Evaluating the Past to Improve Future Techniques: Studying Usability through Heuristic Evaluations

Elise Ransdell
Department of Computer Science
University of Illinois at Springfield
erans@uis.edu

Abstract

In 1990 Nielsen and Molich published nine basic usability heuristics with which to evaluate user interfaces. Using Nielsen and Molich's heuristics as a guideline, two different versions of a commercial piece of software were evaluated for usability problems by a range of evaluators.

The primary purpose of the experiments was to determine how valuable a simple heuristic evaluation would have been had that evaluation been done before the release of the first version. Secondly, the experiments were designed to determine what usability problems were not addressed by the normal testing and release cycle used to produce the second version. Finally, the ideal number of evaluators to use in such tests was investigated.

INTRODUCTION

There are several methods to attempt to determine usability. Each technique finds different problems and deciding which one is best for a particular project requires a balance between insights sought and available resources [2, p.124]. Given unlimited resources, a combination of techniques will yield the best variety and depth of results. However, considering that resources are always limited, this experiment focused on heuristic evaluation techniques.

A heuristic evaluation of an interface is done by examining an interface with a list of heuristics or guidelines in mind and trying to come up with an evaluation about what is good and bad in the interface. Heuristic evaluation has some major advantages that make it a useful technique for an organization that does not have any usability engineering methods already in place. The number one advantage is that heuristic evaluations are inexpensive to implement. In fact, Nielsen commonly refers to heuristic evaluations as a "discount usability- engineering method" [3, p.108]. Nielsen and Molich list three other advantages of heuristic evaluations. They claim the process of doing a heuristic technique is intuitive and thus easy to motivate people to do. It does not require planning in advance. Finally, it can be used early in the development process [5, p.255].

Heuristic guidelines

Development of the heuristics can be difficult. According to Molich and Nielsen, guideline reports of more than 400 pages are common [6, p.338]. However, Molich and Nielsen offer nine categories to use in diagnosing usability problems [6, p.339].

Table 1: Listing of Molich and Nielsen's Heuristic Guidelines

Rule 1	Use simple and natural dialogue
Rule 2	Speak the user's language
Rule 3	Minimize the user's memory load
Rule 4	Be consistent
Rule 5	Provide feedback
Rule 6	Provide clearly marked exits
Rule 7	Provide shortcuts to aid in efficiency of expert users
Rule 8	Provide good error messages
Rule 9	Whenever possible, errors shouldn't occur at all

Molich and Nielsen tested the above nine guidelines on four different interfaces using testers who were not usability experts. They found that in the case where they only used one evaluator on each of the four interfaces, 51%, 38%, 26%, and 20% of known usability problems were found. With 5 evaluators on each interface 90%, 70%, 63%, and 55% of known usability problems were found [5, p.255].

Number of Evaluators

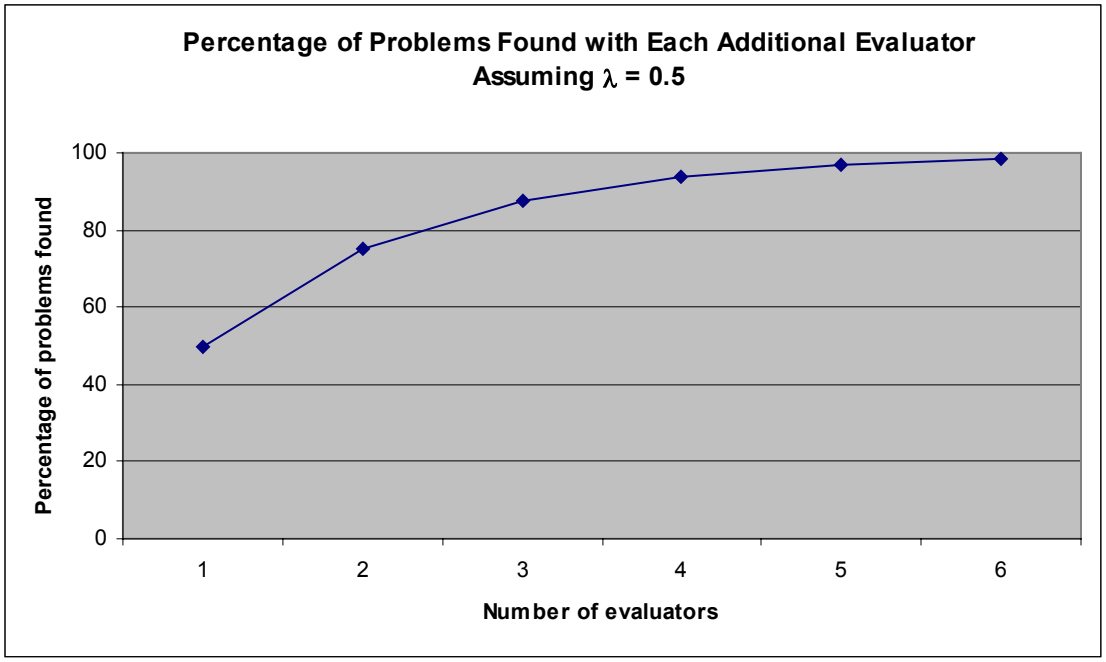
Nielsen and Molich's results raise the question of how many evaluators are needed for an effective heuristic evaluation. Originally, they recommended three to five users [5, p.255]. There is definitely an overlapping area where adding more evaluators does not generate enough additional usability problems to be cost effective. After a while new evaluators will keep finding the same problems as the previous evaluators. In a later paper in 1993, Nielsen and Landauer, using a Poisson model, developed an equation to help companies estimate the number of testers to use. According to their results, the number of usability problems that have been found at least once by i evaluators is:

$$\text{Found}(i) = N(1- (1-\lambda)^{i})$$

In this equation N represents the total number of problems in the interface; λ is the probability of finding the average usability problem when running a single, average heuristic evaluator [4, p.208]. Using this equation, Nielsen and Landauer found that the highest ratio of benefits to costs in medium to large projects calls for 4.4 heuristic evaluators [4, p.212].

Graph 1 displays an example of Nielsen and Landauer's equation with λ set to 0.5. Graphically, it is easy to see what is happening in this equation. After about 4 evaluators the graph dramatically flattens out. This indicates that the additional evaluators are not adding many new problems.

Graph 1: Graph of Landauer and Nielsen's Equation Assuming λ equal to 0.5



Design and Procedures of the Investigation

The Software

In the introduction to Developing User Interfaces, Hix and Hartson state six high risk factors for candidates with usability problems: "It [the interface] was designed by software people, not human-computer interaction specialists. It was developed by strict top-down, functional decomposition. It was not developed to meet written, measurable usability specifications. It was not prototyped as it was being developed. It was not developed by means of an iterative refinement process. And it was not empirically evaluated"[1, p.5]. The software system under evaluation in this paper meets five of these six factors. This makes it an excellent candidate for some type of usability testing. The software does not have a major problem crashing unexpectedly or behaving in a manner that was not planned. It is reasonably reliable. However, does it behave the way the user expects?

The software in this evaluation is a front end to a fairly substantial set of data. At the time of the evaluation, Version 3 of the software was just released. The prior version, Version 2, was being phased out due to serious usability problems.

The Format

Each individual evaluation session lasted between 45 minutes to 1 hour. The sessions began with an informed consent form. The consent form was followed with a general questionnaire. After the evaluator finished the two forms, he/she was presented a brief summary of the history of the application. The summary was designed to provide the tester with a general idea of the goals of the project. After the summary, the list of heuristics was reviewed with each person and he/she was encouraged to ask any questions. The heuristics were followed by a list of possible problem scenarios. The scenarios were examples of information an actual user might try to get from the software. The list of scenarios was merely an aide, not something the tester had to follow. The tester then had 40 to 50 minutes to examine the application. Every time the tester found something that he/she considered a usability problem there was a problem sheet he/she filled out to report the problem. The problem sheet contained a place to mark the heuristic rule(s) broken, the severity of the problem, and a general comment section. The evaluation session was followed by a post evaluation questionnaire.

During each session the tester was continually observed. Any questions the tester had were answered. Additionally, if the tester displayed too much frustration, suggestions were made to help he/she use the application. Following any suggestions, the tester was asked why he/she was feeling frustrated and if the cause of the frustration fit into any of the nine categories.

Participants

Table 1 below displays a layout of the participants in the usability study based on their answers on the general background questionnaire. Four people participated in both a Version 2 and Version 3 test; this is the reason for the inclusion of the total column. There were ten testers of Version 2 and eleven of Version 3 of the software.

Table 2: User Backgrounds

	V2	V3	Total
Background Experience			
Programmer/Systems Person	4	2	4
Computers every day at work	4	6	8
Computers often at Home	1	2	3
Rarely use computers	1	1	2
Age Range			
18-20	0	1	1
21-30	4	2	4
31-45	2	2	3
46-60	1	1	2
61-75	3	5	7
Sex			
Male	7	6	11
Female	3	5	6
Schooling (Highest Level Completed)			
High School	0	1	1
Some College	3	2	5
Undergraduate degree	5	3	5
Master's degree	2	5	6

