

# **POP-11 COMES OF AGE**

**The Advancement of an  
AI Programming Language**

# **POP-11 COMES OF AGE**

## **The Advancement of an AI Programming Language**

*Editor*

**JAMES A. D. W. ANDERSON** B.Sc.(Hons.)  
Lecturer in Computer Science  
University of Reading



**ELLIS HORWOOD LIMITED**  
Publishers · Chichester

First published in 1989 by  
**ELLIS HORWOOD LIMITED**

Market Cross House, Cooper Street,  
Chichester, West Sussex, PO19 1EB, England

*The publisher's colophon is reproduced from James Gillison's drawing of the ancient Market Cross, Chichester.*

**Distributors:**

*Australia and New Zealand:*

**JACARANDA WILEY LIMITED**

GPO Box 859, Brisbane, Queensland 4001, Australia

*Canada:*

**JOHN WILEY & SONS CANADA LIMITED**

22 Worcester Road, Rexdale, Ontario, Canada

*Europe and Africa:*

**JOHN WILEY & SONS LIMITED**

Baffins Lane, Chichester, West Sussex, England

*South-East Asia*

**JOHN WILEY & SONS (SEA) PTE LIMITED**

37 Jalan Pemimpin # 05-04

Block B, Union Industrial Building, Singapore 2057

*Indian Subcontinent*

**WILEY EASTERN LIMITED**

4835/24 Ansari Road

Daryaganj, New Delhi 110002, India

© 1989 J. A. D. W. Anderson/Ellis Horwood Limited

**British Library Cataloguing in Publication Data**

POP-11 comes of age: the advancement of an AI programming language. —  
(Ellis Horwood series in artificial intelligence).

1. Computer systems. Programming languages: POP-11 language

I. Anderson, James A. D. W.

005.13'3

**Library of Congress data available**

ISBN 0-7458-0680-5 (Ellis Horwood Limited)

Printed in Great Britain by Hartnolls, Bodmin

**COPYRIGHT NOTICE**

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 the permission of Ellis Horwood Limited, Market Cross House, Cooper Street, Chichester, West Sussex, England.

## ELLIS HORWOOD SERIES IN ARTIFICIAL INTELLIGENCE

*Joint Series Editors:* Professor JOHN CAMPBELL, Department of Computer Science, University College London, and Dr JEAN HAYES MICHIE, Director of KnowledgeLink Limited; and The Turing Institute, Glasgow

