

**DEVELOPING  
AND MANAGING  
EXPERT SYSTEMS**

**Proven Techniques for  
Business and Industry**

**DAVID S. PRERAU**





# **DEVELOPING AND MANAGING EXPERT SYSTEMS**

**Proven Techniques for  
Business and Industry**

**DAVID S. PRERAU**



**ADDISON-WESLEY PUBLISHING COMPANY**

Reading, Massachusetts ■ Menlo Park, California ■ New York  
Don Mills, Ontario ■ Wokingham, England ■ Amsterdam ■ Bonn  
Sydney ■ Singapore ■ Tokyo ■ Madrid ■ San Juan



---

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. The publisher has made every attempt to supply trademark information about manufacturers and their products mentioned in this book.

**Library of Congress Cataloging-in-Publication Data**

Prerau, David S.

Developing and managing expert systems : proven techniques for business and industry / by David S. Prerau.

p. cm.

Includes index.

ISBN 0-201-13659-7

1. Expert systems (Computer science) I. Title.

QA76.76.E95P74 1990

006.3'3—dc20

89-6801  
CIP

Copyright © 1990 by Addison-Wesley Publishing Company, Inc.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without prior written permission of the publisher.

ABCDEFGHIJ-MA-89



# P R E F A C E

This book presents proven, practical techniques for developing and managing expert systems in business and industry, based on extensive real-world experience.

Expert systems are advanced computer programs that can perform difficult tasks—at a high level of competence—by embodying the knowledge and experience of expert practitioners of the tasks. These programs are coming into widespread commercial use, and as this happens, there is a need for practical information related to their development. My primary purpose in writing this book is to provide some of that practical information based on my broad experience developing expert systems in industry. Therefore this book has a different focus (and, as shall be mentioned, a different organization) from most other books on expert systems.

The book provides a practical step-by-step approach to developing and managing expert systems in the business world. For each step, proven techniques are described and pragmatic advice offered for dealing with the technical and (often overlooked but very important) non-technical problems that must be faced. These techniques are based on my knowledge and experience in developing expert systems—among which is my experience at GTE Laboratories leading the development of one of the largest industrial expert systems, GTE's *COMPASS*.

This book is organized in a unique manner, interweaving general information with concrete examples yet keeping them distinct. Every section of each chapter of the book (other than introductory material) consists of two parts. The first of these parts presents a complete, detailed discussion of the approach, techniques, and recommendations related to a step in expert system development. When appropriate,



generic examples are used to illustrate points. The information in this part generally can be applied to any expert system development project, large or small, and for any application area. This is the information that readers can use—and, I hope, will use—to develop their own expert systems.

The second part of each section appears in a different, slightly smaller type style. It describes how the subject matter of the section was applied by the *COMPASS* project. My objective in recounting the pertinent real-world experiences we encountered in the development of *COMPASS* is fourfold. First, the *COMPASS* experience provides an example of the use of the technique examined in the first part of the section. This should contribute to a better understanding of the points and suggestions made, and thus should be beneficial even for those with no particular interest in *COMPASS* itself. Second, the description of the successful use of a technique or approach should give it added credibility. Third, the discussion of any problems we encountered in *COMPASS* related to the technique or approach might alert the reader to be wary and thus might help in avoiding similar problems. And fourth, for those who might be interested in some of the details of an industrial expert system development project, these parts of the book, taken as a whole, provide a great deal of in-depth, inside information of the development of *COMPASS*, often at a level of detail never before appearing in print about any expert system project.

This novel format should allow readers maximum flexibility in using the book in the way that would be most beneficial for their own particular purposes.

In addition to the section format, there is another feature in the book's organization that I hope will prove useful: At the end of each chapter, I have provided a detailed checklist, highlighting the major points of the chapter. The set of checklists taken together cover all of the steps in expert system development. I have found such checklists to be very useful during my work on *COMPASS* and other expert systems, and I hope they will prove valuable in other such projects. The checklists also provide a succinct summary of the important points of each chapter.

