Mathematicians are very happy to discuss things that don't exist, but engineers are very unhappy about it.

— Maurice V. Wilkes (10 May 1999)

Preface

Computer design is an engineering discipline, not a mathematical one.

— Maurice V. Wilkes[1] (10 May 1999)

It is jokingly said that in any report one should address all of the thorny issues straight away, and ignore them thereafter, whether they have been resolved or not. Therefore we will make the sole mention of Y2K and the so-called "Millennium Bug" in this collection here in the preface.

This book is not about the problems that will/could/might occur on 1 January, 2000, and was never intended to be. However, concerns over that event do make the publication of this collection particularly timely. The amount of publicity that the "Y2K problem" has attracted has brought to people's attention that much of software development has, in the past, been rather *ad hoc*, often careless, and has taken place without consideration for the longevity of software, and the influence that software systems have on so many aspects of our lives. (For technical details on the specification and design of high-quality software, the reader is referred to *High-Integrity System Specification and Design* (Bowen and Hinchey, 1999), published recently, also in the FACIT series.)

Many would argue that software is never truly "engineered" in the sense that buildings, bridges, chemical plants, and, even, hardware systems are. Although they have been available for three decades now, formal methods are still too often seen as a mathematical "toy" exercise for academics. We dispute this fact, however, and we hope that the papers in this collection will help to convince skeptics that formal methods can indeed be applied at a scale suitable for industrial practice. That "proof of concept" was the goal of a previous collection, *Applications of Formal Methods* (Hinchey and Bowen, 1995). This collection goes further, however, in that rather than trying to illustrate that formal methods can be used in industrial practice, chapters in this collection aim to guide the reader in actually applying formal methods to industrial scale projects.

[1] On the occasion of the 50th anniversary of the EDSAC computer at a British Computer Society Computer Conservation Society (BCS-CCS) meeting held at the Science Museum, London.

Chapter 1, by ourselves, the editors, highlights several "sins" often committed by those attempting to apply formal methods. The chapter aims to give some general guidelines that should help in the real-life application of formal methods.

In Chapter 2, Bernard and Laffitte describe their experiences using the B-Method, a tool-supported formal software development approach, to automate systems used in conducting the French Census of Population in 1990. The system was critical in that the statistical data would be used in making political decisions. Based on their experiences, the authors discuss the relevance of B for industrial practice.

Anderson, in Chapter 3, describes a very interesting example of the use of formal methods, namely the application of the BAN logic (developed by Burrows, Abadi and Needham, to help in the formal reasoning about authentication) to an electronic payment system where security considerations are paramount. The result was a highly successful system and an example of how necessary formal methods can be in particular circumstances.

Chapter 4, by Lano, Sanchez and Goldsack describes experiences with applying the B-Method in the development of chemical process control systems. It describes the authors' experiences in applying the method, including how it can be made industrially relevant, and gives excellent guidelines for large-scale application of formal methods in process control environments.

Hardware is the focus of Chapter 5, by Brock and Hunt, which describes the application of ACL2 (A Computational Logic for Applicative Common Lisp) to the verification of the Motorola DSP Complex Arithmetic Processor (CAP). This involves a high level of assurance through the use of automated theorem proving techniques and tool support, a key issue for formalisation.

Chapter 6, by Kesten *et al.*, describes an application of formal methods to Internet security. A combination of techniques was used, which exploits the benefits of model-checking, an increasingly important issue in formal methods.

The topic of tool support is continued in Chapter 7, as Leveson, Heimdahl and Reese describe a successful CAD environment used in the development of safety-critical software systems.

The railway industry provides the application domain for Chapter 8, as Bjørner, George and Prehn report on experiences with applying the RAISE tool-supported formal development approach to the Chinese railway system. This was a particularly large project spanning over a long period of time, and involving collaboration between the railway authorities and Fellows at the United Nations University.

Jacky, in Chapter 9, reports on a large-scale critical system using the Z formal specification notation, namely University of Washington's experiences with the formal specification and development of a radiation therapy machine.