- Anderson, J. (editor) POP-11 COMES OF AGE: The Advancement of an AI Programming Language  
Andrew, A.M. CONTINUOUS HEURISTICS  
Attar, A. KNOWLEDGE ENGINEERING\*  
Bläsius, K.H. and Bürckert, H.-J. DEDUCTION SYSTEMS IN AI\*  
Bramer, M.A. (editor) COMPUTER GAME PLAYING: Theory and Practice  
Campbell, J.A. (editor) IMPLEMENTATIONS OF PROLOG  
Campbell, J.A. and Cuena, J. (editors) PERSPECTIVES IN ARTIFICIAL INTELLIGENCE, Vols. 1 & 2  
Campbell, J.A. & Cox, P. IMPLEMENTATIONS OF PROLOG, Vol. 2\*  
Carter, D. INTERPRETING ANAPHORS IN NATURAL LANGUAGE TEXTS  
Davies, R. (editor) INTELLIGENT INFORMATION SYSTEMS: Progress and Prospects  
Evans, J.B. STRUCTURES OF DISCRETE EVENT SIMULATION  
Farreny, H. AI AND EXPERTISE  
Forsyth, R. & Rada, R. MACHINE LEARNING:  
Applications in Expert Systems and Information Retrieval  
Fraxione, S.G. REPRESENTING CONCEPTS IN SEMANTIC NETS  
Futó, I. & Gergely, T. ARTIFICIAL INTELLIGENCE IN SIMULATION  
Gabbay, D.M. PROGRAMMING IN PURE LOGIC\*  
Hawley, R. (editor) ARTIFICIAL INTELLIGENCE PROGRAMMING ENVIRONMENTS  
Hayes, J.E. & Michie, D. (editors) INTELLIGENT SYSTEMS: The Unprecedented Opportunity  
Levy, D.N.L. & Beal, D.F. (editors) HEURISTIC PROGRAMMING IN ARTIFICIAL INTELLIGENCE:  
The First Computer Olympiad  
Lopez de Mántaras Badía, R. APPROXIMATE REASONING MODELS  
Lukaszewicz, W. NONMONOTONIC REASONING\*  
McGraw, K. & Westphal, C. READINGS IN KNOWLEDGE ACQUISITION:  
Current Practices and Trends\*  
Mellish, C. COMPUTER INTERPRETATION OF NATURAL LANGUAGE DESCRIPTIONS  
Michie, D. ON MACHINE INTELLIGENCE, Second Edition  
Mortimer, H. THE LOGIC OF INDUCTION  
Mozetic, I. MACHINE LEARNING OF QUALITATIVE MODELS\*  
Obermeier, K.K. NATURAL LANGUAGE PROCESSING TECHNOLOGIES IN ARTIFICIAL INTELLIGENCE  
Partridge, D. ARTIFICIAL INTELLIGENCE:  
Applications in the Future of Software Engineering  
Ramsay, A. & Barrett, R. AI IN PRACTICE: Examples in POP-11  
Savory, S.E. ARTIFICIAL INTELLIGENCE AND EXPERT SYSTEMS  
Shanahan, M. & Southwick, R. SEARCH, INFERENCE AND DEPENDENCIES  
IN ARTIFICIAL INTELLIGENCE  
Spacek, L. ADVANCED PROGRAMMING IN PROLOG\*  
Sparck Jones, K. & Wilks, Y. (editors) AUTOMATIC NATURAL LANGUAGE PARSING  
Steels, L. & Campbell, J.A. (editors) PROGRESS IN ARTIFICIAL INTELLIGENCE  
Smith, B. & Kelleher, G. (editors) REASON MAINTENANCE SYSTEMS AND THEIR APPLICATIONS  
Szapkovic, S. LOGIC AND LOGIC GRAMMARS FOR LANGUAGE PROCESSING\*  
Torrance, S. (editor) THE MIND AND THE MACHINE  
Turner, R. LOGICS FOR ARTIFICIAL INTELLIGENCE  
Wallace, M. COMMUNICATING WITH DATABASES IN NATURAL LANGUAGE  
Wertz, H. AUTOMATIC CORRECTION AND IMPROVEMENT OF PROGRAMS\*  
Yazdani, M. (editor) NEW HORIZONS IN EDUCATIONAL COMPUTING  
Yazdani, M. & Narayanan, A. (editors) ARTIFICIAL INTELLIGENCE: Human Effects  
Zeidenberg, M. NEURAL NETWORK MODELS IN ARTIFICIAL INTELLIGENCE AND COGNITION