I have written this book to provide useful information to several types of readers. These include (but are not limited to):

- Inexperienced expert systems developers, who can benefit from step-by-step guidance and practical advice as they proceed through their project.
- Experienced expert systems developers, who would pick up many useful practical pointers.
- Managers and technical leaders of expert system development projects, who would benefit especially from some of the nontechnical points discussed, such as on maneuvering through company politics.
- Upper managers under whom expert systems are being or may be developed, who would learn what is reasonable to expect from this new and exciting (but often oversold) technology.



- Professionals considering developing an expert system, who would find it valuable to see what goes into a serious expert system development.
- People interested in learning about (and, possibly, entering) the expert systems field, who would learn a great deal about all aspects of real expert system development, including some aspects not often discussed.
- Researchers and academics in fields related to expert systems, who would find out about some of the restraints and problems that affect expert system development in the business world.
- Undergraduate and graduate students, who would learn about how artificial intelligence theory is applied in practical expert system development.

I hope that the unique information and organization of this book will allow these different types of readers to find the book a valuable resource.

I thank the many people who have made important contributions to this book. First I must express my great appreciation to the members of the *COMPASS* project team, every one of whom did an outstanding job—Alan Gunderson, Robert Reinke, Alan Lemmon, and Mark Adler of GTE Laboratories, the principal developers of *COMPASS*; W. (Rick) Johnson of GTE-Southwest, the *COMPASS* No. 2 EAX expert; Russ Sivey of GTE Laboratories, the *COMPASS* corporate interface; Charles Rich of MIT, the project consultant; and Ralph Worrest, Roland van der Meer, and Marie Goslin of GTE Laboratories and Scott Schipper of GTE Data Services, who contributed to specific phases of the development of *COMPASS*. Many of these people were coauthors with me of published and unpublished papers, reports, and documentation on *COMPASS*, several of which I have used as sources for this book. I am grateful to the personnel of GTE Data Services and the GTE telephone companies who contributed to the successful transfer, deployment, and operation of *COMPASS*. I also thank the management and staff of GTE Laboratories for providing an excellent working environment in which a project like *COMPASS* could thrive.

A number of people provided me with valuable advice and suggestions on the writing of this book. I especially want to acknowledge the extensive efforts of Mark Adler, Alvah Davis, Alan Gunderson, and Sam Levine of GTE Laboratories and the support of Robert Hoffman of Adelphi University.

Finally, I must give great thanks and much love to my family: to my parents, Lillian and Milton Prerau, who have been my lifelong advisers and supports, and, to the two people who sacrificed the most to allow me to write this book—Gail Prerau, my wife, and Michael Prerau, my son, who provided me with strong, loving support and encouragement and accepted a missing husband and father for much longer than we had expected to allow the completion of this book.



# CONTENTS

## 1

### INTRODUCTION 1

- 1.1 Artificial Intelligence 2
- 1.2 Expert Systems and Knowledge Engineering 3
- 1.3 Benefits of Applying Expert System Technology 3
- 1.4 Expert Systems and Experts 5
- 1.5 Roles of Expert Systems 6
- 1.6 Application Areas of Expert Systems 7
- 1.7 Special Capabilities and Features of Expert Systems 7
- 1.8 Limitations of Expert Systems 8
- 1.9 Developing an Expert System 9

## 2

### BASIC CONCEPTS OF EXPERT SYSTEMS 11

- 2.1 Developing an Expert System 12
- 2.2 Knowledge Acquisition 12
- 2.3 Development of an Expert System Program 16
- 2.4 Structure of an Expert System 17
- 2.5 AI Knowledge Representation Paradigms 18
  - 2.5.1 Production Rules 18
  - 2.5.2 Frames, Inheritance, and Object-Oriented Programming 23



## **3**

### **PLANNING AND MANAGING THE DEVELOPMENT 29**

- 3.1 Initial Phases 30**
  - 3.1.1 Project Start-up 30**
  - 3.1.2 Selection of the Domain 34**
  - 3.1.3 Selection of the Development Environment 36**
- 3.2 Core Development Phases 38**
  - 3.2.1 Development of a Feasibility Prototype System 38**
  - 3.2.2 Development of a Full Prototype System 42**
- 3.3 Final Development and Deployment Phases 44**
  - 3.3.1 Development of a Production System 45**
  - 3.3.2 System Deployment 48**
  - 3.3.3 System Operation and Maintenance 50**

