*Desi*gning for *L*earnability (*DesiL*): An Engine for Informing the Design of Easy-to-Learn Microcomputer Interfaces

Donald J. Winiecki and Terence C. Ahern

College of Education
Texas Tech University

Abstract

Modern software design is focused at maximizing the usability of its products [1, 2, 3]. Once the domain of technical experts and "guru's, we are now told that computers are "for the rest of us"[4].

However, usable software designs still mandate some specialized training in their use [3]. As computers become more prevalent in the workplace, individuals must both maintain their current level of productivity while learning new methods of performing tasks they had previously done either manually or using a different computer software application. Software that minimizes learning time could be of substantial value to deadline pressured workers who must take advantage of advancing technologies but do not have access to special training.

DesiL (*Desi*gning for *L*earnability) is a methodology for informing the design of microcomputer software, that may permit a designer to produce both usable and more easily learned software applications. This methodology is non-deterministic in that it does not make decisions for a designer, but rather informs a designer's options toward creating software that may be more easily learned by the end user.

Keywords — Learnability, training & education, human-computer interface design, human-computer interaction, ethnomethodology, qualitative research.

1. Introduction

The capabilities and flexibility of modern computers is an awesome testimony to the ingenuity of 20th century technology. Television and print advertising declares that mastery of computers will bequeath "David over Goliath" power and an edge over business competitors or fellow students. Once learned, computers provide "one button" (or one mouse-click) access to unlimited skills and knowledge. Once the domain of technical experts and "guru's," we are now told that computers are "for the rest of us" [4].

In the process of making a computer for the rest of us, engineers, scientists, and psychologists have focused much study toward making them more "usable" [2, 3, 5, 6]. However, despite their "easy to use" status, the task of becoming a more efficient computer user has remained a difficult and sometimes threatening task [7]. Thanks to current practice in HCI, *being* a computer user has become an easier task, but the process of *becoming* a computer user, or a more efficient computer user is still an awesome task.

Although publishers of computer software are aware of the need to make their products easier to learn and use [2, 6, 8] current strategies of software design are not always successful in producing an "easy to learn" software application [9]. Evidence of this is seen in the many different "helping" mechanisms computer software publishers have invented including printed manuals, on-line "help" glossaries (software versions of printed manuals), hypertext help systems [6] and on-line demonstrations [10]. Additional evidence is supplied by the successful third party and "self help" publishing industry offering literally hundreds of book, videotape and software titles to assist a computer user in learning to use computer software. However, even the existence of these aids cannot guarantee that a computer user will correctly interpret the instructions as intended by their author [11].

2. Software Design Methodology

DesiL is based on the theoretically and empirically grounded notion that learning is most likely to occur when material to be learned is semantically attached to something already known [12, 13, 14] and that incorporates several parallel "modes of representation" tuned to the intended learner 12]. For this reason, the use of visual and linguistic analogies and metaphors are commonplace in the design of modern software interfaces [5, 8, 14, 15, 16, 17].

Design methodologies exist that bring potential users of software into the software designers world to participate in the iterative process of designing interfaces [18, 19, 20]. These methods permit the designer to solicit actual users for their tacit assessment of the usability of a professionally designed interface

However, despite evidence of these methods advantages to creating "usable" software, they remain largely under the control of specially trained "designers" who may unknowingly limit their design efforts to the "inner environment" [21] of the task domain, and ignore the "outer environment" [21] represented by the user's perception of the task and their perception and interpretation of the context in which it is performed. Because such designs were created out of their context of use, they may be more difficult to learn in the milieu and perhaps chaotic context of actual use.

Newby [22] and Cool's [23] research implies that other factors are necessary for the creation of "learnable" software. These factors transcend the notion of learning promoted by cognitive-science and suggest that a more learnable computer interface should accommodate a computer user's current knowledge, experiences, and perceptions and interpretations of a computer based task and its context of use. Cool [23] delineates a three part design focus that incorporates and extends both "inner" and "outer" [21] factors of the computer user, their environment and the task to be performed, including contextual and linguistic dimensions delineated by Bruner [12] and Vygotsky [24]. This research implies that a computer user's current knowledge, experiences, and perceptions and interpretations of a computer based task and its context of use may be discovered and delineated using ethnomethodological research techniques, and that these findings may be used to inform the design of software that is not only easy to use but also easy to learn.

