**Advances in Human Factors/Ergonomics**

Series Editor: Gavriel Salvendy, Purdue University, West Lafayette, IN 47907, U.S.A.

---

| | |
|---|---|
| Vol. 1. | Human–Computer Interaction (G. Salvendy, Editor) |
| Vol. 2. | Human–Computer Dialogue Design (R.W. Ehrich and R.C. Williges, Editors) |
| Vol. 3. | Expertise Transfer for Expert System Design (J.H. Boose) |
| Vol. 4. | Engineering Physiology: Physiologic Bases of Human Factors/Ergonomics (K.H.E. Kroemer, H.J. Kroemer and K.E. Kroemer-Elbert) |
| Vol. 5. | Human Factors Testing and Evaluation (D. Meister) |
| Vol. 6. | Applications of Fuzzy Set Theory in Human Factors (W. Karwowski and A. Mital, Editors) |
| Vol. 7. | Human Reliability: Analysis, Prediction, and Prevention of Human Errors (K.S. Park) |
| Vol. 8. | Human Aspects of Occupational Vibration (D.E. Wasserman) |
| Vol. 9. | Human Factors Research: Methods and Applications for Architects and Interior Designers (J.E. Harrigan) |
| Vol. 10A. | Social, Ergonomic and Stress Aspects of Work with Computers (G. Salvendy, S.L. Sauter and J.J. Hurrell, Jr., Editors) |
| Vol. 10B. | Cognitive Engineering in the Design of Human–Computer Interaction and Expert Systems (G. Salvendy, Editor) |

Advances in Human Factors/Ergonomics, 10B

# Cognitive Engineering in the Design of Human-Computer Interaction and Expert Systems

Proceedings of the Second International Conference on Human–Computer Interaction, Honolulu, Hawaii, August 10–14, 1987, Volume II

Edited by

**Gavriel Salvendy**
Purdue University, West Lafayette, IN 47907, U.S.A.

ELSEVIER

Amsterdam – Oxford – New York – Tokyo 1987

ELSEVIER SCIENCE PUBLISHERS B.V.
Sara Burgerhartstraat 25
P.O. Box 211, 1000 AE Amsterdam, The Netherlands

*Distributors for the United States and Canada:*

ELSEVIER SCIENCE PUBLISHING COMPANY INC.
52, Vanderbilt Avenue
New York, NY 10017, U.S.A.



ISBN 0-444-42847-X (Vol. 10A)
ISBN 0-444-42848-8 (Vol. 10B)
ISBN 0-444-42849-6 (Set)
ISBN 0-444-42396-6 (Series)



Printed in The Netherlands

# CONTENTS


**I. Theories of Interface Design** 1

The Human-Computer Interface: The Past 35 Years and the Next 35 Years 3
Frederick A. Muckler

What We Know and What We Should Know About Human-Computer Interaction: Strategies for Research and Development 13
Gavriel Salvendy

Adapting Human-Computer Interfaces for Inexperienced Users 21
Robert C. Williges

Individual Differences in Human-Computer Interaction: Concepts and Research Findings 29
Eberhard Ulich

HUFIT (Human Factors in Information Technology) 37
K.-P. Faehnrich, J. Ziegler and D. Davies

The Evaluation of Learning Requirements of IT Systems 45
Tom Boesser

Towards a Truly High-Level and Integrated Human-Computer Interface 53
D. W. Paul and H. R. Wiehle

Multiple Attribute Decision Making and On-Line Information Systems 61
Mark P. O'Shaughnessy, Semra Coskuntuna, Ellen Kantro

Analyzing Human-Computer Dialogue Interfaces 67
Beverly H. Williges

Cognitive Compatibility as a Central Issue in Human-Computer Interaction: Theoretical Framework and Empirical Findings 75
Norbert A. Streitz

Towards Portability of Dialog Systems 83
Jan Stelovsky and Paolo Conti

A Blackboard Architecture to Realize Adaptive Human-Computer Interfaces and Application Systems 89
H. Balzert

Making Computers More Useful and More Usable 97
Gerhard Fischer

Four Human Factors Problems for System Operators as Dependence on Automation Increases 105
Alan D. Swain

A Semantic Paradigm to Solve Natural Language Database Queries 113
J. Gonzalez-Sustaeta and Carlos Martinez

The Role of Natural Language Processing in Human-Computer Interaction 121
V. Raskin

**II. Methodologies of Interface Design** 129

A Computer Method for Specifying Colors By Means of Color Naming 131
S. Tominaga

Methodology for Designing a Normalized User Interface 139
Keith Gregory

The Design of Usable IT Products: The ESPRIT/HUFIT Approach 147
K. D. Eason, C. W. Olphert, F. Novara, N. Bertaggia and N. Allamanno

SQL-U: A Casual-User Oriented Interface for a Relational DBMS on Micro-Computers — 155
G. Taladoire, S. Miranda, N. Le Thanh, and L. Lakhal

Experiments on Human-Computer Interaction Evaluation — 167
Noriko Yamagishi and Motoei Azuma

A Serial-Parallel Integrated Information-Processing Model for Complex Human Problem Solving — 175
Yuichiro Anzai, Hirohiko Mori, Miou Ito, and Yoshio Hayashi

The Design and Implementation of a Data-Driven User Interface — 183
Fuyan Zhang, Shijie Cai, and Shu Wang

**III. Applications of Interface Design** — **191**

An Overview of Research on Electronic Journals — 193
B. Shackel

User Interface Design and Evaluation for an Electronic Encyclopedia — 207
Ben Shneiderman