**Checklist 51**

## **4**

### **FORMING THE TEAM 55**

- 4.1 Project Technical Leader 56**
- 4.2 Project Manager 58**
- 4.3 Domain Selector 58**
- 4.4 Domain Expert 59**
- 4.5 Consulting Domain Expert 60**
- 4.6 Hardware and Software Selector 60**
- 4.7 Knowledge Acquirer 61**
- 4.8 Knowledge Representer 62**
- 4.9 Knowledge Implementer 63**
- 4.10 Systems Engineer 64**
- 4.11 Project Tool Developer 65**
- 4.12 Corporate/Client Interface 65**
- 4.13 Technical Documenter/Writer 67**
- 4.14 System Tester and Evaluator 68**
- 4.15 System Deployer 69**
- 4.16 Trainer 70**
- 4.17 System Operator 70**
- 4.18 System Maintainer 71**
- 4.19 Consultant 72**
- 4.20 End User 73**

**Checklist 74**



**5****SELECTING THE DOMAIN: THE PROCESS 77**

- 5.1 Importance of Domain Selection 78**
- 5.2 A Method for Evaluating Application Domains 79**

Checklist 96

**6****SELECTING THE DOMAIN: DESIRED DOMAIN ATTRIBUTES 97**

- 6.1 Basic Requirements 98**
  - 6.1.1 Need for the Expert System Approach 98**
  - 6.1.2 Existence of Experts 99**
  - 6.1.3 Need to Capture the Expertise 101**
  - 6.1.4 Limited Success Is Acceptable 102**
  - 6.1.5 Payoff 102**
  - 6.1.6 Risk versus Payoff 105**
- 6.2 Type of Problem 106**
  - 6.2.1 Use of Symbolic Reasoning 106**
  - 6.2.2 Use of Heuristics and Other Task Characteristics 107**
  - 6.2.3 Widespread Knowledge and Common Sense 107**
  - 6.2.4 Not Driven by a Particular Technology 108**
  - 6.2.5 Task Definition 110**
  - 6.2.6 Task Inputs 110**
  - 6.2.7 Task Outputs 111**
  - 6.2.8 Similarity to Successful System 112**
- 6.3 Expertise 112**
  - 6.3.1 Experts Are Better than Novices 113**
  - 6.3.2 Availability of Expertise 113**
  - 6.3.3 Utilizing a Single Expert 114**
- 6.4 Task Bounds 114**
  - 6.4.1 Bounds on Task Difficulty 114**
  - 6.4.2 Estimated Lower Bound on Task Knowledge 115**
  - 6.4.3 Estimated Upper Bound on Task Knowledge 115**
  - 6.4.4 Narrowness of Task 116**
- 6.5 Domain Area Personnel and Politics 116**
  - 6.5.1 Domain Personnel's Expectations of Success 116**
  - 6.5.2 Domain Leaders' Task Agreement and Continuing Involvement 117**
  - 6.5.3 Problem Previously Identified 118**
  - 6.5.4 Top-Level Support 118**