\* In preparation

# CONTENTS

PREFACE.....	8
LIST OF TABLES .....	11
LIST OF FIGURES .....	12
LIST OF EXAMPLES .....	13
1 THE ORIGINAL POP .....	16
1.1 HOW I CAME TO CREATE POP-1 .....	16
1.2 THE STANTEC ZEBRA .....	19
1.3 THE DEVELOPMENT OF POP-2.....	22
1.4 THE DEVELOPMENT OF MULTIPOP.....	26
1.5 SOURCE MATERIAL.....	28
1.6 REFERENCES.....	28
2 THE EVOLUTION OF POPLOG AND POP-11 AT SUSSEX UNIVERSITY.....	30
2.1 POP DEVELOPMENTS BEFORE SUSSEX.....	30
2.2 POP DEVELOPMENTS AT SUSSEX.....	32
2.3 THE BIRTH OF POP-11.....	35
2.4 THE MOVE TO UNIX ON THE PDP11 .....	38
2.5 FURTHER DEVELOPMENTS IN THE 70s.....	39
2.6 THE BIRTH OF POPLOG POP-11 .....	40
2.7 THE ADDITION OF PROLOG.....	43
2.8 THE ADDITION OF COMMON LISP.....	44
2.9 DYNAMIC LOCAL EXPRESSIONS .....	45
2.10 OTHER EXTENSIONS TO POP-11 .....	46
2.11 CLASSES AND OBJECT ORIENTED PROGRAMMING .....	47
2.12 SECTIONS AND IDENTIFIERS .....	47
2.13 EXTENSIONS TO THE STORE MANAGER .....	48
2.14 THE TWO-LEVEL VIRTUAL MACHINE .....	49
2.15 MISCELLANEOUS DEVELOPMENTS .....	50
2.16 PORTS TO NEW MACHINES .....	50
2.17 FUTURE DEVELOPMENTS .....	50
2.18 POP-11 AND LISP.....	53

2.19	ACKNOWLEDGEMENTS .....	54
3	REMINISCENCES OF A POP USER.....	56
3.1	GENESIS.....	56
3.2	EXODUS.....	58
3.3	LEVITICUS .....	59
3.4	NUMBERS.....	60
3.5	DEUTERONOMY .....	61
3.6	THE BOOK OF JOSHUA.....	61
4	POP-11: A CASE FOR STANDARDISATION .....	62
4.1	A SYSTEMS ANALYST ON THE ANALYST'S COUCH.....	62
4.2	GOOD ENGINEERING .....	63
4.3	SOME TYPES OF PROGRAMS.....	63
4.4	LONG-LIFE PROGRAMS .....	63
4.5	THE SOFTNESS OF SOFTWARE .....	64
4.6	PORTABILITY.....	64
4.7	COMMERCIAL PRESSURES.....	65
4.8	PRODUCT DIFFERENTIATION.....	65
4.9	CONFIDENCE AND RISK.....	65
4.10	FATUOUS CHOICE.....	66
4.11	BEING RIGHT AND BEING STANDARD.....	66
4.12	WHAT IS A STANDARD? .....	67
4.13	SCOPE OF STANDARDS.....	67
4.14	EFFECTS OF STANDARDS .....	68
4.15	POP-11'S PECULIAR POSITION.....	69
4.16	CONCLUSION .....	70
4.17	REFERENCES.....	70
5	SPEED, SPACE AND STYLE.....	71
5.1	INTRODUCTION.....	71
5.1.1	Overview.....	71
5.1.2	Structure.....	72
5.2	EXPERIENCES .....	72
5.2.1	On Not Counting Costs.....	72
5.2.2	Caveat Lector .....	73
5.2.3	POP-11's Stylistic Flexibility .....	74
5.2.3.1	FORTRAN.....	74
5.2.3.2	PASCAL .....	74
5.2.3.3	C.....	75
5.2.3.4	FORTH.....	75
5.2.3.5	LISP .....	75
5.2.3.6	Summary .....	76
5.2.4	A Case In Point .....	76
5.3	ESTIMATING COSTS .....	77
5.3.1	A Human Scale .....	77