Human Factors in Office Product Design: European Practice — 225
S. Hannigan and V. Herring

Cognition At Your Fingertips: A Cognitive Approach to the Design of Data Entry Devices — 233
Daniel Gopher

A User Interface Case Study of the Macintosh — 241
Jakob Nielsen

Realization of Flexible User-Interface System for Module-Based Simulation System of Nuclear Power Plant — 249
H. Yoshikawa, N. Mizutani, N. Nakaya, M. Ukon, and J. Wakabayashi

Human Decision Making in Computer-Based Scheduling Within a Flexible Manufacturing System: An Experimental Study — 257
N. Nakamura and G. Salvendy

Decision Aids and Risk Taking in Flexible Manufacturing Systems: A Simulation Study — 265
B. Zimolong

Computer Interfaces for the Humanities — 273
Mark H. Chignell

A Methodology for Modeling Multi-Position EV and IV Tasks for Space Station — 281
S. B. Sheppard

Designing Human Factors Design Aids for Designers — 289
A. J. Russell and Margaret D. Galer

A Deep-Reasoning Aid for Deep-Reasoning Fault Diagnosis — 297
Wan C. Yoon and John M. Hammer

**IV. Software Design** — **305**

Software-Ergonomics: History, State-of-the-Art and Important Trends — 307
H.-J. Bullinger, K.-P. Faehnrich, J. Ziegler

I Can't Tell What in the Code Implements What in the Specs — 317
E. Soloway

Software-Design With the Rapid Prototyping Approach: A Survey and Some Empirical Results — 329
C. G. Hoyos, H. Gstalter, V. Strube, and B. Zang

An Integrated Approach to Solving Visual Programming's Problems — 341
Ephraim P. Glinert, Jakob Gonczarowski, Craig D. Smith

Direct Manipulation and Procedural Reasoning — 349
David Owen

Prototyping a Dialogue Interface: A Case Study — 357
B. P. Chao

Rapid Prototyping as a Tool for User Centered Design — 365
Susan Harker

**V. Human Factors in Speech Technology and Telecommunications — 373**

Retrieval of Information from Complex Alphanumeric Displays: Screen Formatting Variables' Effects on Target Identification Time — 375
Carla J. Springer

Design for the Future: A User Interface for Telecommunications Operator Services — 383
Hans Bergman and Kevin o'Cahan

Evaluating Synthetic Speech Devices — 391
Raymond W. Bennett and Steven L. Greenspan

Multimode Interaction in a Telecommunications Testbed: The Case of Memory Dialing — 399
Robert W. Root and Ching-Hua Chow

Telephone Ordering System — 407
James W. Feeney

Mobilanguage — 415
K. G. Engelhardt and Roger Edwards

A "Hands-Off" Workstation — 423
Caroline Fu

Where Does Speech Technology Fit in Human-Computer Interaction? — 431
J. W. Glenn

Conversational Telecommunications Environments — 439
Christopher Schmandt

**VI. Design of Graphic Dialogues — 447**

Understanding Complex Software Systems Using GADD: A Tool for Graphical Animated Design and Debugging — 449
M. C. Moser

A Graphics Tool for Software Design Visualization — 457
Masaya Norifusa, Noriko Hagiwara, and Osamu Shigo

Design of a Graphic Dialogue Without Syntactic Constraints — 465
B. Pavard

Issues in Computer-Assisted Interpretation of Graphs and Quantitative Information — 473
J. A. Campbell and S. P. Ross

Enhancing a Traditional Typeface Design Environment Through the Use of Contemporary Computing Technology — 481
L. Ruggles

**VII. Knowledge Acquisition for Knowledge-Based Systems — 489**

Knowledge Acquisition Research: A Summary of the AAAI Knowledge Acquistion for Knowledge-Based Systems Workshop — 491
John H. Boose and Arthur T. Nagai

Knowledge Acquisition Based on Knowledge Role Understanding 499
Sandra Marcus

The Generic Task Toolset 507
Tom Bylander, B. Chandrasekaran, and John R. Josephson

Knowledge Acquisition Tools for Different Problem-Solving Paradigms: Research at Boeing Computer Services 515
Catherine M. Kitto

User Requirements Gathering Through Verbal Protocol Analysis 523
E. Lueke, P. D. Pagerey, and C. R. Brown

A System Simulation System to Support the Elicitation of Information Requirements by End-Users 529
Oscar Gutierrez

**VIII. Design, Evaluation and Use of Expert Systems** **537**

Behavioral Test and Evaluation of Expert Systems 539
D. Meister

Initial Evaluation of an Intelligent Interface for Operators of Complex Systems 551
D. R. Sewell, N. D. Geddes, and W. B. Rouse

A Knowledge-Based System for Geographical Modeling 559
J. R. Wright and J. T. Diamond

A Framework for Designing Intelligent Human-Computer Dialogues 567
J. Elkerton

An Object-Oriented Reference Model for User Interface Design, Hypertext and Collaboration 575
D. A. Seeley

The Design of FLEX: A Tolerant and Cooperative User Interface to Databases 583
Amihai Motro

**AUTHOR INDEX** **591**

*Cognitive Engineering in the Design of Human-Computer Interaction and Expert Systems,* 329
edited by G. Salvendy
Elsevier Science Publishers B.V., Amsterdam, 1987 — Printed in The Netherlands

# SOFTWARE-DESIGN WITH THE RAPID PROTOTYPING APPROACH: A SURVEY AND SOME EMPIRICAL RESULTS

HOYOS, C. G., GSTALTER, H., STRUBE, V. and ZANG, B.
Lehrstuhl für Psychologie der Technischen Universität München, Lothstr. 17, 8000 München 2

ABSTRACT

Rapid prototyping has become known as an iterative software development procedure. We review the literature on experimental prototyping of human-computer interfaces with respect to different design philosophies, user participation and experimental evaluation methods. Typical elements of the method are illustrated by an example. Hopes connected with the rapid prototyping approach are compared with the results of documented projects. We conclude that iterative design procedures lead to easy-to-use interfaces, reduce the expenses of software development and provide a useful tool in organizational development.

