Remote Usability Evaluation: Can Users Report Their Own Critical Incidents?

José C. Castillo
U S WEST Information Technologies
1801 California St. Suite 1640
Denver, CO 80202 USA
+1 303 965 2946
jxcast2@uswest.com

H. Rex Hartson and Deborah Hix
Department of Computer Science
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061-0106 USA
+1 540 231 4857
hartson@vt.edu, hix@vt.edu

ABSTRACT
In this paper, we briefly introduce the *user-reported critical incident method* (originally called semi-instrumented critical incident gathering [3]) for remote usability evaluation, and describe results and lessons learned in its development and use. Our findings indicate that users can, in fact, identify and report their own critical incidents.

Keywords
Remote usability evaluation, remote evaluation, usability evaluation, critical incidents, user-initiated, usability data

INTRODUCTION
Several methods (e.g., collaborative remote evaluation, remote questionnaire or survey) have been developed for conducting remote usability evaluation, but each suffers from some drawback – e.g., time-consuming data capture, costly data analysis, inapplicability to real users doing real tasks in their normal work environment, or need for direct interaction between user and evaluator during an evaluation session [1]. The goal of our work was to develop and evaluate a cost-effective method for remotely evaluating usability of real-world applications that overcomes these drawbacks.

USER-REPORTED CRITICAL INCIDENT METHOD
Using this method, usability data, centered around critical incidents [4] *self-reported by users,* are captured in day-to-day task situations, while user and evaluator attend the process at different times and different places.

Users, located in their own work environment, are given minimal training to identify critical incidents as they occur during the normal course of on-the-job task performance. Then, whenever usage difficulty is encountered, the user clicks on a *Report Incident* button, a single added object consistently appearing on all screens of the application being evaluated. The click activates an instrumentation routine (outside the application) that captures a contextualized critical incident report consisting of a textual form (in a separate window from the application) for users to enter a structured report about the critical incident encountered, and a video clip showing screen activity immediately prior to clicking the button, to capture the critical incident and events leading up to it. Each contextualized critical incident report is sent asynchronously via the network to a queue from where they are analyzed by evaluators into usability problem descriptions.

FOR PROCEEDINGS ONLY ...

With this method, data capture is cost effective because users do the work of identifying their own critical incidents during task performance. Real-time reporting prevents loss of this perishable data by capturing it immediately as it arises during usage. Data are high quality because they center around critical incidents and therefore are relatively easy to convert into usability problem descriptions.

OVERVIEW OF STUDY
Objective
To gain practical insight and understanding about the strengths and weaknesses of the method, we conducted an empirical study which had as one of its objectives to investigate the feasibility and effectiveness of having users identify and report their own critical incidents during usage.

What user subjects did
After receiving training on identifying and reporting critical incidents, 24 user subjects performed six search tasks using a Web-based application called the Internet Movie Database (http://us.imdb.com). Users self-reported critical incidents identified during task performance using an on-line *Remote Evaluation Report.* Structured questions provided a content-based framework for consistently gathering information on each critical incident.

Critical incident reports
Data gathered in critical incident reports from user subjects included:

- URL (or location) where user encountered critical incident
- Description of user task in progress when critical incident occurred
- Expectations of user about what system was supposed to do when critical incident occurred
- Detailed description of critical incident (what happened and why user thought it happened)
- Whether user could recover from critical incident and, if so, description of how user did so
- Indication of user's ability to reproduce critical incident
- Severity rating of critical incident
- Additional comments, suggestions, or possible solutions to problem

RESULTS AND LESSONS LEARNED
Comparing incidents identified
Results indicated that users, even when working in their daily job environment and lacking interaction with evaluators, are capable of self-reporting high, medium, and

low severity critical incidents encountered during task performance. The experimenter reviewed full videotapes of each user subject's session to identify independently critical incidents that each user encountered (but did not necessarily report).

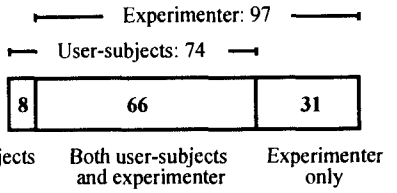

Figure 1. Number of critical incidents identified by user-subjects and experimenter

Across all user subjects, the experimenter found 97 critical incidents (see Figure 1): 66 reported by both experimenter and user-subjects and 31 identified only by the experimenter (mostly of low severity). User-subjects sent a total of 74 critical incident reports (mean: 3.1 reports per user-subject, std. dev.: 1.7). Interestingly, 8 low severity critical incidents were reported by user-subjects in cases where the experimenter did not recognize from review tapes that user-subjects were experiencing a critical incident. The experimenter did not, however, consider these as gratuitous reports sent to please the experimenter, concluding that these critical incidents were known in the minds of the user-subjects but not evident visually in the videotapes. Nevertheless, these 8 reports were not considered during data analysis.