5.3.2	A Computational Cost Model .....	78
5.3.3	An Idealised Store Model .....	78
5.3.3.1	The Cell or Box .....	79
5.3.3.2	The Value Stack .....	79
5.3.3.3	The Call Stack .....	79
5.3.3.4	The Heap .....	80
5.3.3.5	Records .....	80
5.3.3.6	Vectors .....	81
5.3.3.7	Procedures .....	82
5.3.3.8	Processes .....	82
5.3.4	POP-11's Virtual Machine .....	82
5.3.4.1	The Virtual Machine is an Abstraction .....	82
5.3.4.2	Calculating Costs .....	84
5.3.4.3	The Cost of Dynamic Localisation (Dynamic Binding) .....	85
5.3.4.4	The Cost of Lexical Variables (Static Binding) .....	86
5.4	MEASURING COSTS .....	89
5.4.1	Measurement Versus Estimation .....	89
5.4.2	Measuring Time Costs .....	89
5.4.3	Measuring Store Costs .....	90
5.5	STORE TECHNIQUES .....	93
5.5.1	Value Stack Techniques .....	93
5.5.1.1	Value Stack for Multiple Results .....	93
5.5.1.2	Value Stack as Waiting Zone .....	94
5.5.1.3	Value Stack for Argument Passing .....	96
5.5.1.4	Value Stack as Stack .....	96
5.5.1.5	Value Stack as Array .....	97
5.5.2	Call Stack Techniques .....	99
5.5.2.1	Accumulating Parameters .....	99
5.5.2.2	Avoiding Space Leaks .....	100
5.5.2.3	Using Chain .....	101
5.5.3	Heap Techniques .....	102
5.5.3.1	Avoiding Heap Space Leaks .....	102
5.5.3.2	Copy or Share? .....	103
5.5.3.3	Reusing Store and Returning Store .....	105
5.5.4	Garbage Collection Techniques .....	106
5.6	DATATYPE TECHNIQUES .....	109
5.6.1	POP-11's Datatypes .....	109
5.6.2	Number Techniques .....	109
5.6.2.1	Needless Garbage .....	109
5.6.2.2	Using Fast Integer Operations .....	110
5.6.2.3	Equality Tests on Numbers .....	110
5.6.2.4	Numbers as Bit Strings .....	110
5.6.3	Word Techniques .....	110
5.6.4	List Techniques .....	111
5.6.4.1	Lists are Easy To Use but Hard To Control .....	111
5.6.4.2	All Lists are Constructed Dynamically .....	111
5.6.4.3	Use null, not '==', 'nil', or '= nil' .....	113

## CONTENTS

5.6.5	Vector Techniques .....	113
5.6.6	String Techniques .....	114
5.6.6.1	Strings as Byte Arrays.....	114
5.6.6.2	Searching Strings .....	114
5.6.6.3	Reusing Strings .....	114
5.6.7	Procedure Techniques .....	115
5.6.7.1	Representing with Procedures.....	115
5.6.7.2	Repeaters .....	115
5.6.7.3	Properties .....	116
5.6.7.4	Arrays.....	117
5.6.8	Process Techniques .....	118
5.7	ABSTRACT DATATYPES .....	119
5.7.1	Defining Abstract Datatypes.....	119
5.7.2	Sequences.....	119
5.8	VIRTUAL MACHINE CODE PLANTING TECHNIQUES .....	121
5.8.1	Syntax Words.....	121
5.8.2	Controlling the Virtual Machine Flags .....	121
5.8.3	Non-Intrusive Techniques.....	122
5.9	ELEGANT AND EFFICIENT .....	124
5.10	THE EVOLUTION OF POP-11 .....	124
5.10.1	Profiling Tools .....	125
5.10.2	Pattern Matching.....	125
5.10.3	Arrays.....	125
5.10.4	Explicit Parse Tree .....	125
5.10.5	Type Declarations .....	126
5.10.6	Enhancements to the Virtual Machine .....	126
5.10.7	Object Oriented Type System .....	126
5.11	ACKNOWLEDGEMENTS .....	126
5.12	REFERENCES .....	127
6	POP-11 AS A TEACHING TOOL .....	128
6.1	INTRODUCTION .....	128
6.2	WHY TEACH PSYCHOLOGISTS TO PROGRAM? .....	128
6.3	WHY TEACH POP-11? .....	129
6.4	FEATURES OF POP-11 .....	131
6.4.1	Some Introductory Examples.....	131
6.4.2	An AI Example .....	135
6.4.3	Some More Advanced Features .....	136
6.5	WHAT CAN STUDENTS DO WITH POP-11?.....	138
6.6	ACKNOWLEDGEMENTS .....	140
6.7	REFERENCES .....	140
7	EMBEDDING VERY HIGH LEVEL LANGUAGES IN POP-11 .....	142
7.1	INTRODUCTION.....	142
7.2	WHAT ARE VERY HIGH LEVEL LANGUAGES? .....	143
7.3	DESIGNING A VHLL.....	144



