Enhancing the Explanatory Power of Usability Heuristics

Jakob Nielsen
Bellcore
445 South Street
Morristown, NJ 07960
Email: nielsen@bellcore.com (primary) or nielsen.chi@xerox.com (backup)
Electronic business card: nielsen-info@bellcore.com

ABSTRACT

Several published sets of usability heuristics were compared with a database of existing usability problems drawn from a variety of projects in order to determine what heuristics best explain actual usability problems. Based on a factor analysis of the explanations as well as an analysis of the heuristics providing the broadest explanatory coverage of the problems, a new set of nine heuristics were derived: visibility of system status, match between system and the real world, user control and freedom, consistency and standards, error prevention, recognition rather than recall, flexibility and efficiency of use, aesthetic and minimalist design, and helping users recognize, diagnose, and recover from errors.

Keywords: Heuristic evaluation, Usability problems.

INTRODUCTION

Heuristic evaluation [11][13] is a "discount usability engineering" method for evaluating user interfaces to find their usability problems. Basically, a set of evaluators inspects the interface with respect to a small set of fairly broad usability principles, which are referred to as the "heuristics." The original set of usability heuristics used for several early studies was developed with the main goal of making the method easy to teach [12], since it is an important aspect of discount usability engineering that the methods can be widely used and are easy to transfer to new organizations.

In recent years, heuristic evaluation has seen steadily more widespread use, and many users of the method have developed their own sets of heuristics. Also, the user interface literature abounds with lists of general usability principles, even though they are not always explicitly intended for use in heuristic evaluation. Given the many available lists of usability heuristics, it is an open question to what extent one list is better than another and how one could construct an optimal list of usability heuristics. The relative merits of the various lists can only be determined by a shoot-out type comparative test, which is beyond the scope of the present study. Note that it would be insufficient to hand different groups of usability specialists different lists of heuristics and let them have a go at a sample interface: it would be impossible for the evaluators to wipe their minds of the additional usability knowledge they hopefully had, so each evaluator would in reality apply certain heuristics from the sets he or she was supposed not to use.

Instead of finding the "winner" among the existing sets of heuristics, the present study aims at synthesizing a new set of usability heuristics that is as good as possible at explaining the usability problems that occur in real systems. As a seed for this effort, I collected the seven sets of usability heuristics listed in the appendix. As can be seen from the appendix, these sets are very different in scope and nature, and they were indeed selected from the many available lists with the goal of including a wide variety of perspectives on usability.

CHI94-4/94 Boston, Massachusetts USA

RATING THE USABILITY EXPLANATIONS

The usability heuristics were used to explain a database of 249 usability problems collected by the author from 11 earlier projects. Of these 11 projects, 7 were evaluated with heuristic evaluation and 4 with user testing; 4 were evaluated at an early stage of their development lifecycle and 7 were evaluated at a late stage; and 2 had character-based interfaces, 6 had graphical user interfaces, and 3 had telephone-operated interfaces. Each of the 101 usability heuristics was rated for how well it explained each of the 249 usability problems, using the following rating scale:

0 = does not explain the problem at all
1 = may superficially address some aspect of the problem
2 = explains a small part of the problem, but there are major aspects of the problem that are not explained
3 = explains a major part of the problem, but there are some aspects of the problem that are not explained
4 = fairly complete explanation of why this is a usability problem, but there is still more to the problem than is explained by the heuristic
5 = complete explanation of why this is a problem

There is some degree of subjectivity in this kind of rating, so one should not rely on fine distinctions or details in the resulting data. Jeffries [6] found that three usability specialists only had full agreement on about two-thirds of the items in a simple classification of usability problems, and the present rating scale surely also has less than perfect reliabil-

ity. Unfortunately, additional raters were not available as it was necessary to have participated in the original projects in order to assess the degree to which the heuristics explained the usability problems. Thus, it is important not to rely on detailed ratings of individual usability problems.

The appendix gives the mean rating for each usability heuristic, showing how well it was judged to explain the usability problems. It is not reasonable to view this as a kind of competition between the sets of heuristics for several reasons: First, three of the sets were not originally intended for heuristic evaluation (the Star set was intended for interface design, Polson and Lewis' set was limited to improving "guessability," and Carroll and Rosson's set was intended for claims analysis) and these three sets do indeed achieve lower scores than the others. Second, the database of usability problems includes many problems from character-based interfaces and telephone-operated interfaces, which may not be a strength of the Macintosh and SunSoft heuristics since they were probably optimized for graphical user interfaces. Finally, the original set of heuristics no doubt has an advantage since a large part of the database comes from interfaces that were studied as part of the original heuristic evaluation project.

