December 2003

Improving the Usability Evaluation Technique, Heuristic Evaluation, through the Use of Collaborative Software

Paul Lowry
Brigham Young University

Tom Roberts
University of Kansas

IMPROVING THE USABILITY EVALUATION TECHNIQUE, HEURISTIC EVALUATION, THROUGH THE USE OF COLLABORATIVE SOFTWARE

Paul Benjamin Lowry
Marriott School of Management
Brigham Young University
Paul.Lowry@BYU.net

Tom Roberts
KU School of Business
University of Kansas
troberts@ku.edu

Abstract

This paper examines how to increase the productivity of the software usability evaluation technique called heuristic evaluation (HE) through the use of collaborative software. Building off of a brainstorming research, it is theorized that part of HE resembles brainstorming, in that it is a creative process that can benefit from anonymous, parallel production afforded by collaborative software. Approximately 230 participants were involved in a lab study to compare traditional HE to HE conducted in face-to-face and distributed settings using collaborative software. It is found that the addition of collaborative software significantly improves HE by eliminating unnecessary duplication work, increasing overall productivity, and by increasing group awareness and consensus earlier on in HE. It is also found that distributed groups do suffer some process losses due to lack of media richness.

Keywords: Heuristic evaluation, collaborative software, usability, HCI, distributed work

Introduction

This paper examines ways to increase the productivity of the software usability evaluation technique called heuristic evaluation (HE). **Specifically, this research explores the research question of whether or not the use of collaborative software in distributed and face-to-face settings, as opposed to traditional face-to-face HE methods that do not use collaborative software, has any direct impact on the number of usability problems a HE team can find.** Before addressing this question, this paper first addresses the importance of HE, why it can be naturally extended to collaborative work, and the potential benefits that can be gained by having HE teams use collaborative tools instead of non-collaborative tools. Moreover, this paper explains the need for theoretical and empirical research in distributed teams that conduct HE. After this introduction, the paper further builds theory and empirical evidence to address the research question.

HE is a proven, powerful usability evaluation technique because it is fast, it is economical, it is easy to understand, it can be done early in the development process, it can involve non-experts, and it is more effective than other usability evaluation approaches (Nielsen & Molich, 1990). HE is typically based on ten simple heuristics of usability on which novices can be easily trained; thus, HE can effectively leverage non-technical team members from a target business application community to improve usability of a system (Nielsen & Molich, 1990). Although it is one of the few usability techniques that can be performed by non-experts, it still is more effective when conducted by usability experts (Nielsen, 1992). HE generally allows evaluators to find more usability problems than other usability techniques (Jeffries, Miller, Wharton, & Uyeda, 1991; Shaw, 1993).

HE naturally lends itself to group work. Nielsen (Nielsen & Molich, 1990) finds that individuals conducting HE alone only find a minority portion of usability problems in a given application; however, teams conducting HE on the same application find many more usability problems. Nielsen finds that the optimal team size for HE is 3-5 people and that while such teams do find many duplicate usability issues, they find more issues overall (Nielsen & Landauer, 1993). However, once HE teams get larger than three to five members they tend to find too many duplicate usability issues and they fail to find enough additional usability issues to justify the size increases.

Several streams of recent research have investigated how to improve HE. Example studies include changing HE to focus on a design-oriented evaluation (Garzotto, Mainetti, & Paolini, 1995); improving HE through combining them with cognitive walkthroughs (Sears, 1997); extending HE for evaluation of Web pages (Levi & Conrad, 1996); extending HE by adding work-domain experts (Muller, Matheson, Page, & Gallup, 1998); creating heuristics for web-site attractiveness (Sutcliffe, 2001); creating new heuristics for evaluating GroupWare (Baker, Greenberg, & Gutwin, 2001); creating a new instrument for assessing usability through a derivative of HE (Agarwal & Venkatesh, 2002). While this research has made significant strides to usability evaluation research, no research has focused on improving HE using collaborative software tools, or has focused on understanding the differences between distributed and face-to-face groups performing HE.