## 1 INTRODUCTION

Opposed to the rapidly growing technological possibilities, planning, production and evaluation of technical systems in the information and communication area follows a rather traditional course. This design process - sometimes referred to as linear - seems to be a matter of course without alternative ideas for most system designers (Hammond et al., 1984) and organizations (Hoyos and Zang, 1986). Only recently has this procedure been criticized in favour of other approaches (e.g. Clark et al., 1984). An important impulse came up from research on acceptability of computer systems. Analysis of user's complaints showed that systems were often uncomfortable to use, errors were frequent, learning processes long and frustrating: the usability of many systems was insufficient.

Since the rapid growth of information technology in the early fifties the structure of people working with computers has changed from mathematicians, engineers and computer specialists to nearly everyone today including laymen in interaction with information devices. Thus, the discrepancy between the mental models of system designers and people working with their products has become larger and larger. As a result of this development, requirements on system design shift from solving technical problems to producing easy-to-use systems. Recognition of the fact that a man-machine-system is to be optimized (and not a technical one on its own) has strenghtened the role of human factors specialists in the domain of software development. Researchers in "software psychology", "cognitive ergonomics" or "software ergonomics" came up with the demand of considering the resources and needs of the potential users as early as possible in the development process. Therefore an important component of efforts to humanize work is the idea of user participation (e.g. Algera and Koopman, 1984): Workers should be actively engagedin the processes which change their work conditions. But neither user participation nor following guidelines for system design ensures good software products. Therefore the conventional way of producing interactive systems is sometimes demanded to be replaced by iterative design. Meanwhile the designation of (rapid) prototyping has become known for a design process with several iterations. But how can prototypes be evaluated to improve them? Clearly, experimental methods from human factors research seem to be useful here, but are they really applied?

To sum up, our intention is to describe and try out the rapid prototyping approach to the design of an office automation system's interface in a team of psychologists, system designers and users to ensure

- early consideration of user's resources, limitation and intentions by following design guidelines and active user participation
- controlled human-factors evaluation of the prototypes.

This project has only been started by now, but we finished a literature review on the topic. The results are given here and some conclusions are derived. A little example of how we imagine to test our prototypes is included.

## 2 SURVEY OF THE PROTOTYPING LITERATURE

From some thousand references on human-computer-interaction in the international literature we figured out those which were concerned with the development of interfaces for office automation systems. Only a little part of these about 600 articles gave detailed information about software development and -evaluation. 28 articles were left describing experimental approaches with the rapid prototyping idea for software development.

### 2.1 General definition

All these references had two things in common: some kind of user participation and an iterative design process. A system sketch was tested, evaluated and modified with the users under field or laboratory conditions. The first sketch sometimes only existed in a paper version and was tested using paper and pencil methods (Williges et al., 1987). Other authors demand an at least partly functional prototype to be evaluated. A prototype was always assumed to be quickly modifiable and with little effort.

### 2.2 Why prototype?

Various reasons are given in the literature to apply rapid prototyping methods in software design. Most often the wish to achieve systems with high usability is documented. Systems characterized by good ease of use are important because of the different suppositions and requirements of the users. Expense in training users to interact with the system should be kept low to save time and money (Gould and Boies, 1984).

Another important reason to prototype is to clarify the system requirements and specification in cooperation with the users (Riddle, 1984). Through an early and exact adaptation of the interface to the user the expense of system development can be cut down considerably.

Some authors do not only see rapid prototyping as a method to develop products but as a tool to facilitate real user participation and socialization, i.e. as a general tool for organizational development purposes (Cohill et al., 1985; Hollinde and Wagner, 1984). Users are to be motivated and qualified to actively take part in the changes concerning their work environments, tasks, tools etc.

Another argument put forward for prototyping is the acceleration of the software development process. Because of the interaction with the users specifications become more precise and conceptual errors and inefficient lines of development can be avoided in the early development process (Gomaa and Scott, 1981). Costs connected with the correction of wrong design decisions become more expensive the later they are recognized. Thus, the total expense of the software development should be decreased by prototyping in terms of lines of code written, hardware, costs, effort for training as well as maintenance costs (Klausner and Konchan, 1982; Meister, 1985).

Prototyping provides elements of feedback and communication between software designers and users (Floyd, 1984). However, detailed information about the nature of this communication is seldom documented in the literature. Most authors mention interviews or questionnaires (Bjorn-Anderson et al., 1986) without specifying what has being asked and how feedback was given by the users. Sometimes prototyping and user participation is used for acceptability testing and general increase in user's acceptability (Williges et al., 1987). This goal has been criticized and called pseudo-participative (Mambrey and Oppermann, 1985; Müller-Böling, 1986).