6.5.5	Users Want It	118
6.5.6	Introduction of the System	119
6.5.7	Cooperative User Group	120
6.5.8	Political Problems Related to System Control	120
6.5.9	Political Problems Related to System Results	121
6.5.10	Problems Related to the Sensitivity of the Knowledge	121
<b>6.6</b>	<b>Development, Testing, and Deployment</b>	<b>122</b>
6.6.1	Incomplete Coverage	122
6.6.2	Decomposability	123
6.6.3	Teachable Skill	124
6.6.4	Written Material	124
6.6.5	Availability of Test Cases	125
6.6.6	Real-Time and Performance Issues	125
6.6.7	User Interface	126
6.6.8	Long-Term System Need	127
6.6.9	No Alternative	128
6.6.10	Stability	129
6.6.11	Project Dependencies and Milestones	129
6.6.12	Tolerance to Incorrect Results	130
6.6.13	Measurable Payoff	131
6.6.14	Expert Agreement on Correctness	132
	Checklist	134
<b>7</b>	<b>SELECTING THE HARDWARE AND THE SOFTWARE DEVELOPMENT TOOL</b>	<b>139</b>
7.1	A Process for Selecting the Development Environment	141
7.2	Criteria and Parameters for Comparing Software Tools or Hardware	146
7.3	Obtaining the System	147
7.3.1	Availability	147
7.3.2	Cost	148
7.3.3	Legal Arrangements	148
7.3.4	Integration into Environment	149
7.3.5	Uniformity with Others	149
7.4	Background and Prospects	150
7.4.1	Vendor Track Record and Prospects	150
7.4.2	System History	151
7.4.3	Stage of Development	151
7.4.4	System Upgrading	152
7.4.5	System Maintenance by the Vendor	152



<b>7.5</b>	<b>Learning to Use the System</b>	<b>153</b>
7.5.1	Training Materials	153
7.5.2	Training Courses	153
<b>7.6</b>	<b>System Usability and General Features</b>	<b>154</b>
7.6.1	System Interface	154
7.6.2	Supportive Programming Environment	154
7.6.3	Efficiency of the Programming Environment	155
7.6.4	Software Engineering-Related Facilities	155
7.6.5	Documentation	155
7.6.6	Performance	156
7.6.7	Size	157
7.6.8	System Interfaces	157
7.6.9	Security	158
7.6.10	User Support and Consultants	158
<b>7.7</b>	<b>Special Features Related to Software</b>	<b>159</b>
7.7.1	Building or Buying a Software Tool	159
7.7.2	Knowledge Representation Paradigms	160
7.7.3	Incorporation of a Computer Language	162
7.7.4	Match to Problem	162
7.7.5	Access to Source Code	163
<b>7.8</b>	<b>Special Features Related to Hardware</b>	<b>163</b>
7.8.1	Computer Type	164
7.8.2	System Software and File System	164
7.8.3	Support for Multiple Tools	165
<b>7.9</b>	<b>Issues Related to Deployment</b>	<b>165</b>
7.9.1	Use as Deployment Vehicle	165
7.9.2	Easy Transfer to Deployment Vehicle	166
7.9.3	Operation and Maintenance of the Deployed Expert System	167
	<b>Checklist</b>	<b>169</b>

## 8

## SELECTING THE DOMAIN EXPERTS 173

<b>8.1</b>	<b>Responsibility for Selecting the Experts</b>	<b>174</b>
<b>8.2</b>	<b>Attributes of Good Domain Experts</b>	<b>175</b>
8.2.1	Existence of Domain Expertise and Experience	175
8.2.2	Level of Expertise	175
8.2.3	Extensiveness of Experience	176
8.2.4	Reputation	176
8.2.5	Finding Experts with the Right Experience	177



8.2.6	Communication Skills	178
8.2.7	Temperament	179
8.2.8	Cooperativeness	180
8.2.9	Working Relations	181
8.2.10	Availability	181
8.2.11	Management Support for Expert Involvement	182
8.2.12	Computer and AI Background	183
8.2.13	Experts as Domain Representatives	184
8.2.14	Expectations	186
8.3	The Process of Selecting Domain Experts	186
8.4	When No Experts Are Available	190
8.5	The Number of Domain Experts	192
8.5.1	Using a Single Expert	192
8.5.2	Using Multiple Experts	194
8.6	Consulting Domain Experts	195
	Checklist	197

