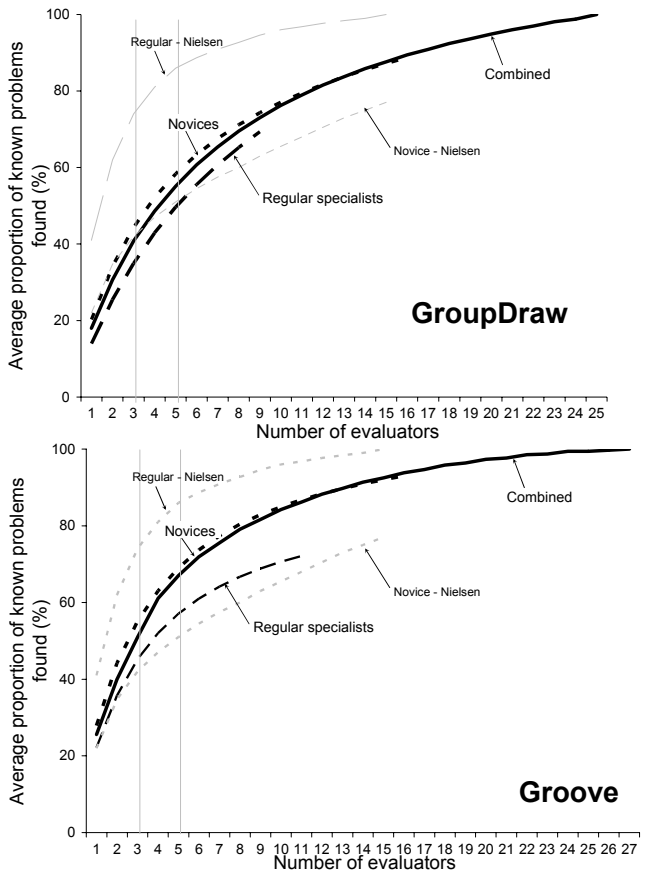

Figure 7: Percentage of problems found by each aggregate of inspectors

inspectors *vs* the number of usability problems uncovered. While increasing the number of inspectors also increases the number of problems uncovered, this is more expensive. In comparison with traditional HE, Nielsen and Molich [14,9] recommend using between three and five regular specialists to find a reasonably high proportion (75%) of usability problems in a given interface; however, they found novice evaluators were not as effective, as at least 5 are needed to discover half the problems.

This is an important result. Uncovering 40-60% of the known teamwork problems with only 3-5 inspectors is quite good if one considering the expense of other groupware evaluation methods.

Characterizing found problems as major or minor

How do inspectors perform in terms of uncovering major *vs* minor problems? Major problems are those that introduce serious obstacles to effective collaboration, whereas end users can typically work around minor problems. Clearly, we would like a method that is good at uncovering major problems as well as minor ones. To answer this, we counted the number of major *vs* minor problems found, and analyzed the inspectors' average performance for uncovering these major and minor problems in both interfaces.

Results. Overall, about 28% of the 105 known teamwork problems are major ones, with the remaining 72% classified as minor ones. This is identical to traditional HE, where Nielsen [9] classified 28% of all 211 found problems across six experiments as major ones. At a finer grain, an individual inspector will tend to find slightly more minor problems than major ones i.e., 6 minor *vs* 4 major problems on average. In contrast, Nielsen [9] reported a higher ratio of 8 minor to 4 major problems. When compared to the total known problems, individual inspectors tend to uncover a higher *proportion* (36%) of all known major problems *vs* a lesser proportion (16%) of the known minor usability problems ($p<.01$). Again, these results are somewhat similar to traditional HE [9]; across 6 case studies, inspectors found an average of 42% of the major problems and 32% of the minor ones.

Discussion. These results suggest two tendencies for groupware HE. First, individual inspectors find a higher proportion of the total number of major *vs* minor usability problems in a given interface. This implies that major problems are easier to find than minor ones, that individuals pay relatively more attention to finding them, and that major problems are more likely to be reported by multiple inspectors. This is advantageous since major usability problems are by definition the most important ones to find and fix.

Second, the individual as well as aggregate results say that inspectors will find more minor than major problems in terms of absolute numbers for a given interface. That is, while the same major problems are often spotted by many inspectors, individual minor problems are found by only a few.

This result is good news for the cost-benefit analysis reported earlier, for it suggests that while 3-5 evaluators will uncover 40-60% of the usability problems, there is a good chance a significant number of these will be major ones, and that most missed problems will be minor ones.

DISCUSSION

While our results are very promising, we suspect that they actually illustrate the low end of what inspectors could do; i.e., that in practice inspectors' performance would be even better.

To explain, we had previously mentioned that we were somewhat surprised when our regular specialists, who all had a good background in groupware, actually performed more poorly than our novices. To understand why this was, we interviewed several of our regular specialists. We found two reasons explaining this poorer performance.