### 2.3 Various approaches to prototyping

Different approaches of prototype construction and use could be found in the literature. In principle, two main philosophies can be distinguished, between which many mixed strategies can be thought of. The first approach is called incremental prototyping. Similar concepts are meant by protoversioning (Wedekind, 1985) or evolutionary design (Bonin, 1984). The first system sketch and its successors are tested (on an average of 3 to 6 iterative cycles); every prototype is improved until some criteria are fulfilled. The last prototype in the series of iterations equals the final product and becomes implemented (Sroka and Rader, 1986). Differences exist within this conceptual framework, e.g. Hallmann (1985) talks of incremental design as the stepwise improvement of particular parts of the system, whereas evolutionary design tries to model and to improve the system as a whole.

This is set apart in principle from the so-called throw-away approach. This means the construction and testing of a prototype which is thrown away after the first iteration and replaced by an entirely new version later on. The main purpose of the prototype is to serve as a model for the development process (Patton, 1983). Clarification of requirements and system specification is to be achieved with the throw-away approach.

A further strategy was found in the literature called flexible approach (Eisfelder, 1983). Here the users themselves created the system in an experimental way by putting together modular units which had been given to their disposal by the designers. Other development procedures involve testing different versions of a system against each other (Francas et al., 1985) or testing only parts of the system in a prototypic fashion and generalizing results to the other components. These approaches are not typical for rapid prototyping and will not be referred to in the rest of this paper.

## 2.4 The user's role

User participation is besides the iterative procedure an important feature of rapid prototyping, but there is great variation concerning the point in time and the nature of participation. To interact with the user as early as possible to adapt the requirements to the user's needs is widely agreed upon (e.g. Wixon et al., 1984). Early participation is necessary if the users are to have a voice in decisions about conceptual issues. Examples can be found where the system is nearly completely based on user's reactions (e.g. Gould and Boies, 1984).

More often the user is confronted with a prototype already defined and functioning. His or her role in the design process will then be restricted to being a subject in a human-factors testing. Besides working at the prototype to process given tasks, additional observations and interviews are often mentioned. Sometimes the users are asked to propose changes in the design(Brice et al., 1983). Supporters of this method often argue that the users have not enough experience and knowledge about system design. Frequently communication problems between designers and users are reported. It would be easier for them to test and criticize given interfaces than to make proposals which are concrete and realistic (Alavi, 1984a). Anticipation and articulation of future system properties is often difficult for the users (Gomaa and Scott, 1981).

But sometimes efforts are taken to qualify users as members of the design team and to give them more credit than only being an expert in the kind of work the system is to facilitate (Cohill et al., 1985).

Who are the 'users' in the prototype evaluation process? It is often stated that they should be a sample of the actual future users (Clark et al., 1984; Cohill et al., 1985). Most references only give 'users' or 'useful subjects' as a description (e.g. Alavi, 1984b; Gill et al., 1982; Gomaa, 1983; Gomaa and Scott, 1981). Probably these samples include unrepresentative subjects like students as is explicitly stated in Boehm et al. (1984) and Bury (1984). Clearly, students are not appropriate subjects of testing office automation interface prototypes.

Even more vague is the group of persons leading, controlling and managing the software development process in the articles describing experimentally oriented prototyping. Examples of labels are designers, design team, experimenter, project manager or even 'people'. But one thing is obvious from the descriptions of what they did: psychologists didn't take part - this was of special interest to us psychologists!

## 2.5 Methods and criteria of prototype evaluation

Trying to gain an overview of the experimental methods applied to prototype evaluation is often made difficult because of abstract or incomplete description of tasks and testing procedures. Methods of data collection sometimes remain undiscussed even if they led to important design decisions. Table 1 nevertheless tries to give a summary of methods used for prototype testing in the literature.

The most common procedure is to ask questions about the user-prototype interaction. Very general statements dominate concerning the evaluation of the system, satisfaction, preferences, ease of use. Where standardized questionnaires have been used they were never documented. Unstandardized methods are sometimes called asking questions (Francas et al., 1985; Mack, 1985), interviews (Lund, 1985) or field interviews (Alavi, 1984a).

The same holds true for the descriptions of observation procedures, which are very general or even absent. Mentioned is search behaviour (Hewett and Meadow, 1986), asking the assistant for help (Neal and Simons, 1984) or 'dialogue' (Green and Wei-Haas, 1985). Often the user is asked to 'think aloud' during his interaction with the system, i.e. to verbalize all his thoughts. The

TABLE 1

Methods described to gather information on evaluation criteria in prototype testing

1=Alavi, 1984a; 2=Alavi, 1984b; 3=Boehm et al., 1984; 4=Brice et al., 1983; 5=Bury, 1984; 6=Clark et al., 1985; 7=Cordes, 1984; 8=Francas et al., 1985; 9=Gomaa and Scott, 1981; 10=Gomaa and Scott, 1980; 11=Gould and Boies, 1984; 12=Green and Wei-Haas, 1985; 13=Hewett and Meadow, 1986; 14=Lund, 1985; 15=Mack, 1985; 16=Marshall, 1984; 17=Neal and Simons, 1984)

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Questionnaires | | | | | | | | | | | | | | | | | | |
| -free | X | X | | | | X | | X | | X | X | X | X | X | X | | X | 11 |
| -standardized | X | | | | X | | | | | X | | | | | | | | 3 |
| observation | | | | X | X | | | | | | X | X | X | X | X | | X | 8 |
| thinking aloud | | | | | | | | | | | | | X | X | X | X | | 4 |
| task analysis | X | | | | X | | | | | | | | | | X | | | 3 |
| protocolls | | | | | | | X | | X | | | X | | X | | X | X | 6 |
| subj.ratings | X | | X | | | | X | X | | | | | | | | | | 4 |
| comparison of variants | | | | | | | | X | | | | | | | | | | 1 |
| intuition | | | | X | | | | | | | | | | | | | | 1 |

idea, of course, is to gain some insight into causes of overt, observable behavior of the subject. The general methodological shortcomings of thinking aloud are documented in the literature on experimental psychology since the beginning of the century. In the present context, results of this method are very difficult to interprete, especially if time required by the user to solve his tasks is measured simultaneously. The dual task of verbalizing one's thoughts clearly interacts with performance criteria in the primary task.

