Designing Expert Systems

A Guide to Selecting Implementation Techniques

Paul J. Kline

Steven B. Dolins

Designing Expert Systems

A Guide to Selecting Implementation Techniques

Paul J. Kline

Steven B. Dolins

Texas Instruments
Dallas, Texas

WILEY

JOHN WILEY & SONS

NEW YORK • CHICHESTER • BRISBANE • TORONTO • SINGAPORE

Library of Congress Cataloging in Publication Data:

Kline, Paul J.
Designing expert systems.

Bibliography: p.
Includes index.
1. Expert systems (Computer science) 2. System design. I. Dolins, Steven B. II. Title.
QA76.76.E95K54 1989 006.3'3 88-37869
ISBN 0-471-50484-X

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

Preface

There is currently a great deal of excitement about the prospects for building new types of computer programs called *expert systems*. While it has long been accepted that computer programs can automate computational and clerical tasks, only with the advent of expert systems has it become possible to automate certain tasks requiring expert knowledge. The expertise of skilled physicians, troubleshooters, designers, analysts, and equipment operators are now being tapped to build expert systems to perform tasks of economic and scientific importance.

The automation of these tasks typically confers a variety of benefits:

- Proliferation of scarce expertise
- Consistent and reliable application of expertise
- Documentation of scarce knowledge

The recent emergence of a variety of software products for expert-system construction has provided widespread access to many of the tools and techniques needed to build expert systems. What is much less widespread is knowledge of how to design expert systems.

This book was written to try to help remedy that problem by providing guidelines that relate problem features to expert-system designs. These guidelines assist expert-system builders in choosing the right expert-system design and the right artificial intelligence (AI) techniques to solve the problem at hand. Since expert-system problems can differ rather dramatically, the guidelines should be useful for designers building their first expert-system or their twenty-first.

Even the experienced expert-system builder who needs no help in developing effective designs for expert systems may find it valuable to have a published source of design guidelines to refer to in design reviews or customer briefings.

We developed the guidelines primarily to assist the practitioner, that is, the expert-system designer. However, the use of quotations from the published literature on expert systems to support the guidelines also makes the book valuable to students of artificial intelligence. This book brings together in one place a wealth of insight and discussion that would otherwise require an extensive study of the professional literature on expert systems.

This discussion of expert systems provides one assessment of the current state of the art in this particular area of artificial intelligence:

- The techniques that have been developed
- The problems those techniques solve
- The pitfalls that might be encountered in using the techniques

We feel fortunate to have been able to collect and record some of the first steps taken in the direction of changing the design of expert systems from an art into a science.

PAUL J. KLINE
STEVEN B. DOLINS

Dallas, Texas
April 1989

Acknowledgments

The research for this book was supported by the Air Force Systems Command, Rome Air Development Center, Griffiss AFB, New York 13441 under contract no. F30602-83-C-0123. This book incorporates a large amount of the material that appeared in the final report of that contract. The preparation of this book was made possible by Texas Instruments Incorporated and its management, who assisted us in numerous ways.

We would like to thank Dr. Northrup Fowler III of Rome Air Development Center for his valuable suggestions during the course of this research. We are very grateful to the expert-system builders who were kind enough to comment on earlier versions of the guidelines: Bruce Buchanan, Penny Nii, Ruven Brooks, John Kunz, Bruce Porter, Robert Drazovich, Frederick Hayes-Roth, Robert Neches, Tim Finin, and Jim Kornell. They are, of course, in no way responsible for any deficiencies of the current set of guidelines.

We would also like to thank the following individuals for their assistance: Byron Davies, Tom Ekberg, Jim Bender, Ken Hill, and Gary Honeycutt at Texas Instruments and Diane Cerra and Wanda Cuevas at John Wiley & Sons.

We are very grateful to the publishers and authors who allowed us to reprint selections from their publications in this book. Rather than acknowledging all the copyright holders here, their names accompany reprinted materials throughout the book.

PAUL J. KLINE
STEVEN B. DOLINS

Contents

Headings in italics denote Design Guidelines.

List of Figures

List of Tables

1 Introduction

A large number of decisions must be made in designing an expert system. For example:

- What knowledge representation technique should be used?
- What problem-solving strategy should be employed?
- How should uncertainty be handled?

Making the right decisions greatly simplifies the development of an expert system and helps to ensure its lasting usefulness. The goal of this book is to help expert-system builders make intelligent design decisions.

Before beginning, it is worth considering the difficulty of making expert-system design decisions: How difficult is it to select the proper artificial intelligence (AI) methods for a particular task? A recent paper by John McDermott, the developer of some of the most successful expert systems ever built, provides an answer:

> Although efforts, some successful, to develop expert systems (application systems that can perform knowledge-intensive tasks) have been going on now for almost 20 years, we are not yet very good at describing the variations in problem-solving methods that these systems use, nor do we have much of an understanding of how to characterize the methods in terms of features of the types of tasks for which they are appropriate [J. McDermott 1988, p. 225].

