Macmillan Computer Science Series

Consulting Editor: Professor F.H. Sumner, University of Manchester

A. Abdellatif, J. Le Bihan, M. Limame, *Oracle – A User's Guide*
S.T. Allworth and R.N. Zobel, *Introduction to Real-time Software Design, second edition*
Ian O. Angell, *High-resolution Computer Graphics Using C*
Ian O. Angell and Gareth Griffith, *High-resolution Computer Graphics Using FORTRAN 77*
Ian O. Angell and Gareth Griffith, *High-resolution Computer Graphics Using Pascal*
M. Azmoodeh, *Abstract Data Types and Algorithms, second edition*
C. Bamford and P. Curran, *Data Structures, Files and Databases, second edition*
Philip Barker, *Author Languages for CAL*
A.N. Barrett and A.L. Mackay, *Spatial Structure and the Microcomputer*
R.E. Berry, B.A.E. Meekings and M.D. Soren, *A Book on C, second edition*
P. Beynon-Davies, *Information Systems Development*
G.M. Birtwistle, *Discrete Event Modelling on Simula*
B.G. Blundell, C.N. Daskalakis, N.A.E. Heyes and T.P. Hopkins, *An Introductory Guide to Silver Lisco and HILO Simulators*
B.G. Blundell and C.N. Daskalakis, *Using and Administering an Apollo Network*
T.B. Boffey, *Graph Theory in Operations Research*
Richard Bornat, *Understanding and Writing Compilers*
Linda E.M. Brackenbury, *Design of VLSI Systems – A Practical Introduction*
Alan Bradley, *Peripherals for Computer Systems*
G.R. Brookes and A.J. Stewart, *Introduction to occam 2 on the Transputer*
J.K. Buckle, *Software Configuration Management*
P.C. Capon and P.J. Jinks, *Compiler Engineering Using Pascal*
J.C. Cluley, *Interfacing to Microprocessors*
J.C. Cluley, *Introduction to Low-Level Programming for Microprocessors*
Robert Cole, *Computer Communications, second edition*
Derek Coleman, *A Structured Programming Approach to Data*
P. Davalo and E. Naim, *Neural Networks*
S.M. Deen, *Principles and Practice of Database Systems*
C. Delannoy, *Turbo Pascal Programming*
Tim Denvir, *Introduction to Discrete Mathematics for Software Engineering*
D. England et al., *A Sun User's Guide, second edition*
A.B. Fontaine and F. Barrand, *80286 and 80386 Microprocessors*
J.S. Florentin, *Microprogrammed Systems Design*
J.B. Gosling, *Design of Arithmetic Units for Digital Computers*
M.G. Hartley, M. Healey and P.G. Depledge, *Mini and Microcomputer Systems*
J.A. Hewitt and R.J. Frank, *Software Engineering in Modula-2 – An Object-oriented Approach*
Roger Hutty, *Z80 Assembly Language Programming for Students*
Roger Hutty, *COBOL 85 Programming*
Roland N. Ibbett and Nigel P. Topham, *Architecture of High Performance Computers, Volume I*
Roland N. Ibbett and Nigel P. Topham, *Architecture of High Performance Computers, Volume II*
Patrick Jaulent, *The 68000 – Hardware and Software*
P. Jaulent, L. Baticle and P. Pillot, *68020–30 Microprocessors and their Coprocessors*
J.M. King and J.P. Pardoe, *Program Design Using JSP – A Practical Introduction*