Results

General Results

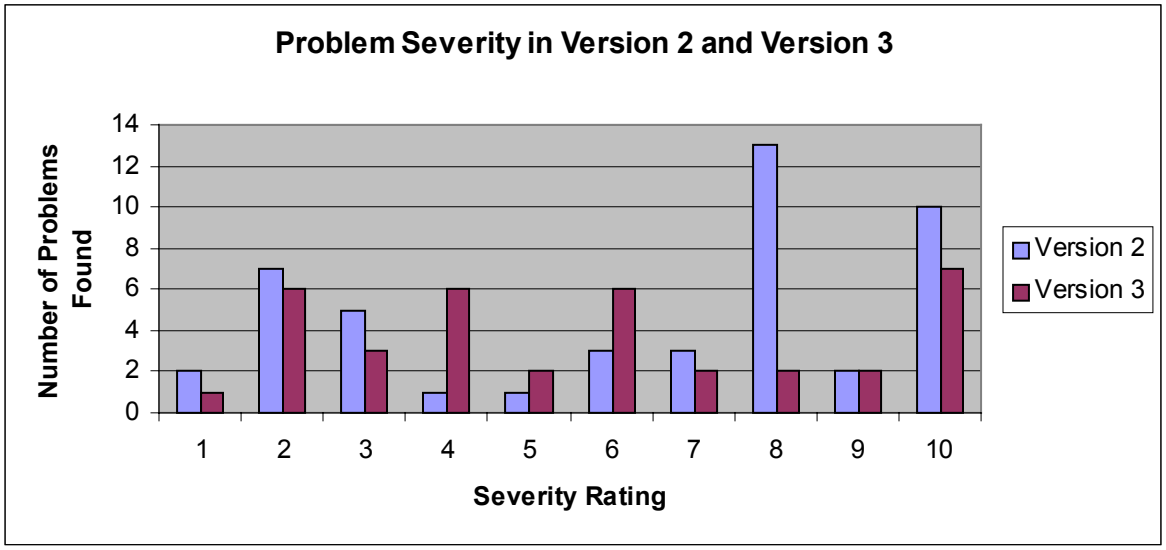
The testers found a total of 87 distinct, valid usability problems in the versions of the software, 49 in Version 2 and 38 in Version 3. Table 3 displays a brief summary of the general results.

Table 3: General Version 2 and Version 3 Results

	V2	V3
Total Valid Problems	49	38
Average Number of Valid Problems Reported Per Tester	6.8	4.63
Standard Deviation of Number of Valid Problems	3.49	2.77
Average Number of Original Problems Reported Per Tester	4.3	3.18
Average Severity of Problems Found	4.7	3.7
Standard Deviation of Severity	4.08	2.26

A closer look at the severity of the problems found per version is shown in graph 2. The severity was rated from 1 to 10 with 1 indicating the problem was almost negligible and 10 equating to a major problem that indicated a major usability problem throughout the application. On the problem report sheet, testers were given the opportunity to rate the severity of the problem themselves. However, that aspect of the problem report generally seemed to cause difficulties because the testers often forgot to fill out the severity portion. When the testers did remember, their suggestions for severity often did not seem to take into account the problem found with respect to the general goals of the application. Therefore, since the severity results obtained from the user were incomplete and often very biased, for continuity the author rated all of the problems with a severity. Thus the results shown in this section are based on the author's rating of severity of the problems testers found, not the tester's individual ratings.

Graph 2: Severity of Problems Found



There were 15 problems reported in Version 2 that were still in Version 3. One-fifth of the 15 problems were found again in the testing of Version 3. The 15 problems had an average severity of 6.45. The standard deviation of the severity was 3.29. Also, the 15 problems were found by an average of 1.3 users in Version 2. The 3 problems that were reported again in Version 3 were found by an average of 2.67 testers.

There were not very many problems reported that were not actually problems. In Version 2 there were 6 false problems reported by 4 users. In Version 3 there were 4 false problems reported by 3 users.

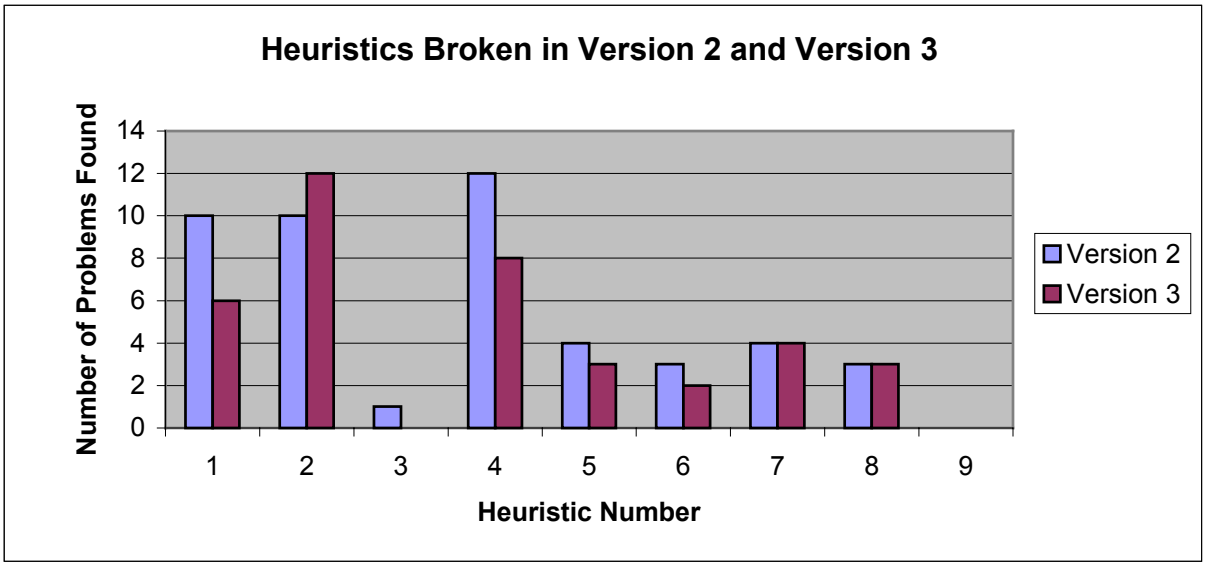
There were 19 problems that were reported by multiple users, 12 in Version 2 and 7 in Version 3. In Version 2 the multiple problems were reported by an average of 2.58 testers. In Version 3 the average was 2.9 testers per problem.