1. Our regular specialists were not highly motivated. Their only incentive was their willingness to assist in the research. In contrast, novice evaluators performed the evaluations as a part of a (graded) course requirement and thus had a high incentive to do the task diligently.
2. Our regular specialists also participated in another related study during the same time interval, while the novice evaluators only did this one task. Because of their extra time commitments, some of the regular

specialists said they only spent a modest amount of time performing the actual HE.

Given this insight, we can claim that our method has achieved fairly promising results *in spite of* using novices and weakly-motivated (and time-stressed) regular specialists. Thus we should achieve even better performance in practice; we expect industrial practitioners to be highly motivated, where they would devote a reasonable amount of time and effort to do a thorough inspection.

CONCLUSION

Our study shows that our new groupware heuristics can be integrated into a low-cost practical methodology for identifying teamwork-oriented usability problems related to real-time collaboration within a shared workspace. The method directly parallels Nielsen's traditional heuristic evaluation, and our analysis shows similar performance results.

We saw that individual inspectors uncover approximately one-fifth of the known teamwork problems for two separate systems, and that these include major as well as minor problems. Given the high cost of most other groupware evaluation methods, even a minimal evaluation by one inspector is clearly worth doing.

While one is better than none, we also saw the heuristic evaluation will produce better results if several inspectors are used. To quantify this, we found a reasonable and important tradeoff in the cost of employing multiple inspectors *vs* the number of usability problems uncovered. Three to five inspectors will find 40-60% of the total known teamwork problems. Of course, the precise numbers of inspectors to use is ultimately a trade-off between the cost of using additional inspectors versus the cost of leaving usability problems unfound; this tradeoff can be looked up in Figure 7.

While very promising, we are convinced that heuristic evaluation of groupware can do even better. We speculate that a few inspectors could find even more teamwork problems if they had better training and more practice evaluating groupware (ours had fairly minimal training). We also believe that the groupware heuristics in Table 1 could be improved to make them easier to learn and easier to apply, and that they could be fine-tuned or added to give even better coverage of potential teamwork problems. Nielsen's heuristics evolved over a time and across different systems; so will ours.

In summary, we previously contributed a new set of groupware heuristics [2]. In this paper, we showed that inspectors can effectively apply these heuristics within a low-cost heuristic evaluation methodology for real-time groupware systems containing shared visual spaces.

Acknowledgements. This work was partially funded by NSERC, ASERC, Microsoft Research, and NIST.

REFERENCES

1. Baker, K. (2002) *Heuristic evaluation of groupware based on the mechanics of collaboration*. MSc thesis, Dept Computer Science, University of Calgary, Canada.
2. Baker, K., Greenberg, S. & Gutwin, C. (2001) Heuristic evaluation of groupware based on the mechanics of collaboration. In M. Little and L. Nigay (Eds) *Engineering for Human-Computer Interaction*, LNCS Vol 2254, p123-139, Springer.
3. Clark, H. (1996) *Using language*. Cambridge University Press, Cambridge.
4. Cox, D. & Greenberg, S. (2000). Supporting collaborative interpretation in distributed groupware. *Proc ACM CSCW'00,* 289-298.
5. Greenberg, S., Fitzpatrick, G., Gutwin, C. & Kaplan, S. (1999). Adapting the Locales framework for heuristic evaluation of groupware. *Proc. OZCHI'99*, 28-30.
6. Grudin, J. (1988) Why CSCW applications fail: Problems in the design and evaluation of organizational interfaces. *Proc ACM CSCW'88*, 85-93.
7. Gutwin, C. and Greenberg, S. (2002). A descriptive framework of workspace awareness for real-time groupware. *J. CSCW*, Kluwer Academic Press.
8. Gutwin, C. & Greenberg, S. (2000). The mechanics of collaboration: Developing low cost usability evaluation methods for shared workspaces. *Proc 9th IEEE WETICE: Infrastructure for Collaborative Enterprises*.
9. Nielsen, J. (1992). Finding usability problems through heuristic evaluation. *Proc ACM CHI'92*, 372-380.
10. Nielsen, J. (1993). *Usability Engineering*. Academic Press.
11. Nielsen, J. (1993) Usability heuristics. In [10].
12. Nielsen, J. (1994). Heuristic Evaluation. In [13], 25-62.
13. Nielsen, J. & Mack, R., Eds. (1994) *Usability Inspection Methods.* John Wiley and Sons, New York.
14. Nielsen, J. & Molich, R. (1990). Heuristic evaluation of user interfaces. *Proc ACM CHI '90*, 249-256.
15. Pinelle, D. & Gutwin, C. (2000) A review of groupware evaluations. *Proc 9th IEEE WETICE Infrastructure for Collaborative Enterprises*.
16. Roseman, M. & Greenberg, S. (1996). Building real time groupware with GroupKit, a groupware toolkit. *ACM TOCHI,* 3(1), 66-106.
17. Steves, M., Morse, E., Gutwin, C. & Greenberg, S. (2001). A comparison of usage evaluation and inspection methods for assessing groupware usability. *Proc ACM Group'01*, ACM Press. 37.
18. Tang, J. (1991). Findings from Observational Studies of Collaborative Work. *Int J Man-Machine Studies* 34(2), 143-160.