Chapter 10, by Hall, is a fascinating paper, first published in *IEEE Software*, describing experiences in the development of an air traffic control system for the London airports, and pointing out many issues that anyone embarking on using formal methods would be wise to consider.

The use of formal methods in tandem with other (less formal) notations is a growing area of interest for the formal methods community. Chapter 11, by Sem-

mens and Bryant, describes their success in using the Rigorous Review Technique, whereby formal methods (specifically the Z notation) are used to enable great insights into specifications and designs produced using structured analysis techniques and notations.

Z is also the focus of Chapter 12. Craigen, Meisels and Saaltink describe the use of the Z/EVES tool for analysing Z specifications and proving properties of them. While Z can be used simply as a notation to capture specifications unambiguously, in practical industrial use, tool-support is required to check and reason about these formal descriptions in order to help avoid human errors when large specifications are involved.

Moore, Klinker and Mihelcic describe, in Chapter 13, how to approach formal specification and verification in such a way that those certifying systems will be convinced of the validity of the approach.

Chapter 14, by Ardis and Mataga, addresses the importance of domain engineering and describes how to the tackle the ever-difficult problem of technology transfer.

Verification, and in particular tool-supported verification, is also the subject of Chapter 15, by Borälv and Stålmarck, who report on their many years of successful application of formal methods in a variety of industries.

Finally, Chapter 16, by Linger and Trammell, presents the Cleanroom approach to software engineering. The approach places high emphasis on quality, with components that fail to meet strict criteria being discarded.

Our thanks to all those who contributed to this collection, especially when one of us tormented them over PostScript figures, deadlines, and other issues that at the end of the day seem so trivial. And our special thanks to Michael Jackson for kindly agreeing to write a foreword to the collection and to Maurice Wilkes for permission to quote him at the start of the book.

We are very grateful to all at Springer-Verlag for their assistance and patience during the prepartion of this book. In particular we would like to thank Rebecca Mowat, Karen Barker, and especially Rosie Kemp. Additionally, Jane Bowen kindly proofread Chapter 1 for us.

We hope that the collection will prove to be useful to you, the reader, whether you are a student, an academic, or an industrialist wanting to know more about formal methods or how to put formal methods into practice on an industrial scale. Relevant information about this book will be kept up to date online at:

http://www.fmse.cs.reading.ac.uk/isfm/

M.G.H. J.P.B.
Omaha Reading

Contents

Foreword

Henry Buckle, the nineteenth-century English historian, asked himself why historians had been unable to discover general laws and principles and calculi of the kind and quality that mathematicians and physicists and chemists had found with such spectacular success. He concluded that historians were simply inferior to the scientists "…no one having devoted himself to history who in point of intellect is at all to be compared with Kepler, Newton, or many others …". But he also believed that within another century history would assimilate the methods and principles of natural science and would itself become a respectable science.

The earliest and most impassioned advocates of formal methods sometimes sounded like Buckle. Software development practitioners, they implied, were simply the intellectual inferiors of the researchers and academics who had devised mathematically sound development methods. Practitioners should learn and adopt formal methods and their difficulties would melt away. But these advocates, like Buckle, had failed to understand the nature of the work.

Certainly, many aspects of software development, as of history, can profitably draw on formal reasoning and calculation. But serious software developments are about the real world, and there is more in heaven and earth than is dreamt of in the philosophy of mathematical formalisms. The formal system embodied in the computer and its software must interact with the inherently informal world of human beings and physical nature, where its whole purpose resides. A very large part of the software engineering task is to analyse and describe that inherently informal world, and the system's purpose within it, well enough to ensure the practical achievement of that purpose. This is not a work of formal reasoning or refinement or calculation: it is a work of formalisation and description — the necessary prelude to the formal parts of the work.

Today's advocates of formal methods understand this much better than their impassioned predecessors, and their work is correspondingly more convincing and more valuable. The editors themselves write in their introductory chapter: "One of the most difficult aspects [in the application of formal techniques] is learning to model reality with sufficient accuracy". They recognise that "Often it is best to use formal methods with a light touch, applying them only when extra assurance is required for the development of a difficult part of a large system", and that "The process of producing the formalisation is as important as, or perhaps even more important than, the resulting specification itself …".

