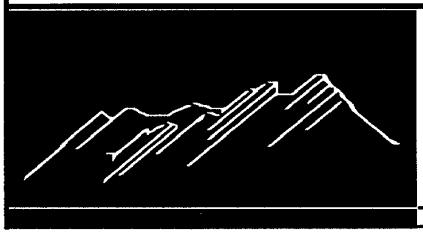
INSTITUTE OF COGNITIVE SCIENCE



Technical Report

University of Colorado, Boulder

The Role of Psychological Theory in Usability Inspection Methods

Cathleen Wharton & Clayton Lewis

Institute of Cognitive Science
University of Colorado
Boulder, Colorado 80309-0344

Technical Report #93-06

The Role of Psychological Theory in Usability Inspection Methods

Cathleen Wharton

Clayton Lewis

University of Colorado at Boulder

Department of Computer Science

and Institute of Cognitive Science

Campus Box 344

Boulder, CO 80309-0344

ICS Technical Report #CU-ICS-93-06

This paper will appear as a book chapter in "Usability Inspection Methods" edited by J. Nielsen and R. Mack. The publisher is John Wiley & Sons (New York) and the book will be published in 1994.

The Role of Psychological Theory in Usability Inspection Methods

Cathleen Wharton

Clayton Lewis

University of Colorado at Boulder

Someone once told one of us that "the only good user is a dead user." At the other extreme one could say that "software can only be successful if extensive user testing and detailed analyses of a user's cognitive skills are undertaken prior to a product's release." But constraints in development practices, schedules, resources, and time-based competition in industry do not currently allow for such extensive usability work. This creates the context in which the first remark was formulated.

From a researcher's perspective the context of real usability work provides a challenge. How can the theoretical ideas of the researcher aid a practitioner who is always short of time, and for whom usability is only one of many important design and development goals, not all of which can be met?

In this paper we consider what can be done to bring psychological theory to bear on actual design problems, in real software development organizations. We begin with a very quick sketch of psychological theory in which we point out some of its aspects that are potentially relevant to software design. We then consider how these theoretical ideas might be applied in practical settings.

What psychological theory has to offer

Psychologists have established a number of important facts about the mental machinery of people which are pertinent to design problems. Psychological theory aims to connect these and other facts into a predictive and explanatory framework, in much the same way that a physical theory, like Newtonian mechanics, aims to predict and explain the behavior of physical systems. (What will happen when we let go of the neck of this balloon? Why?) To provide background for our subsequent discussion we list here some of the facts that a psychological theory might encompass, stressing those facts that we think designers need to apply.

Some things are easier to see or hear than others. Everybody knows this, but there are models that provide quantitative data on the effects of contrast, loudness, size, and like factors.

Some things don't look or sound the way you would think. For example, the same color looks very different against different backgrounds. This means that in using color to identify areas of a map that share some common feature one may actually have to adjust the colors to get different patches that are supposed to be the same color look the same.

Only some of the contents of a complex display are likely to be seen. What is seen depends on size, color, organization, and movement as well as where the viewer is looking, what the user knows about the structure of the scene, and what the user is trying to do. As a simple example, suppose you have two sets of data displayed, each in a different color. To the user you have two groups of data and if the elements are in close proximity, this association will be even stronger (Tullis, 1988).

Precise movements take longer than gross movements. (Fitts and Posner, 1967). For example, if an interface makes use of numerous buttons or sequences of button presses, the distance between buttons in the sequence, and their size becomes an issue. There are good

quantitative models which predict how long a movement of a given length and requiring a given precision will take.

Mental operations take time. It takes time to recall information from memory or to make a decision. It is possible to make reasonable quantitative estimates of the times involved, at least for fairly simple operations, and it is possible to break up complex operations into simple parts for purposes of estimation.

People can perform some mental and physical operations in parallel. People can operate a keyboard while talking or listening, for example. Quantitative models can be used here: Gray, John, and Atwood (1992) made a very accurate analysis of the parallelism in the use of a telephone operator's workstation, for example.

People get faster the more often they perform a task. The Power Law of Practice (Newell and Rosenbloom, 1981) states that the more frequently a task is performed, the quicker it will be accomplished. The rate of improvement is rapid at first, dropping off as the task becomes performed more often.