2.1. DesiL Methodology

DesiL is based on sociologically grounded ethnomethodological inquiry in which a software designer observes and conducts interviews with members of a population in situ to determine what *methods* they use to perform their daily duties. Ethnomethodology has as its purpose to "analyze everyday activities as members' methods for making those same activities visibly-rational-and-reportable-for-all-practical-purposes" [25].

Traditionally, ethnomethodology has confined itself to the understanding of singular events with the belief that, because the circumstances surrounding events are fundamentally non-repeatable, so the analysis of these events are non-generalizable in the classic sense. However, it is becoming increasingly acceptable to perform ethnographically grounded studies of work sites with the intent of informing the design of human-computer interactions [26, 27, 28, 29] DesiL continues this effort but focuses on using the ethnomethodological data for improving the "learnability" of a microcomputer interface.

In particular this study was performed through the onsite observation of workplace activities and guided and open ended interviews similar in format to the anthropologically grounded "elicitation interview" [30, 31] and the "20 statements test" [32].

Field notes and interview transcripts were analyzed using the constant comparative analysis technique [33]. These analyses were used to create "contact summaries" [34] of the observations and transcripts and as guides in coding the data to describe the participant's perception and interpretation of specific human-computer interaction tasks and how these were effected by the context of their work.

These codes were used by the software designer during initial design phases, to better “fit” the evolving design to the participant’s understanding of the task domain and context of work.

3. Research Venue and Population

This study was conducted with four office workers in an administrative office located in a Southwestern, USA university. This office has recently been switched from using stand-alone microcomputers, paper-based files and remote access to the university's mainframe computer to being connected to a new Local-Area-Network (LAN).

This office is comprised of five staff members and their supervisor, and is dedicated to administrating the college's undergraduate, transfer student and post baccalaureate enrollment into teacher certification programs, college of education student graduations and to assist in the data processing necessary for assigning student teachers to local schools for fieldwork experience. The transition of data processing tasks to a LAN-based database management system introduces a small but "authentic" problem for these workers, the transfer of their knowledge and understanding of an "old way" of doing their jobs, to a "new way" of performing the same jobs with different tools.

3.1. The Participants

With the exception of the office's supervisor, all staff members of this office are adult women ranging in age from 28 to over 50. The research participant's formal instruction in the use of microcomputers has been limited to "short courses" in the use of applications software (word processor, spreadsheet, ...) at the university's computer center. Currently, each participant uses several common application software packages in her individual job duties.

Each of the participants in this study has a distinct "specialization" in the office. Although each is capable of filling in for one another to a limited extent, for the purposes of this study, each person's individual responsibilities do not overlap. As I have observed, the many and varied responsibilities of this office can occasionally result in scenes similar to hospital "triage" as students come and go, with their own individual questions and crises.

4. DesiL's Contribution to Software Design

As indicated, DesiL is a non-deterministic design methodology. This means that interface designers are free to analyze and interpret the field based data in terms of their own personal design philosophy and in conjunction with accepted industry and psychologically grounded standards for HCI design [6, 8, 16, 19, 35]. However, from the ethnomethodological inquiry, an improved understanding of the context of human-computer interaction tasks may be gained, and used by the designer to more fully "tune" his or her HCI design for the target population.

In particular, during our conductance of this research, it was noted that the qualitative data collection and analyses greatly facilitated the conductance of learnability focused "cognitive walkthroughs" [2, 36, 37, 38] of interim interface designs before they were ever committed to code. It is in this phase of the design process that we see the greatest benefit for the DesiL design methodology. Additionally, the effectiveness of such a priori learnability checks may improve as a designer becomes more accustomed to the use of DesiL.

An additional benefit of the ethnographically based DesiL methodology, over other methods for ensuring software learnability [2, 36, 37, 38] is that it does not require any special training for the designer other than an "eye" for the [33] and a diligence in data analyses!

4.1. Testing of Interfaces Designed with DesiL

As of this writing (6/95) final analyses of the HCI design has not been completed. A combination of both qualitative and quantitative data collection and analysis methods is used to assess the learnability of prototype interfaces.

Verbal protocol [39, 40, 41] and elicitation interviews will be conducted with each participant at two day intervals following installation of the software. Resulting field notes and interview transcripts are analyzed for the existence of "awareness contexts" [42] and "flow" states [43]. Awareness contexts delineate a taxonomy that indicates a person's level of understanding of their partner in an interaction, and "flow" is a psychological construct indicating a merging of a person's skill and knowledge of a situation.