continued overleaf

E.V. Krishnamurthy, *Introductory Theory of Computer Science*
 V.P. Lane, *Security of Computer Based Information Systems*
 Graham Lee, *From Hardware to Software – An Introduction to Computers*
 Lightfoot, *Formal Specification Using Z*
 A.M. Lister and R.D. Eager, *Fundamentals of Operating Systems, fourth edition*
 Elizabeth Lynch, *Understanding SQL*
 Tom Manns and Michael Coleman, *Software Quality Assurance*
 A. Mével and T. Guéguen, *Smalltalk-80*
 R.J. Mitchell, *Microcomputer Systems Using the STE Bus*
 R.J. Mitchell, *Modula-2 Applied*
 Y. Nishinuma and R. Espesser, *UNIX – First contact*
 Pim Oets, *MS-DOS and PC-DOS – A Practical Guide, second edition*
 Pham Thu Quang and C. Chartier-Kastler, *MERISE in Practice*
 A.J. Pilavakis, *UNIX Workshop*
 Christian Queinnec, *LISP*
 E.J. Redfern, *Introduction to Pascal for Computational Mathematics*
 Gordon Reece, *Microcomputer Modelling by Finite Differences*
 F.D. Rolland, *Programming with VDM*
 W.P. Salman, O. Tisserand and B. Toulout, *FORTH*
 L.E. Scales, *Introduction to Non-Linear Optimization*
 Peter S. Sell, *Expert Systems – A Practical Introduction*
 A.G. Sutcliffe, *Human-Computer Interface Design*
 Colin J. Theaker and Graham R. Brookes, *A Practical Course on Operating Systems*
 M.R. Tolhurst et al., *Open Systems Interconnection*
 A.J. Tyrrell, *COBOL from Pascal*
 M.J. Usher, *Information Theory for Information Technologists*
 B.S. Walker, *Understanding Microprocessors*
 Colin Walls, *Programming Dedicated Microprocessors*
 I.R. Wilson and A.M. Addyman, *A Practical Introduction to Pascal – with BS6192, second edition*

Non-series

Roy Anderson, *Management, Information Systems and Computers*
 I.O. Angell, *Advanced Graphics with the IBM Personal Computer*
 B.V. Cordingley and D. Chamund, *Advanced BASIC Scientific Subroutines*
 N. Frude, *A Guide to SPSS/PC+*
 Percy Mett, *Introduction to Computing*
 Tony Royce, *COBOL – An Introduction*
 Tony Royce, *Structured COBOL – An Introduction*
 Barry Thomas, *A PostScript Cookbook*

Programming with VDM

F.D. Rolland

*Senior Lecturer,
Department of Computing,
Manchester Polytechnic*

M
MACMILLAN

© F.D. Rolland 1992

All rights reserved. No reproduction, copy or transmission of this publication may be made without written permission.

No paragraph of this publication may be reproduced, copied or transmitted save with written permission or in accordance with the provisions of the Copyright, Designs and Patents Act 1988, or under the terms of any licence permitting limited copying issued by the Copyright Licensing Agency, 90 Tottenham Court Road, London W1P 9HE.

Any person who does any unauthorised act in relation to this publication may be liable to criminal prosecution and civil claims for damages.

First published 1992 by
THE MACMILLAN PRESS LTD
Houndmills, Basingstoke, Hampshire RG21 2XS
and London
Companies and representatives
throughout the world

ISBN 978-1-349-12694-1 ISBN 978-1-349-12692-7 (eBook)
DOI 10.1007/978-1-349-12692-7

A catalogue record for this book is available
from the British Library.

Typeset by
Ponting-Green Publishing Services, Sunninghill, Berks

Contents

<i>Preface</i>	<i>vii</i>
1 Formal Specification and Object-Oriented Design	1
1.1 The need for software that works	1
1.2 Object-oriented software design	2
1.3 Object-oriented programming languages	4
1.4 The Vienna Development Method	6
1.5 Summary	9
2 Mathematical Foundations	11
2.1 Boolean logic	11
2.2 Set theory	15
2.3 Quantifiers and free variables	18
3 Functions	21
3.1 Function headings and definitions	21
3.2 Truth-valued functions	23
3.3 Implicit specifications	25
3.4 Summary	30
4 Operations	31
4.1 Functions and operations	31
4.2 Operations and states	32
4.3 The coding of operations	34
5 Composite Objects	42
5.1 Basic notation	42
5.2 Invariants	44
5.3 Specification using a simple composite object	44
5.4 Implementation of a simple composite object	46
5.5 Summary	52
6 Further Structured Data Types	53
6.1 Recursively defined structured data types	53
6.2 Sequences	61