Heuristic Results

This section comprises the heuristic rules broken and the problems found that did not fit into any of the rules. It also includes the misdiagnosed problems. Misdiagnosed problems were problems where the testers correctly identified a usability problem in the application, however they did not correctly identify which heuristic rule was broken. The heuristic rules are referred to as numbers here based on the order in Table 1.

As with severity, the author found it necessary to take a look at each individual problem found for the rules that were broken. Each problem was rated with a "best fit" heuristic rule that was broken to simplify the results. Additionally, a rating of "approximate fits" for each problem was necessary because many problems broke multiple rules. For example, if an error in the program was found with an error message of "Unidentified Object or With Block Set," this obviously broke rule 9 because an error occurred. However, at the same time it also broke rule 8 because the error message was extremely unhelpful. The results shown in Graph 3 display which of the heuristic rules were broken in each problem for the two versions according to my best fit.

Graph 3: Heuristics Broken in Each Version.



Two problems did not fit any of Nielsen's nine guidelines. One tester on Version 2 of the software reported these two problems. The two problems were both visual display usability problems.

The testers misdiagnosed the heuristic broken in 36 problems. It was determined that a rule was misdiagnosed if the tester's report of the rule broken did not match any of the approximate fits. The analysis was generous in diagnosing approximate fits and gave the tester the benefit of the doubt whenever possible. 22 problems were misdiagnosed in Version 2 and 14 in Version 3.

Background Results

There were six different categories filled out on the background questionnaire: age group, computer experience, energy industry experience, program experience, sex, and school level. Computer experience proved to be the most influential factor impacting results. There were 4 different levels of computer experience represented in the result sets. The first level comprises the computer professionals including programmers and systems people. The second level includes people who use computers and computer applications every day at work. These people are experienced with dealing with computer applications to get their jobs done. The third level is comfortable with using computers at home. The fourth level rarely uses computers at work or at home. Table 4 displays a summary of their combined results.

Table 4: Results by Computer Experience – Combined Results

Computer Experience	Programmer/Systems	Work User	Home User	Rare User
Total Users	4	8	3	2
Ave. Problems Reported	8	5.75	2.33	3.5
Ave. Misdiagnosed	2.5	2.25	0.33	0.5
Ave. False	1	0.375	0.33	0
Highest No. By One User	11	13	5	4
Lowest No. By One User	6	2	0	3
Ave. Severity	5.66	7.1	7.43	7.43
Std. Dev. Severity	3.18	2.79	2.37	2.76

Discussion

General

The overall goal of this project was to determine whether a heuristic evaluation of usability could be successfully applied to a production piece of software. Multiple aspects of the results indicate that the heuristic evaluation had merit in this situation. The sheer number of problems found indicates that doing a heuristic evaluation is a good idea. For instance, in the results for Version 2, 49 problems were found. If these had been fixed 3 years before Version 2 went out for the first time, a much more usable piece of software would have been released.

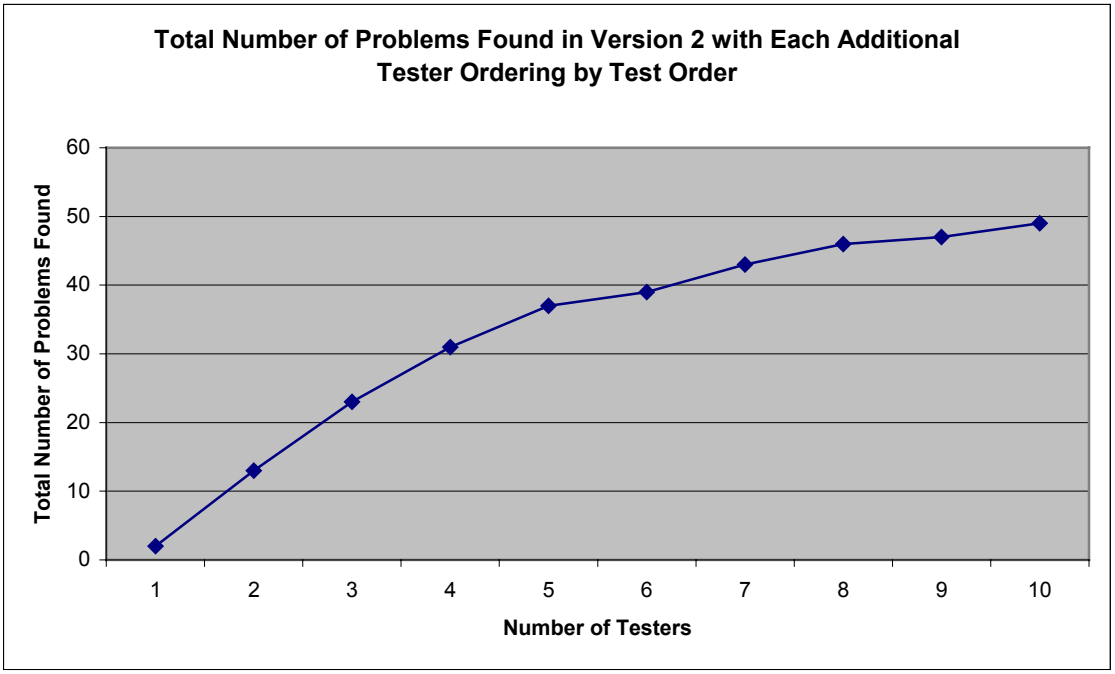
From the feedback of customers and the general evolution of the application, by the release of Version 3, 34 of the problems reported by testers of Version 2 had been found and fixed. However, another result that is very interesting and supports using heuristic evaluations is that 31% of the problems found in the testing of Version 2 still remain in the application to this date. Not only does this indicate an area that needs to be worked on in the software, it also indicates an area where usability in the software has obviously been misdiagnosed consistently. Additionally, as was stated in the result section, the average severity of these remaining problems is rather high at 6.45. In fact, four of the problems were listed as having a severity of 10.