While HE has been shown to be more effective when conducted by groups, traditional HE techniques use non-collaborative technologies such as spreadsheets or word processors to capture the results. **HE could likely be improved by collaborative technologies that are designed to improve group results.** To understand how HE can be improved by collaborative technologies, the key steps that groups perform in HE must be explained. The standard process of HE involves three steps that are conducted with non-collaborative tools such as spreadsheets or word processors (Nielsen & Molich, 1990):

> *(1) Step one: All team members evaluate the interface(s) separately without talking to each other to avoid bias. During this time team members asses the interface(s) using the 10 heuristics to try to creatively find and categorize problems in the interface.*
>
> *(2) Step two: The team members meet face-to-face to compare and discuss their results. They also sort out duplicate bugs, make a combined list and even assign priority ratings to each verified bug.*
>
> *(3) Step 3: The team communicates the results to the formal design team.*

Steps one and two provide the most striking improvement opportunities for research.

Step one is designed to be conducted individually in groups, on the assumption that separate work helps prevent typical biases and domination that occur in face-to-face groups (Nielsen & Molich, 1990). This assumption reveals a significant improvement opportunity: While such biases and domination tend to occur in face-to-face groups, these negative effects are greatly mitigated by the use of collaborative technologies. For example, a separate stream of research in brainstorming has shown that group performing brainstorming can benefit from collaborative technologies that allow groups to anonymously and simultaneously produce ideas in a shared, collaborative environment (Gallupe et al., 1992). Not only do such technologies help overcome bias, but they also help increase idea production. The positive results of improving face-to-face brainstorming with anonymous, collaborative tools raises the important question as to whether such technology could similarly benefit HE. Currently, no empirical research addresses this question.

The other opportunity that arises from applying group research to HE is the question of how physical proximity may affect the productivity results of HE teams. This is an increasingly pertinent research question, because distributed group work is becoming increasingly important due to the advent of the Internet and the continued globalization of business. While software development teams are becoming increasingly dispersed and global, no one has published empirical research that examines the effects of dispersion on HE teams.

Given these opportunities, this research focuses on presenting theory and empirical results on performing HE with collaborative software in distributed and face-to-face settings, as compared to the traditional approach of using non-collaborative software in a face-to-face setting. This research addresses whether HE can be improved by using collaborative software in a simultaneous, anonymous production process. Furthermore, it compares the results of traditional HE groups, face-to-face collaborative software groups, and distributed collaborative software groups to better understand how differences in proximity affect results in such groups.

Theory Building

This section builds the base of theory that is used to predict the outcomes of using different tool and proximity choices in conducing HE. **In predicting the outcomes of HE, it is imperative to tie separate theory to steps one and two, because each step fosters different group processes and outcomes.** Thus, each step should be affected uniquely by different proximity and tool choices of HE groups. This section proposes theory for steps one and steps two.

Step One: Creative Bug Searching

In step one, the task of finding the usability bugs within a given interface is not simply a cognitive process; it is also a creative process that can benefit from simultaneous (or parallel), anonymous production provided through collaborative software. In step one, HE participants are not just trying to solve the cognitive problem of matching the ten known heuristics to specific usability problems, but they are trying to think creatively of how the heuristics should apply to a specific screen—since a screen's usability problems are not obvious, structured, or entirely known. When a participant finds a usability bug he/she tries to creatively find similar bugs that may exist in other parts of the system. Due to the creative and unstructured nature of this step, non-expert participants often find usability bugs that have not been considered by usability experts (Nielsen, 1992).

Since participants build creatively on their own work during traditional HE in step one, they would likely benefit from building on the ideas of all their group members, as long as this can be done in parallel (simultaneously) anonymously, and without involving face-to-face discussions that can slow down productivity—as seen in brainstorming research (Gallupe et al., 1992; Sosik, Avolo, & Kahai, 1998). Such an improvement in usability evaluation would have the added benefits of providing those who are caught in mental ruts to receive fresh ideas, to give implicit guidance and examples to inexperienced group members, and to help avoid duplication of work. Similar to what is found in brainstorming research, productivity results will likely increase with larger groups (Gallupe et al., 1992), but also will reach a point of diminishing marginal returns.