7.4	CONCEPTUAL BASIS .....	145
7.4.1	Diagnostic Expert Systems .....	145
7.4.2	Grammar Formalisms .....	145
7.4.3	Knowledge Representation Languages .....	146
7.4.4	Linguistic Aspects of a VHLL .....	146
7.5	IMPLEMENTING A VHLL .....	148
7.5.1	Reusable Facilities .....	148
7.5.2	Language Building Tools .....	149
7.6	TWO EXAMPLES OF IMPLEMENTING VHLL's .....	149
7.6.1	An Expert System Shell .....	149
7.6.2	A Grammatical Formalism .....	153
7.7	CONCLUSION .....	158
7.8	REFERENCES .....	158
8	POP RULES OK! (A Tool For Knowledge Elicitation) .....	160
8.1	INTRODUCTION .....	160
8.2	THE APPLICATION .....	160
8.3	THE RULES SYSTEM .....	161
8.3.1	Definition of Training Sets .....	163
8.3.2	The Induction Process .....	163
8.3.3	The Output of RULES .....	165
8.4	USER ENVIRONMENT .....	167
8.4.1	The Command Interpreter .....	167
8.4.2	Presentation of Induced Rules .....	167
8.4.3	Examining Training Sets .....	168
8.4.4	Testing Environment .....	168
8.5	RULES ON POPLOG .....	168
8.6	CONCLUDING REMARKS .....	168
8.7	CONCLUSION .....	170
8.8	REFERENCES .....	170
8.9	APPENDIX: CONCEPT DEFINITION SYNTAX .....	170
9	IDEAS - FOR EXPERT SYSTEMS .....	172
9.1	INTRODUCTION .....	172
9.2	KNOWLEDGE REPRESENTATION .....	173
9.2.1.1	Frame_check .....	174
9.2.1.2	Frame_update .....	174
9.2.1.3	Create_frame .....	175
9.2.1.4	Put_frame_value .....	176
9.2.1.5	Get_frame_value .....	177
9.2.1.6	Efficiency Limitations .....	178
9.2.1.7	Associative Networks .....	178
9.2.2	The IDEAS KRL - RBFS .....	180
9.2.2.1	RBFS Frame Structure .....	181
9.2.2.2	RBFS Frame Handling Language (FRL) .....	182
9.2.2.3	RBFS Production Rules .....	187

	9.2.2.4	RBFS Inferencing .....	188
9.3		KNOWLEDGE ACQUISITION.....	190
	9.3.1	The Knowledge Acquisition Problem.....	190
	9.3.2	IDEAS for Knowledge Acquisition .....	190
	9.3.2.1	VEGAN - Visual Editor for the Generation of Associative Networks .....	191
	9.3.2.2	KET - a Knowledge Encoding Tool. ....	192
9.4		APPLICATIONS.....	194
9.5		ACKNOWLEDGEMENTS .....	195
9.6		REFERENCES.....	195
10		ON-LINE KNOWLEDGE-BASED INTERPRETATION OF INDUSTRIAL ULTRASONIC IMAGES .....	197
	10.1	INTRODUCTION.....	197
	10.2	NON-DESTRUCTIVE TESTING OF MATERIALS .....	198
	10.3	THE BLACKBOARD ARCHITECTURE .....	200
	10.4	INTERPRETATION OF AN ULTRASONIC IMAGE.....	201
	10.5	THE ARBS RULE STRUCTURE .....	202
	10.5.1	External procedures and functions .....	205
	10.5.2	The use of VALOF .....	205
	10.6	INTERNAL CONTROL ISSUES.....	206
	10.6.1	Data-driven control .....	207
	10.6.2	Goal-driven control .....	207
	10.7	MORE COMPLEX ULTRASONIC IMAGES.....	209
	10.8	CONCLUSION .....	211
	10.9	SOFTWARE AVAILABILITY .....	212
	10.10	ACKNOWLEDGEMENTS .....	212
	10.11	REFERENCES .....	212
	10.12	APPENDIX: A-SCAN RULE-BASE .....	213
11		SIMULATING MECHANICAL DEVICES .....	217
	11.1	INTRODUCTION.....	217
	11.2	MODELLING MECHANICAL DEVICES.....	218
	11.2.1	The Structural Description .....	218
	11.2.2	What Sort of Simulation?.....	222
	11.2.3	Force Outcome Decision Rules.....	223
	11.2.4	Outcome Propagation Rules.....	224
	11.3	THE FLAVOURS LIBRARY.....	225
	11.3.1	What are FLAVOURS? .....	225
	11.3.2	Defining a new flavour .....	227
	11.3.3	Instances.....	227
	11.3.4	Sending a Message.....	228
	11.3.5	Implementing Device Simulation Using FLAVOURS.....	229
	11.4	SIMULATING MECHANICAL DEVICES IN FLAVOURS .....	229
	11.4.1	Building a Device Description.....	229
	11.4.2	Running a Simulation.....	231
	11.4.3	Generating a Complete State Map .....	231