7	Specification using Maps	67
7.1	Specifying a simple map	68
7.2	Creating, deleting and retrieving items in a map	69
7.3	Updating items in a map	74
7.4	Retrieving information from a map	76
7.5	Summary	78
8	A Simple Case Study	79
8.1	The Quick 'n Easy Email System	79
8.2	Specifying the Quick 'n Easy Email System using VDM	80
8.2.1.	The State Definition	
8.2.2	Operations on the Email state	
8.2.3	Function specifications for the Email system	
8.3	Implementation of the Email System	85
8.4	Summary	104
	Epilogue	105
	<i>Appendices:</i>	
	VDM Specification Language Syntax	107
	The Email System specified in Ada	116

Preface

There is presently much interest in industry and education in the use of Formal Methods of software development. There are many who believe, rightly in my view, that the use of Formal Methods can greatly improve the quality of software. Software quality is an increasingly important issue with the continual growth in the range and number of computer applications.

This book looks specifically at the Vienna Development Method (VDM). This is a well-documented and defined methodology that is currently subject to an international standardisation process in anticipation of its widespread commercial use. Many higher education courses in the UK use VDM as their main example of a formal methodology, and various leading software companies are making extensive use of it.

VDM is essentially a method of program specification using a mathematical form of notation. The notation is very powerful and can be used across a very wide range of applications. This text is not intended, however, to be an exhaustive study of VDM. Its intention is to provide a simple introduction to the main features of the methodology with programmed examples in Modula-2. A number of texts already exist that outline the major aspects of one or more Formal Methods of software specification, but fail to make these concrete in the form of program code derived from the specification. In my experience, it is this last stage where we move from the formal specification to actual program code that most interests and causes the major initial difficulty for new users of VDM. I aim in this text to enable readers not only to write and interpret simple VDM specifications, but also to use a simple set of guidelines for transforming these specifications into a high-level language similar to Modula-2. Although the book title is *Programming with VDM - An Introduction*, it should not be taken as intended for readers with no programming experience. The introduction is to VDM and how it can be used by the programmer. It is not an introduction to programming itself. Readers should ideally have a working knowledge of Modula-2, but those familiar with block-structured languages such as Pascal or Ada should also be able to follow the examples. A special appendix has been added for the benefit of Ada programmers.

Some consideration is also given to object-oriented programming. There are many good reasons for the increasing general interest in object-oriented

methods of software development, and VDM can certainly be used in an object-oriented manner. Modula-2 is not an object-oriented programming language as such, but it does respond very profitably to taking an object-oriented approach to program design.

It is intended that by the end of this book, the reader will be able to:

- 1 Read and understand simple VDM specifications;
- 2 Transform them into Modula-2 programs;
- 3 Describe and use some of the basic concepts of object-oriented programming;
- 4 Progress, with further study, to a full mastery of VDM.

I should point out here that the VDM syntax used in the text is a deliberate simplification of the full formal notation. There is an appendix that gives the proposed international standard for a subset of the formal syntax. The syntax used in the text is fully compatible with this and conversion to it should present no problems to the reader who has understood the text.

It is this stage that I should like to pay tribute to past and present students and colleagues at Manchester Polytechnic, whose reactions in the form of comments, criticisms, compliments, encouragements and occasional howls of dissent have played such a part in the shaping of this text. In particular, Steve Oakey kindly gave helpful comments on the opening chapter. I would like to thank the numerous anonymous reviewers commissioned by the publishers for their comments on various draft manuscripts. Thanks are also due to Tim Denvir who furnished me with the relevant BSI/IST documents regarding the VDM standard. I would also like to acknowledge the seminal influence of Sunhil Vadeera who first introduced me to the power and richness of VDM.

F.D. Rolland
Buxton, 1991