To support anonymous, parallel production in HE, a HE process needs to incorporate collaborative software that supports parallel production. Parallel production through collaborative software has been shown to increase productivity in groups (Nunamaker Jr., Briggs, Mittleman, Vogel, & Balthazard, 1997). Parallel production provided by collaborative software has also been shown to benefit other forms of group work because it allows simultaneous production without common process losses such as production blocking, domination, side tracking (Nunamaker Jr., Dennis, Valachich, Vogel, & George, 1991). Collaborative software can also help with information overload by creating a better form of structured, group memory (Nunamaker Jr. et al., 1991). Recently, these benefits have been shown in software inspection (a task that shares many similarities with usability evaluation) with inspection groups using used collaborative software (Rogers & Dean, 2003; Tyran & George, 2002). Specifically, HE participants must be able to see the bugs that other participants are reporting, to build on these contributions, to receive guidance, to develop new ideas of bugs that may exist based on these, and to avoid duplicate work. For example, collaborative software used by teams performing software inspection has been shown to save time because inspectors are not looking for problems already found by other inspectors (Rogers & Dean, 2003). Such software has also been shown to increase the deliberate efforts of team members because of the visibility of the contributions of others, increased attention, and the improvement in overall process discipline and group member motivation (Rogers & Dean, 2003). In summary, HE groups using collaborative software will likely find less overall bugs than non-collaborative software groups, because they will not have nearly as many duplicates; however, they should find more legitimate bugs:

> *Hypothesis 1a: Face-to-face groups conducting HE with collaborative software will find less overall bugs than traditional face-to-face HE groups, when including errors and duplicates.*
>
> *Hypothesis 1b: Face-to-face groups conducting HE with collaborative software will report fewer erroneous bugs than traditional groups.*
>
> *Hypothesis 1c: Face-to-face groups conducting HE with collaborative software will report fewer duplicate bugs than traditional groups.*
>
> *Hypothesis 1d: Face-to-face groups conducting HE with collaborative software will report more legitimate bugs than traditional groups.*

Turning from collaborative software, the other manipulation of this research involves the use of face-to-face groups versus distributed groups. **It is likely that proximity choices will cause no measurable differences in the productivity of face-to-face groups versus distributed groups that are working in step one.** The primary reason for this claim is that the traditional implementation of step one in HE does not involve participants working together; furthermore, implementations of HE using collaborative software in either face-to-face or distributed work modes do not involve participants directly communicating with each other. Thus, the only manipulation that will likely affect step one is that of tool choice (collaborative versus non-collaborative). Building on hypothesis 1, distributed groups using collaborative software should still outperform face-to-face groups using traditional processes:

Hypothesis 2a: Distributed groups conducting HE with collaborative software will find less overall bugs than traditional face-to-face HE groups, when including errors and duplicates.

Hypothesis 2b: Distributed groups conducting HE with collaborative software will report fewer erroneous bugs than traditional groups.

Hypothesis 2c: Distributed groups conducting HE with collaborative software will report fewer duplicate bugs than traditional groups.

Hypothesis 2d: Distributed groups conducting HE with collaborative software will report more legitimate bugs than traditional groups.

Step Two: Cognitive Bug Categorization

While proximity should not affect step one, it should have dramatic effects on step two because step two involves communication among team members. Groups that meet face-to-face should have communication advantages over distributed groups, because distributed groups need to use distributed communication technologies for discussion. Research shows that the results of distributed groups tend to be diminished by less media richness and socialization than face-to-face groups (Burke & Chidambaram, 1996). Media Richness Theory predicts that because distributed work requires non-face-to-face communication technologies, it will cause less rich media interactions (Burke & Chidambaram, 1996). Social Presence Theory predicts that that distributed work will cause worsened social presence in work groups (Burke & Chidambaram, 1996). Because distributed HE groups will have worse media richness and social presence than face-to-face groups, they will be less productive in step two of HE. Thus, distributed groups will properly categorize fewer bugs into their heuristic categories in this step and will perform less overall work than groups working face-to-face, as measured in direct activity in bug clean up (elimination of duplicates and errors, and correct bug categorization).