11.4.4	Performing Simple Diagnosis From The Model .....	232
11.5	POPLOG - THE RIGHT SOLUTION? .....	234
11.6	FURTHER ASPECTS OF THE RESEARCH .....	234
11.7	REFERENCES .....	235
11.8	APPENDIX: SIMULATION OF MASTIC GUN .....	235
12	POPLOG FOR WIMPS .....	238
12.1	INTRODUCTION .....	238
12.2	OVERVIEW OF WIMP-BASED INTERFACES .....	239
12.2.1	The Macintosh Toolbox .....	240
12.2.2	The Symbolics User Interface .....	242
12.3	BUILDING BETTER INTERFACES IN POPLOG .....	243
12.3.1	The ved interface .....	243
12.3.2	The POPLOG Window Manager .....	244
12.3.3	The PWM Made Easy .....	245
12.4	USING THE PWM TO BUILD WIMPS .....	246
12.4.1	Graphical Flavour Browser .....	246
12.4.2	The Design .....	247
12.4.3	The Implementation .....	247
12.4.4	System Operation .....	248
12.5	DEVICE MODEL BUILDER .....	249
12.5.1	The Design .....	250
12.5.2	The Implementation .....	250
12.5.3	System Operation .....	251
12.6	CAN THE PWM BE A WIMP? .....	252
12.7	CONCLUSION .....	252
12.8	ACKNOWLEDGEMENTS .....	252
12.9	REFERENCES .....	252
	INDEX .....	253

## PREFACE

POP was officially born in 1968 with the publication of 'The POP-2 reference manual' in *Machine Intelligence 2*. Today POP-11 is the main POP Language and, following the custom of computer languages, though not of people, it shares its ancestor's birth and celebrates its 21st Birthday this year.

The language was first conceived by Robin Popplestone as COWSEL, but following a palace revolution, was re-named POP and then POP-1. POP-1 and MULTIPOP lead to the birth of POP-2 and proliferation into a large number of versions. This early history is told in chapter one by Robin drawing on a number of contemporary internal reports. This chapter conveys something of the flavour of the times and the development of high level languages on machines with a main memory of just 56 words. Today this feat is almost unimaginable.

In chapter two, Aaron Sloman describes the development of POP-11 from these early beginnings and shows why POP-11 is a superior language to LISP in many important ways. Indeed it was one of Robin Popplestone's ambitions in writing the early POP languages to overcome some of the intrinsic design faults of LISP. One aspect of the superiority of POP-11 is its tremendous flexibility whilst retaining computational efficiency. This allowed, even *provoked*, the development of the POPLOG programming environment which includes a full COMMON LISP, PROLOG, POP-11 and VED - a powerful visual editor.

