EXPERT SYSTEMS: THE USER INTERFACE

edited by
James A. Hendler

EXPERT SYSTEMS: THE USER INTERFACE

edited by

James A. Hendler
University of Maryland

ABLEX PUBLISHING CORPORATION
Norwood, New Jersey 07648

Library of Congress Cataloging-in-Publication Data

Expert systems.

(Human/computer interaction)
Bibliography: p.
Includes index.
1. Expert systems (Computer science) I. Hendler, James A. II. Series: Human/computer interaction (Norwood, N.J.)
QA76.76.E95E985 1987 006.3'3 87-18675
ISBN 0-89391-429-0

Ablex Publishing Corporation
355 Chestnut St.
Norwood, NJ 07648

EXPERT SYSTEMS: THE USER INTERFACE

HUMAN/COMPUTER INTERACTION

A Series of Monographs, Edited Volumes, and Texts

SERIES EDITOR

BEN SHNEIDERMAN

Series Editor's Preface

Ben Shneiderman

The Controversy

Enthusiasts see expert systems as a replacement for scarce human experts, the embodiment of a human expert's wisdom, a repository for the collective knowledge of many experts, or a tool to enable novices to behave like experts. Critics complain that the promises are more attractive than the reality and that the term "expert systems" is seductive and possibly misleading.

Some prefer the term "knowledge-based" systems to convey the idea that knowledge is being stored on the computer, not just data or algorithms. A still less flamboyant term—"rule-based" systems describes the programming language style of using multiple, nonsequential IF-THEN groupings. Many hours of debate have been wasted in discussing whether an expert system can be written without a rule-based language.

In spite of the confusion over terminology, it is clear that something new has emerged. The promise of advanced systems that support less-structured decision-making situations has engaged the energy of thousands of researchers, product developers, and commerical designers. Hardware manufacturers have come forward with high-speed machines to support rule-based programming and large screen displays to provide visibility for the complex world of action in many of the applications.

But while there has been tremendous progress in programming methods and hardware, there seems to be little emphasis on the user interface. Designers of early expert systems postulated that a natural language front end would handle all the problems and that a human-like dialogue would be most effective. Of course, these naîve visions generated serious problems with the implementation and the usability of practical systems. Natural language front ends that look good during demonstrations are not sufficiently robust to handle the diversity of input in realistic applications. Human-like dialogue in which the computer prompts the user for answers to a long series of questions are too rigid, tedious, and machine-centered.

Real users of expert systems are not data entry clerks. They have a large body of knowledge about the problem, the computer, and the real world. In many cases they are willing to learn some compact notation to speed problem entry, but they also want more flebility in the sequence of actions. They also want to understand what is happening and be in control. Therefore they are often unsatisfied with the generated explanations of why each rule was invoked, but prefer to be in charge along the way, directing the machine and developing an effective cognitive model of the task domain that reduces their dependency on the expert system.

Turning Toward a Scientific Approach

These unsupported conjectures may not apply to many situations and users, but are meant as an alternate view of the users. Rather than waste further hours of debate, it seems possible to develop a more scientific approach and study explicitly the user interface for expert systems. Instead of vague arguments about the user friendliness of competing approaches, we can turn to controlled psychologically oriented experiments or carefully planned observational and thinking aloud studies of user behavior. Separate measurements are made of the learning time, speed of performance on benchmark tasks, rates and distribution of errors, and retention over time for a variety of user communities. In addition, subjective satisfaction scales and informal comments can provide insight to the problems users experience.

The benefits of these approaches in advanced basic research and in commercial environments have been amply demonstrated in other applications. Usability laboratories and thorough user interface testing have become part of the landscape for successful software companies.

Expert Systems Programmer Interfaces

Programmers, knowledge engineers, and maintainers or expert systems also need effective user interfaces. When there are 1,000 rules of 20 lines each and 80 are invoked to produce a result, tracing and debugging become fighteningly difficult. Successful tools must support rapid and comprehensible browsing of rules and facts, graphic display of relationships, convenient dynamic execution, meaningful documentation facilities, and elaborate version control. Still more powerful features are needed when multiple graphics displays or direct manipulation programming is involved.

Creating a Community

With explicit attention to the user interface the advantages of rule-based programming will be more easily and successfully applied in many situations. To foster more attention to the user and programmer interface, we began to organize a workshop. We wanted to bring together the small number of people who had already identified the problems and were working on solutions. In early 1986, about 20 people were contacted to present their work during 2-day in June 1986. The organizing committee included James Hendler, Dana Nau, James Reggia, and Roland Simon.

With support from the University of Maryland Institute for Advanced Computer Studies (UMIACS) (Larry Davis, acting director) and the Department of Computer Science (Victor Basili, chair) we were able to invite the people we wanted and play host to a stimulating workshop. Johanna Weinstein did a superb job with the administrative issues, resulting in a smooth running workshop.