The general downtrend in both average number of problems found per tester and severity of problems found between Version 2 and Version 3 corresponds with what has been heard from clients presently using the software. The software has improved. However, as the numbers indicate, there is still definite room for more improvement. Examining the results from the problems that are in both Version 2 and Version 3 of the product, it is interesting that only 3 of the 15 problems found in the Version 2 test were found by the Version 3 testers. Part of the reason for this is that the application is just so large many testers did not get to everything. Another possible explanation is that although the problems may technically still exist in Version 3, because of the layout change perhaps some are not as obvious or as important. However, close examination of the problems does not seem to suggest that most of the problems are not as obvious. It seems that the testers simply did not notice them or recognize the problems as usability problems. The fact that there are at least 12 known usability problems, some rather severe, in the Version 3 software that were not found in the user testing, does not give a high level of confidence that the heuristic evaluation found most of either the Version 2 or the Version 3 usability problems.

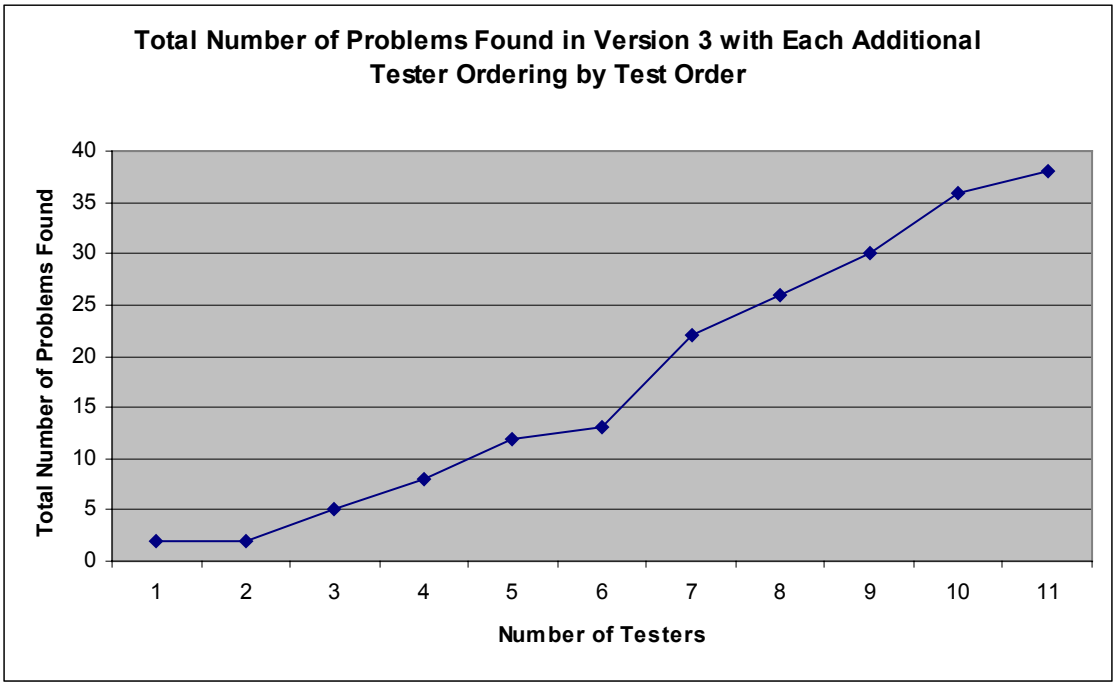
In Nielsen and Molich's results from their tests of four different interfaces using heuristic evaluation, they reported a low percentage of cases where the tester incorrectly labeled something as a usability problem [5, p.253]. Similarly, a very low percentage of the problems reported by testers here were not actual problems. Only 8% of the problems reported in Version 2 and 7% in Version 3 fell into this category. This does not seem to be a major problem with heuristic evaluations.