FACTOR ANALYSIS

A principal components analysis of the data shows that it is not the case that a few factors account for most of the variability in the usability problems. The two most important factors account for about 6% of the variance each. The seven factors that account for more than 3% of the variance each only add up to 30% of the variance. Indeed, there is a gradual decline in the significance of the factors, with no particular sharp drop-off point that might indicate that a core factor set had been found. There are 25 factors that account for 1% or more of the variance each, and these 25 factors together account for 62% of the variance.

The following is a list of the seven most important factors from the factor analysis. Each factor was given a descriptive name in order to summarize the underlying usability phenomenon that seems to be covered by most of the heuristics that are highly loaded for that factor. For each factor, the list states the proportion of the total variance in the usability problem ratings accounted for by that factor. Finally, the heuristics with loadings of .25 or more are listed for each factor (the codes in front of the heuristics refer to the appendix where many of them are explained in more detail).

Factor 1: Visibility of system status	**6.1%**
A5 Feedback: keep user informed about what goes on	.81
C8 Provide status information	.70
F7 Feedback: show that input has been received	.70
E13 Features change as user carries out task	.69
G4 Feedback provided for all actions	.56
G5 Feedback timely and accurate	.48
E10 Indicate progress in task performance	.46
F2 Direct manipulation: visible objects, visible results	.39
D3 Identity cues system response vs. user's goals	.34
C13 Show icons and other visual indicators	.32
F5 WYSIWYG: do not hide features	.32
E15 What incorrect inferences are most likely	.27
Factor 2: Match between system and real world	**5.9%**
A2 Speak the user's language	.78
C7 Contains familiar terms and natural language	.71
G2 Speak the user's language	.67
F1 Metaphors from the real world	.63
B1 Familiar user's conceptual model	.62
E7 Use of user's background knowledge	.51
C6 Learnable through natural, conceptual model	.47
G18 Follow real-world conventions	.45
B3 Screen representation matches non-computer	.37
E2 Encourage users to import pre-existing tasks	.35
D2 Identity cues between actions and user's goals	.31
G3 Understand the user's language	.27
Factor 3: User control and freedom	**4.6%**
G23 Undo and redo should be supported	.89
D4 Obvious way to undo actions	.86
F8 Forgiveness: make actions reversible	.75
C18 Ability to undo prior commands	.64
A6 Clearly marked exits	.52
C19 Ability to re-order or cancel tasks	.45
B7 Modeless interaction	.31
F6 User control: allow user to initiate/control actions	.30
F11 Modelessness: allow users to do what they want	.27
Factor 4: Consistency and standards	**4.2%**
A4 Consistency: express same thing same way	.87
B5 Consistency	.87
F4 Consistency: same things look the same	.86
C3 Uniform command syntax	.57
G19 Conform to platform interface conventions	.46
C4 Consistent key definitions throughout	.34
B4 Universal commands: a few, generic commands	.33
C5 Show similar info at same place on each screen	.31
Factor 5: Error prevention	**3.7%**
A9 Prevent errors from occurring in the first place	.83
G22 System designed to prevent errors	.73
G3 Understand the user's language	.54
E6 What planning mistakes are most likely?	.37
E9 What slips are most likely?	.35
D2 Identity cues between actions and user's goals	.30
Factor 6: Recognition rather than recall	**3.1%**
F3 See-and-point instead of remember-and-type	.72
D1 Make the repertoire of available actions salient	.68
B2 Seeing and pointing: objects and actions visible	.57
G16 All user needs accessible through the GUI	.53
E12 What features often missed and at what cost?	.52
C10 Provide lists of choices and picking from lists	.42
A3 Minimize the users' memory load	.37
F2 Direct manipulation: visible objects, visible results	.33
E8 Easy or difficult to perform (execute) task?	.32
E1 Evoke goals in the user	.31
C20 Allow access to operations from other apps.	.30
A6 Clearly marked exits	.29
C13 Show icons and other visual indicators	.29
G20 Integrated with the rest of the desktop	.27
Factor 7: Flexibility and efficiency of use	**2.8%**
G14 Accelerators should be provided	.80
A7 Shortcuts: Accelerators to speed up dialogue	.80
B8 User tailorability to speed up frequent actions	.62
F6 User control: allow user to initiate/control actions	.43
G12 System should be efficient to use	.42
G17 User interface should be customizable	.42
C19 Ability to re-order or cancel tasks	.28
G21 Keyboard core functions should be supported	.26
G11 Physical interaction with system feels natural	.26