Likewise, little difference will likely exist in step two between face-to-face traditional groups and face-to-face groups using collaborative technologies. This is primarily because at this point tools are simply used to record group decisions. It is possible that collaborative software could accelerate the consolidation process by allowing the participants to more readily identify, categorize, and summarize bugs. However, step two involves direct communication among participants, which will likely be the primary productivity bottleneck for any group. For example, it was found that software inspection groups that work face-to-face tend to be slowed down because conversations tend to be between two people at a time (Votta, 1993). Thus, the discussion pace of HE groups will likely set the maximum productivity rate, so face-to-face groups conducting HE with traditional tools will have the same productivity in step two as face-to-face groups recording their work with collaborative software.

Hypothesis 3a: Distributed HE groups, using non-face-to-face communication technologies, will properly categorize fewer bugs in step two than groups that have face-to-face discussions.

Hypothesis 3b: Distributed HE groups, using non-face-to-face communication technologies, will properly eliminate fewer errors in step two than groups that have face-to-face discussions.

Hypothesis 3c: Distributed HE groups, using non-face-to-face communication technologies, will properly eliminate fewer duplicates in step two than groups that have face-to-face discussions.

Hypothesis 3d: Distributed HE groups, using non-face-to-face communication technologies, will do less overall work in step two than groups that have face-to-face discussions.

Hypothesis 4a: Traditional face-to-face groups will properly categorize as many bugs in step two as face-to-face groups using collaborative software.

Hypothesis 4b: Traditional face-to-face groups will properly eliminate as many errors in step two as face-to-face groups using collaborative software.

Hypothesis 4c: Traditional face-to-face groups will properly eliminate as many duplicates in step two as face-to-face groups using collaborative software.

Hypothesis 4d: Traditional face-to-face groups will do as much work in step two as face-to-face groups using collaborative software.

Method

Design

The research design of this study involves three conditions: HE performed via the traditional method, HE performed face-to-face using collaborative software, and HE performed in a distributed setting using collaborative software and a non-face-to-face communication technology. Table 1 overviews these control and treatment conditions. Each condition involved participants being randomly assigned to their condition in groups of three. The control groups performed HE face-to-face using traditional processes; thus, these groups did not conduct step one of HE in parallel or anonymously. Instead, they recorded individually their bugs using Microsoft Word™. In step two, control groups discussed their results face-to-face and combined them into one document in Word™. The first treatment performed HE face-to-face in step one using anonymous, parallel production, using a collaborative tool, called Collaboratus (Lowry, Albrecht, Nunamaker Jr., & Lee, 2002). In step two, these groups discussed their results face-to-face and combined them into one document in Collaboratus. The second treatment performed HE in step one in a distributed-synchronous work mode using anonymous, parallel production, using Collaboratus as the collaborative tool. In step two, these distributed treatment groups discussed their results using the chat features of Microsoft NetMeeting™, and their results were combined into one Collaboratus document.

Table 1. Experiment Treatment and Control Conditions

Condition	Mode	Tools
(A) Control	Face-to-face	Word™
(B) Treatment	Face-to-face	Collaboratus
(C) Treatment	Distributed synchronous	Collaboratus + NetMeeting

Participants

The participants were all members of a 200-level IS class at a large Midwestern university. Approximately 300 students were enrolled in the class and all were required to participate in the HE experiment for graded course credit. Of these, around 270 students actually participated, but another 40 participants' data was dropped because they did not adhere to experimental procedures or were in a session that did not meet control standards (this occurred in one of the distributed labs). The change in participants caused the design to be unbalanced, as shown in Table 2. A total of 228 students provided the following demographic data: age (M=20.2, SD=2.0); GPA (M=3.3, SD=.46), years of education (M=13.7, SD=1.2.); years of work (M=4.2, SD=2.5); gender (57 % male, 43% female).