Chapter three is an amusing vignette by Christopher Longuet-Higgins on the development of POP-11 or, as he prefers, POP-10.5. Following a life time's work in science, Christopher is now the honorary President of the Poplog and Pop Languages User Group. (He is also an emeritus Professor and Fellow of the Royal Society.)

In chapter four, Jonathan Laventhol attempts to reign-in POP-11 and show how it might be standardised. He argues that the language should be established in its own right as a useful high level language, regardless of its history in AI programming and teaching. In fact POP-11 *is* being standardised and should re-appear in 1991 as Pop91.

The prime mover of the standardisation effort is Steve Knight who wrote chapter five. This is a significant chapter on efficiency issues in AI programming in

general, though, of course, it deals mainly with POP-11. Steve also gives a careful appraisal of LISP, giving it due commendation where it is superior to POP-11. Though, if Steve's proposals are taken up, some version of POP will have an explicit parse tree to match LISP's.

Chapter six, by Mike Burton, describes the role of POP-11 in teaching. This is a good place for the reader unfamiliar with POP to start: it gives clear examples and references a number of introductory texts.

Chapters seven to eleven describe applications of POP-11.

Chapter seven, by Allan Ramsay, develops one of the themes of POP-11: its flexibility. It describes how very high level languages can be implemented in POP-11. This chapter is an updated version of his chapter which appeared in 'Artificial Intelligence Programming Environments', ed. Robert Hawley, Ellis-Horwood 1987. That book also describes POP-11 and the POPLOG environment.

Chapter eight, by Colin Shearer and Karen Osbourne, describes a commercial development of an induction system implemented in POP-11 which can generate decision trees in PASCAL, C, POP-11, LISP and PROLOG, based on examples given in an internal format or in POP-11 procedures or PROLOG predicates.

Chapter nine, by Graham Winstanley, John Kellet, Jeffrey Best and Niall Teskey describes the result of many years research in the development of knowledge elicitation tools for expert systems. They give both a 'beginners' introduction to the field and indicate how effective and efficient tools can be built. This chapter shows some 'working' POP-11 code, rather than just artificial, polished examples.

Chapter ten, by Adrian Hopgood, Nicholas Hallam & Neil Woodcock, describes an expert system for the on-line interpretation of industrial ultra-sound images. In practice, this capitalises on POP-11's ability to make call-outs to external languages, such as C and FORTRAN which are supported automatically, or to arbitrary external code if the interface is hand-crafted from the general purpose tools. In recent years the external interface has proved very significant in the acceptance of POP-11 by industrial research groups of all kinds.

Chapter eleven, by Chris Price and John Hunt, describes a simulator of mechanical objects using FLAVOURS, a POP-11 library which provides an object-oriented programming language.

The remaining chapter describes an area of great future potential for POP-11.

Chapter twelve, by John Hunt and Chris Price, describes the support POP-11 gives to WIMP programming. WIMPs (Windows, Icons, Mice and Pop-up menus) have transformed the man-machine interface and are likely to lead to the ubiquitous acceptance of workstations. The existing POP-11 tools need further development before they can fully support this area.

This highlights a problem with POP-11. Because POP-11 is so flexible it attracts developments in many of the fashionable areas of AI and Computer Science, but it takes time for these to be reflected in the design of the language itself. POP-11 is unusual in this respect, that the demands of popular applications programs are supported by developments in the core language. This is nowhere more clear than in the two-level virtual machine, described in chapter two, which has instructions to support the popular applications PROLOG and LISP!

In the coming years POP-11 will face the competing demands for flexibility and computational efficiency which it has dealt with so effectively in the past. I wish POP-11 another 21 years of successful development.

Alex Morrison proposed this book, but business at his company, which sells ALPHAPOP, became so great that he could not carry the book through. I stepped in and did it.