Novice users may perform tasks differently than expert users. There are individual differences in the way that knowledge is mentally represented between novices and experts. Such knowledge is a direct reflection of the way that way that a task is both understood and organized by people at different levels of expertise (Egan, 1988).

It takes time to learn things well. If only a little time is available for learning, only a little bit of information can be learned well enough to be remembered even a short time later. It is possible to make quantitative estimates of the time required to learn a skill to a reasonable criterion, based on a decomposition of the skill into small parts (Kieras and Polson, 1985).

Prior knowledge can be beneficial. Information related to prior knowledge is easier to learn and less likely to be forgotten than completely new information.

Recognition is easier than recall. Under usual conditions it is easier to recognize something when you see it or hear it than to recall it. This was one of the basic design principles of the Star design (Smith, et al., 1982; Johnson, et al., 1989) which triggered the decisive shift toward graphical interfaces, in which interaction is dominated by recognizing and selecting depictions of objects and actions rather than by recalling and typing names and commands.

People forget things, especially if they have not had ample time to rehearse the material, and if the material is not meaningfully related to things they already know. It is hard to keep arbitrary information in mind for even a short time while performing a task. So for example remembering and following advice from a help screen will be difficult; using the advice will be easy only if it can be kept visible while it is used.

Behavior is often guided by goals. People choose actions that they believe will accomplish their goals. If an action appears to lead away from a goal, they will not select it. A strategy called "label following" is often seen: people pick actions whose labels are obviously related to their goals (Engelbeck, 1986). For example, if people have the goal of archiving a file they will readily choose an action labelled *archive* and less readily one labelled *disk maintenance.*

Alternative methods can cause problems. Problems in which many alternative methods seem plausible are harder than problems for which only a few alternatives seem plausible.

People try to assess progress. If they do not seem to be making progress towards a goal people may abandon a task, seek to undo earlier actions, or find an alternative method. So enabling them to make an accurate assessment is important.

Other summaries of much of this material, and more, can be found in Card, Moran, and Newell's seminal book *The Psychology of Human-Computer Interaction* (1983) and in Olson and Olson (1990).

This list is far from complete. For instance, we haven't included facts about interference, stimulus-response compatibility, and other useful ideas. But we think it will serve to indicate the sort of thing psychological theory might supply to designers.

How might system designers become aware of and apply this knowledge in their work?

We consider some possible answers to this question in a hypothetical dialog with a software developer.

This is all common sense, so there's no problem. This just isn't true. Some of the subtle points, such as that the same color patch will look very different when presented against different backgrounds, are known by very few people. Beyond these, even some of the "obvious" points were not reflected in design until fairly recently, and are probably not widely known even now. Take the superiority of recognition over recall: this entered design practice (and as we noted quickly became dominant) only circa 1980 (Smith, et al., 1982). If it were common sense we would have seen designs reflecting this principle from the earliest days.

OK, this isn't all common sense but designers all know it by now. We doubt it. Our experience in discussing this material with designers is that few of them have encountered many of the concepts here, such as the role of goals in guiding behavior.

Well, they don't really know these things, but they act as if they do: current design practice already reflects these insights, so designers that follow established style are doing OK. There is much truth in this claim. It is true that designers who adapt the features of successful applications are able to produce designs which on the average are far superior to those prevailing several years ago. But one still sees many design flaws resulting from conflict with these ideas. Sometimes this happens because designers do not follow a successful model, sometimes because the model they are using has to be extended or

adapted, and sometimes because some of the principles can't be applied without considerable analysis.

Consider the importance of goals. Designs often fail because they (implicitly) assume that users have particular goals that they don't in fact have. For example, if one wants to forward calls on our university phones one must first adopt the goal of cancelling forwarding. This is a bad flaw, because nobody brings this goal with them until they learn they need it by painful experience. But a flaw like this, or the avoidance of it, isn't visible on the surface of an interface. One must examine the action sequences required for various tasks carefully to spot it.