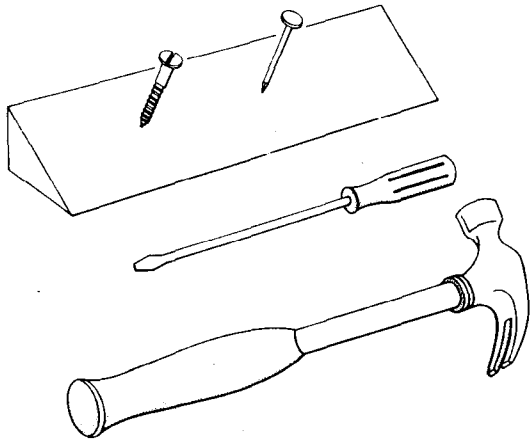

Figure 1.1. It is easy to select the right tool for driving nails and screws. Expert-system development software is often referred to as *toolkits*, which suggests, incorrectly, that it is easy to choose the right AI tool to solve an expert-system design problem.

Despite McDermott's suggestion that it can be difficult to make design decisions, it would be possible for the casual observer to get the impression that it is easy to select the appropriate AI technique to solve a particular problem. For example, the vendors of software systems that support the construction of expert systems often refer to their products as "toolkits." The use of the word "toolkit" to describe this software conjures up images of hammers, screwdrivers, saws, and similar hardware items. The implied suggestion is that selecting the right AI tool for an expert-system problem is roughly as difficult as deciding whether a hammer, screwdriver, or saw is the right tool for a particular repair job. As suggested by Figure 1.1, selecting the appropriate household tool is often easy. Even without experience with these two tools, it is easy to see that the hammer is the right tool for driving the nail and the screwdriver the right tool for driving the screw.

Unfortunately, the task facing the expert-system builder is considerably more difficult than this analogy to selecting real tools suggests. A better analogy is the problem of selecting the appropriate household chemicals to solve cleaning and stain-removal problems. Figure 1.2 offers two alternatives for removing stains, rust stains in one case and mustard stains in the other. *Hints from Heloise* [1980, pp. 380f.] provides the right answer: The lemon juice will remove the rust stains and the vinegar will remove the mustard stains. (In case the correct answer

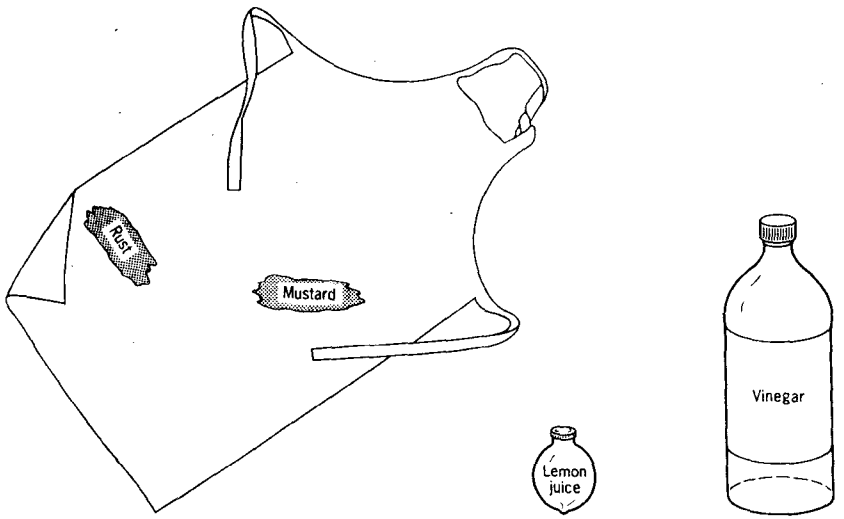

Figure 1.2. Selecting the right household chemical to remove rust stains and mustard stains is harder than the hammer–screwdriver problem depicted in Figure 1.1. Choosing the right AI technique to solve an expert-system problem is more like this stain-removal problem.

to this problem is obvious, perhaps you can also explain why Heloise provides the stern warning "Never use ammonia" to try to remove a mustard stain.)

It is not easy to guess that lemon juice is effective for removing rust stains and vinegar for removing mustard stains, even with a lifetime of familiarity with these chemicals. The fact that there are many such "nonobvious" tools for cleaning has kept Heloise busy for many years churning out household hints.

One way to view this book is as a version of *Hints from Heloise* for the expert-system builder. Simply knowing about Rule interpreters, Frame systems, and Blackboard systems does not guarantee that it is easy to see which of these should be used for a particular expert-system problem—no more than knowing about vinegar and lemon juice guarantees that it is easy to see which of them should be used on the stains.

The examination of the design guidelines for building expert systems found in Chapters 2 through 7 of this book supports the argument that selecting the right AI technique to solve a problem is more like choosing the right stain remover than it is like choosing the right tool from a toolbox.