Comparing severity rankings

Users rated severity of each critical incident on a well-defined scale of one (lowest) through five (highest). The experimenter independently ranked severity of the reported critical incidents, compressing the ratings into three ranks: low, medium, and high. Across all 24 user-subjects, the experimenter's rankings agreed with those of users for 55 out of 66 (83%) of the critical incidents reported. Breaking this figure down, user-subjects agreed with the ranking of 21 out of 28 (75%) of the critical incidents identified by the experimenter as high severity. They also agreed with 19 out of 24 (79%) of the medium severity critical incidents and 15 out of 45 (33%) of the low severity critical incidents as ranked by the experimenter. Six reports were given lower severity than the experimenter, five higher. Thus, we conclude that users can rate self-reported critical incidents with reasonable accuracy compared to an expert evaluator.

Lessons learned

We expected user activity for reporting a critical incident to be structured (i.e., that users would perform a task, encounter a critical incident, and immediately report it). However, this was not the case. One particular pattern of deviation from the expected sequence led to perhaps our most significant observation — a delay in reporting following the occurrence of many critical incidents. It is likely that some of these delays were due to the user's perceived need to wait long enough to understand the nature of a critical incident and to gather enough information for a complete report. Delays roughly corresponded with severity of the critical incident. The shortest delays occurred when reporting low severity critical incidents (mean: 0.4 min., std. dev.: 0.5), and the longest delays occurred for high severity critical incidents (mean: 4.5 min., std. dev.: 6.6). Our presumption is that a more severe critical incident requires more information to report and, therefore, results in a larger delay before reporting.

The user-reported critical incident method includes automatic and continuous scan-converted (or digital) video capture of screen activity during task performance. From this, the system extracts contextualized critical incidents, short video clips of screen activity just moments preceding the point at which a critical incident is reported. We expected these clips, when reviewed by an evaluator in conjunction with reading the reports, to help explain the critical incidents. However, because of the reporting delay just described, video clips often were irrelevant to the critical incident. A solution to this problem is to de-couple the time of critical incident occurrence from the time of reporting, and to associate the video clip with the occurrence itself to ensure that the clip contains data relevant to the critical incident. In this approach, users would click a button (e.g., *Begin Incident Capture*) when they believe they are beginning to experience a critical incident. The retrospective video clip would be captured at this point. Users would subsequently click on a *Report Incident* button when ready to complete a detailed report of the critical incident.

We expected that users would want to report critical incidents anonymously, but they indicated they did not mind being identified with their reports, if, in a real-world setting, it meant they could receive acknowledgment of receipt of their reports, plus feedback from evaluators.

Interestingly, the experimenter discovered by manual inspection of data that some of the users who gave the longest and most detailed reports also said they felt that reporting critical incidents did not interfere much with performing the tasks. We had feared that self-reporting might be perceived as burdensome to users.

We found, in sum, that users could, in fact, with minimal training, recognize and report critical incidents effectively, that they could rank their severity reasonably, and that they did not find this self-reporting to interfere with getting real work done.

REFERENCES

1. Castillo, J.C. (1997). *The User-Reported Critical Incident Method for Remote Usability Evaluation.* MS Thesis, Department of Computer Science, Virginia Tech, Blacksburg, VA 24061 USA.
2. Castillo, J.C. & Hartson, H.R. (1997). Remote Usability Evaluation Home Page. Internet WWW page at: <http://hci.ise.vt.edu/~josec/remote_eval/index.html > (Last updated: 12/01/97; last accessed: 12/31/97.).
3. Hartson, H.R., Castillo, J.C., Kelso, J., Kamler, J., & Neale, W.C. (1996). Remote Evaluation: The Network as an Extension of the Usability Laboratory. *Proceedings of CHI '96 Conference on Human Factors in Computing Systems.* New York: ACM, 228-235.
4. Shattuck, L.W. & Woods, D.D. (1994). The Critical Incident Technique: 40 Years Later. *Proceedings of the 38th Annual Meeting of the Human Factors Society,* 1080-1084.