An other method is the analysis of the steps of the subject working at thetask. This can e.g. be done by registration of keystrokes. Registration of the data is easily done by a computer and as underlying theoretical foundations ('keystroke level model'; Card, Moran and Newell, 1983) exist, this method has become rather popular.

Protocolls or recordings can involve video recording of the entire dialogue between user and prototype (Lund, 1985), measurement of times (Marshall, 1984), registration of errors (Neal and Simons, 1984), comments by the subject (Gomaa and Scott, 1981) or combinations of these data. Often a second computer is used for these recordings. Time and error values have the appeal of being exact, objective and directly comparable variables.

Subjective measures include Likert-scales (Alavi, 1984a), ratings of subjective preferences (Boehm et al., 1984; Francas et al., 1985), scales to measure user satisfaction (Francas et al., 1985) and rank orderings of different alternatives (Francas et al., 1985).

It often remains unclear, what kind of evaluation criteria are connected with the methods of data collection and analysis described above. Overall subjective assessments of system characteristics functionality (Boehm et al., 1984), ease of use (Alavi, 1984a; Cordes, 1984) occur in the references as well as measurement of special facts like different times (Neal and Simons, 1984), classification and frequency of errors (Hewett and Meadow, 1986), utility of help facilities (Lund, 1985; Neal and Simons, 1984) or statements about problems occurring during the work with the prototype. Criteria vary considerably with respect to a 'hard-soft' dimension, reaching from measurements of time in msec to intuitive comments and proposals. In total, the not quantifiable evaluation criteria dominate and seem to be more useful in reaching at an overall assessment of the quality of human-computer-interaction. Most hints can be gained by dialogue registration and -analysis. The recording of the dialogue using the methods above is recommended in every case. Details from the dialogue structure can be useful to rate the quality of the interaction according to many different evaluation criteria (Neal and Simons, 1984).

TABLE 2

Selected evaluation criteria used for prototype evaluation

| GENERAL EVALUATION CRITERIA | MORE DETAILED CRITERIA |
|---|---|
| - functionality | - utility of the system for special tasks<br>- output accuracy<br>- output utility |
| - ease of use/usability | - performance times<br>- tasks-related errors<br>- frequency of different commands<br>- time spent reading documentations<br>- comments, suggestions and preferences of users |
| - self-explanatoryness<br>- easy to learn<br>- flexibility<br>- can be adapted to fit individual needs<br>- general satisfaction<br>- satisfaction with the system | |
| - dialogue | - difficulties/problems<br>- behavioral sequences<br>- search strategies<br>- number of tasks solved/time<br>- efficiency: relation between necessary and actually executed keystrokes<br>- asking (the experimenter) for help<br>- frequences and types of errors<br>- time until the first keystroke is done |

## 3 AN EXAMPLE: EVALUATING THE INTERFACE OF AN OFFICE AUTOMATION SYSTEM

A team of technicians and psychologists designed a prototype of an interface of a system with typical office functions like printing, mailing, documentation etc. This sketch was to be tested on different aspects of usability featuring ease of learning and self-explanatoryness. The system was designed using the desk-top-metaphor. Software was written in Interlisp D and implemented on a Siemens 5815.

To test the prototype we constructed a pool of typical office tasks which our subjects had to solve in interaction with the simulated system. The subjects were people working in offices and had no prior experience with advanced information processing devices. After a short explanation of the system the subjects got a 2-page manual and started to work on the problems. During the experiment one experimenter wrote a protocoll of the session using a standardized observation procedure. A second experimenter simulated the non-implemented system-functions. Errors, working times and comments given by the subjects were recorded.

The test of the first system sketch could be stopped after only 4 users had been in the experiment. Parts of the system showed a lack of consistency and reliability; names of functions were ambiguous, menues were not adapted to the task. The results were discussed and led to a modification of the prototype.

The second experiment was conducted with 12 users belonging to the same population as above. The tasks remained unchanged, their order was systematically permuted. Time measurements had to be given up for technical reasons. The "manual" was revised and at the end of each session a standardized interview about the system was added. Of course, no details on the results can be given here. In total, the usability of the new prototype was considerably improved. Analysis of the recorded data nevertheless showed shortcomings which would have to be removed in further iterations.

Our conclusions from that little experiment were: 1. The users enjoyed the sessions and tried to help us giving comments on the system and their typical tasks; but only few made proposals for system improvements. 2. The prototypes could be evaluated quickly with respect to usability criteria using the collected data. 3. Resulting prototype assessment could easily be translated into modifications. 4. Shortcomings in the prototype-user interaction had to be discovered by empirical data; they could <u>not</u> have been predicted by deviations from guidelines. 6. Cooperation between psychologists and system designers was very instructive and initiated several other projects.