I still don't think designers need to know any more about these things than they do now. They can fix these flaws by doing user testing of their designs, or by getting more user input during design. These are both good things to do. We would never argue that ideas from cognitive theory can be more than a useful supplement to these approaches. But we do argue for that much: relying only on testing and user interaction is a slow way home. If you start testing with a design with serious flaws testing takes a long time to work them all out. This is true partly because problems can mask each other in testing: you may not be able to get data on problem B before you have fixed problem A, if problem A throws users off the track before they get to B.

You've sold me on the need to do something about getting these ideas into practice. I'll adopt a set of design guidelines right away. This approach will work for some of these principles but not for others. Consider the problem with our telephones that we discussed earlier. There's no simple guideline for avoiding this problem that can be applied to an interface just by looking at it. Some process that looks at the steps required to perform tasks with the interface, and critiques them, is needed.

I can see that this problem is harder than I thought. Designers should build complete process models of what is needed to use their systems. Then they'll smoke out any problems that exist, no matter how far below the surface they are. This is a good idea. In some situations it can be done and

provides very useful results. An excellent example is the work of Gray, John, and Atwood (1992) in modelling the interface for a telephone operators' workstation, in which they were able to identify subtle but extremely important flaws in a new design.

But there are two common problems with building models. First, only some of the principles above are understood in enough detail to support accurate modelling. What is well understood tends to be the quantitative aspects of the principles: how long it takes to make a movement, for example. And even for these quantitative aspects only the behavior of highly skilled or highly constrained users can be modelled accurately.

A second problem is that building a complete process model is a major undertaking. It can approach or even exceed in effort the task of building the interface itself. Worse, before beginning to build a model one needs to invest considerable time in mastering the modelling framework and any tools that may be available to support it. In some situations, like that of Gray, John, and Atwood (1992), where small changes in interface performance are worth a great deal, this investment may pay back handsomely. It may be that a sober analysis of the economics of usability, if we knew enough to make such a analysis, would show that the investment would be worthwhile in many common design situations. But it is clear that in most current design environments the payoff won't be recognized, even if it's there, and the investment just won't be made.

Let's forget about this problem. It's insoluble. Not so fast. We can envision a method for analyzing designs that does some of what building a model would do but at much less cost. For instance, in the Cognitive Walkthrough method, discussed in Wharton, Rieman, Lewis, & Polson (Chapter ----), a design evaluator builds and applies a simple process model in real time while examining an interface. Any problems encountered in doing this are noted as the evaluation proceeds. We consider this approach more fully in the next section.

You're right, I gave up too soon. And I can see other solutions, too: how about hiring some cognitive psychologists and getting them to look over my

designs? They already know the theory. This isn't a bad idea. But there are some things to worry about. One is finding cognitive psychologists who know enough about user interfaces to be able to apply the principles. Another is the "kibitzer problem": it can cause trouble in your organization to have people whose only job is to criticize somebody else's work, or even to make supposedly helpful and constructive suggestions about somebody else's work. Besides, it's expensive enough to have designers, let alone kibitzers.

I'll just have the psychologists do the design. Again, not a bad idea. But the supply of psychologists who can do the design is even smaller than for kibitzers. For example, in your organization they probably have to understand the X Window System to get started.

How about teaching some cognitive theory to computer people so they can do a good job on interface design? Another good idea. We should do that, but we also need to do something in the meantime.

Can theory be reflected in inspection methods?

Having considered some general approaches to involving psychological theory in design, let us zero in on the specific idea of using inspection methods, including the Cognitive Walkthrough, as a way to bring theory into design. We'll do this by continuing our dialog with the hypothetical software developer, but now presuming that he or she has attended a workshop at the CHI conference on inspection methods, or has read the introductory chapters of this book.

I'm really taken with the idea of inspection methods. They are cost-effective and fast. And I can get my existing design personnel to use them. Actually the data are mixed on both of these points. Heuristic Evaluations are fast (Molich and Nielsen, 1990; Nielsen and Molich, 1990), but Cognitive Walkthroughs (Lewis, Polson, Wharton, and Rieman, 1990; Polson, Lewis, Rieman, and Wharton, 1992), at least using the original procedures, are not. And there's evidence that usability expertise helps when applying these methods (Jeffries, Miller, Wharton, and Uyeda, 1991; Wharton, Bradford, Jeffries, and Franzke, 1992; Nielsen,

1992). It remains unclear whether these methods will be competitive with other approaches in an organization like yours with few experienced usability people. Anyway, why do you think other people in your organization will share your enthusiasm and be willing to put time into using these methods? That doesn't always happen in organizations we know about.