## **9**

### **ACQUIRING THE KNOWLEDGE 199**

9.1	Considering Knowledge Acquisition at the Project's Beginning	201
9.1.1	Selecting the Domain with a View toward Knowledge Acquisition	201
9.1.2	Selecting Experts with a View toward Knowledge Acquisition	202
9.2	Knowledge Acquisition Meetings	203
9.2.1	Maximizing Access to the Experts	203
9.2.2	Allowing the Experts to Demonstrate their Expertise	204
9.2.3	Minimizing Interruptions	205
9.2.4	Accessing the Implementation	206
9.2.5	Locating Meetings at the Project Team's Site	207
9.3	Beginning the Knowledge Acquisition	208
9.3.1	Meeting Atmosphere	208
9.3.2	Focusing the Knowledge Acquisition	209
9.3.3	Getting Background Domain Knowledge	209
9.3.4	Preparing a Tutorial Document	210
9.3.5	Giving the Domain Experts Some AI Background	211
9.3.6	Using Written Materials for Initial Knowledge	211
9.3.7	Initial Steps	212
9.4	Documenting the Knowledge	213
9.4.1	Using Quasi-English Knowledge Acquisition Rules	213



9.4.2	The Knowledge Document	214
9.4.3	Readability of the Knowledge Documentation	216
9.4.4	Terminology Used in the Knowledge Documentation	216
9.4.5	Devising Terminology for Documentation and Discussion	217
9.4.6	Identifying the Knowledge Acquisition Rules and Procedures	219
9.4.7	Organizing the Knowledge Acquisition Rules and Procedures	220
9.4.8	Utilizing an Explanatory Clause	220
9.4.9	Domain Description and Glossary	221
9.5	Acquiring the Knowledge	221
9.5.1	Basic Knowledge Acquisition Cycle before Implementation	221
9.5.2	Basic Knowledge Acquisition Cycle after Implementation Has Begun	223
9.5.3	Using Test Cases to Elicit Expert Knowledge	225
9.5.4	Using Acquired Knowledge to Guide Related Knowledge Acquisition	226
9.5.5	Using Knowledge Acquisition Formalisms Directly	227
9.5.6	Updating the Knowledge Documentation	227
9.5.7	Deferring Specification of Certain Details	227
9.5.8	Generating Test Cases from Test Cases	228
9.5.9	Establishing a Default for Close Decisions	229
9.5.10	Finding the Extent of Rules	229
9.6	Recording the Knowledge	230
9.6.1	Flexibility	230
9.6.2	Recording Reminders	231
9.6.3	Recording Benefits of the Expert System	231
	Checklist	233

## **10** REPRESENTING THE SYSTEM KNOWLEDGE 237

10.1	Decomposing the Problem	238
10.2	Paradigm(s) for Representation	248
10.3	Frames, Inheritance, and Demons	249
10.4	Production Rules	254
10.5	Object-Oriented Programming	257
10.6	Programming Language Code	258
10.7	Representing and Rerepresenting	259
	Checklist	263



## **11**

### **IMPLEMENTING THE SYSTEM 265**

- 11.1 Expert System Implementation Compared with the Implementation of Conventional Programs 266**
  - 11.1.1 Implementation without a Full Specification 266**
  - 11.1.2 Reimplementing the Implementation 269**
- 11.2 Some Techniques for Knowledge Implementation 277**
  - 11.2.1 Correspondence of Knowledge Acquisition Rules and Implementation Rules 277**
  - 11.2.2 Grouping Rules in Rulesets 278**
  - 11.2.3 Implementing and Debugging 279**
  - 11.2.4 Documentation 280**
- 11.3 Managing the Implementation and Promoting Maintainability 282**
  - 11.3.1 Software Modularity and Task Assignment 283**
  - 11.3.2 Uniformity of Style 284**
  - 11.3.3 Use of Paradigms that Promote Maintainability 286**
  - 11.3.4 Configuration Management and Control 288**
  - 11.3.5 Data Flow and Access 290**
  - 11.3.6 Control Flow 292**
  - 11.3.7 Input/Output 293**
- Checklist 296**

## **12**

### **TESTING AND EVALUATING THE SYSTEM 299**

- 12.1 Validation and Verification during Expert System Development 300**
  - 12.1.1 Validation and Verification as Inherent Parts of Knowledge Acquisition 300**
  - 12.1.2 Validation Testing by Consulting Experts 302**
- 12.2 Validation of the Developed Expert System 303**
  - 12.2.1 Validating Absolutely 303**
  - 12.2.2 Validating against Expert Performance 306**
  - 12.2.3 Validation by Field Testing 307**
- 12.3 Verification of an Expert System Program 308**
- 12.4 Evaluation Effort and Standards 310**
- Checklist 312**