The last three factors in the top ten, each accounting for about 2% of the variance, can be described as aesthetic and minimalist design, well-structured features that are easy to discriminate, and use of default values so that the user does not have to re-enter information. Note, by the way, that the labels used to describe the factors are the author's subjective

attempt to abstract the main usability thrust of each factor. It would have been possible to use other names instead.

The difference between factors 1 and 6 seems to be that "visibility of system status" deals mostly with revealing what is happening in the system, whereas "recognition rather than recall" deals mostly with making the user's future options salient. The difference between factors 3 and 7 seems to be that "user control and freedom" is focused on minimizing the extent to which the system traps the user in a specific state from which there is no escape, whereas "flexibility and efficiency of use" is focused on allowing the user additional options to sidestep the regular interaction techniques.

The factors revealed by the factor analysis do seem to cover fundamental usability principles, but unfortunately it was not possible to account for a reasonably large part of the variability in the usability problems with a small, manageable set of usability factors. In other words, usability problems are due to a broad variety of underlying phenomena.

EXPLANATORY COVERAGE

53 factors are needed to account for 90% of the variance in the usability problems. This is too much for practical heuristic evaluations where evaluators are asked to compare each interface element against the list of heuristics. Thus, instead of finding a set of usability factors that account for all usability phenomena, we will have to reduce our ambitions to finding a set of usability heuristics that account reasonably well for the majority of the usability problems. It is likely that the seven (or ten) factors listed in the previous section could be used as such a set, but we do not currently have empirical evidence to confirm the value of this new set of heuristics.

Instead, we will look at the explanatory coverage that is possible by various combinations of the existing heuristics for which we do have data. Since we have seen that perfection is impossible with a reasonably small set of heuristics, we will consider a usability problem to be "explained" by a set of heuristics if it has achieved an explanation score of at least 3 ("explains a major part of the problem, but there are some aspects of the problem that are not explained") from at least one of the heuristics in the set. With this scoring method, a set of heuristics did not get additional credit for having multiple heuristics that explained a problem. This was done because it is currently an open issue to what extent it is better to have a good match between a usability problem and a single heuristic (meaning that the evaluator has it pegged) or to have a match between the problem and several heuristics (meaning that more aspects of the problem are known). The appendix lists the proportion of usability problems "explained" by each heuristic as well as the proportion of problems explained by each set of heuristics.

The widest explanatory coverage will be realized by first choosing the heuristic that explains the most usability problems, then adding the heuristic that explains the most of the remaining problems (i.e., those that have not already been explained), and so on. The top part of Table 1 lists the ten heuristics that taken together explain the most usability problems as assessed by this approach.

Top Heuristics to Explain All the Usability Problems		
A4 Consistency: same thing, same way	23%	23%
A2 Speak the user's language	16%	39%
F7 Feedback: show receipt of user's input	13%	52%
B2 Seeing/pointing vs. remembering/typing	7%	59%
F10 Aesthetic integrity, keep design simple	7%	65%
A7 Shortcuts and accelerators	6%	71%
G18 Real-world conventions	4%	76%
E18 Help error recognition/recovery	4%	80%
F8 Forgiveness: reversible computer actions	3%	83%
D1 Salient repertoire of available actions	2%	85%

Top Heuristics to Explain the Serious Usability Problems		
B2 Seeing/pointing vs. remembering/typing	22%	22%
F4 Consistency: same thing looks the same	18%	40%
G5 Feedback timely and accurate	17%	57%
D1 Salient repertoire of available actions	12%	70%
F8 Forgiveness: reversible computer actions	7%	77%
B1 Familiar user's conceptual model	5%	82%
F7 Feedback: show receipt of user's input	5%	87%
A9 Prevent errors from occurring	4%	90%
D5 Easy to discriminate action alternatives	2%	93%
B7 Modeless interaction	2%	95%