The Workshop

The thoughtful presentations, lively discussions, and joy at discovering colleagues with shared interests made for a successful workshop. Many of the chapters in this volume were prepared especially for the workshop. James Hendler organized the reviewing and editing process and diligently pursued the researchers who were working on key problems but could not attend the workshop. The reviewers made numerous suggestions that strengthened the individual efforts and the cohesiveness of the collection. We hope that the appearance of this book will stimulate further research in this area.

The Future

Our workshop was just a beginning. We were excited with the enthusiastic response of participants and the numerous intriguing directions for research. I believe that greater attention to the user interface will be extremely beneficial to the success of expert systems. The work will be challenging, but the payoff substantial. I was very satisfied to see that researchers have come to recognize the central role of user interface design.

The user interface is not the paint put on at the end of the project, but the steel frame on which to hang the details. Designers who are concerned about good user interfaces are finding the ideas to build quality into their systems. I believe that we are getting closer to the goal of building interactive systems that are comprehensible, predictable, safe, and attractive. I think users of well-designed systems experience a higher level of competence, mastery, clarity, potency, and satisfaction.

Preface

James Hendler

As Ben has mentioned in his Preface, this book was originally to be a presentation of the set of papers given at a workshop held at the University of Maryland entitled "Expert systems: The user interface." I was asked to edit because of my interest in the area and my involvement in the workshop. As time progressed, however, it became clear that interest in the book was so high that just presenting a set of workshop papers wouldn't be enough. New papers were sought to try to present a balanced coverage of the field of interface design for expert systems. We felt it important to represent as many of the different people working in the expert systems field as possible. Our goal was to present papers from academics and those working in industry; from diagnosis, classification, and domain modeling; and from the perspective of knowledge engineer, expert, and end user. Thus, I tried my best to provide a broad coverage of the field with no particular viewpoint emphasized.

I wasn't aware how successful this attempt was until I tried to organize the gathered chapters into subsections for this volume. There were so many different overlaps that almost any grouping of the papers provided some sort of classification. The order these papers now appear corresponds in some rough way to a "spectrum" of the field, with "practice" at one end and "theory" at the other. Unfortunately for this ordering, (although fortunately for the reader), this was an artificial distinction at best. Papers on theory cite specific systems and papers on developed systems aim to explain the theory behind their operation.

Each of the authors contributing to this volume was asked not only to contribute a chapter, but also to read and review at least two other papers. The production schedule was frantic and authors were asked to produce these reviews, read their own reviews, and update their papers in a timely manner. Finding time in busy schedules to perform these tasks was a monumental effort for each of the authors, and their labors are most appreciated. Several outside reviewers were also used. In particular, Kate Erhlich of Symbolics Inc., Lorin Wilde of Lisp Machines Inc., and Joy Bush of the University of Maryland took the time to read several articles each and produce useful reviews for me and the authors. Their efforts, too, are greatly appreciated.

The workshop at which several of these papers were first presented was jointly sponsored by the University of Maryland Institute for Advanced Computer Studies (UMIACS) and the University of Maryland Computer Science Department. I am grateful to the help of those organizing the conference technically: James Reggia, Dana Nau, Ben Shneiderman, and Roland Simon, and administratively: Larry Davis (acting director of UMIACS), Victor Basili (chairman of the Computer Science Department). A special thanks to Johanna Weinstein who did much of the administrative work.

Finally, a great many people expressed interest in contributing articles to this book. It was a difficult task to make these decisions, and I must apologize again to all those doing outstanding work whose articles do not appear here. Thanks for your graciousness and understanding.

Jim Hendler
University of Maryland

Contents

ONE

Introduction: Designing Interfaces for Expert Systems

James Hendler

Department of Computer Science
University of Maryland

Clayton Lewis

Department of Computer Science
University of Colorado

Over the past decade the field of expert systems has grown from a few small projects to a major field of both academic and industrial endeavor. The systems have gone from academic laboratories, through industrial development, and are now reaching a substantial user population. In other areas of computer science such explosive growth has often led to systems which are difficult to learn and painful to use. Will expert systems suffer this same fate? In this chapter we compare expert systems with more traditional computer systems and discuss the special needs of this new field. We do this by presenting several short questions, and the corresponding long answers.

Question 1:

Does Fantastic New AI Technology Avoid Traditional System Usability Issues?

When a new technology makes the transition from laboratory to industry, there is a tendency to treat it as if it has a magical nature. Since it solves many previously unsolvable problems one begins to believe it solves all of them. *Expert systems* are no exception to this. The seeming natural language interfaces of many early systems, the windowing and animation facilities of some of the newer systems, the built-in help facilities of present-day Lisp Machines and the like appear to be the necessary tools to cause user interaction problems to all but disappear. Would that it were so.

Unfortunately, interface design is more complicated than just putting up the windows on the screen. The designer must consider many aspects of computer usage ranging from cognitive models of the users' thought processes to the ergonomic issues of body motions and comfort. The designer must concentrate on many aspects of usability, including a focus on users and their tasks, getting empirical evidence about effectiveness, and stressing iteration between designers, implementers, and users (Gould & Lewis, 1985).

