USER INTERFACE EVALUATION IN THE REAL WORLD: A COMPARISON OF FOUR TECHNIQUES

Robin Jeffries *James R. Miller*

Hewlett-Packard Laboratories
1501 Page Mill Road
Palo Alto, California 94304

Cathleen Wharton

Dept. of Computer Science and
Institute of Cognitive Science
University of Colorado
Boulder, Colorado 80309
and
Hewlett-Packard Laboratories

Kathy M. Uyeda

Corporate Engineering
Hewlett-Packard
3155 Porter Drive
Palo Alto, California 94304

ABSTRACT

A user interface (UI) for a software product was evaluated prior to its release by four groups, each applying a different technique: heuristic evaluation, software guidelines, cognitive walkthroughs, and usability testing. Heuristic evaluation by several UI specialists found the most serious problems with the least amount of effort, although they also reported a large number of low-priority problems. The relative advantages of all the techniques are discussed, and suggestions for improvements in the techniques are offered.

KEYWORDS: Evaluation, guidelines, usability testing, cognitive walkthrough

INTRODUCTION

Currently, most user interfaces are critiqued through techniques that require UI expertise. In *heuristic evaluation*, UI specialists study the interface in depth and look for properties that they know, from experience, will lead to usability problems. (Our use of the term "heuristic evaluation" is somewhat different than that of [6], as will be noted later.) In addition, they may carry out *usability testing*, in which the interface is studied under real-world or controlled conditions, with evaluators gathering data on problems that arise during its use. These tests can offer excellent opportunities for observing how well the situated interface supports the users' work environment.

Under proper circumstances, these methods can be effective. However, several factors limit their use. People with adequate UI experience to carry out these evaluations are scarce. The techniques are difficult to apply before an interface exists; consequently, any recommendations come at a late stage in development, often too late for substantive changes to be made. If the UI specialists are not part of the development team, they may not be aware of technical limitations on the design or why certain design or implementation decisions were made. Technical and organizational gulfs can arise between the development team and the UI specialists, which impede the communication and correction of the problems discovered during the evaluation. In addition, usability testing is generally expensive and time-consuming.

Alternative means of evaluating interfaces have been proposed that address some of these problems. They try to structure the evaluation process so that interface developers, not UI specialists, can carry out the evaluation, potentially increasing the number of people who can do evaluations and avoiding some of the problems mentioned earlier. One technique is the use of published *guidelines*, which provide evaluators with specific recommendations about the design of an interface, such as how the contents of a screen should be organized or how items should be arranged in a menu (e.g., [1, 2, 8, 9]).

The *cognitive walkthrough* method [4, 5] combines software walkthroughs with a cognitive model of learning by exploration [4]. In this methodology, the developers of an interface walk through the interface in the context of core tasks a typical user will need to accomplish. The actions and feedback of the interface are compared to the user's goals and knowledge, and discrepancies between the user's expectations and the steps required by the interface are noted.

Beyond the advocacy of these techniques by their creators, little is known about how well they work, especially in comparison to one other: what kinds of interface problems they are best-suited to detect, whether developers who are

not UI specialists can actually use them, and how they compare in cost/benefit terms. The experiment described in this paper was designed to provide such a test.

THE EXPERIMENT

In brief: We obtained a pre-release version of a forthcoming software product, and organized groups to evaluate its interface with the four techniques described above: heuristic evaluation, usability testing, guidelines, and cognitive walkthroughs. Each of the evaluation groups reported the problems they encountered on a common form, so that the numbers and kinds of problems detected by the different groups could be compared.

A primary goal for this experiment was for all the evaluations to occur in conditions as close as possible to those that might exist in a real product evaluation. We used the results from the actual usability tests that were done for this product. Similarly, we used researchers in HP Laboratories who are frequently called upon to perform heuristic evaluations for real clients. The set of guidelines and the cognitive walkthrough technique we used have not been used in their current form on enough real products to determine what realistic usage patterns would be. Accordingly, we worked with the developers of these methodologies to set up procedures that were consistent with the ways that the technique developers intended them to be used.

The goal of realism means that these evaluations suffered from all the real-world problems of interface evaluation. We were limited in the number of people who could participate in the evaluations and the amount of time they could devote to them. We had limited access to the developers, who were hundreds of miles away and busy producing the final version of the product. However, we believe that these evaluations are quite typical of what goes on in product development, and therefore our results should be a good measure of how these techniques will work when applied in the real world.

The Interface.