As was reported in the results, 12 problems in Version 2 and 7 problems in Version 3 were found by multiple testers. This indicates that 28% of the problems reported in Version 2 were repeats and 25.5% of the problems reported in Version 3 were repeats. Graphs 6 and 7 below demonstrate the number of new problems added with each additional tester in Version 2 and Version 3 respectively. It should be noted that the shape of these graphs is dependent upon the order in which the evaluations are processed. For the following two graphs, the evaluations were processed in the order in which the tests were run.

Graph 4: Number of Additional Problems Found in Version 2 with Each Additional Tester in the Order in Which the Tests Were Run



Graph 5: Number of Additional Problems Found in Version 3 with Each Additional Tester in the Order in Which the Tests Were Run

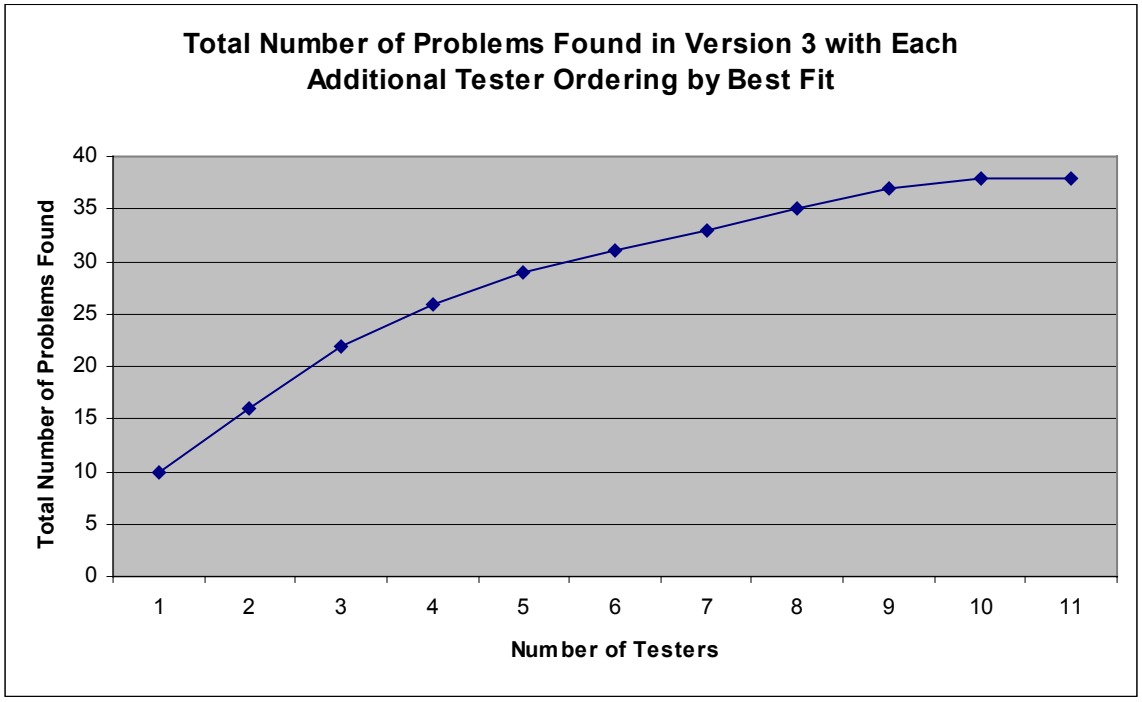


It is interesting to compare graphs above with graph 1. Graph 1 shows the basis for Landauer and Nielsen's estimate that the "best number" of evaluators to use is 4.4. The graph for Version 2 is similar to the Nielsen and Landauer graph. If we were to stop at the 5th evaluator, we would still have found 37 of the 49 total usability problems in

Version 2. This is about 76% of the total problems. Thus, the next 5 testers only accounted for 24% of the usability problems found. This supports Landauer and Nielsen's findings. However, graph 5 depicts a different story. If we were to have stopped after 5 evaluators in Version 3, we would only have found 12 of the 38 problems, or 32%. The last 6 testers accounted for 68% of the problems found.

We can optimize the order in which we process the results to achieve graphs that closely resemble graph 1. For brevity only the graphs for Version 3 is shown. Optimizing the results for Version 2 has the first 5 evaluators finding 38 of the 49 total problems found, approximately 78%. It is interesting to note that this is not much better than using the order in which the actual tests were run. The optimized graph for Version 3, Graph 8, indicates the first 5 evaluators finding 29 of the 38 total problems, approximately 76%. This is considerably better than the actual test order. The graph of Version 3 is steep in the beginning and flatter as the total number of evaluators increases. Mathematically this would indicate a high value for λ.