Table 2. Participants by Condition

Group	Mode	# of Participants
(1) Control	Face-to-face	99
(2) Treatment	Face-to-face	78
(3) Treatment	Distributed synchronous	51

Procedures

Several strict, experimental procedures were followed to increase the control and accuracy of results. An entire class session was dedicated to providing consistent training to all participants on how to conduct HE. Many examples and screen shots were provided that showed compliant and non-compliant interfaces, in terms of the Nielsen's (Nielsen & Molich, 1990)10 basic heuristics. Students were given handouts, with examples, to study to reinforce the training. Next, over a period of a week, students were required to attend their normal class laboratory sessions during which their assigned conditions were executed. To avoid mixing experimental conditions in the same sessions, a given lab session (usually consisting of 20-30 students) was

dedicated to only one experiment condition. Students were not informed that other conditions were being executed in other labs, and were asked not to discuss the assignment with other students so that they would not receive unfair advantages when conducting the assignment.

To further ensure control and consistency, participants were given carefully scripted training for each condition. The participants were given brief training on the tools they were to use (i.e. Word™, Collaboratus, NetMeeting™), depending on the condition each participant was assigned to. Training was then given on step one, after which all participants were given exactly 30 minutes to conduct this step. After this step, brief training was given on how to conduct step two, after which participants were given 10 minutes to conduct this step. All procedures were strictly scripted, without participant involvement. Participants were not allowed to ask the facilitator questions at any time, so that all groups would be receiving exactly the same information. Finally, participants were required to complete a post-test questionnaire to gather demographic data.

Measures

All measures reported for this study involve derivations of the construct of productivity. Most of the measures were evaluated by a panel of three judges, as summarized in Table 3.

Results and Discussion

This final section presents and discusses the results of this study. After the empirical results are overview, the contributions of this research to theory and practice are presented. Next, the limitations and simplifying assumptions as addressed. Last, future research opportunities are overviewed.

Table 3. HE Productivity Measures Used for This Experiment

Productivity Measure	Description
Gross bugs produced	The total number of bugs reported by a group, regardless of incorrectly reported bugs or duplicate bugs (derived from) (Hertzum & Jacobsen, 2001).
Error bugs	The total number of bugs that are incorrectly reported by a group (derived from) (Hertzum & Jacobsen, 2001).
Duplicate bugs	The total number of bugs that are duplicates.
Real bugs	Measure 1 minus measure 2 minus measure 3. Or, the total number of bugs reported by a group that are correct bugs and do not include duplicates (derived from) (Hertzum & Jacobsen, 2001; Nielsen & Molich, 1990).
Net correct heuristics	How many real bugs were correctly categorized into their heuristic categories (1 of 10 categories), removing duplicate bugs (based on measure 4).
Net incorrect heuristics	How many real bugs were incorrectly categorized into their heuristic categories (1 of 10 categories), removing duplicate bugs (based on measure 4).
Net work	The net amount of actual changes and additional work between sessions 1 and 2.

Several significant results were found in the experiment that validated several of the hypotheses. These results are summarized in Table 4. The key findings are that the traditional, face-to-face groups (A condition) did produce more gross bugs in steps 1 and steps 2 than the face-to-face groups using Collaboratus (B condition) and the distributed groups (C condition). It is appears the primary reason for the greater gross production is that the A groups had significantly more duplicate bugs—no differences were seen in the number of incorrectly identified bugs. Interestingly, despite the massive number of duplicates created in the A condition, the overall number of legitimate bugs was not statistically different across conditions. Although, finding all the legitimate, non-duplicate bugs for the A condition was a highly difficult task, as there were many more duplicates to sort through.