The interface that we evaluated was *HP-VUE*, a visual interface to the UNIX operating system.[1] It provides graphical tools for manipulating files, starting and stopping applications, requesting and browsing help, controlling the appearance of the screen, etc. We evaluated a "beta-test" version of HP-VUE. Thus, some usability problems had been identified and corrected in earlier versions of the interface, and the final product version of the interface underwent further changes, based on these and other evaluations.

[1]The Hewlett-Packard Visual User Environment (HP-VUE 2.0). HP-VUE is a trademark of the Hewlett-Packard Company. UNIX is a trademark of AT&T. The X Window System is a trademark of Massachusetts Institute of Technology. Motif is a trademark of the Open Software Foundation, Inc.

The techniques and the evaluators.

The heuristic evaluation technique used here differs somewhat from that described by [6], who proposed that software developers apply the technique. In our experience, heuristic evaluation is most commonly done by UI specialists, who are called in as the available "experts" to critique an interface. The four heuristic evaluators, members of a research group in human-computer interaction, had backgrounds in behavioral science as well as experience in providing usability feedback to product groups. We gave the evaluators a two-week period in which to do their evaluation, along with all their other job-related tasks. They spent whatever amount of time they chose within that period, and reported the time spent at the conclusion of their evaluation.

The usability tests were conducted by a human factors professional, for whom product usability testing is a regular part of his job. Six subjects took part in the test; they were regular PC users, but not familiar with UNIX. They spent about three hours learning HP-VUE and two hours doing a set of ten user tasks defined by the usability testing team.

The guidelines group used a set of 62 internal HP-developed guidelines [3], based on established human factors principles and sources (e.g., [8]). These are general guidelines that can be applied across a wide range of computer and instrument systems, and are meant to be used by software developers and evaluators.

When applying the cognitive walkthrough method to this interface, several changes were made to the method as described in [5]. First, the walkthrough was done by a group of evaluators to make it more consistent with the usual procedures for software walkthroughs. Second, since the method has primarily been applied to "walk up and use" interfaces, which have not required distinctions among users with different amounts of knowledge about the interface and its domain, a new procedure for capturing the assumptions of the various kinds of users was needed. This procedure was specified by one of the investigators of the original cognitive walkthrough work. Third, even though the cognitive walkthrough method is task-based, the method does not indicate how tasks are to be selected. In this study, the experimenters selected the walkthrough tasks and provided them to the evaluators. Finally, a pilot experiment using the cognitive walkthrough method allowed us to refine both the procedure and tasks prior to the actual experiment.

Both guidelines and cognitive walkthroughs are intended to be used by the actual designers and implementers of the software being tested. Since we did not have access to the original HP-VUE designers for this study, we used teams of three software engineers. They were members of a research organization (HP Laboratories), but most of them had product experience. They were required to have substantial familiarity with UNIX and X Windows (the computational

platform for HP-VUE), and to have designed and implemented at least one graphical UI. All of the evaluators spent time familiarizing themselves with HP-VUE before doing the evaluation.

The problem report form.
To standardize the reporting of UI problems across techniques, we asked each evaluator or team to report every usability problem they noticed on a special problem report form. The usability test evaluator and each heuristic evaluator submitted separate requests; the guidelines and cognitive walkthrough groups submitted team reports. We defined a usability problem as "anything that impacts ease of use — from a core dump to a misspelled word." On this form, evaluators were asked to briefly describe the problem. We encouraged evaluators to report problems they found even if the technique being used did not lead them to the problem, and to note how they found the problem.

RESULTS

Data refinement.
Evaluators filled out 268 problem report forms; Table 1 shows their distribution across the techniques. We examined the reports for problems that were not directly attributable to HP-VUE. Four such categories were identified: *underlying system* (problems caused by conventions or requirements of one of the systems HP-VUE is built on: UNIX, X Windows, and Motif), *evaluator errors* (misunderstandings on the part of the evaluator), *non-repeatable/system-dependent* (problems that could not be reproduced or were due to aspects of a particular hardware configuration), and *other* (reports that did not refer to usability defects).

Table 1: Total problems found by problem type and evaluation technique.

	Heur Eval	Usa-bility	Guide-lines	Cog Walk	Total
Total	152[2]	38	38	40	268
Underlying system	15	3	3	0	21
Evaluator error	7	0	0	3	10
Non-repeatable	6	3	0	2	11
Other	3	0	0	0	3
Core	121	32	35	35	223
Core, no duplicates	105	31	35	35	206

The categorization was done by three raters, who worked independently and then reconciled their differences as a group. Overall, 45 (17%) of the problems were eliminated, approximately equally across groups. The 223 *core* problems that remained were ones that addressed usability issues in HP-VUE.