It seems to me that a method like the Cognitive Walkthrough will definitely be a step in the right direction for us, anyway, since we'll get the benefit of some psychological knowledge we now lack. We hope you are right. But it's unclear whether all of the ideas we hope to import from theory can really be brought in by an inspection method. Consider the ideas about perception and attention in the our inventory of useful ideas at the beginning of this paper. Both the Cognitive Walkthrough and Heuristic Evaluation allow for judgements about perception or attention to be made, but they do not help inexperienced people to make them. They don't, for example, inform people about facts about perception or attention that they may not already know.

Couldn't these methods be extended to provide more coverage? Perhaps. But can we do this while keeping the methods acceptably simple and fast, and without requiring extensive pre-training in the concepts involved? The beauty of the Heuristic Evaluation approach is its simplicity, and we've seen that the original Cognitive Walkthrough was too complex and slow. Extending these methods won't be easy.

There may be problems, I agree, but this seems to be the most promising approach. Cognitive modelling is never going to be practical in my organization. You may be right. Remember, though, that for the right problem cognitive modelling is already very effective: if you are dealing with highly-skilled users on an application with very high sensitivity to user performance, modelling will pay off. Beyond that, it may be possible to build cognitive models for *classes* of interfaces, as Peter Polson (personal communication, November, 1992) has suggested. (cf. Heuristic Estimation which estimates training times for classes of interfaces (Polson and Olson, 1992; Polson, Rieman, Wharton, and Olson, 1992).) Then you and your people could pick

up a generic model, do a little tailoring, and have a good model of your design. Or maybe some design questions could be settled by the model at the level of the whole class of interfaces, so that you wouldn't even need to tailor the model. To some extent this is already happening: there are good theoretical analyses of different menu selection mechanisms, for example, that could be applied by designers without requiring new modelling to be done.

But if this kind of thing is done couldn't the results be incorporated in inspection methods, in the form of questions to ask about the features of an interface? That's a good idea. It could lead to inspection methods specialized to interfaces of a given class. The first versions of the Cognitive Walkthrough suffered from not doing this: people evaluating a graphical user interface had to answer questions that were unlikely to be relevant except for telephone interfaces. Targeting inspection methods on particular classes of interfaces could be a way of extending inspection methods without making them too complicated, since the method that would be used in a given situation would include only a subset of all the possibly relevant issues. But then there's the problem of organizing a lot of different inspection methods for lots of specific settings.

My organization could do that for the kinds of interfaces we build. Maybe so.

Another advantage of inspection methods for us is that we can use them very early in design. We can't do that with user testing. That's another point that needs more exploration. You can do user testing very early if you use paper and pencil or quick and dirty screen mockups, but we don't have any comparison of inspection methods with this kind of early test.

A problem we are concerned about is getting fixes to the problems an inspection method turns up. In principle that is a big advantage to having a theoretical grounding for the inspection. The theory ought to tell you *why* something is a problem, not just that it *is* a problem. And knowing why something is a problem is a big step toward fixing it. For example, knowing that a color patch looks wrong *because* its

appearance is influenced by the surrounding context is much more helpful than just knowing that it looks wrong. But we don't have data on how well this works in practice.

Conclusion

We conclude the above hypothetical discussion.

We are going to try inspection methods in my organization. We'll try to push the methods to connect more with psychological theory, by using methods that offer some "why" information as part of the evaluation. We'll share our contributions and also work with the research community, and our industry colleagues, to keep up the flow of ideas and data on these methods and how they compare with others. Sounds good to us.

Acknowledgements

We thank all participants in the CHI'92 Usability Inspection Methods Workshop for their comments and helpful discussions. We also thank the following people for their helpful discussions and detailed comments on earlier drafts of this paper: Randolph Bias, Louis Blatt, Barbara Diekmann, George Engelbeck, Peter Polson, and John Rieman.

References

Card, S.K., Moran, T.P., and Newell, A. (1983) *The psychology of human-computer interaction*. Hillsdale, NJ: Erlbaum.