As predicted, B groups were able to more effectively categorize bugs than C groups in step 2, highlighting the differences in proximity between the groups. Also, as predicted A and B groups virtually had the same productivity in step 2 in terms of producing correctly categorized bugs, incorrectly categorized bugs, error bugs, and duplicate bugs

Table 4. Experiment Results

Hypothesis	Step	Measure	In Predicted Direction?	Hypothesis Support?	F-Statistic	Sig.	A Mean (stdv)	B Mean (stdv)	C Mean (stdv)
Ia: A > B	1	gross bugs	yes	yes	* 7.7	< 0.01	47.7 (12.3)	39.3 (12.2)	n/a
Ib: A > B	1	error bugs	yes	no	0.0	0.96	10.3 (4.5)	10.2 (5.3)	n/a
Ic: A > B	1	duplicate bugs	yes	yes	* 25.0	< 0.01	14.5 (7.1)	7.3 (4.5)	n/a
Id: A < B	1	legitimate bugs	no	no	0.43	0.51	29.9 (6.9)	21.8 (5.8)	n/a
IIa: A > C	1	gross bugs	yes	yes	* 18.6	< 0.01	47.7 (12.3)	n/a	32.7 (8.9)
IIb: A > C	1	error bugs	yes	no	0.62	0.44	10.3 (4.5)	n/a	9.1 (4.4)
IIc: A > C	1	duplicate bugs	yes	yes	??	< 0.01	14.5 (7.1)	n/a	4.3 (2.7)
IId: A < C	1	legitimate bugs	no	no	3.5	0.07	29.9 (6.9)	n/a	19.2 (7.1)
IIIa: A > C	2	correctly categorized	yes	no	1.0	0.33	7.6 (4.3)	n/a	6.5 (2.7)
IIIa: A < C	2	incorrectly categorized	no	no	2.9	0.09	14.9 (6.5)	n/a	12.0 (5.7)
IIIa: B > C	2	correctly categorized	yes	yes	3.6	0.06	n/a	8.7 (3.6)	6.5 (2.7)
IIIa: B < C	2	incorrectly categorized	no	no	0.34	0.56	n/a	13.0 (4.8)	12.0 (5.7)
IIIb: A > C	2	error bugs	yes	no	0.05	0.82	9.0 (4.3)	n/a	8.7 (4.6)
IIIb: B > C	2	error bugs	yes	no	0.94	0.34	n/a	10.0 (5.3)	8.7 (4.6)
IIIc: A > C	2	duplicates	yes	yes	* 12.4	< 0.01	8.8 (5.2)	n/a	4.0 (2.8)
IIIc: B > C	2	Duplicates	yes	yes	* 4.3	0.04	n/a	6.9 (4.6)	4.0 (2.8)
IIId: A > C	2	Work	yes	yes	* 43.2	< 0.01	29.2 (16.1)	n/a	5.3 (11.1)
IIId: B > C	2	Work	no	no	0.74	0.39	n/a	2.1 (5.2)	5.3 (11.1)
IVa: A = B	2	correctly categorized	yes	yes	1.3	0.26	7.6 (4.3)	8.7 (3.6)	n/a
IVa: A = B	2	incorrectly categorized	yes	yes	1.6	0.21	14.9 (6.5)	13.0 (4.8)	n/a
IVb: A = B	2	error bugs	yes	yes	0.80	0.38	9.0 (4.3)	10.0 (5.3)	n/a
IVc: A = B	2	duplicates	yes	yes	2.5	0.12	8.8 (5.2)	6.9 (4.6)	n/a
IVd: A = B	2		no	no	* 73.4	< 0.01	29.2 (16.1)	2.1 (5.2)	n/a

While A groups produced significantly more work than C groups in Step 2, the surprise finding was that A groups produced significantly more work than B groups in Step 2. In examining the data, both the groups using Collaboratus (B and C groups) virtually stop doing any work in Step 2, while the A groups made significant changes throughout step 2. This may be partially due to Collaboratus groups have far less duplicate bugs, which caused less need to sort out the duplicate bugs in Step 2. Also, a deeper phenomenon may be occurring in that Collaboratus likely encouraged more consensus earlier in the group process (in Step 1), because of the increased group awareness caused by the parallel work and by group members seeing what other group members were doing; thus, by the time the B and C groups had entered Step 2, they had implicit agreement on how to categorize their bugs, while the A groups had no implicit agreements or understanding. This further highlights the potential productivity benefits of using collaborative software in heuristic evaluation.