This, then, is not a book of dogma. It is a fruit of many substantial and successful applications of formal methods to serious and often safety-critical developments. The application areas include process control, population census, railway signalling, air traffic control, telecommunications and radiotherapy. The methods used include B, Z, VDM and CSP. Customers for the systems include large organisations already possessing a large store of software development experience.

The authors of the contributed chapters show a wide appreciation of the range of factors that may be important in a development. They recognise that development must be supported by convincing justification of the formal model; that formal reasoning must be made intelligible to the customer; and that in a world that is informal — not bounded *a priori* — testing is absolutely necessary and can never be dispensed with in favour of complete reliance on formal proofs. They also recognise that the development process itself is subject to human error, and that the resulting systems must therefore be analysed, like the products of established engineering disciplines, for their potential modes of failure.

In short, both experience and advocacy of formal methods are coming of age. This book is a rich record of much that has been learned in the progression from the naivete of childhood to the practical common sense of more mature years.

Michael A. Jackson
London, June 1999

List of Contributors

Ross J. Anderson, University of Cambridge Computer Laboratory, Cambridge, UK

Mark Ardis, Software Production Research Department, Bell Laboratories, Lucent Technologies, NJ, USA

Pascal Bernard, Philips Consumer Communications, Le Mans, France

Dines Bjørner, Technical University of Denmark, Department of Information Technology, Lyngby, Denmark

Arne Borälv, Prover Technology AB, Stockholm, Sweden

Jonathan P. Bowen, The University of Reading, Department of Computer Science, Reading, UK

Bishop C. Brock, IBM Corporation, Austin, TX, USA

Tony Bryant, Leeds Metropolitan University, Leeds, UK

Chris George, United Nations University, International Institute for Software Technology, Macau

Stephen Goldsack, Imperial College, Department of Computing, London, UK

Dan Craigen, ORA Canada, Ottawa, Ontario, Canada

Anthony Hall, Praxis, Bath, UK

Mats P.E. Heimdahl, University of Minnesota, Department of Comptuer Science and Engineering, Minneapolis, MN, USA

Mike Hinchey, University of Nebraska-Omaha, Department of Computer Science, Omaha, NE, USA

Warren A. Hunt, Jr., IBM Corporation, Austin, TX, USA

Jonathan Jacky, University of Washington, Department of Oncology, Seattle, WA, USA

Yonit Kesten, Ben Gurion University, Department of Communication Systems, Beer-Sheva, Israel

Amit Klein, Perfecto Technologies Limited, Herzelia, Israel

J. Eric Klinker, Naval Research Laboratories, Washington DC, USA

Guy Laffitte, Institut National de la Statistique et des Études Èconomique (INSEE), Nantes, France

Kevin Lano, Imperial College, Department of Computing, London, UK

Nancy G. Leveson, Department of Aeronautics and Astronautics, Massachussetts Institute of Technology, Cambridge, MA, USA

Richard C. Linger, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, USA

Peter Mataga, Software Production Research Department, Bell Laboratories, Lucent Technologies, NJ, USA

Irwin Meisels, ORA Canada, Ottawa, Ontario, Canada

David M. Milhelcic, Naval Research Laboratories, Washington D.C., USA

Andrew P. Moore, Naval Research Laboratories, Washington D.C., USA

Søren Prehn, TERMA Elektronik AS, Birkerod, Denmark

Amir Pnueli, Weizmann Institute of Science, Rehovot, Israel

Gil Raanan, Perfecto Technologies Limited, Herzelia, Israel.

Jon D. Reese, Safeware Engineering Corporation, Seattle, WA, USA

Mark Saaltink, ORA Canada, Ottawa, Ontario, Canada

Arturo Sanchez, Imperial College, Department of Chemistry, London, UK

Lesley Semmens, Leeds Metropolitan University, Leeds, UK

Gunnar Stålmarck, Prover Technology AB, Stockholm, Sweden

Carmen J. Trammell, CTI-PET Systems, Inc., Knoxville, TN, USA