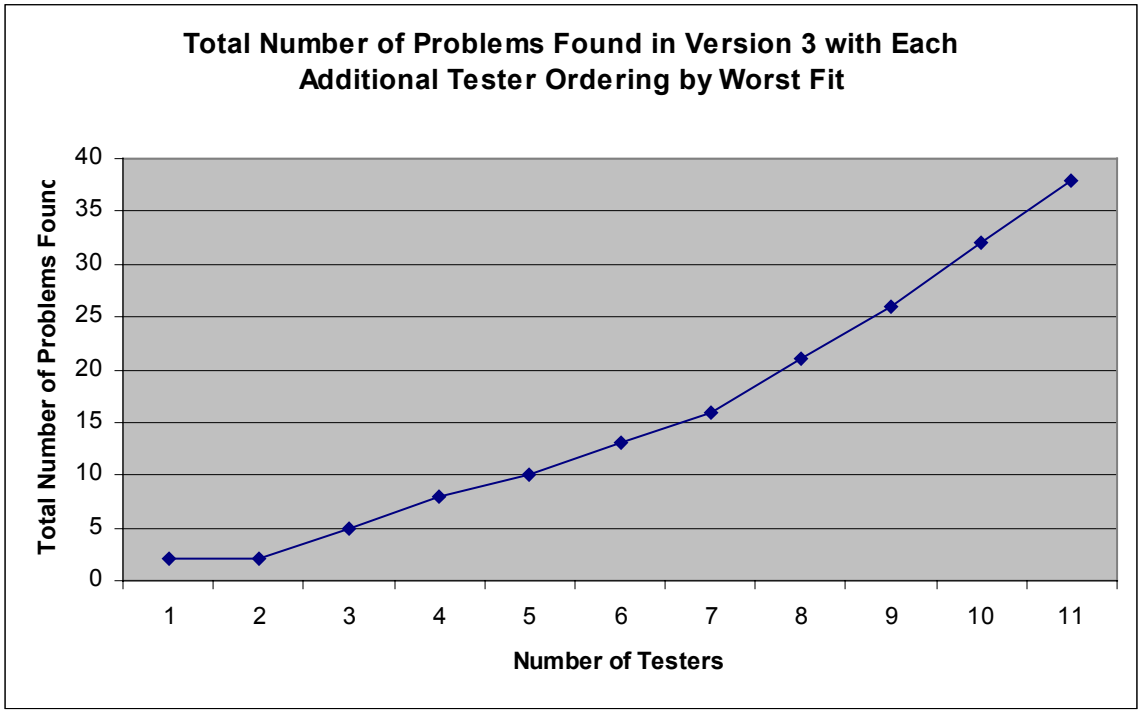
Graph 6: Number of Additional Problems Found in Version 3 with Each Additional Tester Utilizing the Best Ordering Scheme Possible



It is also possible to reorder the results and use the worst ordering possible. Using the worst ordering possible for Version 2, only 18 of the 49 total problems were found by the 5th evaluator. This is approximately 38% of the total problems. Graph 7 shows the worst ordering of evaluators for Version 3. In this graph, 10 of the 38 problems are found by the 5th evaluator. This is approximately 26% of the total problems found. This graph would indicate lower values of λ. The graph is increasing and does not appear as if it is

approaching a value that would represent 100% of the total number of usability problems in the application.

Graph 7: Number of Additional Problems Found in Version 3 with Each Additional Tester Utilizing Worst Ordering Scheme Possible



The differences between the graphs of the best and worst ordering imply that the order of the evaluators dominates the shape of the graphs, not any inherent quality of the results themselves. This is something extremely important when trying to apply Nielsen and Landauer's equation to determine λ and N from individual results. Nielsen and Landauer do discuss some of the assumptions they are making in order to simplify the equation. For instance, they assume that each of the evaluators finds the same number of problems [4, p.208].

A factor, which might be causing the extreme differences in the graphs of the best and worst ordering, might be user type. As is indicated in the results section background, computer experience tended to be a major factor in determining the average number of problems found by a single evaluator in this project. An additional consideration with these graphs is that the users who found the most problems, computer systems people, also found the least severe problems. Finding many problems is important; however a user cannot be discounted who find only one problem, if that problem keeps him/her from using the application at all.

Heuristic

One of the most obvious questions to ask in a heuristic evaluation is how difficult are the heuristics to use? The facts that 32% of the problems in Version 2 and 27% of the problems in Version 3 were misdiagnosed suggest that the rules are somewhat difficult for testers to apply. The question is, do the heuristics still serve a valuable purpose by providing guidance on how to identify usability problems? In the post evaluation form, multiple testers stated that the heuristics were a good starting point in evaluating the usability of the application. However, many noted frustration with the rules, stating that usability problems were difficult to pigeonhole into one (or more) of the nine categories.

One tester, even though he had an explanation of the end goal of the project, absolutely refused to fill in a rule that was broken. In fact, he turned the problem sheet over and reported all of the problems he found on the back of the sheet. Most of the other testers were more receptive; however, a majority of them displayed at least some frustration with the heuristic rules.