## 4 CONCLUSIONS: ADVANTAGES AND PROBLEMS CONNECTED WITH THE RAPID PROTOTYPING APPROACH FOR SOFTWARE DEVELOPMENT

Although there still is a considerable lack of knowledge about iterative system design, especially concerning adequate evaluation methods for prototypetesting, we come to some preliminary conclusions, derived from the literature review and our own experiences. Some problems associated with prototyping will be outweighted by lots of positive results; but this may be distorted by the tendency of most researchers or journal editors only to write about or publish successful projects and to neglect unsuccessful ones.

Firstly, prototyping was found to be a valuable method in a social sense and can be recommended for purposes of organizational development. It increased work motivation and user satisfaction, promoted communication between users, designers and technicians, reduced feelings of anxiety and uncertainty about what will happen in the organization in the near future, proved to be a valuable concept in user qualification and clearly improved the acceptability of the final products (Alavi, 1984a; Alavi, 1984b; Bonin, 1984; Clark et al., 1984; Gomaa, 1983; Gomaa and Scott, 1981; Green and Wei-Haas, 1985; Lund, 1985). Of course, an organization which works with prototyping for the first time may have to pay for its experiences. This can be due to unrealistic anticipations on the users' side, to a lack of management strategies needed for the concept or an overestimation of the users' ability to articulate their needs and to anticipate future requirements (Alavi, 1984a; Bonin, 1984; Cohill et al., 1985; Floyd, 1984; Gomaa, 1983).

What about the quality of the prototyped systems? There is agreement that the resulting man-computer interfaces were highly satisfactory both in terms of the systems' functionality and its usability or ease of use. Solutions were generally called flexible, precise and elegant (Boehm et al., 1984; Brice et al., 1983; Eisfelder, 1983; Gomaa, 1983; Gould and Boies, 1984; Lund, 1985). This is attributed to the prototyping's ability to detect errors and misconceptions through prototype testing and user feedback early in the software development process, the possibility of trying various ideas instead of only one and the vividness of prototype versions, which help to clarify requirements. Some anothers limit the application of the prototyping method to systems with low complexity (Alavi, 1984a; Bewley et al., 1984; Boehm et al., 1984; Gomaa and Scott, 1981).

What is the effort needed for prototyping? With a few exceptions there is no straightforward way to compare the expense of traditional versus iterative design. But estimations by experts and comparisons with similar products's developments unequivocally show that prototyping was less expensive in terms of money, manpower, lines of codes written and development time (Boehm et al., 1984; Bury, 1984; Clark et al, 1984; Cohill et al., 1984; Francas et al., 1985; Gehani, 1982; Gill et al., 1982; Gomaa and Scott, 1981; Neal and Simons, 1984; Klausner and Konchan, 1982). Very few exceptions are documented (Eisfelder, 1983; Lund, 1985). Shortcomings in the current methodology are mainly associated with the evaluation methods of the prototypes: 'Users' are seldom representative; tests are very short (we found a maximum of 4 hours with an average of 2 hours)providing a kind of snapshot rather than long term effects on learning processes or mental load problems; maintenance costs couldn't be considered empirically until now; evaluation criteria and methods have to be improved in operationalization and standardization. To us, necessary improvements in rapid prototyping methodology seem profitable.

## 5 REFERENCES


Alavi, M., 1984. The evolution of information systems development approach: some field observations. Data Base, Spring '84, pp. 19-24.

Alavi, M., 1984. An assessment of the prototyping approach to information systems development. Communications of the ACM 27, pp. 556-563.

Algera, J.A. and Koopman, P.L., 1984. Automation: Design process and implementation. In: Drenth et al. (Eds.), Handbook of work and organizational psychology. Wiley, New York, pp. 1037-1066.

Bewley, W.L., Roberts, Th., Schroit, D. and Verplank, W.L., 1984. Human factors testing in the design of Xerox 8010 'Star' office work station. In: A. Janda, Human factors in computing systems. North-Holland, Amsterdam, pp. 72-78.

Bjorn-Anderson, N., Cason, K. and Robey, D. (Eds.), 1986. Managing computer impact. An international study of management and organizations. Ablex Publ. Corp., Norwood.

Boehm, B., Gray, T. and Seewaldt, T., 1984. Prototyping versus specifying: a multiproject experiment. IEEE: Transactions on Software Engineering, Vol. SE 10(3), May 1984, pp. 290-301.

Bonin, H., 1984. Prototyping. ÖVD/Online, 5, pp. 74-78.

Brice, L., Connell, J. and Shafer, D., 1983. Using 'Ingres' as a rapid prototyping device during development of management informations applications. IEEE: Proceedings of the symposium on application and assessment of automated tools for software development, pp. 34-43.

Bury, K.F., 1984. The iterative development of usable computer interfaces. In: B. Shackel (Ed.), Human computer-interaction - Interact '84. Conference papers, pp. 743-748.

Card, S.K., Moran, T.P. and Newell, A., 1983. The psychology of human computer interaction. Lawrence Erlbaum, Hillsdale, N.Y.

Clark, F., Drake, P., Kapp, M. and Wong, P., 1984. User acceptance of information technology through prototyping. In: B. Shackel (Ed.), Human-computer-interaction - Interact '84. Conference papers, pp. 703-708.

Cohill, A.M., Harper-O'Donnell, L. and Curran, F.P., 1985. User partnership in system design and data administration. Swezey, R. (Ed.), Proceedings of the Human Factors Society, 29th Annual Meeting, pp. 480-484.