Summarizing the key contributions of this research, the findings show that the introduction of collaborative software into traditional HE groups does significantly alter the results of small HE groups. Two significant alterations are particularly notable: (1) Collaborative software eliminates duplicate bugs, (2) Collaborative software appears to drive earlier consensus in the HE process to where little if any additional work is necessary in step 2. However, despite these improvements collaborative software groups still did not improve on decreasing HE errors and categorization problems; thus, collaborative software groups still find about as many errors overall. On the other hand, the results of the collaborative software teams are much more readily usable by software design teams, because they do not have to sort through all the duplicate bugs produced by the traditional teams. Thus, the direct implication for practice with actual design teams is that collaborative software will improve consensus, decrease duplicates, decrease the overall time needed for Step 2, and decrease the overall time the design team needs to sort through the results for invalid duplicates before the results can be used.

Furthermore, this research shows that lack of proximity appears to have a significant affect on how distributed HE groups perform. In general, distributed HE groups performed reasonably well, especially in producing fewer duplicates and incorrect bug categorizations than traditional face-to-face teams. However, the data suggests that traditional groups tend to outperform distributed teams in producing overall legitimate bugs. Hence, it appears in Step 2 that the decreases in media richness for distributed groups (through decreased communication capabilities) have a partial negative effect on distributed groups. One finding that cannot be completely explained is that in Step 1 distributed groups produced fewer legitimate bugs than traditional groups; while in Step 1 Collaboratus face-to-face groups did not produce fewer legitimate bugs than traditional groups. This is difficult to explain because none of the groups were able to communicate during this step.

The contributions of this research need to be examined in light of its limitations and its simplifying assumptions. The primary limitations can be expressed in terms of limitations due to the experimental control, limitations due to the tools used, and limitations due to the artificial nature of the task. Clearly, by using a controlled laboratory environment, with tightly scripted conditions, the results of this research have limited generalizability. This generalizability is further limited by the use of students who are non-experts in usability analysis. On the other hand, this also highlights the significance of the results—the results may have been more significant with the use of usability experts. HE was also designed specifically to be used by non-experts; thus, students provide useful surrogates for non-experts. However, the most important factor in the use of students is that the students were graded on their productivity and accuracy, so that they would treat the experiment seriously. This grading procedure may have introduced evaluation pressure in the experiment that would not exist in professional environments.

Another limited element of the controlled environment was the simulation of distributed-synchronous work. Because real-life distributed conditions lack control, this experiment had members of the same distributed team work in the same room, but where they could not see each other or communicate, except through NetMeeting™. This approximates a fairly realistic environment of distributed-synchronous work; however, it does not directly address asynchronous-distributed HE, for which it is much more difficult to control.

The experiment may have also suffered from artifacts that were the direct result of the tools used. The control groups used Microsoft Word™, of which most participants had several months or years of exposure. Another **limitation is that the task did not involve a real system and real business scenarios.** However, this did improve control of the experiment.

Given the contributions and limitations of this research, much promising future research is possible. First, given the limited generalizability of these findings, several streams of research could be conducted to improve the generalizability. Second, future research can explore variations in tool choices to further improve generalizability. Third, this stream of research would also likely benefit from exploration of longitudinal measures. Fourth, testing variations of group size with HE groups that use collaborative software could also be highly useful. Fifth, another important research possibility is to further examine the effects of collaborative software use on step one of HE.

In conclusion, this research provided several advances to human-computer interaction research; particularly, in finding ways to better understand and improve the usability evaluation technique, heuristic evaluation. This research shows that HE groups can have their results improved by using collaborative software that is designed for group awareness and anonymous, parallel production. Furthermore, while performing HE in distributed groups can be effective, it does appear to suffer from the effects of decreased media richness. These findings open up an exciting array of future research possibilities in improving usability techniques.

References

Agarwal, R., and Venkatesh, V. (2002). "Assessing a Firm's Web Presence: A Heuristic Evaluation Procedure for the Measurement of Usability." *Information Systems Research (ISR), 13*(2), 168-186.

Baker, K., Greenberg, S., and Gutwin, C. (2001). "Heuristic Evaluation of Groupware Based on the Mechanics of Collaboration." *Lecture Notes in Computer Science, 199*(2254), 123-140.