We then looked for duplicate problems within evaluation groups. Three raters examined all problems and noted those that, in their judgment, were duplicates. Conflicts were resolved in conference. Combining the 17 sets of within-group duplicates produced a set of 206 problems, on which the analyses below are based.

Problem identification.
Table 1 shows the total number of problems found by each of the techniques. It is clear that, of all the groups of evaluators, the combined set of heuristic evaluators found the largest number of problems, more than 50% of the total problems found. A further analysis looked at how the problems were found by the evaluators: whether by applying the technique as intended, as a side effect of applying the technique (e.g., while applying a guideline about screen layout, a problem with menu organization might be noted), or from prior experience with the system. (See Table 2.) By definition, all of the problems found by the heuristic evaluators were found by heuristic evaluation.

Table 2: Core problems found by technique and how the problem was found.

	Heuristic Eval	Usability	Guide-lines[3]	Cognitive Walkthru
via technique	105 (100%)	30 (97%)	13 (37%)	30 (86%)
side effect	—	1 (3%)	8 (23%)	5 (14%)
prior experience	—	0	12 (34%)	0

This analysis showed that a few problems were found by side effect during both the cognitive walkthrough evaluation and usability testing. However, the most noteworthy finding is that problems found by the guidelines group fell about equally into all three categories. This may indicate that part of the value of a guidelines-based approach lies in its forcing a careful examination of the interface as in the particular guidelines used. The large number of problems found by guidelines evaluators as a result of prior HP-VUE experience is likely due to the fact that two of the three evaluators had worked with HP-VUE previously, and had 40 and 20 hours experience with the system prior to the evaluation.

[2]The individual heuristic evaluators found from 26 to 54 problems.

[3]Only 33 of the 35 guidelines problems are represented here: in the two remaining cases, evaluators neglected to report how they found the problem.

Severity analyses.
Seven individuals rated the severity of the 206 core problems on a scale from 1 (trivial) to 9 (critical). Raters were told to take into account the impact of the problem, the frequency with which it would be encountered, and the relative number of users that would be affected. The raters included four UI specialists and three people with a moderate amount of HCI experience.

Table 3: Mean problem severity by technique.

Heuristic Eval	Usability	Guide-lines	Cognitive Walkthru
3.59	4.15	3.61	3.44

The mean ratings of the different groups (Table 3) varied significantly ($F(3,18)=5.86$, $p<.01$). The higher rating for usability testing may reflect a bias on the part of the raters. While evaluators in the other groups stated their problems in personal or neutral terms, the usability tester used phrases such as "users had trouble ...". Thus, it was easy to tell which problems came from the usability test. Of course, attributing greater severity to problem reports that are backed by data is a reasonable strategy, both for our raters and for developers receiving usability reports.

We also examined severity by ordering the problems by mean rated severity and splitting them into thirds (most severe: 3.86 or more; least severe: 2.86 or less). One-third of the most severe problems can be credited to heuristic evaluators, but so can two-thirds of the least severe (Table 4).

Table 4: Number of problems found by technique and severity.

	Heuristic Eval	Usability	Guide-lines	Cognitive Walkthru
most severe	28	18	12	9
least severe	52	2	11	10

A different notion of problem severity is the certainty that a problem really is one in need of repair. For instance, a missing label marking a numeric field as measuring minutes or seconds is clearly a problem needing to be fixed; the suggestion that animation would be a better way to show hierarchical directory relationships than a tree diagram is more a matter of taste. Preliminary results from ratings of this "real vs. taste" dimension suggest that the heuristic evaluation and cognitive walkthrough groups included more "taste" problems. If confirmed, this finding may reflect the inclusion criteria of the different evaluators, or it may suggest something more fundamental about these evaluation techniques.

We used the severity scores and information about the time spent on each technique to produce a benefit/cost analysis across the four techniques. The summed severity scores of the core problems served as a measure of the value of each evaluation. As a measure of cost, we took the number of person-hours spent by the evaluators for each technique. This time was made up of three components — time spent on the analysis itself, on learning the technique, and on becoming familiar with HP-VUE.

The benefit/cost ratios (problems found per person-hour) were computed in two ways, and are shown in Table 5. The first set of ratios is based on the sums of the times noted, and indicate that heuristic evaluation has a distinct advantage — a four-to-one advantage — over the other methods. The second set of ratios omits the time spent on HP-VUE familiarization by the guidelines and cognitive walkthrough evaluators: if the methods had been applied as ideally intended — by the HP-VUE software developers — they would not require additional time to become familiar with the interface. The same general results obtain in this analysis: heuristic evaluation has about a two-to-one advantage.[4]

Table 5: Benefit/cost ratios for the four techniques.