Additionally, the software has been designed to unobtrusively collect each individual user's software command inputs. Tracking discrete command inputs during each session with the software results in an "action transcript" of each human-computer interaction. Stochastic analyses performed on these "action transcripts" permit the identification of patterns of interaction as the participants use the software. In terms of this research, the existence of a rapid decrease in entropy over time, and a statistically significant regularity [44] in these "action transcripts" is considered as evidence of an easily learned interface design.

5. Contribution to the Discipline

The discipline of computer software design has two purposes: (a) to codify a set of instructions to guide a computer in the performance of some task and (b) to give a computer user relevant control over that task. By better understanding how persons bring their existing contextual knowledge and skills into an HCI and use them in completing their duties, this research may inform software designers on how to better design and develop their products to accommodate user's perceptions and interpretations of that software. The resulting products may be easier to learn.

Additionally, the use of both qualitative and quantitative data to mutually support each other, may further inform the practice of software design for learnability by adding a unique methodology to the researcher's set of tools for assessing software designs.

References

1. Nielsen, J. & Levy, J. (1994). Measuring usability: Preference vs. performance. Communications of the ACM, 37 (4), 66-75.

2. Nielsen, J. & Mack, R. L. (1994). Usability inspection methods. Brisbane: John Wiley & Sons, Inc.

3. Price, R. V. & Winiecki, D. J. (1995). Trends in undergraduate microcomputer education. forthcoming.

4. Rubin, J. (1994). Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests. New York: John Wiley & Son's, Inc.

5. Raymond, E. S. (1994). The New Hacker's Dictionary (2nd. ed.). Cambridge, MA: MIT Press.

6. Laurel, B. (1990). The art of human-computer interface design. New York: Addison-Wesley.

7. Shneiderman, B. (1992) Designing the user interface: Strategies of effective human-computer interaction. (2nd. Ed.). New York: Addison-Wesley Publishing Company.

8. Price, R. V. & Winiecki, D. J. Student characteristics and trends in computer education. (forthcoming).

9. Apple Computer, Inc. (1987). Human interface guidelines: The apple desktop interface. Cupertino, CA: Author.

10. Gery, G. (1991). Electronic performance support systems: How and why to remake the workplace through the strategic application of technology. Boston, MA: Weingarten Publications.

11. Microsoft Corporation. (1994). Microsoft® Word® for Windows™, version 6.0a. [computer software]. Redmond, WA: Author.

12. Lewis, C. & Mack, R. (1982). Learning to use a text processing system: Evidence from 'thinking aloud' protocols. Proceedings of Human Factors in Computer Systems (pp. 387-392). Gaithersberg, MD: ACM Press.

13. Bruner, J. (1966). Towards a theory of instruction. Cambridge, MA: Belknap Press.

14. Lin, S. L. (1993). The effects of elaboration and placement of analogies on student learning and attitude toward BASIC programming using computer-assisted-instruction. Unpublished doctoral dissertation, Texas Tech University, College of Education, Lubbock.

15. MacLean, A, Bellotti, V, Young, R & Moran, T. (1991). Reaching through analogy, A design rationale perspective on roles of analogy. Proceedings of the ACM CHI'91 Conference (pp. 167-172). ACM Press: Author.

16. Horton, W. (1994). The icon book: Visual symbols for computer systems and documentation. Brisbane: Wiley.

17. Microsoft Corporation. (1992). The Windows™ interface, an application design guide. Redmond, WA: Microsoft Press.

18. Togzannini, B. (19XX). Tog on interface. New York: Addison-Wesley.

19. Gould, J. D., Boies, S. J. & Lewis, C. (1991). Making usable, useful, productivity-enhancing computer applications. Communications of the ACM, 34 (1), 74-85.

20. Hix, D. & Hartson, R. (1993). Developing user interfaces: Ensuring usability through product and process. New York: Wiley & Sons.

21. Rettig, M. (1994). Prototyping for tiny fingers. Communications of the ACM, 37 (4), 21-27.

22. Simon, H. (1984). The Sciences of the Artificial. Boston, MA: MIT Press.

23. Newby, G. (1989). User models of information retrieval: Applying knowledge about human communication to computer interface design. Proceedings of the 52nd Annual Meeting of ASIS (pp. 71-74). Medford, NJ: Learned Information, Inc.

24. Cool, C. (1993). Information retrieval as symbolic interaction: Examples from humanities scholars. Proceedings of the 56th Annual Meeting of ASIS (pp. 274-277). White Plains, NY.: Knowledge Industry Publications.