Cordes, R.E., 1984. Software ease-of-use evaluation using magnitude estimation. Allusi, M. et al. (Eds.), Proceedings of the Human Factors Society, 28th Annual Meeting, pp. 157-160.

Eisfelder, H., 1983. Prototyping mit modularer Anwendungs-Software. In: Handbuch der modernen Datenverarbeitung, pp. 69-75.
Floyd, C., 1984. A systematic look at prototyping. In: R. Budde et al., Approaches to prototyping. Springer, Berlin, pp. 1-18.
Francas, M., Goodman, D. and Dickison, J., 1985. A reliability study of task walk-through in the computer communications industry. Human Factors, 27, pp. 601-605.
Galitz, W.O., 1984. Humanizing office automation. QED Information Sciences, Inc., Wellesley, Mass.
Gehani, N.H., 1982. A study in prototypng. ACM Sigsoft Software Engineering Notes, Special Issue on Rapid Prototyping, 7, pp. 71-74.
Gill, H., Lindvall, R., Rosin, O., Sandevall, E., Sörensen, H. and Wigertz, O., 1982. Experience from computer supported prototyping for information flow in hospitals. ACM Sigsoft Software Engineering Notes, Special Issue on Rapid Prototyping, 7, pp. 67-70.
Gomaa, H., 1983. The impact of rapid prototyping on specifying user requirements. ACM Sigsoft Software Engineering Notes, 8, pp. 17-28.
Gomaa, H. and Scott, D., 1980. An APL-prototype of a management and control system for a semiconductor fabrication facility. Proceedings of the APL Users Meeting, pp. 573-583.
Gomaa, H. and Scott, D., 1981. Prototyping as a tool in the specification of user requirements. Proceedings of the 5th International Conference on Software Engineering, March 9-12, San Diego, Cal., ACM Order No. 592810, pp. 333-339.
Gould, J.D. and Boies, S.J., 1984. Human factors of the 1984 olympic message system. Allusi, M. et al. (Eds.), Proceedings of the Human Factors Society, 28th Annual Meeting, pp. 547-551.
Green, P. and Wei-Haas, L., 1985. The rapid development of user interfaces: experience with the wizard of oz method. Swezey, R. (Ed.), Proceedings of the Human Factors Society, 29th Annual Meeting, pp. 470-474.
Hallmann, M., 1985. Klassifizierung von Prototypingansätzen in der Software-Entwicklung. Forschungsbericht 209. Dortmund: Lehrstuhl für Software-Technologie der Universität Dortmund.
Hammond, N., Jorgensen, A., MacLean, A., Barnard, P. and Long, J., 1984. Design practice and interface usability: evidence from interviews with designers. In: A. Janda (Ed.), Human factors in computing systems. North-Holland, Amsterdam, pp. 40-44.
Hewett, T.T. and Meadow, C.T., 1986. On designing for usability: An application of four key principles. Mantei, M. et al. (Eds.), Proceedings of CHI '86 Human Factors in Computing Systems, ACM, pp. 247-252.
Hollinde, I. and Wagner, K.A., 1984. Experience with prototyping in command information system. In: R. Budde et al., (Eds.), Approaches to prototyping. Springer, Berlin, pp. 80-91.
Hoyos, C. Graf and Zang, B., 1986. Einführung rechnergestützter Systeme im Bürobereich: Arbeitsfeld für Psychologen. In: H. Methner (Ed.), Psychologie in Betrieb und Verwaltung. Bericht über die 28. Fachtagung zur Arbeits- und Betriebspsychologie, Wiesbaden 1986. Deutscher Psychologen Verlag, Bonn.
Klausner, E. and Konchan, T.E., 1982. Rapid prototyping and requirements specification using PDS. ACM Sigsoft Software Engineering Notes, Special Issue on Rapid Prototyping, 7, pp. 96-105.
Lund, M.A., 1985. Evaluating the user interface: The candid camera approach. Borman, L. et al. (Eds.), Proceedings of CHI '85 Human Factors in Computing Systems, ACM, pp. 107-113.
Mack, R., 1985. Identifying and designing toward new user expectations in a prototype text-editor. Borman, L. et al. (Eds.), Proceedings of CHI '85 Human Factors in Computing Systems, ACM, pp. 139-141.
Mambrey, P. and Oppermann, R., 1985. Partizipation von Benutzern bei der Softwareentwicklung - Fortschritte und Rückschlage. In: Gesellschaft für Informatik: Softwaretechnik-Trends. Mitteilungen der Fachgruppe "Software-Engineering", Heft 5-2, pp. 47-62.