	Heuristic Eval	Usability	Guide-lines	Cognitive Walkthru
	Sum of severity scores			
	433	133	130	120
	Time spent on analysis (in person-hours)			
analysis time	20	199	17	27
technique training	—	—	5	10
HP-VUE training	15	—[5]	64	6
	Benefit/cost ratios: severity / time			
total time	12	1	2	3
total time minus VUE training			6	3

Content analyses.
We were also interested in the content of the problem reports, in terms of their comments about the interface. We carried out three analyses, each splitting the problems along a different dimension:

- *Consistency*: Did the problem claim that an aspect of HP-VUE was in conflict with some other portion of the system, or not? Overall, about 25% of the problems raised consistency issues.

[4]Note that this second set of ratios gives the guidelines and cognitive walkthrough evaluators credit for all the problems they found, even those not found by applying their respective techniques (see Table 2).

[5]The usability tester had worked with the HP-VUE developers over a period of months; that time is not included in these analyses.

However, only 6% of the problems identified by usability testing were consistency problems.[6]

- *Recurring*: Is this problem one that only interferes with the interaction the first time it is encountered, or is it always a problem? Seventy percent of the problems found by guidelines and usability testing were recurring problems, versus only 50% of those found by heuristic evaluation and cognitive walkthroughs.

- *General*: Did this problem point out a general flaw that affects several parts of the interface, or was the problem specific to a single part? About 40% of the problems were general ones. Sixty percent of the problems found by guidelines were general; usability testing found about equal numbers of both types, and heuristic evaluation and cognitive walkthroughs found a greater number of specific problems than general ones.

DISCUSSION

This study addressed the relative effectiveness of four techniques for evaluating user interfaces, and it offers some insights into each (Table 6).

Overall, the heuristic evaluation technique as applied here produced the best results. It found the most problems, including more of the most serious ones, than did any other technique, and at the lowest cost. However, it is dependent upon having access to *several* people with the knowledge and experience necessary to apply the technique. Our heuristic evaluators were skilled UI professionals, with advanced degrees and years of experience in evaluating interfaces.[7] Such people are a scarce resource and their time is valuable, especially since multiple evaluators are necessary to obtain the kinds of results found here: no individual heuristic evaluator found more than 42 core problems. Another limitation of heuristic evaluation is the large number of specific, one-time, and low-priority problems found and reported.

Usability testing did a good job of finding serious problems: it was second only to heuristic evaluation and was very good at finding recurring and general problems, and at avoiding low-priority problems (although this may be more attributable to the skills of this particular tester than the technique itself). However, this performance came at a significant cost: it was the most expensive of the four techniques. In addition, despite this cost, there were many serious problems that it failed to find.

Table 6: Summary of the study's findings.

	Advantages	Disadvantages
Heuristic evaluation	Identifies many more problems Identifies more serious problems Low cost	Requires UI expertise Requires several evaluators
Usability testing	Identifies serious and recurring problems Avoids low-priority problems	Requires UI expertise High cost Misses consistency problems
Guidelines	Identifies recurring and general problems Can be used by software developers	Misses some severe problems
Cognitive Walk-through	Helps define users' goals and assumptions Can be used by software developers	Needs task definition methodology Tedious Misses general and recurring problems

The guidelines evaluation was the best of the four techniques at finding recurring and general problems. A well-designed set of guidelines serves as a focusing device, forcing evaluators to take a broad look at the interface, rather than limiting their evaluation to a subset of the interface's properties. Such was the case here: in the post-evaluation questionnaire, the guidelines evaluators were more confident that they had covered the entire interface than the heuristic evaluators. However, the guideline-based evaluators missed a large number of the most severe problems, especially if only those problems actually found through the application of the guidelines technique are counted. It is important to remember, however, that the people administering the guidelines were software engineers, rather than UI specialists. If UI specialists are not available, application of the guidelines by developers is a reasonable alternative.

The cognitive walkthrough technique was roughly comparable in performance to guidelines. This was the first time that this technique was applied by a group of evaluators, and to a non-"walk-up-and-use" interface, so these results should not be considered definitive. The findings offer some suggestions for the future development of the cognitive walkthrough technique. A method for defining these tasks, driven by the model underlying the walkthrough methodology, would be a useful addition. While the walkthrough evaluators said that defining user goals and explicitly stating assumptions was a useful exercise, they also felt that the technique was tedious and sometimes required too much detail. In general, the

[6] All differences described in this section are significant at the $p<.05$ level, by a chi-square test.