Table 1 *The ten heuristics that achieve the widest coverage with respect to explaining usability problems. The top list are heuristics to explain the complete database of 249 usability problems and the bottom list are heuristics to explain the 82 serious usability problems. For each heuristic, the first percentage indicates the proportion of problems it explains (that have not already been explained by a higher-ranked heuristic), and the second percentage indicates the cumulative proportion of usability problems explained by at least one element of the list of heuristics.*

Concentrating on Major Usability Problems

It is often noted that a very large proportion of the usability problems found by heuristic evaluation tends to be minor problems [7]. This preponderance of minor problems is seen as a drawback by many [2], even though it is still possible to focus on the serious problems by using a severity rating method [8][11] to prioritize the list of usability problems found by a heuristic evaluation of a given interface. In any case, it is probably desirable to increase the proportion of serious usability problems found by heuristic evaluation.

Of the 249 usability problems in the database used for the present analysis, 82 can be classified as serious usability problems in that they have high potential for causing major delays or preventing the users from completing their task [11]. The bottom part of Table 1 lists those heuristics that give maximum explanation coverage for this set of serious usability problems. It can be seen from the table that the major usability problems are somewhat more concentrated around a few heuristics than is the group of usability problems as a whole: the four heuristics with the widest explanatory coverage explain 70% of the major problems but only 65% of the full set of problems (which is dominated by the 167 minor problems).

It can be seen from Table 1 that there is not much difference between the heuristics that explain the full database and those that explain the major problems. Most principles occur in both lists, either as exact duplicates or in slightly alternative wordings. The main difference seems to be that the list of heuristics covering the serious problems gives more weight to usability principles associated with making things visible and salient in the interface (to the extent that there are *two* feedback rules on the list—the reason this can happen is that these rules were described differently in the source documents, meaning that there is some degree of non-overlap in the problems they explain).

Comparing the heuristics explaining the major problems with those explaining the minor problems (not listed for reasons of space), shows that the heuristics in the top-10 for the major problems that are not in the top-10 for the minor problems are D1 (make the repertoire of available actions salient), B1 (familiar user's conceptual model), A9 (prevent errors), D5 (easy to discriminate available action alternatives), and B7 (modeless interaction). Thus, closer attention to these heuristics may help increasing the proportion of serious usability problems found by heuristic evaluation.

Among the five heuristics with the widest explanatory coverage of minor usability problems, three do not occur on the top-10 list for major problems: A2 (speak the user's language), F10 (aesthetic integrity), and A7 (shortcuts and accelerators). One might argue that these heuristics should be disregarded in the future since they tend to find minor problems. Even so, F10 and A7 should be kept since aesthetic integrity is important for subjective satisfaction and sales and shortcuts and accelerators are relevant for expert user performance. These qualities are important for overall usability even though any individual usability problem in these categories will not cause the system to be unusable which is why they tend not to be classified as major.

CONCLUSIONS

Almost all of the seven usability factors found above are represented in the lists of top-10 heuristics in Table 1. The exceptions are that factor 5 (error prevention) is not represented in the set of heuristics to explain the full database and factor 7 (flexibility and efficiency of use) is not represented in the set of heuristics to explain the major usability problems. Given the above comments that efficiency issues are important even though they were often not classified as major problems, it would seem that Table 1 indicates the potential for the seven usability factors to form the backbone of an improved set of heuristics. Two important heuristics from Table 1 are left out from the usability factors: F10 (aesthetic integrity) and E18 (help users to recognize, diagnose, and recover from errors). Error handling and aesthetic integrity should probably be added as the eight and ninth heuristics to the set of factors.

The analysis in this paper has thus resulted in a candidate set of nine heuristics: visibility of system status, match between system and the real world, user control and freedom, consistency and standards, error prevention, recognition rather than recall, flexibility and efficiency of use, aesthetic and minimalist design, and helping users recognize, diagnose, and recover from errors. These heuristics seem to be excellent for *explaining* previously found usability problems. It remains to be seen to what extent they are also good for *finding* new problems, which of course is the main goal of heuristic evaluation.

Acknowledgments

The author would like to thank Alan McFarland and nine *CHI'94* referees for comments on earlier versions of this manuscript.