Do the heuristic rules keep testers from writing or even noticing some usability problems? As was stated in the results, two problems did not fit any of Nielsen's nine guidelines. One tester on Version 2 of the software reported these two problems. The two problems were both visual display usability problems. They definitely qualified as usability problems. For example, the first one was that the shade of blue used is difficult for colorblind people to distinguish. This was definitely a valid problem since the blue shade indicates different attributes of the program that are not indicated in any other manner. For example, the calculated numbers are blue in the output, and the numbers taken directly from the government data are black. If a user cannot distinguish the difference between the blue and black shades, then there is no way for him/her to tell whether the value was calculated or not.

When observing the testers, while a certain level of frustration was evidenced categorizing the results, there were not many cases where the tester simply gave up and did not write a report of a valid problem because he/she could not categorize them. To minimize the chances of problems being missed because they don't fit one of the nine rules, a tenth category could have been set up to catch anything that might slip through the cracks. However, that seemed somewhat dangerous because testers might not have tried to fit the problem into any of the previous nine categories.

Background

Computer experience seemed to be the biggest determining factor in how effective the heuristic evaluation was. The severity of problems found by the users who use computers often at home, at work, or who rarely use computer was similar. However, on average the systems people who work with computers professionally found much less severe problems. Additionally the standard deviation of the severity is greater for the systems people. This happened because the problems found were either very severe bugs or fairly inconsequential lack of consistency with common Windows functions. For

instance, systems people often noticed that it was not possible to multi-select items from the list box using the control keys.

In observation of the testing, it was found that technical people often dismissed the usability problems they found. They would begin the evaluation by running through the application to solve a problem once. The next time through it was much easier for them. Their end analysis was usually that it was possible to figure out how the software worked, so the software was usable. The problem was that the experienced testers did not realize that since they have much more experience with computers, they catch on to how products work much more quickly than the average user. There were multiple instances where something would trip everyone up the first time; the experienced people would dismiss it as inconsequential the second time, but the general tester often never did figure out what was happening. Computer professionals also tended to derive more satisfaction from finding software flaws rather than usability problems. One systems person even wrote a comment on the post evaluation form that the experience was valuable because "I was able to find a few problems [bugs]."

It is also interesting to note how many more average problems per tester systems people found compared to the other levels of computer experience. Work users also found many more average problems per tester than home or rare users. This can probably be explained by noting the more experienced a tester, the more expectation he/she has from the program. Inexperienced users were also usually so busy simply coping with the application that they did not notice minor problems as often.

Observations

The author observed a disturbing trend for testers to blame themselves for problems. Even after being quite frank and admitting that the reason for this study is that there are serious usability problems with the software, the author heard several comments such as, "Well, if I were smarter, I would have figured that out," and "I guess I am too stupid to use this application." The attitude of self-blame kept some users from noting areas of the software that were problematic for them.

However, more positively, some users seemed to see the list of heuristics as an avenue of escape from self-blame. They looked at problems in the application, and since they could categorize them, they stopped blaming themselves for some problems they have with applications.

Another interesting observation was that some testers seemed to hesitate to critique the product. They did not want to say negative things. One tester even switched to positive things about the application after the first page of problem reports. The tester actually started pointing out and writing down ways the program was helping usability, in the meantime completely ignoring the text on the problem report sheet.

Conclusion

In conclusion, based on the sheer number of problems found, especially in Version 2, a heuristic evaluation certainly proved to be a valuable aide in assessing the software's usability. The usability testing even found problems that were in both versions of the software. It seems like a relatively painless way to catch some major usability issues. Although, based on the problems not found in Version 3 that were known to still exist in the product based on the Version 2 testing, the heuristic evaluation was missing a substantial number of problems in the product.

The ideal number of testers to use seemed to be based more on the type of tester used. It was difficult to draw conclusions based on Neilsen and Landauer's equation. However, noting the results for the systems people suggests that unless they are trained in usability assessment they have difficulty even being aware of the problems that typical users face.

References

1. Hix, Deborah, H.R. Hartson. Developing User Interfaces: Ensuring Usability Through Product and Process, John Wiley & Sons, New York, 1993.

2. Jefferies, R., J.R Miller, C. Wharton, K.M. Uyeda. 1991. "User interface evaluation in the real world: A comparison of four techniques," *Proceedings of CHI '91*, ACM, New Orleans.

3. Nielsen, J. 1990. "Big paybacks from 'discount' usability engineering," *IEEE Software*. May 1990, 107-108.

4. Nielsen, Jakob, Thomas Landauer. 1993. "A mathematical model of the finding of usability problems," *Proc. Of INTERCHI'93*, ACM, Amsterdam, 206-213.

5. Nielsen, Jakob, R. Molich. 1990 "Heuristic evaluation of user interfaces," *Proc. Of CHI'90*, ACM, New York, 249 – 256.

6. Molich, R, J. Nielsen. 1990 "Improving a Human-Computer Dialogue," *Communications of the ACM* 33, 3 (March), 338-348.

Acknowledgements

I would acknowledge all of my testers for their sincere effort on this project.

I would also like to acknowledge Dr. Keith Miller at the University of Illinois at Springfield for his suggestions and comments.