[7] Because of this difference in evaluators, we cannot directly address Nielsen and Molich's (1990) use of software developers as heuristic evaluators.

problems they found were less general and less recurring than those found by other techniques; future versions of the walkthrough methodology should reformulate the question set to encourage the identification of more general, recurring problems.

The cognitive walkthrough evaluators were only able to complete the analysis of seven tasks during their evaluation session, which was the longest of all the groups. More tasks would be necessary to cover all of HP-VUE, and, had these tasks been completed, more problems would surely have been found. However, once the additional time required to analyze these tasks had been added into the benefit/cost ratios, it is not clear that the method would have fared any differently. In addition, the amount of time required by such an analysis would very likely make the technique unattractive to most software developers.

It should be noted that some of the most important results of the walkthrough analysis — the knowledge that users are assumed to have and the internal states of the system that are relevant to users' interaction with it — would be of significant value to designers and other members of a development team, such as documentation writers, although they fall outside of the scope of the present analysis.

In general, our results show that guidelines and cognitive walkthroughs can be used by software engineers to identify some important usability problems when UI specialists are not available. However, heuristic evaluation and usability testing have advantages over those techniques. Many of the most severe problems found in the study simply could not be identified by guidelines or cognitive walkthroughs. For instance, in this early version of HP-VUE, accidentally deleting your home directory made it impossible to log in at a later time. This problem (since corrected) was found by the usability test, through the inadvertent actions of one of the subjects. It is hard to conceive of a guideline or walkthrough task that would detect this problem without being targeted precisely to that problem, thus failing to be applicable to the design of a wide range of interfaces. Loosely structured techniques, such as heuristic evaluation, run the opposite risk of finding many problems, some of which may not be the most important to correct. Deciding between these techniques requires careful consideration of the goals of the evaluation, the kinds of insights sought, and the resources available. We believe that heuristic evaluation and usability testing draw much of their strength from the skilled UI professionals who use them. The importance of these peoples' knowledge and experience cannot be underestimated.

REFERENCES

1. Apple Computer, Inc. Human interface guidelines: The Apple desktop interface. Cupertino, California, 1989.

2. Brown, C. M. *Human-computer interface design guidelines.* Norwood, N.J.: Ablex Publishing Corporation, 1988.

3. Hewlett-Packard Company, Corporate Human Factors Engineering. *SoftGuide: Guidelines for usable software,* in press, 1990.

4. Lewis, C., & Polson, P. Theory-based design for easily-learned interfaces. *Human-Computer Interaction,* 1990, *5,* 2-3, 191-220.

5. Lewis, C., Polson, P., Wharton, C., & Rieman, J. Testing a walkthrough methodology for theory-based design of walk-up-and-use interfaces. In *Proceedings of CHI'90,* (Seattle, Washington, April 1-5, 1990), ACM, New York, pp. 235-242.

6. Nielsen, J., & Molich, R. Heuristic evaluation of user interfaces. In *Proceedings of CHI'90,* (Seattle, Washington, April 1-5, 1990), ACM, New York, pp. 249-256.

7. Polson, P., Lewis, C., Rieman, J., & Wharton, C. (1990). *Cognitive walkthroughs: A method for theory-based evaluation of user interfaces.* Manuscript submitted for publication.

8. Smith, S. L., & Mosier, J. N. *Guidelines for designing user interface software.* Report MTR-10090, The MITRE Corporation, Bedford, Massachusetts, 1986.

9. Shneiderman, B. *Designing the user interface: Strategies for effective human-computer interaction.* Reading, Massachusetts: Addison-Wesley, 1986

ACKNOWLEDGEMENTS

Thanks are due to the people responsible for the technologies and methodologies under study here — the HP Interface Technology Operation (HP-VUE), HP Corporate Human Factors Engineering (guidelines) and Peter Polson and Clayton Lewis (cognitive walkthrough). In addition, we thank the participant evaluators and raters for their work on the study and comments throughout the experiment: Lucy Berlin, Wendy Fong, Warren Harris, Audrey Ishizaki, Jeff Johnson, Nancy Kendzierski, Brett Kessler, Jay Lundell, Peter Marvit, Bonnie Nardi, Vicki O'Day, Andreas Paepcke, Steve Whittaker, and Craig Zarmer.