References

1. Apple Computer. *Macintosh Human Interface Guidelines.* Addison-Wesley, Reading, MA, 1992.
2. Brooks, P. Adding value to usability testing. In Nielsen, J., and Mack, R. L. (Eds.), *Usability Inspection Methods*, John Wiley & Sons, New York, NY, 1994, 253–270.
3. Carroll, J. M., and Rosson, M. B. Getting around the task-artifact cycle: How to make claims and design by scenario. *ACM Trans. Infor. Systems* **10**, 2 (April 1992), 181–212.
4. Holcomb, R., and Tharp, A. L. An amalgamated model of software usability. In Knafl, G. (Ed.), *Proceedings of the 13th IEEE COMPSAC International Conference*, IEEE Computer Society, Washington, D.C., 1989.
5. Holcomb, R., and Tharp, A. L. What users say about software usability. *International Journal of Human–Computer Interaction* **3**, 1 (1991), 49–78.
6. Jeffries, R. Usability problem reports: Helping evaluators communicate effectively with developers. In Nielsen, J., and Mack, R. L. (Eds.), *Usability Inspection Methods*, John Wiley & Sons, New York, NY, 1994, 271–292.
7. Jeffries, R. J., Miller, J. R., Wharton, C., and Uyeda, K. M. User interface evaluation in the real world: A comparison of four techniques. *Proc. ACM CHI'91 Conf.* (New Orleans, LA, 28 April 28–3 May), 119–124.
8. Karat, C. A comparison of user interface evaluation methods. In Nielsen, J., and Mack, R. L. (Eds.), *Usability Inspection Methods*, John Wiley & Sons, New York, NY, 1994, 203–232.
9. Molich, R., and Nielsen, J. Improving a human–computer dialogue. *Communications of the ACM* **33**, 3 (March 1990), 338–348.
10. Nielsen, J. *Usability Engineering.* Academic Press, Boston, MA, 1993.
11. Nielsen, J. Heuristic evaluation. In Nielsen, J., and Mack, R. L. (Eds.), *Usability Inspection Methods*, John Wiley & Sons, New York, NY, 1994, 25–64.
12. Nielsen, J., and Molich, R. Teaching user interface design based on usability engineering. *ACM SIGCHI Bulletin* **21**, 1 (July 1989), 45–48.
13. Nielsen, J., and Molich, R. Heuristic evaluation of user interfaces. *Proc. ACM CHI'90 Conf.* (Seattle, WA, 1–5 April 1990), 249–256.
14. Polson, P. G., and Lewis, C. H. Theory-based design for easily learned interfaces. *Human–Computer Interaction* **5**, 2&3 (1990), 191–220.
15. Rohn, J. A. *Usability Engineering: Improving Customer Satisfaction While Lowering Development Costs.* Brochure, SunSoft, Inc., Mountain View, CA, 1993.
16. Smith, D. C., Irby, C., Kimball, R., Verplank, B., and Harslem, E. Designing the Star user interface. *BYTE* 7, 4 (April 1982), 242–282.

APPENDIX: LIST OF SEVEN SETS OF HEURISTICS FROM THE USER INTERFACE LITERATURE

In most cases, the sets of heuristics suggested by other authors have been rewritten for the sake of brevity and to achieve a consistent format. The exact wording of these heuristics as printed here is therefore the responsibility of the present author and does not necessarily correspond to the way the original authors would have edited their principles.

For each heuristic, the table lists its mean explanatory score across the 249 usability problems in the sample. The explanatory power of each heuristic was scored on a 0–5 scale for each usability problem, with 0 indicating that the heuristic did not explain the problem at all and 5 indicating that the heuristic provided a complete explanation of why the user interface issue in question constituted a usability problem. The table also lists the proportion of the usability problems that were explained at a level of 3 or more, with a score of 3 indicating that the heuristic explained a major part of the problem while leaving some aspects of the problem unexplained.

For each full set of heuristics (indicated by **boldfaced** type), the table lists the mean across usability problems of the best explanation provided by any heuristic in the group as well as the proportion of problems for which the set had at least one heuristic explaining the problem at a level of at least 3.