Furthermore, far from alleviating the design burden, expert systems bring up new design issues which must be addressed. Developing an expert system usually contains three separate, but highly interacting, components: knowledge capture, programming and debugging the system, and, finally, placing the system before an active user community. Thus, the interface designer must take new factors into consideration in the design of tools for making these stages more efficient and for the development of systems which can be used by the various personnel involved in this process. The designer must now consider:

1. *The issues involved in providing tools for the different personnel involved in each of these stages.* The designer is forced to examine who is involved at each stage. What are their particular needs? How are these needs best addressed in the design of the system?
2. *The special needs of expert system users.* The user community for expert systems is often different from those using editors, operating systems, and other traditional systems. What are the special needs of these professional users who, despite being computer novices, are often experts in their own field of endeavor?
3. *The efficacy of these interfaces.* The design process requires getting empirical evidence about the effectiveness of the tools. The designer must therefore consider how to evaluate the interfaces designed for expert systems. How do we demonstrate that these systems are beneficial to the users? How do we present a rule base such that the eventual user is able to test it.

In short, instead of avoiding the design issues for standard interfaces, expert systems appear to need all these considerations, plus more. This leads us to our next question:

Question 2:

Are Expert Systems Different From More Traditional Technologies in Their Interface Needs?

Along with the new design issues listed above, expert systems create a different set of demands on the interface. An expert system, unlike a traditional computer program, is not just a *tool* that implements a process, but rather it is a *representation* of that process. Further, many of these processes correspond to judgments that can have critical real world consequences. The user interface must often present not only conclusions, but an explication of the processes by which those conclusions are reached.

Compare the behavior of an expert system to that of a more traditional software system, say a compiler. The compiler user cares about ease of input, clarity, and timeliness of error reports, etc. but is willing to trust the compiler designer's decisions as to optimizations in the code, translations, etc. It is a rare user who demands that the interface show the internal workings of the compiler or produce a represention of the algorithms used.

This is not the case, however, for the typical expert system. These systems are often not merely used as a tool for performing a task, but rather as a support tool for performing some decision-making process. Thus, the interface designer must be able to provide support for a knowledge engineer who is trying to enter and debug the representation of the process, and also support, through the same or a different interface, the user who wishes to know how some decision is reached. Issues involving explanation of the reasoning process and display of this reasoning become paramount in the acceptance of the system by users and thus may become paramount to the designer. This implies that the expert systems interface designer has a difficult task on his/her hands. Not only must the designer concentrate on the issues that lead to acceptance in traditional interfaces (which are often overlooked in the design of expert systems), but he/she must also concentrate on these new design areas of import in the system's acceptance.

Question 3:

What Are the Traditional Design Areas Which Are Often Overlooked in the Design of Expert Systems?

As mentioned above, one of the key features an expert system designer must worry about is the acceptance of the system by the intended users. Most of the thrust in many AI systems today is to get the knowledge in and make it work. The interfaces may be designed in advance, as opposed to allowing them to follow from the data, but often the end user of the system is not considered. The system, a perfectly usable one with a well-designed interface, is sent out to a customer, where it sits on the shelf and is never used.

There are many reasons this may occur. One reason may be simply the natural skepticism about *artificial intelligence* that is still seen in many fields and endeavors. These users need much convincing that the underlying technology is sound and that the knowledge contained is correct. Further, many of these users also insist that the knowledge be changeable by themselves. They are not content with someone else's judgments being promoted as "the answer." They want to make sure they will agree. Typical users in this group would be doctors, lawyers, physicists, managers, and others who are traditionally viewed as *experts* in their own field.

These users make many demands on the designer of an expert system interface. The system must be able to explain its behavior and must be able to be updated, using some interaction technique which is natural to these experts. Further, these experts will demand to see data that will convince them of the efficacy of the system. The issue of exactly what needs evaluation and how such evaluation is to be performed is beyond the scope of "interface design," and yet, the interface designer must be sensitive to such issues—the product developer, or the users themselves, will make demands on the interface during this evaluation process.

Further, there is another, probably larger, group of users for whom expert systems are being designed which is not as subject to this "AI fear." These users, often not the experts themselves, but "consumers of expertise," usually *will* trust the results of the system. Yet they too, in many current situations, are refusing to use the new technologies.

In these cases, we believe, it is often the interface, and thus, the interface designer, which is to blame. If the interface doesn't take into account the global needs of the user, that is those beyond the use of the system itself, the program will not be used. Unfortunately, it is these considerations which are often ignored during the design process.

One of the key factors ignored by many designers is the users' environment. The expert system must be designed to match the office or workplace of the system's users. The designer must take into account the users' access to the system, their time to use it, and the compatibility of the expert system with other interfaces the users have been exposed to. Too often, these factors are not taken into account.

Take, for example, a user with some form of PC on his/her desk. If an expert system is imported which must run on a Lisp Machine located