Egan, D. E. (1988) "Individual differences in human-computer interaction." *Handbook of Human-Computer Interaction*. Chapter 18. Helander, M. (editor). Elsevier Science Publishers B.V. (North-Holland), New York. 543-568.

Engelbeck, G.E. (1986) "Exceptions to generalizations: Implications for formal models of human-computer interaction." Unpublished Masters Thesis, Department of Psychology, University of Colorado, Boulder, CO.

Fitts, P. M., and Posner, M. I. (1967) *Human performance.* Basic Concepts in Psychology Series. Belmont, CA: Wadsworth Publishing Company, Inc. Brooks/Cole Publishing Company.

Gray, W. D., John, B. E., and Atwood, M. E. (1992) "The pr'ecis of project Ernestine or an overview of a validation of GOMS." In *Proceedings of CHI, 1992* (Monterey, CA, May 3- May 7, 1992) ACM, New York, 307-312.

Jeffries, R., Miller, J. R., Wharton, C., and Uyeda, K. M. (1991) "User interface evaluation in the real world: A comparison of four techniques." In *Proceedings of CHI, 1991* (New Orleans, LA, April 27 - May 2, 1992) ACM, New York. 119-124.

Johnson, J., Roberts, T. L., Verplank, W., Smith, D. C., Irby, C. H., Beard, M., and Mackey, K. (1989) "The Xerox Star: A retrospective." *IEEE Computer,* 22, 9, September, 11-29.

Kieras, D. E. and Polson, P. G. (1985) "An approach to the formal analysis of user complexity." *International Journal of Man-Machine Studies,* 22, 365-394.

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

Molich, R. and Nielsen, J. (1990) "Improving a human-computer dialogue: What designers know about traditional interface design." *Communications of the ACM,* 33, 338-348.

Newell, A. and Rosenbloom, P. S. (1981) "Mechanisms of skill acquisition and the law of practice." In J. R. Anderson (Ed.). *Cognitive Skills and Their Acquisition.* Hillsdale, NJ: Lawrence Erlbaum Associates, Publishers. 1-55.

Nielsen, J. (1992) "Finding usability problems through heuristic evaluation." In *Proceedings of CHI, 1992* (Monterey, CA, May 3- May 7, 1992) ACM, New York. 373-380.

Nielsen, J. and Molich, R. (1990) "Heuristic evaluation of user interfaces." In *Proceedings of CHI, 1990* (Seattle, WA, April 1-5, 1990) ACM, New York. 249 - 256.

Olson, J. R. and Olson, G. M. (1990) "The growth of cognitive modeling in human-computer interaction since GOMS." *Human-computer interaction*, 5, 2 & 3, 221-265.

Polson, P., Lewis, C., Rieman, J., and Wharton, C. (1992) "Cognitive walkthroughs: A method for theory-based evaluation of user interfaces." *International Journal of Man-Machine Studies*, 36, 741-773.

Polson, P. G. and Olson, J. S. (1992). "An approximate method for estimating total learning time: An example of a usability inspection method derived from cost/benefit considerations." Paper presented at *Human-Computer Interaction Consortium Annual Winter Conference*, Pittsburgh, PA, January 28, 1992.

Polson, P., Rieman, J., Wharton, C., and Olson, J. (1992) "Usability inspection methods: Rationale and examples." In *Proceedings of 8th Symposium on Human Interface* (Kawasaki, Japan, October 21-23, 1992). 377-384.

Smith, D. C., Irby, C., Kimball, R., Verplank, W., and Harslem, E. "Designing the Star user interface." *Byte*, 7, 4, April 1982, 242-282.

Tullis, T. S. (1988) "Screen design." *Handbook of Human-Computer Interaction*. Chapter 18. Helander, M. (editor). Elsevier Science Publishers B.V. (North-Holland), New York. 377-411.

Wharton, C., Bradford, J., Jeffries, R., and Franzke, M. (1992) "Applying Cognitive Walkthroughs to more complex user interfaces: Experiences, issues, and recommendations." In *Proceedings of CHI, 1992* (Monterey, CA, May 3- May 7, 1992) ACM, New York. 381-388.

[THIS PAGE INTENTIONALLY LEFT BLANK]