Code	*Usability Heuristic*	*Mean explanation rating*	*Problems explained at 3 or more*
A	**The ten usability heuristics explained in detail in [10]. This is a slightly modified version of the original heuristics used by Molich and Nielsen [9][13]**	**3.72**	**82%**
A1	*Simple and natural dialogue:* Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility. All information should appear in a natural and logical order.	.78	10%
A2	*Speak the user's language:* The dialogue should be expressed clearly in words, phrases and concepts familiar to the user, rather than in system-oriented terms.	1.04	20%
A3	*Minimize the users' memory load:* The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.	.53	10%
A4	*Consistency:* Users should not have to wonder whether different words, situations, or actions mean the same thing.	1.14	23%
A5	*Feedback:* The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.	.70	12%
A6	*Clearly marked exits:* Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue.	.28	6%
A7	*Shortcuts:* Accelerators—unseen by the novice user—may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users.	.41	8%
A8	*Good error messages:* They should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.	.51	10%
A9	*Prevent errors:* Even better than good error messages is a careful design that prevents a problem from occurring in the first place.	.64	11%
A10	*Help and documentation:* Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, be focused on the user's task, list concrete steps to be carried out, and not be too large.	.23	4%
B	**The usability principles used in the design of the Star user interface [16]**	**2.38**	**49%**
B1	Familiar user's conceptual model: Use analogies and have the user interact with concrete objects	.40	7%
B2	Seeing and pointing versus remembering and typing: Make objects and actions visible. Allow users to create new objects by copying and editing old ones.	.49	10%
B3	What you see is what you get: Screen representation of objects matches their non-computer representation	.47	6%
B4	Universal commands: A few, basic generic commands used throughout the system	.22	4%
B5	Consistency.	1.08	22%
B6	Simplicity: Simple things should be simple; complex things should be possible.	.40	6%
B7	Modeless interaction: Follow the noun-verb syntax. Have each mechanism serve one purpose.	.19	3%
B8	User tailorability: Allow speed-up of frequently performed operations (e.g., document templates, meta-operations) and changes in interface appearance (e.g., change file sort order).	.21	4%

Code	Usability Heuristic	Mean explanation rating	Problems explained at 3 or more
C	**Usability principles studied by Holcomb and Tharp [4][5]**	**2.90**	**64%**
C1	Able to accomplish the task for which the software is intended.	.10	2%
C2	Perform tasks reliably and without errors.	.15	3%
C3	Uniform command syntax.	.51	10%
C4	Consistent key definition throughout.	.23	4%
C5	Show similar information at the same place on each screen.	.36	6%
C6	Learnable through natural, conceptual model.	.24	4%
C7	Contains familiar terms and natural language.	.69	14%
C8	Provide status information.	.54	11%
C9	Don't require information entered once to be re-entered.	.14	3%
C10	Provide lists of choices and allow picking from the lists.	.08	0%
C11	Provide default values for input fields.	.04	0%
C12	Prompt before destructive operations.	.10	2%
C13	Show icons and other visual indicators.	.11	2%
C14	Immediate problem and error notification.	.18	4%
C15	Messages that provide specific instructions for actions.	.49	9%
C16	On-line help system available.	.07	1%
C17	Informative, written documentation.	.10	2%
C18	Ability to undo results of prior commands.	.14	2%
C19	Ability to re-order or cancel tasks.	.29	6%
C20	Allow access to operations from other applications/operating system from within the interface	.05	1%
D	**Design principles for successful guessing suggested by Polson and Lewis [14]**	**2.31**	**47%**
D1	Make the repertoire of available actions salient.	.42	9%
D2	Use identity cues between actions and user goals.	.52	12%
D3	Use identity cues between system responses and user goals.	.80	13%
D4	Provide an obvious way to undo actions.	.28	6%
D5	Make available action alternatives easy to discriminate.	.32	6%
D6	Offer few alternatives: This increases the chance of guessing the correct one.	.38	7%
D7	Tolerate at most one hard-to-understand action in a repertoire from which the user has to select.	.13	2%
D8	Require as short a chain of choices as possible to complete an action.	.17	4%
E	**Artifact claims analysis questions listed by Carroll and Rosson [3]**	**1.99**	**44%**
E1	How does the artifact evoke goals in the user?	.41	7%
E2	How does the artifact encourage users to import pre-existing tasks?	.21	2%
E3	How does the artifact suggest that a particular task is appropriate or inappropriate?, simple or difficult?, basic or advanced?, risky or safe?	.29	5%
E4	What inappropriate goals are most likely?, most costly?	.10	1%
E5	What distinctions must be understood in order to decompose a task goal into methods?, how are these distinctions conveyed by the artifact?	.39	7%
E6	What planning mistakes are most likely?, most costly?	.23	3%
E7	How does the artifact encourage the use of background knowledge (concepts, metaphors, skills) in planning a task?	.29	4%
E8	How does the artifact make it easy or difficult to perform (execute) a task?	.25	3%
E9	What slips are most likely?, most costly?	.30	6%
E10	How does the artifact indicate progress in task performance?	.10	2%
E11	What are the most salient features of the artifact?, what do these features communicate to the user?	.18	2%
E12	What features are commonly missed and at what cost?	.17	3%
E13	What features of the artifact change as users carry out a task?, what do these changes communicate to the user?	.26	4%
E14	How does the artifact guide the user to make correct inferences?	.24	4%
E15	What incorrect inferences are most likely?, most costly?	.13	2%