Marshall, C.R., 1984. System ABC: A case study in the design and evaluation of a human-computer dialogue. In: B. Shackel (Ed.), Human Computer Interaction - Interact '84. Conference papers, pp. 571-575.
Meister, D., 1985. Behavioral analysis and measurement methods. Wiley, New York
Müller-Böling, D., 1986. Akzeptanz und Partizipation - sind Systemgestalter lernfähig? Informatik Fachberichte 123. Springer, Berlin, pp. 153-167.
Neal, A.S. and Simons, R.M., 1984. Playback: a method for evaluating the usability of software and its documentation. In: A. Janda: Human factors in computing systems. North-Holland, Amsterdam, pp. 78-83.
Patton, B., 1983. Prototyping - A nomenclature problem. ACM Sigsoft Software Engineering Notes, 8, pp. 14-16.
Riddle, W.E., 1984. Advancing the state of the art in software system prototyping. In: R. Budde, et al. (Ed.): Approaches to prototyping. Springer, Berlin, pp. 19-26.
Smith, S.L. and Mosier, J.N., 1984. Design guidelines for user-system-interface software. The Mitre Corporation, MTR 9420, Bedford, Mass.
Sroka, J.M. and Rader, M., 1986. Prototyping increases change of systems acceptance. Data Management, March, pp. 12-20.
Wedekind, H., 1985. Ein Experiment zum rapid prototyping. Angewandte Informatik 27, pp. 58-61.
Williges, R.C., Williges, B.N. and Ellerton, J., 1987. Software interface design. In: G. Salvendy (Ed.), Handbook of human factors, Wiley, New York, pp. 1416-1449.
Wixon, D., Witheside, J., Good, M. and Jones, S., 1984. Building a user-defined interface. In: A. Janda: Human factors in computing systems. North-Holland, Amsterdam, pp. 24-28.

# AUTHOR INDEX


| | |
|---|---|
| Allamanno, N. | 147 |
| Anzai, Y. | 175 |
| Azuma, M. | 167 |
| Balzert, H. | 89 |
| Bennett, R. W. | 391 |
| Bergman, H. | 383 |
| Bertaggia, N. | 147 |
| Boesser, T. | 45 |
| Boose, J. H. | 491 |
| Brown, C. R. | 523 |
| Bullinger, H.-J. | 307 |
| Bylander, T. | 507 |
| Cai, S. | 183 |
| Campbell, J. A. | 473 |
| Chandrasekaran, B. | 507 |
| Chao, B. P. | 357 |
| Chignell, M. H. | 273 |
| Chow, C.-H. | 399 |
| Conti, P. | 83 |
| Coskuntuna, S. | 61 |
| Davies, D. | 37 |
| Diamond, J. T. | 559 |
| Eason, K. D. | 147 |
| Edwards, R. | 415 |
| Elkerton, J. | 567 |
| Engelhardt, K. G. | 415 |
| Faehnrich, K.-P. | 37, 307 |
| Feeney, J. W. | 407 |
| Fischer, G. | 97 |
| Fu, C. | 423 |
| Galer, M. D. | 289 |
| Geddes, N. D. | 551 |
| Glenn, J. W. | 431 |
| Glinert, E. P. | 341 |
| Gonczarowski, J. | 341 |
| Gonzalez-Sustaeta, J. | 113 |
| Gopher, D. | 233 |
| Greenspan, S. L. | 391 |
| Gregory, K. | 139 |
| Gstalter, H. | 329 |
| Gutierrez, O. | 529 |
| Hagiwara, N. | 457 |
| Hammer, J. M. | 297 |
| Hannigan, S. | 225 |
| Harker, S. | 365 |
| Hayashi, Y. | 175 |
| Herring, V. | 225 |
| Hoyos, C. G. | 329 |
| Ito, M. | 175 |
| Josephson, J. R. | 507 |
| Kantro, E. | 61 |
| Kitto, C. M. | 515 |
| Lakhal, L. | 155 |
| Le Thanh, N. | 155 |
| Lueke, E. | 523 |
| Marcus, S. | 499 |
| Martinez, C. | 113 |
| Meister, D. | 539 |
| Miranda, S. | 155 |
| Mizutani, N. | 249 |
| Mori, H. | 175 |
| Moser, M. C. | 449 |
| Motro, A. | 583 |
| Muckler, F. A. | 3 |
| Nagai, A. T. | 491 |
| Nakamura, N. | 257 |
| Nakaya, N. | 249 |
| Nielsen, J. | 241 |
| Norifusa, M. | 457 |
| Novara, F. | 147 |
| o'Cahan, K. | 383 |
| Olphert, C. W. | 147 |
| Owen, D. | 349 |
| O'Shaughnessy, M. P. | 61 |
| Pagerey, P. D. | 523 |
| Paul, D. W. | 53 |
| Pavard, B. | 465 |
| Raskin, V. | 121 |
| Root, R. W. | 399 |
| Ross, S. P. | 473 |
| Rouse, W. B. | 551 |
| Ruggles, L. | 481 |
| Russell, A. J. | 289 |
| Salvendy, G. | 13, 257 |
| Schmandt, C. | 439 |
| Seeley, D. A. | 575 |

| | |
|---|---|
| Sewell, D. R. | 551 |
| Shackel, B. | 193 |
| Sheppard, S. B. | 281 |
| Shigo, O. | 457 |
| Shneiderman, B. | 207 |
| Smith, C. D. | 341 |
| Soloway, E. | 317 |
| Springer, C. J. | 375 |
| Stelovsky, J. | 83 |
| Streitz, N. A. | 75 |
| Strube, V. | 329 |
| Swain, A. D. | 105 |
| Taladoire, G. | 155 |
| Tominaga, S. | 131 |
| Ukon, M. | 249 |
| Ulich, E. | 29 |
| Wakabayashi, J. | 249 |
| Wang, S. | 183 |
| Wiehle, H. R. | 53 |
| Williges, B. H. | 67 |
| Williges, R. C. | 21 |
| Wright, J. R. | 559 |
| Yamagishi, N. | 167 |
| Yoon, W. C. | 297 |
| Yoshikawa, H. | 249 |
| Zang, B. | 329 |
| Zhang, F. | 183 |
| Ziegler, J. | 37, 307 |
| Zimolong, B. | 265 |