ALPHAPOP can be bought in the United Kingdom from Cognitive Applications, 4 Sillwood Terrace, Brighton, BN1 2LR. ALPHAPOP and POPLOG, which includes POP-11, can be bought in the United States from Computable Functions Inc, 35 South Orchard Drive, Amherst, MA 01002. POPLOG can be bought in the United Kingdom from, Integral Solutions Ltd, Unit 3, Campbell Court, Bramley, Basingstoke, Hampshire, RG6 5EG. The Poplog and Pop Languages User Group can also be contacted at this address.

James Anderson

# LIST OF TABLES

1.1	COWSEL Built In Functions .....	18
2.1	For Syntax .....	51
5.1	Data Layout of a Recordclass Object.....	81
5.2	Data Layout of a String .....	81
5.3	Execution Time of Factorial .....	84
6.1	Categories of Student Projects, 1984 .....	138
10.1	ARBS Rule Syntax.....	205

## LIST OF FIGURES

5.1	Reduction of a list to a binary tree .....	106
5.2	Heap map .....	108
8.1	Operation of RULES .....	162
8.2	Generated decision tree for isgoodpub.....	164
9.1	A simple associative network .....	178
9.2	A generic model instantiated to a specific statement.....	181
9.3	N-inheritance.....	186
9.4	Z-inheritance .....	186
9.5	VEGAN display of Blocks World relationships.....	191
9.6	KET diagram of Blocks World relationships.....	193
10.1	ARBS overview .....	198
10.2	Some possible reflections of ultrasonic waves .....	201
10.3	Schematic filtered and digitised a-scan.....	201
10.4	A-scan knowledge evolving on the blackboard .....	203
10.5	Rule dependency graph for a-scan rule set .....	208
10.6	B-scan image.....	210
10.7	Lines fitted by Hough transform .....	210
10.8	Line information added to blackboard .....	210
10.9	Final conclusions.....	210
11.1	Mastic application gun .....	218
11.2	Inheritance relationship between flavours .....	226
11.3	Graphical description of mastic gun .....	230
11.4	A single state of the mastic gum .....	232
11.5	Part of the mastic gun's state map .....	233
12.1	An example of the Mackintosh interface .....	240
12.2	Some example dialogue boxes and scroll bars.....	241
12.3	An example of the Symbolics interface .....	242
12.4	An example of the PWM on a graphics workstation .....	244
12.5	The inheritance lattice for window flavours .....	245
12.6	The graphical flavour browser .....	247
12.7	The simple door lock.....	248
12.8	The device model builder.....	249
12.9	The objects in the model builder.....	251



## LIST OF EXAMPLES

1.1	member(item, list) in COWSEL .....	18
1.2	member(item, list) in ZEBRA COWSEL .....	19
1.3	Partial application with free variables.....	24
1.4	Partial application with bound variables.....	24
1.5	Fragment of a garbage collector.....	25
2.1	Pattern matching .....	36
2.2	Lexical scoping versus partial application .....	45
2.3	Subsyntax .....	51
2.4	Nested conditionals.....	53
5.1	Factorial in FORTRAN.....	74
5.2	Factorial in PASCAL .....	74
5.3	Factorial in C.....	75
5.4	Factorial in FORTH .....	75
5.5	Factorial in LISP .....	75
5.6	eat_store().....	77
5.7	A recordclass declaration.....	81
5.8	Push/pop elimination.....	83
5.9	Factorial in POP-11.....	84
5.10	Type 1 lexical binding.....	86
5.11	Factorial in POP-11 with lexical variables .....	86
5.12	first_thought_of(x) with Type 3 lexicals .....	87
5.13	first_thought_of(x) with Type 1 reference.....	87
5.14	first_thought_of(x) with Type 1 variables .....	87
5.15	Type 3 lexicals create garbage .....	88
5.16	Type 2 lexicals are garbage free .....	89
5.17	total_size(item).....	91
5.18	room(structure).....	92
5.19	lookup(index, list, default).....	93
5.20	skip() .....	94
5.21	flatten(tree).....	95
5.22	compose(f, g) .....	96
5.23	poor_compose(f, g).....	96
5.24	revzip(list).....	97