Code	Usability Heuristic	Mean explanation rating	Problems explained at 3 or more
E16	How does the artifact encourage the use of background knowledge in making inferences?	.06	0%
E17	How does the artifact convey completion of a task?	.03	0%
E18	How does the artifact help users to recognize, diagnose, and recover from errors?	.45	10%
E19	How does the artifact encourage elaboration and retrieval of task goals and methods?	.11	1%
F	**Human interface principles listed in the Macintosh Human Interface Guidelines [1]**	**3.09**	**66%**
F1	*Metaphors* from the real world to take advantage of people's knowledge of the world.	.31	6%
F2	*Direct manipulation:* objects on screen remain visible while user performs physical actions on them, and the impact of these operations is immediately visible.	.24	3%
F3	*See-and-point* instead of remember-and-type: users act by choosing between visible alternatives	.43	8%
F4	*Consistency:* same thing looks the same, same actions are done the same way.	1.11	22%
F5	*WYSIWYG* (what you see is what you get): do not hide features (unless there is a way to make hidden things visible)	.28	3%
F6	*User control:* allow the user to initiate and control actions.	.46	7%
F7	*Feedback:* immediately show that user's input has been received and is being operated on. Inform users of expected delays. Also, tell the user how to get out of the current situation.	.76	14%
F8	*Forgiveness:* make computer actions reversible. Always warn people before they lose data.	.32	6%
F9	*Perceived stability:* finite set of objects that do not go away (but may be dimmed).	.35	5%
F10	*Aesthetic integrity:* things should look good, keep graphic design simple, follow the graphic language of the interface without introducing arbitrary images to represent concepts.	.77	12%
F11	*Modelessness:* allow people to do whatever they want whenever they want it.	.20	3%
F12	*Accessibility* for users who differ from the "average" user (cognitive or physical limitations, different culture and language of worldwide users)	.12	2%
G	**SunSoft usability guidelines [15]**	**3.31**	**73%**
G1	Core functionality should be understandable within an hour	.04	1%
G2	System should speak the user's language	.78	14%
G3	System should understand the user's language	.29	6%
G4	Feedback should be provided for all actions	.32	6%
G5	Feedback should be timely and accurate	.57	12%
G6	UNIX® concepts should be minimized (in general, minimize underlying system concepts)	.18	2%
G7	User sensibilities should be considered	.29	6%
G8	Functions should be logically grouped	.10	1%
G9	Interface should be logically ordered	.12	2%
G10	Core functionality should be clear	.16	3%
G11	Physical interaction with the system should feel natural	.21	2%
G12	System should be efficient to use	.31	6%
G13	Reasonable defaults should be provided	.07	1%
G14	Accelerators should be provided	.31	6%
G15	Users should not have to enter system-accessible information	.12	2%
G16	Everything the user needs should be accessible through the GUI (or, in general, through whatever interface style is chosen for the interface)	.13	3%
G17	The user interface should be customizable	.11	2%
G18	System should follow real-world conventions	.72	15%
G19	System should follow platform interface conventions	.50	10%
G20	System should be effectively integrated with the rest of the desktop	.06	2%
G21	Keyboard core functions should be supported	.17	3%
G22	System should be designed to prevent errors	.49	8%
G23	Undo and redo should be supported	.21	4%
G24	Good visual design: There is no substitute for a good graphic artist	.54	7%

UNIX is a registered trademark of Unix System Laboratories.