Burke, K., and Chidambaram, L. (1996). "Do Mediated Contexts Differ in Information Richness? A Comparison of Collocated and Dispersed Meetings," paper presented at the Twenty-Ninth Hawaii International Conference on System Sciences, Maui, Hawaii.

Gallupe, R. B., Dennis, A. R., Cooper, W. H., Valacich, J. S., Bastianutti, L. M., and Nunamaker Jr., J. F. (1992). "Electronic Brainstorming and Group Size." *Academy of Management Journal, 35*(2), 350-369.

Garzotto, F., Mainetti, L., and Paolini, P. (1995). "Hypermedia Design, Analysis, and Evaluation Issues." *Communications of the ACM 38*(8), 74-86.

Hertzum, M., and Jacobsen, N. E. (2001). "The Evaluator Effect: A Chilling Fact About Usability Evaluation Methods." *International Journal of Human-Computer Interaction, 13*(4), 421-443.

Jeffries, R., Miller, J. R., Wharton, C., and Uyeda, K. M. (1991). "User Interface Evaluation in the Real World: A Comparison of Four Techniques," paper presented at the SIGCHI conference on Human factors in computing systems: Reaching through Tchnology, New Orleans.

Levi, M. D., and Conrad, F. G. (1996). "A Heuristic Evaluation of a World Wide Web Prototype." *Interactions, July-August*, 50-61.

Lowry, P. B., Albrecht, C. C., Nunamaker Jr., J. F., and Lee, J. D. (2002). "Evolutionary Development and Research on Internet-Based Collaborative Writing Tools and Processes to Enhance eWriting in an eGovernment Setting." *Decision Support Systems 34*(3), 229-252.

Muller, M. J., Matheson, L., Page, C., and Gallup, R. (1998). "Participatory Heuristic Evaluation." *Interactions, September-October*, 13-18.

Nielsen, J. (1992, May 3-7). "Finding Usability Problems through Heuristic Evaluation," paper presented at the Computer Human Interaction.

Nielsen, J., and Landauer, T. K. (1993). "A Mathematical Model of the Finding of Usability Problems," paper presented at the Interchi.

Nielsen, J., and Molich, R. (1990). "Huristic Evaluation of User Interfaces," paper presented at the Computer Human Interaction.

Nunamaker Jr., J. F., Briggs, R. O., Mittleman, D. D., Vogel, D. R., and Balthazard, P. A. (1997). "Lessons from a Dozen Years of Group Support Systems Research: A Discussion of Lab and Field Findings." *Journal of Management Information Systems, 13*(3), 163-207.

Nunamaker Jr., J. F., Dennis, A., Valachich, J., Vogel, D., and George, J. (1991). "Electronic Meeting Systems to Support Group Work." *Communications of the ACM 34*(7), 40-61.

Rogers, T. L., and Dean, D. L. (2003). "Increasing Inspection Efficiency through Group Support Systems: An Examination of Causal Factors," Working Paper, KU School of Business, University of Kansas.

Sears, A. (1997). "Heuristic Walkthroughs: Finding the Problems Without the Noise." *Journal of Human-Computer Interaction, 9*(3), 213-234.

Shaw, D. (1993). "CD-ROM Interfaces for Information Retrieval: Heuristic Evaluation and Observations of Intended Users," paper presented at the 14th National Online Meeting, New York, NY.

Sosik, J. J., Avolo, B. J., and Kahai, S. S. (1998). "Inspiring Group Creativity: Comparing Anonymous and Identified Electronic Brainstorming." *Small Group Research, 29*(1), 8-31.

Sutcliffe, A. (Ed.). (2001). *Heuristic Evaluation of Website Attractiveness and Usability*. Berlin: Springer-Verlag.

Tyran, C. K., and George, J. F. (2002). "Improving Software Inspections with Group Process Support." *Communications of the ACM 45*(9), 87-92.

Votta, L. G. J. (1993). "Does Every Inspection Need a Meeting?," paper presented at the SIGSOFT'93 - First ACM SIGSOFT Symposium on Software Development Engineering.