**13****TRANSFERRING AND DEPLOYING THE SYSTEM 315**

- 13.1** Developing an Expert System with Technology Transfer and Deployment in Mind **316**
  - 13.1.1** Issues Related to Domain Selection **316**
  - 13.1.2** Issues Related to Knowledge Representation and Implementation **318**
- 13.2** Technology Transfer **319**
- 13.3** Organizational Roles and Activities in Technology Transfer **321**
- 13.4** Transferring AI Expertise **324**
  - 13.4.1** General Training Requirements **324**
  - 13.4.2** AI Techniques, Tools, and Systems Training **325**
  - 13.4.3** Domain Training **326**
  - 13.4.4** Training on the Expert System Program **327**
  - 13.4.5** Knowledge Acquisition Training **328**
- 13.5** Deployment **328**
  - 13.5.1** Deployment Environment **328**
  - 13.5.2** Reliability, Maintainability, and Security **331**
  - 13.5.3** Integration of the Expert System **333**
- 13.6** Gaining User Acceptance of the Deployed Expert System **334**
  - 13.6.1** Educating the Users **334**
  - 13.6.2** Introducing the Expert System **336**
  - 13.6.3** Use of the Expert System **337**
- Checklist 339**

**14****EXPERT SYSTEM TRENDS AND DIRECTIONS 341**Glossary **347**Index **353**



# CHAPTER

# 1

## Introduction

- What is artificial intelligence?
- What are expert systems?
- What is knowledge engineering?
- In what kinds of situations might an expert system be used?
- What benefits might be derived from an expert system?
- Can expert systems ever perform better than the best experts?
- What expert roles can an expert system fill?
- In what fields have expert systems been developed?
- What special features and capabilities do expert systems have that distinguish them from conventional programs?
- What are the limitations of current state-of-the-art expert systems?



Interest in and excitement about the use of artificial intelligence (AI) for applications in business and industry have been growing dramatically. Practical use of AI is still relatively new, but almost all major corporations and many smaller ones are involved in AI to some extent. Many commercial and government organizations have set up their own expert system development groups; others have purchased AI products.

Probably the primary area of commercial interest in AI is the field of expert systems. Expert systems technology offers great possibilities for solving difficult real-world problems—problems that might not be amenable to solution by any other technology. By utilizing expert systems, organizations have gained valuable benefits, such as lowered costs, improved quality of products and services, increased profitability, the development or expansion of competitive advantage, and the generation of new products and services.

This book describes and discusses proven techniques for developing expert systems. Let us begin the discussion of expert system development with some background on AI and expert systems.

---

## 1.1 ARTIFICIAL INTELLIGENCE

Although AI has burst onto the commercial scene relatively recently, it has been an active discipline in an academic environment for some years; the term *artificial intelligence* was first formally used in the mid-1950s. **Artificial intelligence** can be defined as the science of making machines behave in a way that would generally be accepted as requiring human intelligence. It attempts to provide ways for modeling and mechanizing intelligent processes that otherwise could not be automated. In addition to conventional computer approaches, work in AI is often based on symbolic, nonalgorithmic methods of problem solving and the uses of computers for reasoning with concepts rather than calculating with numbers.

AI is a field of computer science. Aspects of it are closely related to several other disciplines, such as psychology, cognitive science, computational linguistics, data processing, decision support systems, and computational modeling, and it draws from all of these disciplines.

There are several important subfields of AI, including robotics, computer vision, speech synthesis and recognition, automated reasoning and theorem proving, natural language processing, automatic programming, automated learning, neural networks—and expert systems. All of these subfields are under study in universities and certain industrial laboratories, but only some have reached the stage of commercial applicability. Of these, expert systems is the subfield of AI that has evoked the most interest in business and industry and has resulted in the greatest number of practical applications.