25. Vygotsky, L. S. (1986). Thought and language. A. Kozulin (Ed.). Cambridge, MA: The MIT Press.

26. Garfinkel, H. (1967). Studies in ethnomethodology. Englewood Cliffs, NJ: Prentice-Hall.

27. Hughes, J., King, V., Rodden, T. & Andersen, H. (1994). Moving out of the control room: Ethnography in system design. In R. Furuta & C. Neuwirth (Eds.), Proceedings of the Conference on Computer Supported Cooperative Work, CSCW '94 (pp. 429-439). ACM Press: Author.

28. Hughes, J., King, V., Rodden, T. & Andersen, H. (1995). The role of ethnography in interactive systems design. Interactions: New Visions of Human-Computer Interaction, 11 (2). 56-65.

29. Shapiro, D. (1994). The limits of ethnography: Combining social sciences for CSCW. In R. Furuta & C. Neuwirth (Eds.), Proceedings of the Conference on Computer Supported Cooperative Work, CSCW '94 (pp. 417-428). ACM Press: Author.

30. Shapiro, D. Ferrets in a sack? Ethnographic studies and task analysis in CSCW. To appear in D. Shapiro, M. Tauber & R. Traunmuller (Eds.) The Design of Computer Supported Cooperative Work and Groupware Systems. Amsterdam: Elsevier Science.

31. Black, M. B. (1969). Eliciting folk taxonomy in Ojibwa. In S. A. Tyler (Ed.), Cognitive Anthropology. (pp. 165-189). New York: Holt, Rinehart & Winston, Inc.

32. Black, M. B. & Metzger, D. (1969). Ethnographic description and the study of law. In S. A. Tyler (Ed.). Cognitive Anthropology. (pp. 137-165). New York: Holt, Rinehart & Winston, Inc.

33. Kuhn, M. H. & McPartland, T. S. (1972). An empirical investigation of self-attitudes, in Symbolic interaction: A reader in social psychology. (J. G. Manis & B. N. Meltzer, eds.) Boston, MA.: Allyn and Bacon, Inc.

34. Lincoln, Y. S. & Guba, E. G. (1985). Naturalistic Inquiry. Newbury Park, CA: Sage Publications.

35. Miles, B. B. & Huberman, A. M. (1994). Qualitative data analysis: An expanded sourcebook (2nd. ed.). Thousand Oaks, CA: Sage Publications.

36. Card, S., Moran, T. P. & Newell, A. (1983). The psychology of human-computer interaction. Hillsdale, NJ: Lawrence Erlbaum, Assoc., Inc.

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

38. Wharton, C. (1992). Cognitive walkthroughs: Instructions, forms, and examples. (Technical Report CU-ICS-92-17). University of Colorado: Boulder, CO: Institute of Cognitive Science.

39. Lewis, C., Polson, P., Wharton, C. & Rieman, J. (1990). Testing a walkthrough methodology for theory-based design of walk-up-and-use interfaces. Proceedings of the ACM CHI'90 Conference (pp. 235-242). ACM Press: Author.

40. Bainbridge, L. (1979). Verbal reports as evidence of the process operator's knowledge. International Journal of Man-Machine Studies, 11 411-436.

41. Ericsson, K. A. & Simon, H. A. (1980). Verbal reports as data. Psychological Review, 87 (3). 215-251.

42. Lewis, C. (1982). Using the "thinking aloud" method in cognitive interface design. Research report RC9265, IBM T. J. Watson Research Center. Yorktown Heights, NY: Author.

43. Glaser, B. & Strauss, A. (1972). Awareness contexts & social interaction. In J. Manis & B. Meltzer (Eds.). Symbolic interaction: A reader in social psychology (2nd. ed.) (pp. 429-444). Boston, MA: Allyn & Bacon, Inc.

44. Csikszentmihalyi, M. (1988). Introduction. In Optimal experience: Psychological studies of flow in consciousness. M. Csikszentmihalyi & I. S. Csikszentmihalyi (Eds.). New York, NY: Cambridge University Press. (pp. 3-15).

45. LeFever, R. D. (1990). Markov: A BASIC program for numerical analysis of sequential data on the microcomputer. Computers & Geosciences, 16 (2), 141-152.

Authors' Addresses

Donald J. Winiecki and Terence C. Ahern: Texas Tech University, College of Education, Box 41071, Lubbock, TX 79409-1071. {ibdon, DWTCA} @ttacs.ttu.edu.